aws

Panduan Pengguna

Amazon ElastiCache

Versi API 2015-02-02

Amazon ElastiCache: Panduan Pengguna

Table of Contents

Apa itu Amazon ElastiCache?

Selamat datang di Panduan ElastiCache Pengguna Amazon. Amazon ElastiCache adalah layanan web yang memudahkan pengaturan, pengelolaan, dan skala penyimpanan data dalam memori terdistribusi atau lingkungan cache di cloud. Layanan ini menyediakan solusi caching beperforma tinggi, dapat diskalakan, dan hemat biaya. Layanan ini juga membantu menghilangkan kompleksitas yang terkait deployment dan manajemen lingkungan cache terdistribusi.

Anda dapat mengoperasikan Amazon ElastiCache dalam dua format. Anda dapat memulai dengan cache nirserver atau memilih untuk merancang klaster cache Anda sendiri.

> **Note**
>
> Amazon ElastiCache bekerja dengan mesin Valkey, Memcached, dan Redis OSS. Jika Anda tidak yakin mesin yang ingin digunakan, lihat Membandingkan cache yang dirancang sendiri Valkey, Memcached, dan Redis OSS dalam panduan ini.

Caching nirserver

ElastiCache menawarkan caching tanpa server, yang menyederhanakan penambahan dan pengoperasian cache untuk aplikasi Anda. ElastiCache Tanpa server memungkinkan Anda membuat cache yang sangat tersedia dalam waktu kurang dari satu menit, dan menghilangkan kebutuhan untuk menyediakan instance atau mengkonfigurasi node atau cluster. Pengembang dapat membuat cache Tanpa Server dengan menentukan nama cache menggunakan ElastiCache konsol, SDK atau CLI.

ElastiCache Tanpa server juga menghilangkan kebutuhan untuk merencanakan dan mengelola kapasitas caching. ElastiCache terus-menerus memonitor memori cache, komputasi, dan bandwidth jaringan yang digunakan oleh aplikasi Anda, dan skala untuk memenuhi kebutuhan aplikasi Anda. ElastiCache menawarkan pengalaman endpoint sederhana bagi pengembang, dengan mengabstraksi infrastruktur cache dan desain cluster yang mendasarinya. ElastiCache mengelola penyediaan perangkat keras, pemantauan, penggantian node, dan patching perangkat lunak secara otomatis dan transparan, sehingga Anda dapat fokus pada pengembangan aplikasi, daripada mengoperasikan cache.

ElastiCache Serverless kompatibel dengan Valkey 7.2, Memcached 1.6.21 dan di atasnya, dan Redis OSS 7.1 dan di atasnya.

Merancang ElastiCache cluster Anda sendiri

Jika Anda memerlukan kontrol halus atas cluster Anda, Anda dapat memilih untuk merancang ElastiCache cluster Valkey, Memcached, atau Redis OSS Anda sendiri. ElastiCache ElastiCache memungkinkan Anda untuk mendesain cluster Anda, dengan memilih tipe node, jumlah node, dan penempatan node di seluruh AWS Availability Zones untuk cluster Anda. Karena ElastiCache merupakan layanan yang dikelola sepenuhnya, ia secara otomatis mengelola penyediaan perangkat keras, pemantauan, penggantian node, dan penambalan perangkat lunak untuk cluster Anda.

Merancang ElastiCache cluster Anda sendiri menawarkan fleksibilitas dan kontrol yang lebih besar atas cluster Anda. Misalnya, Anda dapat memilih untuk mengoperasikan klaster dengan ketersediaan AZ Tunggal atau ketersediaan multi-AZ tergantung pada kebutuhan Anda. Anda juga dapat memilih untuk menjalankan Valkey, Memcached, atau Redis OSS dalam mode cluster yang memungkinkan penskalaan horizontal, atau tanpa mode cluster hanya untuk penskalaan secara vertikal. Saat merancang klaster Anda sendiri, Anda bertanggung jawab untuk memilih jenis dan jumlah simpul dengan benar untuk memastikan bahwa cache Anda memiliki kapasitas yang cukup seperti yang dipersyaratkan oleh aplikasi Anda. Anda juga dapat memilih kapan harus menerapkan patch perangkat lunak baru ke cluster Valkey atau Redis OSS Anda.

Saat merancang ElastiCache cluster Anda sendiri, Anda dapat memilih untuk menjalankan Valkey 7.2 dan di atasnya, Memcached 1.4 ke atas, atau Redis OSS 4.0 hingga 7.1 dan di atasnya.

Layanan terkait

MemoryDB

Saat memutuskan apakah akan menggunakan ElastiCache atau MemoryDB pertimbangkan perbandingan berikut:

- ElastiCache adalah layanan yang biasa digunakan untuk cache data dari database lain dan penyimpanan data menggunakan Valkey, Memcached, atau Redis OSS. Anda harus mempertimbangkan ElastiCache beban kerja caching di mana Anda ingin mempercepat akses data dengan database utama atau penyimpanan data yang ada (kinerja baca dan tulis mikrodetik). Anda juga harus mempertimbangkan ElastiCache untuk kasus penggunaan di mana Anda ingin menggunakan struktur data Valkey atau Redis OSS dan APIs untuk mengakses data yang disimpan dalam database utama atau penyimpanan data.
- ElastiCache juga dapat membantu Anda menghemat biaya database dengan menyimpan data yang sering diakses dalam cache. Jika aplikasi Anda memiliki persyaratan throughput baca yang

tinggi, Anda dapat mencapai skala tinggi, kinerja cepat, dan menurunkan biaya penyimpanan data dengan menggunakan ElastiCache, alih-alih menskalakan basis data dasar Anda.

- MemoryDB adalah database dalam memori yang tahan lama untuk beban kerja yang membutuhkan database primer yang sangat cepat. Ini kompatibel dengan Valkey dan Redis OSS. Anda harus mempertimbangkan untuk menggunakan MemoryDB jika beban kerja Anda memerlukan basis data durabel yang memberikan performa sangat cepat (latensi baca mikrodetik dan penulisan satu digit milidetik). MemoryDB mungkin juga cocok untuk kasus penggunaan Anda jika Anda ingin membangun aplikasi menggunakan struktur data Valkey atau Redis OSS dan APIs dengan database primer yang tahan lama. Terakhir, Anda harus mempertimbangkan untuk menggunakan MemoryDB untuk menyederhanakan arsitektur aplikasi Anda dan menurunkan biaya dengan mengganti penggunaan basis data dengan cache untuk durabilitas dan performa.

Amazon Relational Database Service

ElastiCache dapat membantu Anda menghemat biaya database dengan menyimpan data yang sering diakses dalam cache. Jika aplikasi Anda memiliki persyaratan throughput baca yang tinggi, Anda dapat mencapai skala tinggi, kinerja cepat, dan menurunkan biaya penyimpanan data dengan menggunakan ElastiCache, alih-alih menskalakan basis data dasar Anda.

Untuk informasi latar belakang lebih lanjut tentang layanan Amazon Relational Database Service terkait, lihat Amazon RDS

ElastiCache dapat membantu Anda menghemat biaya database dengan menyimpan data yang sering diakses dalam cache. Jika aplikasi Anda memiliki persyaratan throughput baca yang tinggi, Anda dapat mencapai skala tinggi, kinerja cepat, dan menurunkan biaya penyimpanan data dengan menggunakan ElastiCache, alih-alih menskalakan basis data dasar Anda.

Bagaimana cara ElastiCache kerja

Di sini Anda dapat menemukan ikhtisar komponen utama ElastiCache penerapan.

Mesin cache dan caching

Cache adalah penyimpanan data dalam memori yang dapat Anda gunakan untuk menyimpan data yang di-cache. Biasanya, aplikasi Anda akan menyimpan data yang sering diakses dalam cache untuk mengoptimalkan waktu respons. ElastiCache menawarkan dua opsi penerapan: Cluster tanpa server dan yang dirancang sendiri. Lihat Memilih antara opsi deployment.

> **Note**
>
> Amazon ElastiCache bekerja dengan mesin Valkey, Memcached, dan Redis OSS. Jika Anda tidak yakin mesin yang ingin digunakan, lihat Membandingkan cache yang dirancang sendiri Valkey, Memcached, dan Redis OSS dalam panduan ini.

Topik

Bagaimana cara ElastiCache kerja

ElastiCache Nirserver

ElastiCache Tanpa server memungkinkan Anda membuat cache tanpa khawatir tentang perencanaan kapasitas, manajemen perangkat keras, atau desain cluster. Anda cukup memberikan nama untuk cache Anda dan Anda menerima satu titik akhir yang dapat Anda konfigurasi di klien Valkey, Memcached, Redis OSS Anda untuk mulai mengakses cache Anda.

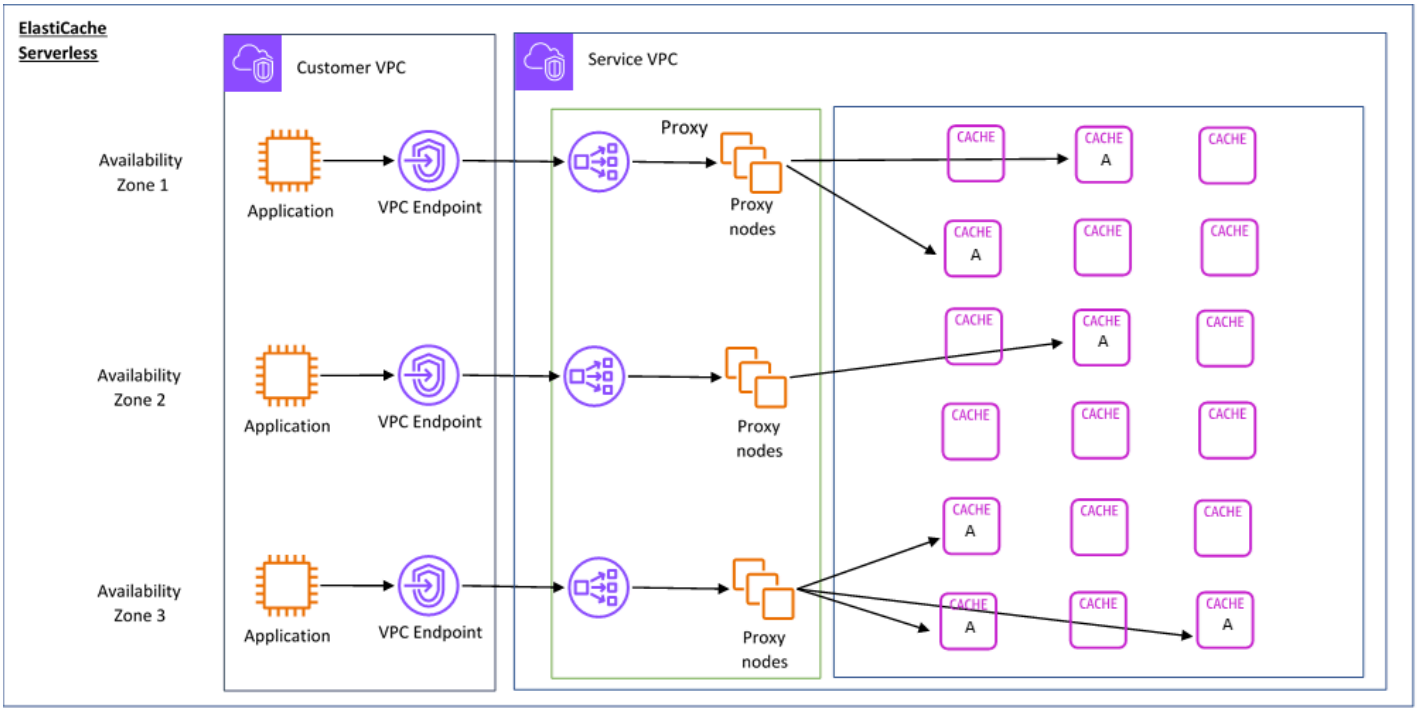
> **Note**
>
> - ElastiCache Tanpa server menjalankan Valkey, Memcached, atau Redis OSS dalam mode cluster dan hanya kompatibel dengan klien yang mendukung TLS.

Manfaat Utama

- Tidak ada perencanaan kapasitas: ElastiCache Tanpa server menghilangkan kebutuhan Anda untuk merencanakan kapasitas. ElastiCache Tanpa server terus memantau memori, komputasi, dan pemanfaatan bandwidth jaringan cache Anda dan menskalakan baik secara vertikal maupun horizontal. Hal ini memungkinkan simpul cache untuk tumbuh dalam ukuran, sekaligus secara paralel memulai operasi menskalakan ke luar untuk memastikan bahwa cache dapat diskalakan untuk memenuhi persyaratan aplikasi Anda setiap saat.

- Pay-per-use: Dengan ElastiCache Tanpa Server, Anda membayar untuk data yang disimpan dan menghitung yang digunakan oleh beban kerja Anda pada cache. Lihat Dimensi harga.
- Ketersediaan tinggi: ElastiCache Tanpa server secara otomatis mereplikasi data Anda di beberapa Availability Zone (AZ) untuk ketersediaan tinggi. Layanan ini secara otomatis memantau simpul cache yang mendasarinya dan menggantinya jika terjadi kegagalan. Hal ini menawarkan SLA ketersediaan 99,99% untuk setiap cache.
- Upgrade perangkat lunak otomatis: ElastiCache Tanpa server secara otomatis meningkatkan cache Anda ke versi perangkat lunak minor dan patch terbaru tanpa dampak ketersediaan apa pun pada aplikasi Anda. Ketika versi utama baru tersedia, ElastiCache akan mengirimkan pemberitahuan.
- Keamanan: Nirserver selalu mengenkripsi data bergerak dan diam. Anda dapat menggunakan kunci yang dikelola layanan atau menggunakan Kunci Dikelola Pelanggan Anda sendiri untuk mengenkripsi data diam.

Diagram berikut menggambarkan bagaimana ElastiCache Serverless bekerja.



Saat Anda membuat cache tanpa server baru, ElastiCache buat Titik Akhir Virtual Private Cloud (VPC) Virtual Private Cloud (VPC) di subnet pilihan Anda di VPC Anda. Aplikasi Anda dapat terhubung ke cache melalui Titik Akhir VPC ini.

Dengan ElastiCache Tanpa Server Anda menerima satu titik akhir DNS yang terhubung dengan aplikasi Anda. Saat Anda meminta koneksi baru ke titik akhir, ElastiCache Tanpa Server menangani semua koneksi cache melalui lapisan proxy. Lapisan proksi membantu mengurangi konfigurasi klien yang kompleks, karena klien tidak perlu menemukan kembali topologi klaster jika terjadi perubahan pada klaster yang mendasarinya. Lapisan proksi adalah sekumpulan simpul proksi yang menangani koneksi menggunakan penyeimbang beban jaringan.

Saat aplikasi Anda membuat koneksi cache baru, permintaan dikirim ke simpul proksi oleh penyeimbang beban jaringan. Ketika aplikasi Anda mengeksekusi perintah cache, simpul proksi yang terhubung ke aplikasi Anda mengeksekusi permintaan di simpul cache di cache Anda. Lapisan proksi mengabstraksi topologi klaster cache dan simpul dari klien Anda. Hal ini memungkinkan ElastiCache untuk secara cerdas memuat keseimbangan, skala keluar dan menambahkan node cache baru, mengganti node cache ketika mereka gagal, dan memperbarui perangkat lunak pada node cache, semua tanpa dampak ketersediaan ke aplikasi Anda atau harus mengatur ulang koneksi.

Cluster yang dirancang sendiri ElastiCache

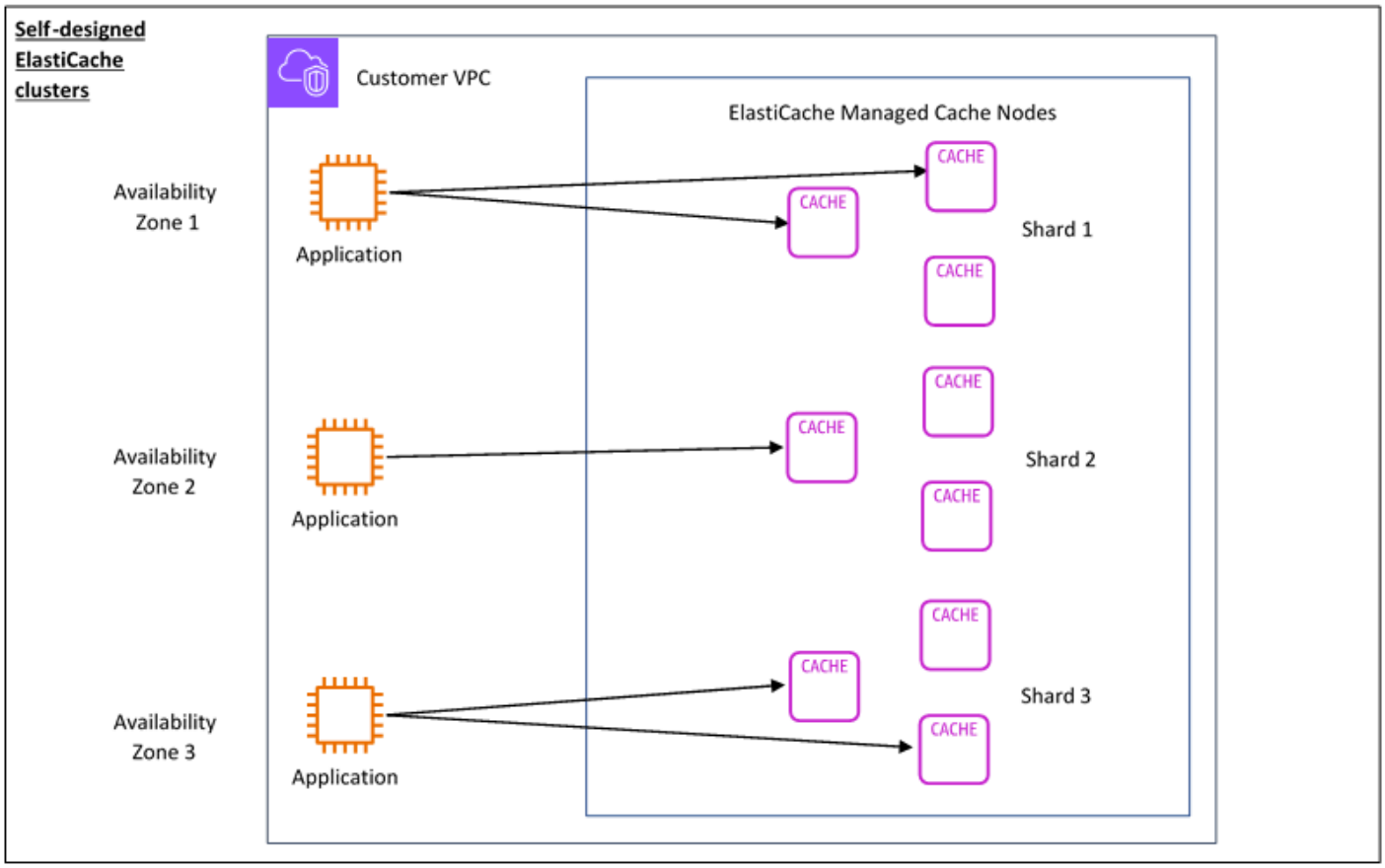
Anda dapat memilih untuk mendesain ElastiCache cluster Anda sendiri dengan memilih n cache node keluarga, ukuran, dan jumlah node untuk cluster Anda. Merancang klaster Anda sendiri memberi Anda kontrol yang lebih terperinci dan memungkinkan Anda memilih jumlah serpihan di cache Anda dan jumlah simpul (primer dan replika) di setiap serpihan. Anda dapat memilih untuk mengoperasikan Valkey atau Redis OSS dalam mode cluster dengan membuat cluster dengan beberapa pecahan, atau dalam mode non-cluster dengan pecahan tunggal.

Manfaat Utama

- Desain cluster Anda sendiri: Dengan ElastiCache, Anda dapat mendesain cluster Anda sendiri dan memilih di mana Anda ingin menempatkan node cache Anda. Misalnya, jika Anda memiliki aplikasi yang ingin menukar ketersediaan tinggi dengan latensi rendah, Anda dapat memilih untuk menerapkan simpul cache Anda dalam satu AZ. Atau, Anda dapat mendesain cluster Anda dengan node di beberapa AZs untuk mencapai ketersediaan tinggi.
- Kontrol terperinci: Saat mendesain klaster sendiri, Anda memiliki kontrol yang lebih besar dalam menyempurnakan pengaturan cache Anda. Misalnya, Anda dapat menggunakan Parameter Valkey dan Redis OSS atau Parameter spesifik Memcached mengkonfigurasi mesin cache.
- Menskalakan secara vertikal dan horizontal: Anda dapat memilih untuk menskalakan klaster Anda secara manual dengan menambah atau mengurangi ukuran simpul cache saat diperlukan. Anda juga dapat menskalakan secara horizontal dengan menambahkan serpihan baru atau menambahkan lebih banyak replika ke serpihan Anda. Anda juga dapat menggunakan fitur Auto-

Scaling untuk mengonfigurasi penskalaan berdasarkan jadwal atau penskalaan berdasarkan metrik seperti penggunaan CPU dan Memori pada cache.

Diagram berikut menggambarkan bagaimana cluster yang ElastiCache dirancang sendiri bekerja.



Dimensi harga

Anda dapat menerapkan ElastiCache dalam dua opsi penerapan. Saat menerapkan ElastiCache Tanpa Server, Anda membayar penggunaan untuk data yang disimpan dalam GB-jam dan menghitung di ElastiCache Unit Pemrosesan (ECPU). Saat memilih untuk mendesain ElastiCache cluster Anda sendiri, Anda membayar per jam penggunaan node cache. Lihat detail harga di sini.

Penyimpanan data

Anda membayar data yang disimpan dalam ElastiCache Serverless yang ditagih dalam gigabyte-hours (GB-HRs). ElastiCache Tanpa server terus memantau data yang disimpan dalam cache Anda, mengambil sampel beberapa kali per menit, dan menghitung rata-rata per jam untuk menentukan

penggunaan penyimpanan data cache dalam GB-jam. Setiap cache ElastiCache Tanpa Server diukur untuk minimal 1 GB data yang disimpan.

ElastiCache Unit Pengolahan (ECPUs)

Anda membayar permintaan yang dijalankan aplikasi Anda di ElastiCache Tanpa Server di Unit ElastiCache Pemrosesan (ECPUs), unit yang mencakup waktu vCPU dan data yang ditransfer.

- Bacaan dan penulisan sederhana membutuhkan 1 ECPU untuk setiap kilobyte (KB) data yang ditransfer. Misalnya, perintah GET yang mentransfer hingga 1 KB data mengonsumsi 1 ECPU. Permintaan SET yang mentransfer 3,2 KB data akan mengkonsumsi 3,2 ECPUs.
- Dengan Valkey dan Redis OSS, perintah yang mengkonsumsi lebih banyak waktu vCPU dan mentransfer lebih banyak konsumsi data ECPUs berdasarkan yang lebih tinggi dari dua dimensi. Misalnya, jika aplikasi Anda menggunakan perintah HMGET, mengkonsumsi 3 kali waktu vCPU sebagai SET/GET perintah sederhana, dan mentransfer 3,2 KB data, itu akan mengkonsumsi 3.2 ECPU. Atau, jika mentransfer hanya 2 KB data, itu akan mengkonsumsi 3 ECPUs.
- Dengan Valkey dan Redis OSS, perintah yang membutuhkan waktu vCPU tambahan akan mengkonsumsi lebih banyak secara proporsional. ECPUs Misalnya, jika aplikasi Anda menggunakan perintah Valkey atau Redis OSS HMGET, dan mengkonsumsi 3 kali waktu vCPU sebagai perintah SET/GET sederhana, maka itu akan mengkonsumsi 3. ECPUs
- Dengan Memcached, perintah yang beroperasi pada beberapa item akan mengkonsumsi secara proporsional lebih banyak. ECPUs Misalnya, jika aplikasi Anda melakukan multiget pada 3 item, itu akan mengkonsumsi 3 ECPUs.
- Dengan Memcached, perintah yang beroperasi pada lebih banyak item dan mentransfer lebih banyak data mengkonsumsi ECPUs berdasarkan yang lebih tinggi dari dua dimensi. Sebagai contoh, jika aplikasi Anda menggunakan perintah GET, mengambil 3 item, dan mentransfer 3,2 KB data, maka aplikasi tersebut akan menghabiskan 3,2 ECPU. Atau, jika mentransfer hanya 2 KB data, itu akan mengkonsumsi 3 ECPUs.

ElastiCache Tanpa server memancarkan metrik baru yang disebut `ElastiCacheProcessingUnits` yang membantu Anda memahami ECPUs konsumsi beban kerja Anda.

Jam simpul

Anda dapat memilih untuk mendesain cluster cache Anda sendiri dengan memilih keluarga EC2 node, ukuran, jumlah node, dan penempatan di seluruh Availability Zones. Saat mendesain sendiri klaster Anda, Anda membayar per jam untuk setiap simpul cache.

ElastiCache cadangan

Cadangan adalah point-in-time salinan cache tanpa server, atau cluster yang dirancang sendiri Valkey atau Redis OSS. ElastiCache memungkinkan Anda untuk mengambil cadangan data Anda kapan saja atau mengatur cadangan otomatis. Cadangan dapat digunakan untuk memulihkan cache yang sudah ada atau melakukan seeding cache baru. Cadangan terdiri dari semua data dalam cache ditambah beberapa metadata. Untuk mengetahui informasi selengkapnya, lihat Melakukan snapshot dan pemulihan.

Memilih antara opsi deployment

Amazon ElastiCache memiliki dua opsi penerapan:

- Caching nirserver
- Klaster yang dirancang sendiri

Untuk daftar perintah yang didukung untuk keduanya, lihatPerintah Valkey, Memcached, dan Redis OSS yang didukung dan dibatasi.

Caching nirserver

Amazon ElastiCache Serverless menyederhanakan pembuatan cache dan menskalakan secara instan untuk mendukung aplikasi pelanggan yang paling menuntut. Dengan ElastiCache Tanpa Server, Anda dapat membuat cache yang sangat tersedia dan dapat diskalakan dalam waktu kurang dari satu menit, menghilangkan kebutuhan untuk menyediakan, merencanakan, dan mengelola kapasitas cluster cache. ElastiCache Tanpa server secara otomatis menyimpan data secara berlebihan di tiga Availability Zone dan menyediakan Perjanjian Tingkat Layanan (SLA) ketersediaan 99,99%. Cadangan dari cluster Valkey atau Redis OSS yang dirancang sendiri dapat dikembalikan ke konfigurasi tanpa server.

Klaster yang dirancang sendiri

Jika Anda memerlukan kontrol halus atas cluster Valkey, Memcached, atau Redis OSS Anda, Anda dapat memilih untuk mendesain cluster Anda sendiri. ElastiCache ElastiCache memungkinkan Anda untuk mengoperasikan cluster berbasis node, dengan memilih tipe node, jumlah node, dan penempatan node di seluruh AWS Availability Zones untuk cluster Anda. Karena ElastiCache merupakan layanan yang dikelola sepenuhnya, ini membantu mengelola penyediaan perangkat keras, pemantauan, penggantian node, dan penambalan perangkat lunak untuk cluster Anda. Cluster

yang dirancang sendiri dapat dirancang untuk menyediakan SLA ketersediaan hingga 99,99%. Cadangan dari cache Valkey atau Redis OSS tanpa server dapat dikembalikan ke cluster yang dirancang sendiri.

Memilih antara opsi deployment

Pilih caching nirserver jika:

- Anda membuat cache untuk beban kerja yang baru atau sulit diprediksi.
- Anda memiliki lalu lintas aplikasi yang tak terduga.
- Anda ingin cara termudah untuk memulai dengan cache.

Pilih untuk mendesain ElastiCache cluster Anda sendiri jika:

- Anda sudah menjalankan ElastiCache Serverless dan menginginkan kontrol yang lebih halus atas jenis node yang menjalankan Valkey, Memcached, atau Redis OSS, jumlah node, dan penempatan node tersebut.
- Anda mengharapkan lalu lintas aplikasi Anda relatif dapat diprediksi, dan Anda ingin kontrol yang baik atas kinerja, ketersediaan, dan biaya.
- Anda dapat memperkirakan persyaratan kapasitas Anda untuk mengontrol biaya.

Membandingkan caching tanpa server dan cluster yang dirancang sendiri

Fitur	Caching nirserver	Klaster yang dirancang sendiri
Pengaturan cache	Buat cache hanya dengan nama dalam waktu kurang dari satu menit	Menyediakan kontrol halus atas desain cluster cache. Pengguna dapat memilih tipe node, jumlah node, dan penempatan di seluruh zona ketersediaan AWS
ElastiCache Versi yang didukung	Valkey 7.2 dan lebih tinggi, Redis OSS versi 7.1 dan lebih tinggi, Memcached 1.6.21 dan lebih tinggi	Valkey 7.2 dan lebih tinggi, Redis OSS versi 4.0 dan lebih tinggi, Memcached 1.4 dan lebih tinggi

Fitur	Caching nirserver	Klaster yang dirancang sendiri
Mode Cluster (Valkey dan Redis OSS)	Mengoperasikan mesin `cluster mode enabled` hanya di. Klien harus mendukung `cluster mode enabled` untuk terhubung ke Tanpa ElastiCache Server.	Dapat dikonfigurasi untuk beroperasi dalam mode cluster diaktifkan atau mode cluster dinonaktifkan.
Penskalaan	Secara otomatis menskalakan mesin baik secara vertikal maupun horizontal tanpa manajemen kapasitas apa pun.	Memberikan kontrol atas penskalaan, sementara juga membutuhkan pemantauan untuk memastikan kapasitas saat ini cukup memenuhi permintaan. Untuk Valkey dan Redis OSS, Anda dapat memilih untuk menskalakan secara vertikal dengan menambah atau mengurangi ukuran node cache saat diperlukan. Anda juga dapat menskalakan secara horizontal, dengan menambahkan pecahan baru atau menambahkan lebih banyak replika ke pecahan Anda. Kemampuan ini tidak tersedia untuk Memcached. Dengan fitur Auto-Scaling, Anda juga dapat mengonfigurasi penskalaan berdasarkan jadwal, atau skala berdasarkan metrik seperti penggunaan CPU dan Memori pada cache.

Fitur	Caching nirserver	Klaster yang dirancang sendiri
Koneksi klien	Klien terhubung ke satu titik akhir. Ini memungkinkan topologi node cache yang mendasarinya (penskalaan, penggantian, dan peningkatan) berubah tanpa memutuskan sambungan klien.	Klien terhubung ke setiap node cache individu. Jika node diganti, klien menemukan kembali topologi cluster dan membangun kembali koneksi.
Konfigurasi	Tidak ada konfigurasi berbutir halus yang tersedia. Pelanggan dapat mengkonfigurasi pengaturan dasar termasuk subnet yang dapat mengakses cache, apakah backup otomatis diaktifkan atau dimatikan, dan batas penggunaan cache maksimum.	Cluster yang dirancang sendiri menyediakan opsi konfigurasi berbutir halus. Pelanggan dapat menggunakan grup parameter untuk kontrol berbutir halus. Untuk mengetahui tabel nilai parameter ini berdasarkan jenis simpul, lihat Parameter spesifik mesin.
Multi-AZ	Data direplikasi secara asinkron di beberapa Availability Zone untuk ketersediaan yang lebih tinggi dan latensi baca yang lebih baik.	Menyediakan opsi untuk mendesain cluster dalam satu Availability Zone atau di beberapa Availability Zones (AZs). Saat menggunakan Valkey atau Redis OSS, menyediakan cluster multi-AZ dengan data yang direplikasi secara asinkron di beberapa Availability Zone untuk ketersediaan yang lebih tinggi dan latensi baca yang lebih baik.

Fitur	Caching nirserver	Klaster yang dirancang sendiri
Enkripsi diam	Selalu diaktifkan. Pelanggan dapat menggunakan Kunci yang dikelola AWS atau kunci yang dikelola pelanggan AWS KMS.	Opsi untuk mengaktifkan atau menonaktifkan enkripsi saat istirahat. Ketika diaktifkan, pelanggan dapat menggunakan Kunci yang dikelola AWS atau kunci yang dikelola pelanggan AWS KMS.
Enkripsi dalam perjalanan (TLS)	Selalu diaktifkan. Klien harus mendukung konektivitas TLS.	Opsi untuk mengaktifkan atau menonaktifkan.
Pencadangan	Mendukung backup cache otomatis dan manual tanpa dampak kinerja. Cadangan Valkey dan Redis OSS kompatibel silang, dan dapat dikembalikan ke cache Tanpa ElastiCache Server atau cluster yang dirancang sendiri.	Mendukung backup otomatis dan manual untuk Valkey dan Redis OSS. Cluster mungkin melihat beberapa dampak kinerja tergantung pada memori cadangan yang tersedia. Untuk informasi selengkapnya, lihat Mengelola memori cadangan untuk Valkey dan Redis OSS. Cadangan Valkey dan Redis OSS kompatibel silang, dan dapat dikembalikan ke cache Tanpa ElastiCache Server atau cluster yang dirancang sendiri.

Fitur	Caching nirserver	Klaster yang dirancang sendiri
Pemantauan	Mendukung metrik tingkat cache termasuk tingkat hit cache, tingkat kehilanga n cache, ukuran data, dan ECPUs konsumsi. ElastiCache Tanpa server mengirim peristiwa menggunakan EventBridge saat peristiwa penting terjadi di cache Anda. Anda dapat memilih untuk memantau, menelan, mengubah, dan menindaklanjuti ElastiCache acara menggunakan Amazon EventBridge. Untuk informasi selengkapnya, lihat Peristiwa cache nirserver.	ElastiCache Cluster yang dirancang sendiri memancark an metrik di setiap tingkat node, termasuk metrik tingkat host dan metrik cache. Cluster yang dirancang sendiri memancarkan pemberitahuan SNS untuk acara penting. Lihat Metrik untuk Memcached dan Metrik untuk Valkey dan Redis OSS.
Ketersediaan	99,99% ketersediaan Perjanjia n Tingkat Layanan (SLA)	Cluster yang dirancang sendiri dapat dirancang untuk mencapai ketersediaan hingga 99,99% Perjanjian Tingkat Layanan (SLA), tergantung pada konfigurasi.

Fitur	Caching nirserver	Klaster yang dirancang sendiri
Peningkatan dan penambalan perangkat lunak	Secara otomatis memutakhirkan perangkat lunak cache ke versi minor dan patch terbaru, tanpa dampak aplikasi. Pelanggan menerima pemberitahuan untuk peningkatan versi utama, dan pelanggan dapat meningkatkan ke versi utama terbaru kapan pun mereka mau.	Cluster yang dirancang sendiri menawarkan layanan mandiri yang diaktifkan pelanggan untuk peningkatan versi minor dan patching, serta peningkatan versi utama. Pembaruan terkelola diterapkan secara otomatis selama jendela pemeliharaan yang ditentukan pelanggan. Pelanggan juga dapat memilih untuk menerapkan upgrade versi minor atau patch sesuai permintaan.
Toko Data Global	Tidak didukung	Mendukung Global Data Store, yang memungkinkan replikasi lintas wilayah dengan penulisan wilayah tunggal dan pembacaan multi-wilayah
Tingkat Data	Tidak didukung	Cluster yang dirancang menggunakan node dari keluarga r6gd memiliki data berjenjang antara memori dan penyimpanan SSD lokal (solid state drive). Tiering data menyediakan opsi harga-kinerja untuk beban kerja Valkey dan Redis OSS dengan memanfaatkan solid state drive (SSDs) berbiaya lebih rendah di setiap node cluster, selain menyimpan data dalam memori.

Fitur	Caching nirserver	Klaster yang dirancang sendiri
Model penentuan harga	Pay-per-use, berdasarkan data yang disimpan dalam GB-jam dan permintaan di Unit ElastiCache Pemrosesan (ECPU). Lihat detail harga di sini.	Pay-per-hour, berdasarkan penggunaan node cache. Lihat detail harga di sini.

Topik terkait:

- Merancang dan mengelola ElastiCache cluster Anda sendiri

ElastiCache Sumber daya Amazon untuk pengguna pertama kali

Kami menyarankan agar pengguna pertama kali memulai dengan membaca bagian berikut, dan merujuknya sesuai kebutuhan.

- Sorotan dan harga layanan - Halaman detail produk memberikan gambaran umum produk ElastiCache, sorotan layanan, dan harga.
- ElastiCache video - ElastiCache Video Bagian ini memiliki video yang memperkenalkan Anda ke Amazon ElastiCache. Video mencakup kasus penggunaan umum ElastiCache dan demo cara penggunaan ElastiCache untuk mengurangi latensi dan meningkatkan throughput untuk aplikasi Anda.
- Memulai – Bagian Memulai dengan Amazon ElastiCache mencakup informasi tentang membuat klaster cache. Ini juga mencakup cara memberikan otorisasi akses ke klaster cache, terhubung ke simpul cache, dan menghapus klaster cache.
- Kinerja dalam skala besar — Kinerja pada skala dengan ElastiCache whitepaper Amazon membahas strategi caching yang membantu aplikasi Anda berkinerja baik dalam skala besar.

Setelah Anda menyelesaikan bagian terdahulu, baca bagian ini:

- Memilih ukuran simpul Anda

  Anda menginginkan simpul Anda cukup besar untuk mengakomodasi semua data yang ingin dijadikan cache. Pada saat yang sama, Anda tidak ingin mengeluarkan biaya untuk kelebihan

cache yang tidak diperlukan. Anda dapat menggunakan topik ini untuk membantu Anda dalam memilih ukuran simpul yang terbaik.

- ElastiCache praktik terbaik dan strategi caching

  Identifikasi dan atasi masalah yang dapat mempengaruhi efisiensi klaster Anda.

Jika Anda ingin menggunakan AWS Command Line Interface (AWS CLI), Anda dapat menggunakan dokumen-dokumen ini untuk membantu Anda memulai:

- AWS Command Line Interface dokumentasi

  Bagian ini memberikan informasi tentang mengunduh AWS CLI, mendapatkan AWS CLI pekerjaan pada sistem Anda, dan memberikan AWS kredensi Anda.

- AWS CLI dokumentasi untuk ElastiCache

  Dokumen terpisah ini mencakup semua ElastiCache perintah AWS CLI for, termasuk sintaks dan contoh.

Anda dapat menulis program aplikasi untuk menggunakan ElastiCache API dengan berbagai bahasa pemrograman populer. Berikut adalah beberapa sumber daya:

- Alat untuk Amazon Web Services

  Amazon Web Services menyediakan sejumlah kit pengembangan perangkat lunak (SDKs) dengan dukungan untuk ElastiCache. Anda dapat membuat kode untuk ElastiCache menggunakan Java, .NET, PHP, Ruby, dan bahasa lainnya. Ini SDKs dapat sangat menyederhanakan pengembangan aplikasi Anda dengan memformat permintaan Anda ElastiCache, mengurai respons, dan memberikan logika coba lagi dan penanganan kesalahan.

- Menggunakan ElastiCache API

  Jika Anda tidak ingin menggunakan AWS SDKs, Anda dapat berinteraksi dengan ElastiCache langsung menggunakan Query API. Pada bagian ini, Anda dapat menemukan tips pemecahan masalah dan informasi tentang membuat dan melakukan autentikasi atas permintaan serta menangani tanggapan.

- Amazon ElastiCache API Referensi

  Dokumen terpisah ini mencakup semua operasi ElastiCache API, termasuk sintaks dan contoh.

AWS Wilayah dan Zona Ketersediaan

Sumber daya komputasi cloud Amazon berlokasi di fasilitas pusat data dengan ketersediaan tinggi di berbagai wilayah di dunia (misalnya, Amerika Utara, Eropa, atau Asia). Setiap lokasi pusat data disebut AWS Region.

Setiap AWS Wilayah berisi beberapa lokasi berbeda yang disebut Availability Zone, atau AZs. Setiap Zona Ketersediaan dirancang agar terisolasi dari kegagalan di Zona Ketersediaan yang lain. Masing-masing dirancang untuk menyediakan konektivitas jaringan latensi rendah yang murah ke Availability Zone lainnya di Wilayah yang sama. AWS Dengan meluncurkan instans di Zona Ketersediaan yang terpisah, Anda dapat melindungi aplikasi Anda dari kegagalan di satu lokasi. Untuk informasi selengkapnya, lihat Memilih wilayah dan zona ketersediaan.

Anda dapat menjalankan klaster Anda di beberapa Zona Ketersediaan, yang merupakan opsi yang disebut deployment Multi-AZ. Saat Anda memilih opsi ini, Amazon secara otomatis menyediakan dan memelihara instans simpul siaga sekunder pada Zona Ketersediaan yang berbeda. Instans simpul primer Anda direplikasi secara asinkron di seluruh Zona Ketersediaan ke instans sekunder. Pendekatan ini membantu memberikan redundansi data dan dukungan failover, menghilangkan I/O pembekuan, dan meminimalkan lonjakan latensi selama pencadangan sistem. Untuk informasi selengkapnya, lihat Meminimalkan waktu henti ElastiCache untuk Valkey dan Redis OSS dengan Multi-AZ.

Kasus ElastiCache Penggunaan Umum dan Bagaimana ElastiCache Dapat Membantu

Baik ketika menyajikan berita terkini, papan peringkat 10 teratas, katalog produk, maupun menjual tiket suatu acara, kecepatan adalah aspek yang terpenting. Keberhasilan situs web dan bisnis Anda sangat dipengaruhi oleh kecepatan Anda dalam menyediakan konten.

Dalam artikel "For Impatient Web Users, an Eye Blink Is Just Too Long to Wait", New York Times menjelaskan bahwa pengguna dapat merasakan perbedaan 250 milidetik (1/4 detik) di antara beberapa situs yang bersaing. Pengguna cenderung lebih memilih situs yang lebih cepat daripada situs yang lambat. Pengujian yang dilakukan di Amazon, yang dikutip dalam artikel How Webpage Load Time Is Related to Visitor Loss, mengungkapkan bahwa untuk setiap 100 milidetik (1/10 detik) penambahan waktu pemuatan, penjualan turun 1 persen.

Jika seseorang menginginkan data, Anda dapat mengirimkan data tersebut lebih cepat jika disimpan dalam cache. Hal tersebut berlaku baik untuk halaman web maupun laporan yang mendorong keputusan bisnis. Apakah bisnis Anda mampu menampilkan halaman web Anda dengan latensi yang sesingkat mungkin tanpa menyimpannya dalam cache?

Tanpa perlu dipikir lagi, Anda pasti ingin meng-cache item Anda yang paling banyak diminta. Namun, mengapa tidak meng-cache item yang kurang sering diminta? Bahkan kueri atau panggilan API jarak jauh ke basis data yang paling dioptimalkan sekalipun akan terasa lebih lambat daripada mengambil satu kunci datar dari cache dalam memori. Pemuatan yang terasa lebih lambat cenderung membuat pelanggan beralih ke bisnis lain.

Contoh berikut menggambarkan beberapa cara penggunaan ElastiCache dapat meningkatkan kinerja keseluruhan aplikasi Anda.

Topik

Penyimpanan Data Dalam Memori

Tujuan utama dari penyimpanan nilai-kunci dalam memori adalah untuk menyediakan akses ultracepat (latensi submilidetik) dan murah ke salinan data. Kebanyakan penyimpanan data memiliki area data yang sering diakses namun jarang diperbarui. Selain itu, kueri basis data selalu lebih lambat dan menghabiskan lebih banyak daya komputasi daripada menemukan kunci dalam cache pasangan nilai kunci. Beberapa kueri basis data menghabiskan sangat banyak daya komputasi untuk dijalankan. Contohnya adalah kueri yang memerlukan operasi join terhadap beberapa tabel atau kueri dengan penghitungan intensif. Dengan meng-cache hasil kueri tersebut, Anda membayar harga kueri hanya sekali. Kemudian, Anda dapat dengan cepat mengambil data beberapa kali tanpa harus menjalankan kembali kueri tersebut.

Apa yang Harus Saya Simpan dalam Cache?

Saat menentukan data apa yang harus di-cache, pertimbangkan beberapa faktor ini:

Kecepatan dan biaya – Selalu akan lebih lambat dan lebih mahal untuk mendapatkan data dari basis data dibandingkan dari cache. Beberapa kueri basis data secara inheren lebih lambat dan lebih mahal daripada yang lain. Misalnya, kueri yang melakukan operasi join terhadap beberapa tabel akan jauh lebih lambat dan lebih mahal daripada kueri tabel tunggal sederhana. Jika data yang diperlukan hanya dapat diperoleh menggunakan kueri yang lambat dan mahal, berarti data ini kemungkinan cocok untuk caching. Jika data dapat diambil dengan kueri yang relatif cepat dan sederhana, data tersebut mungkin masih dapat menjadi kandidat untuk caching, tergantung pada beberapa faktor lainnya.

Data dan pola akses – Untuk menentukan apa yang perlu di-cache, diperlukan juga pemahaman terhadap data itu sendiri dan pola aksesnya. Misalnya, tidak masuk akal untuk meng-cache data yang berubah dengan cepat atau jarang diakses. Agar proses cache menyediakan manfaat nyata, data harus relatif statis dan sering diakses. Contohnya adalah profil pribadi di situs media sosial. Di sisi lain, Anda tidak ingin meng-cache data jika caching tidak menyediakan kecepatan atau keuntungan biaya. Misalnya, tidak masuk akal untuk meng-cache halaman web yang menampilkan hasil pencarian karena kueri dan hasilnya biasanya unik.

Staleness – Menurut definisi, data yang di-cache adalah data usang (stale). Meskipun dalam keadaan tertentu data tidak usang, namun data ini harus selalu dianggap dan diperlakukan sebagai data usang. Untuk mengetahui apakah data Anda adalah kandidat untuk caching, tentukan toleransi aplikasi Anda untuk data usang.

Aplikasi Anda mungkin dapat menoleransi data usang dalam satu konteks, tetapi tidak untuk yang lain. Misalnya, anggaplah situs Anda menyajikan harga saham yang diperdagangkan secara publik.

Pelanggan Anda mungkin bisa menerima beberapa data usang jika diberi klarifikasi bahwa data harga mungkin tertunda n menit. Namun, jika Anda menyajikan harga saham tersebut kepada pialang saham yang melakukan penjualan atau pembelian, maka Anda menginginkan data waktu nyata.

Pertimbangkan untuk meng-cache data Anda jika hal berikut ini berlaku:

- Data Anda lambat atau mahal untuk diperoleh jika dibandingkan dengan pengambilan dari cache.
- Pengguna sering mengakses data Anda.
- Data Anda tetap relatif sama, atau jika data berubah dengan cepat, "staleness" bukanlah masalah besar.

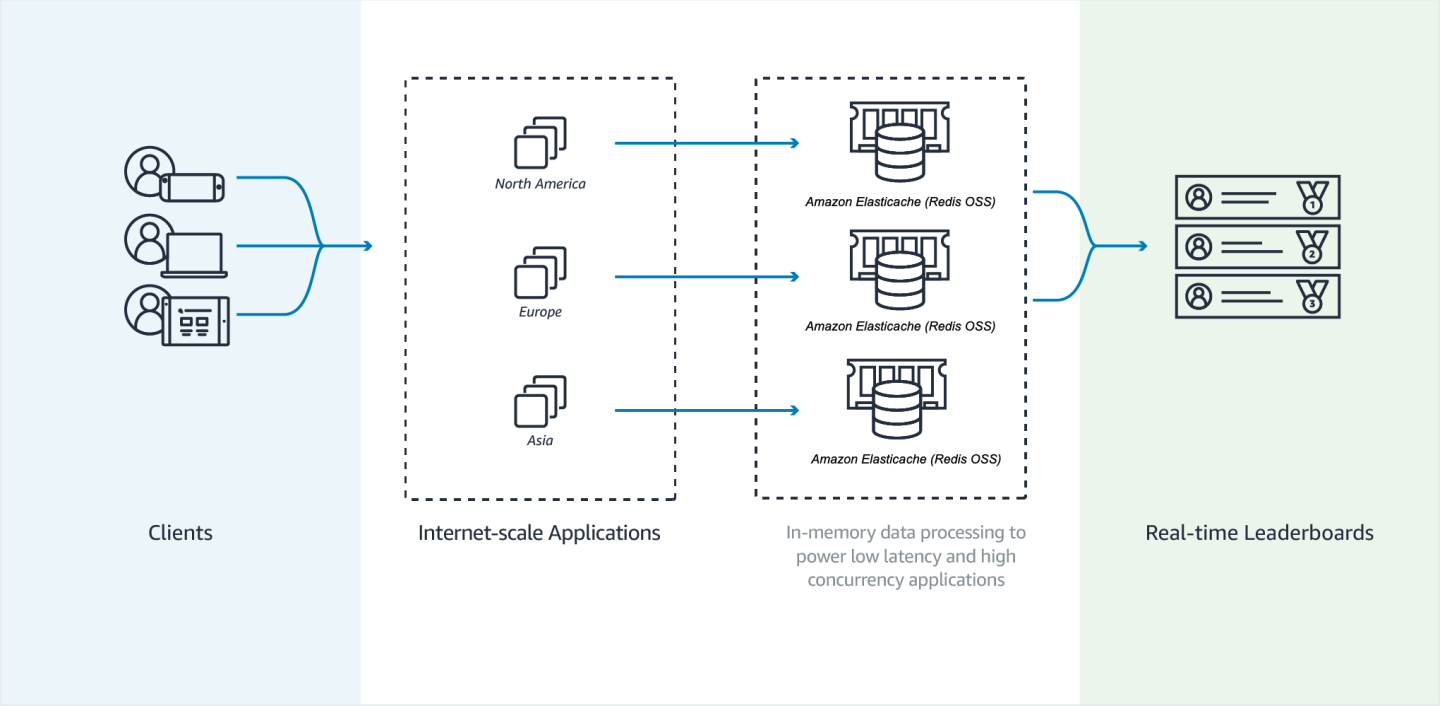
Untuk informasi selengkapnya, lihat Strategi cache untuk Memcached

Papan Peringkat Permainan

Dengan set Valkey atau Redis OSS yang diurutkan, Anda dapat memindahkan kompleksitas komputasi papan peringkat dari aplikasi Anda ke cluster Anda.

Papan peringkat, seperti 10 skor teratas untuk sebuah game memiliki komputasi yang rumit. Hal ini terutama terjadi jika ada banyak pemain sekaligus dan skor terus berubah. Set yang diurutkan Valkey dan Redis OSS menjamin keunikan dan urutan elemen. Dengan set yang diurutkan, setiap kali elemen baru ditambahkan ke set yang diurutkan, itu di-reranked secara real time. Elemen tersebut kemudian ditambahkan ke set dalam urutan numerik yang benar.

Dalam diagram berikut, Anda dapat melihat cara kerja papan peringkat ElastiCache game.



Example Papan Peringkat Valkey atau Redis OSS

Dalam contoh ini, empat pemain game dan skor mereka dimasukkan ke dalam daftar berurut menggunakan ZADD. Perintah ZREVRANGEBYSCORE mengurutkan pemain sesuai skor mereka, dari tinggi ke rendah. Selanjutnya, ZADD digunakan untuk memperbarui skor milik June dengan menimpa entri yang sudah ada. Terakhir, ZREVRANGEBYSCORE mengurutkan pemain sesuai skor mereka, dari tinggi ke rendah. Daftar ini menunjukkan bahwa June telah naik dalam peringkat.

```
ZADD leaderboard 132 Robert
ZADD leaderboard 231 Sandra
ZADD leaderboard 32 June
ZADD leaderboard 381 Adam

ZREVRANGEBYSCORE leaderboard +inf -inf
1) Adam
2) Sandra
3) Robert
4) June

ZADD leaderboard 232 June

ZREVRANGEBYSCORE leaderboard +inf -inf
1) Adam
2) June
```

```
3) Sandra
4) Robert
```

Perintah berikut menunjukkan peringkat June di antara semua pemain. Karena peringkat adalah berbasis nol, ZREVRANK menampilkan nilai 1 untuk June, yang berada di posisi kedua.

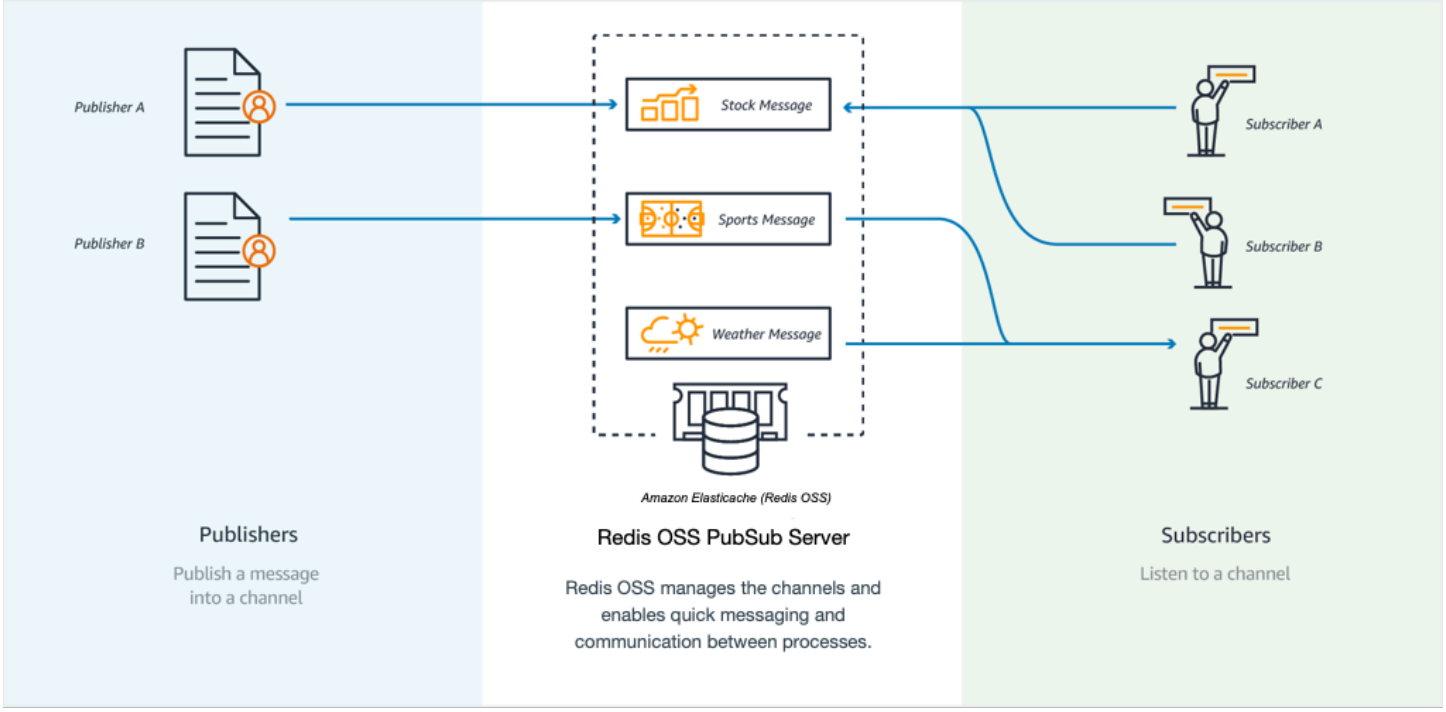
```
ZREVRANK leaderboard June
1
```

Untuk informasi selengkapnya, lihat dokumentasi Valkey tentang set yang diurutkan.

Pesan (Pub/Sub)

Ketika Anda mengirim pesan email, Anda mengirimkannya ke satu atau beberapa penerima tertentu. Dalam paradigma pub/sub Valkey dan Redis OSS, Anda mengirim pesan ke saluran tertentu yang tidak tahu siapa, jika ada, yang menerimanya. Orang-orang yang mendapatkan pesan adalah mereka yang berlangganan saluran tersebut. Misalnya, anggaplah Anda berlangganan saluran news.sports.golf. Anda dan semua orang yang berlangganan saluran news.sports.golf mendapatkan semua pesan yang dipublikasikan ke news.sports.golf.

Fungsionalitas PUB/sub tidak ada hubungannya dengan ruang kunci apa pun. Oleh karena itu, fungsi ini sama sekali tidak memberikan pengaruh. Dalam diagram berikut, Anda dapat menemukan ilustrasi ElastiCache pesan dengan Valkey dan Redis OSS.



Berlangganan

Untuk menerima pesan pada saluran, Anda berlangganan saluran tersebut. Anda dapat berlangganan satu saluran, beberapa saluran tertentu, atau semua saluran yang sesuai dengan pola. Untuk membatalkan langganan, Anda berhenti berlangganan dari saluran yang ditentukan saat Anda berlangganan. Atau, jika berlangganan menggunakan pencocokan pola, Anda berhenti berlangganan menggunakan pola yang sama seperti yang Anda gunakan sebelumnya.

Example - Langganan ke Satu Saluran

Untuk berlangganan satu saluran, gunakan perintah SUBSCRIBE yang menentukan saluran yang diinginkan. Pada contoh berikut, klien berlangganan saluran news.sports.golf.

```
SUBSCRIBE news.sports.golf
```

Setelah beberapa saat, klien ini membatalkan langganannya ke saluran tersebut menggunakan perintah UNSUBSCRIBE yang menentukan saluran yang akan dihentikan langganannya.

```
UNSUBSCRIBE news.sports.golf
```

Example - Berlangganan ke Beberapa Saluran Tertentu

Untuk berlangganan beberapa saluran tertentu, tampilkan daftar saluran dengan perintah SUBSCRIBE. Pada contoh berikut, klien berlangganan saluran news.sports.golf, news.sports.soccer, dan news.sports.skiing.

```
SUBSCRIBE news.sports.golf news.sports.soccer news.sports.skiing
```

Untuk membatalkan langganan ke saluran tertentu, gunakan perintah UNSUBSCRIBE dan tentukan saluran yang akan dihentikan langganannya.

```
UNSUBSCRIBE news.sports.golf
```

Untuk membatalkan langganan ke beberapa saluran, gunakan perintah UNSUBSCRIBE dan tentukan saluran-saluran yang akan dihentikan langganannya.

```
UNSUBSCRIBE news.sports.golf news.sports.soccer
```

Untuk membatalkan semua langganan, gunakan UNSUBSCRIBE dan tentukan setiap saluran. Atau gunakan UNSUBSCRIBE dan jangan menentukan saluran.

```
UNSUBSCRIBE news.sports.golf news.sports.soccer news.sports.skiing
```

atau

```
UNSUBSCRIBE
```

Example - Langganan Menggunakan Pencocokan Pola

Klien dapat berlangganan semua saluran yang cocok dengan sebuah pola menggunakan perintah PSUBSCRIBE.

Pada contoh berikut, klien berlangganan semua saluran olahraga. Anda tidak menampilkan daftar semua saluran olahraga satu per satu, seperti yang Anda gunakan dengan SUBSCRIBE. Namun, dengan perintah PSUBSCRIBE Anda menggunakan pencocokan pola.

```
PSUBSCRIBE news.sports.*
```

Example Membatalkan Langganan

Untuk membatalkan langganan ke saluran tersebut, gunakan perintah PUNSUBSCRIBE.

```
PUNSUBSCRIBE news.sports.*
```

Important

- String saluran yang dikirim ke perintah [P]SUBSCRIBE dan ke perintah [P]UNSUBSCRIBE harus cocok. Anda tidak dapat menjalankan PSUBSCRIBE ke news.* dan PUNSUBSCRIBE dari news.sports.* atau UNSUBSCRIBE dari news.sports.golf.
- PSUBSCRIBEdan tidak PUNSUBSCRIBE tersedia untuk Tanpa ElastiCache Server.

Publikasi

Untuk mengirim pesan ke semua pelanggan saluran, gunakan perintah PUBLISH, dengan menentukan saluran dan pesan. Contoh berikut memublikasikan pesan, "It’s Saturday and sunny. I’m headed to the links." ke saluran news.sports.golf.

```
PUBLISH news.sports.golf "It's Saturday and sunny. I'm headed to the links."
```

Klien tidak dapat mempublikasikan ke saluran yang berlangganan.

Untuk informasi selengkapnya, lihat Pub/Sub di dokumentasi Valkey.

Data Rekomendasi (Hash)

Menggunakan INCR atau DECR di Valkey atau Redis OSS membuat rekomendasi kompilasi menjadi sederhana. Setiap kali pengguna "menyukai" produk, Anda menaikkan nilai penghitung item:productID:like. Setiap kali pengguna "tidak menyukai" produk, Anda menaikkan nilai penghitung item:productID:dislike. Menggunakan hash, Anda juga dapat mempertahankan daftar semua orang yang menyukai atau tidak menyukai suatu produk.

Example - Suka dan Tidak Suka

```
INCR item:38923:likes
HSET item:38923:ratings Susan 1
INCR item:38923:dislikes
HSET item:38923:ratings Tommy -1
```

ElastiCache Testimonial Pelanggan

Untuk mempelajari cara bisnis seperti Airbnb, PBS, Esri, dan lainnya menggunakan Amazon ElastiCache untuk mengembangkan bisnis mereka dengan pengalaman pelanggan yang lebih baik, lihat Cara Orang Lain Menggunakan Amazon. ElastiCache

Anda juga dapat menonton video Tutorial untuk kasus penggunaan ElastiCache pelanggan tambahan.

Memulai dengan Amazon ElastiCache

Gunakan tutorial langsung di bagian ini untuk membantu Anda memulai dan mempelajari lebih lanjut tentang menggunakan. ElastiCache

Topik

Menyiapkan ElastiCache

Untuk menggunakan layanan ElastiCache web, ikuti langkah-langkah ini.

Topik

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.

2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimi Anda email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .
2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

  Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuknya, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.
2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Memberikan akses programatis

Pengguna membutuhkan akses terprogram jika mereka ingin berinteraksi dengan AWS luar. AWS Management Console Cara untuk memberikan akses terprogram tergantung pada jenis pengguna yang mengakses AWS.

Untuk memberi pengguna akses programatis, pilih salah satu opsi berikut.

Pengguna mana yang membutuhkan akses programatis?	Untuk	Oleh
Identitas tenaga kerja	Gunakan kredensyal sementara untuk menandata ngani permintaan terprogra	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan.

Pengguna mana yang membutuhkan akses programatis?	Untuk	Oleh
(Pengguna yang dikelola di Pusat Identitas IAM)	m ke AWS CLI,, AWS SDKs atau. AWS APIs	• Untuk AWS CLI, lihat Mengkonfigurasi yang akan AWS CLI digunakan AWS IAM Identity Center dalam Panduan AWS Command Line Interface Pengguna. • Untuk AWS SDKs, alat, dan AWS APIs, lihat Autentikasi Pusat Identitas IAM di Panduan Referensi Alat AWS SDKs dan Alat.
IAM	Gunakan kredensyal sementara untuk menandatangani permintaan terprogram ke AWS CLI,, AWS SDKs atau. AWS APIs	Mengikuti petunjuk dalam Menggunakan kredensyal sementara dengan AWS sumber daya di Panduan Pengguna IAM.

Pengguna mana yang membutuhkan akses programatis?	Untuk	Oleh
IAM	(Tidak direkomendasikan) Gunakan kredensyal jangka panjang untuk menandatangani permintaan terprogram ke AWS CLI,, AWS SDKs atau. AWS APIs	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan. • Untuk mengetahui AWS CLI, lihat Mengautentikasi menggunakan kredensyal pengguna IAM di Panduan Pengguna.AWS Command Line Interface • Untuk AWS SDKs dan alat, lihat Mengautentikasi menggunakan kredensi jangka panjang di Panduan Referensi Alat AWS SDKs dan Alat. • Untuk AWS APIs, lihat Mengelola kunci akses untuk pengguna IAM di Panduan Pengguna IAM.

Topik terkait:

- Apa itu IAM? dalam Panduan Pengguna IAM.
- AWS Kredensyal Keamanan dalam Referensi AWS Umum.

Siapkan izin Anda (hanya ElastiCache pengguna baru)

Untuk memberikan akses dan menambahkan izin bagi pengguna, grup, atau peran Anda:

- Pengguna dan grup di AWS IAM Identity Center:

Buat rangkaian izin. Ikuti instruksi di Buat rangkaian izin di Panduan Pengguna AWS IAM Identity Center .

- Pengguna yang dikelola di IAM melalui penyedia identitas:

  Buat peran untuk federasi identitas. Ikuti instruksi dalam Buat peran untuk penyedia identitas pihak ketiga (federasi) dalam Panduan Pengguna IAM.
- Pengguna IAM:
  - Buat peran yang dapat diambil pengguna Anda. Ikuti instruksi dalam Buat peran untuk pengguna IAM dalam Panduan Pengguna IAM.
  - (Tidak disarankan) Lampirkan kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti petunjuk dalam Menambahkan izin ke pengguna (konsol) dalam Panduan Pengguna IAM.

Amazon ElastiCache membuat dan menggunakan peran terkait layanan untuk menyediakan sumber daya dan mengakses sumber AWS daya dan layanan lain atas nama Anda. ElastiCache Untuk membuat peran terkait layanan untuk Anda, gunakan kebijakan AWS-managed bernama. `AmazonElastiCacheFullAccess` Peran ini dilengkapi sebelumnya dengan izin yang diperlukan layanan untuk membuat peran terkait layanan untuk Anda.

Anda mungkin memutuskan untuk tidak menggunakan kebijakan default dan sebagai gantinya menggunakan kebijakan terkelola khusus. Dalam hal ini, pastikan bahwa Anda memiliki izin untuk memanggil `iam:createServiceLinkedRole` atau Anda telah membuat peran tertaut layanan ElastiCache.

Untuk informasi lain, lihat yang berikut ini:

- Membuat Kebijakan Baru (IAM)
- AWS kebijakan terkelola untuk Amazon ElastiCache
- Menggunakan Peran Tertaut Layanan untuk Amazon ElastiCache

Mengatur EC2

Anda perlu mengatur EC2 instance dari mana Anda akan terhubung ke cache Anda.

- Jika Anda belum memiliki EC2 instance, pelajari cara menyiapkan EC2 instance di sini: Panduan EC2 Memulai Amazon.

- EC2 Instans Anda harus dalam VPC yang sama dan memiliki pengaturan grup keamanan yang sama dengan cache Anda. Secara default, Amazon ElastiCache membuat cache di VPC default Anda dan menggunakan grup keamanan default. Untuk mengikuti tutorial ini, pastikan EC2 instans Anda ada di VPC default dan memiliki grup keamanan default.

Berikan akses jaringan dari grup keamanan Amazon VPC ke cache Anda

ElastiCache Cluster yang dirancang sendiri menggunakan port 6379 untuk perintah Valkey dan Redis OSS, dan tanpa ElastiCache server menggunakan port 6379 dan port 6380. Agar berhasil menghubungkan dan menjalankan perintah Valkey atau Redis OSS dari EC2 instans Anda, grup keamanan Anda harus mengizinkan akses ke port ini sesuai kebutuhan.

ElastiCache untuk Memcached menggunakan port 11211 dan 11212 untuk menerima perintah Memcached. Agar berhasil menghubungkan dan menjalankan perintah Memcached dari EC2 instans Anda, grup keamanan Anda harus mengizinkan akses ke port ini.

1. Masuk ke AWS Command Line Interface dan buka EC2 konsol Amazon.
2. Pada panel navigasi, di bagian Jaringan & Keamanan, pilih Grup Keamanan.
3. Dari daftar grup keamanan, pilih grup keamanan untuk Amazon VPC Anda. Kecuali Anda membuat grup keamanan untuk ElastiCache digunakan, grup keamanan ini akan diberi nama default.
4. Pilih tab Inbound, lalu:

    a. Pilih Edit.

    b. Pilih Tambahkan aturan.

    c. Pada kolom Jenis, pilih Aturan TCP kustom.

    d. Jika menggunakan Valkey atau Redis OSS, maka di kotak rentang Port, ketik. `6379`

    Jika menggunakan Memcached, maka di kotak rentang Port, ketik. `11211`

    e. Di kotak Sumber, pilih Di mana saja yang memiliki rentang port (0.0.0.0/0) sehingga EC2 instans Amazon apa pun yang Anda luncurkan dalam VPC Amazon Anda dapat terhubung ke cache Anda.

    f. Jika Anda menggunakan ElastiCache tanpa server, tambahkan aturan lain dengan memilih Tambahkan aturan.

    g. Di kolom Jenis, pilih Aturan TCP kustom.

    h. Jika menggunakan ElastiCache untuk Redis OSS, maka di kotak rentang Port, ketik. `6380`

Jika menggunakan ElastiCache untuk Memcached, maka di kotak rentang Port, ketik. `11212`

i. Di kotak Sumber, pilih Di mana saja yang memiliki rentang port (0.0.0.0/0) sehingga EC2 instans Amazon apa pun yang Anda luncurkan dalam VPC Amazon Anda dapat terhubung ke cache Anda.

j. Pilih Simpan.

Unduh dan atur akses baris perintah

Unduh dan instal utilitas valkey-cli.

Jika Anda menggunakan ElastiCache untuk Valkey, maka Anda mungkin menemukan utilitas valkey-cli berguna. Jika Anda menggunakan Redis OSS dengan redis-cli, pertimbangkan ElastiCache untuk beralih ke valkey-cli karena berfungsi untuk Redis OSS juga.

1. Connect ke EC2 instans Amazon Anda menggunakan utilitas koneksi pilihan Anda. Untuk petunjuk tentang cara menyambung ke EC2 instans Amazon, lihat Panduan EC2 Memulai Amazon.
2. Unduh dan instal utilitas valkey-cli dengan menjalankan perintah yang sesuai untuk pengaturan Anda.

   Amazon Linux 2

   ```
   sudo amazon-linux-extras install epel -y
   sudo yum install gcc jemalloc-devel openssl-devel tcl tcl-devel -y
   wget https://github.com/valkey-io/valkey/archive/refs/tags/8.0.0.tar.gz
   tar xvzf valkey-8.0.0.tar.gz
   cd valkey-8.0.0
   make BUILD_TLS=yes
   ```

> **Note**
>
> - Ketika Anda menginstal paket redis6, paket ini akan menginstal redis6-cli dengan dukungan enkripsi default.
> - Penting untuk memiliki dukungan build untuk TLS saat menginstal valkey-cli atau redis-cli. ElastiCache Tanpa server hanya dapat diakses saat TLS diaktifkan.

- Jika Anda terhubung ke klaster yang tidak terenkripsi, Anda tidak memerlukan opsi `Build_TLS=yes`.

Buat cache tanpa server Valkey

Pada langkah ini, Anda membuat cache baru di Amazon ElastiCache.

AWS Management Console

Untuk membuat cache baru menggunakan ElastiCache konsol:

1. Masuk ke AWS Management Console dan buka https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi di sisi kiri konsol, pilih cache Valkey.
3. Di sisi kanan konsol, pilih Buat cache Valkey
4. Pada Pengaturan cache, masukkan Nama. Anda juga dapat memasukkan deskripsi untuk cache.
5. Biarkan pengaturan default dipilih.
6. Klik Buat untuk membuat cache.
7. Setelah cache berada dalam status "AKTIF", Anda dapat mulai melakukan operasi baca dan tulis ke cache.

AWS CLI

AWS CLI Contoh berikut membuat cache baru menggunakan create-serverless-cache.

Linux

```
aws elasticache create-serverless-cache \
    --serverless-cache-name CacheName \
    --engine valkey
```

Windows

```
aws elasticache create-serverless-cache ^
    --serverless-cache-name CacheName ^
```

```
    --engine valkey
```

Perhatikan bahwa nilai bidang Status diatur ke `CREATING`.

Untuk memverifikasi bahwa ElastiCache telah selesai membuat cache, gunakan `describe-serverless-caches` perintah.

Linux

```
aws elasticache describe-serverless-caches --serverless-cache-name CacheName
```

Windows

```
aws elasticache describe-serverless-caches --serverless-cache-name CacheName
```

Setelah membuat cache baru, lanjutkan ke Membaca dan menulis data ke cache.

Membaca dan menulis data ke cache

Bagian ini mengasumsikan bahwa Anda telah membuat EC2 instance Amazon dan dapat terhubung dengannya. Untuk petunjuk tentang cara melakukannya, lihat Panduan EC2 Memulai Amazon.

Bagian ini juga mengasumsikan bahwa Anda memiliki pengaturan akses VPC dan pengaturan grup keamanan untuk EC2 instance dari mana Anda terhubung ke cache Anda, dan mengatur valkey-cli pada instance Anda. EC2 Untuk informasi selengkapnya tentang langkah tersebut lihat Menyiapkan ElastiCache.

Selain langkah-langkah di bawah ini, jika Anda memiliki aplikasi besar atau global, Anda dapat sangat meningkatkan kinerja baca dengan membuat dan membaca dari replika. Untuk informasi lebih lanjut tentang langkah yang lebih maju ini lihatPraktik Terbaik untuk menggunakan Read Replicas.

Menemukan titik akhir cache Anda

AWS Management Console

Untuk menemukan titik akhir cache Anda menggunakan ElastiCache konsol:

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi di sisi kiri konsol, pilih cache Valkey.

3. Di sisi kanan konsol, klik nama cache yang baru saja Anda buat.
4. Dalam Detail cache, cari dan salin titik akhir cache.

AWS CLI

AWS CLI Contoh berikut menunjukkan untuk menemukan titik akhir untuk cache baru Anda menggunakan describe-serverless-caches perintah. Setelah Anda menjalankan perintah, cari bidang "Titik Akhir".

Linux

```
aws elasticache describe-serverless-caches \
  --serverless-cache-name CacheName
```

Windows

```
aws elasticache describe-serverless-caches ^
  --serverless-cache-name CacheName
```

Connect ke Valkey Cache (Linux)

Sekarang setelah Anda memiliki titik akhir yang Anda butuhkan, Anda dapat masuk ke EC2 instance Anda dan terhubung ke cache. Dalam contoh berikut, Anda menggunakan utilitas valkey-cli untuk terhubung ke cluster. Perintah berikut menghubungkan ke cache (catatan: ganti cache-endpoint dengan titik akhir yang Anda ambil pada langkah sebelumnya).

```
src/valkey-cli -h cache-endpoint --tls -p 6379
set a "hello"          // Set key "a" with a string value and no expiration
OK
get a                  // Get value for key "a"
"hello"
```

Connect ke Valkey Cache (Windows)

Sekarang setelah Anda memiliki titik akhir yang Anda butuhkan, Anda dapat masuk ke EC2 instance Anda dan terhubung ke cache. Dalam contoh berikut, Anda menggunakan utilitas valkey-cli untuk terhubung ke cluster. Perintah berikut terhubung ke cache. Buka Command Prompt dan ubah ke direktori Valkey atau Redis OSS dan jalankan perintah (catatan: ganti Cache_Endpoint dengan titik akhir yang Anda ambil pada langkah sebelumnya).

```
c:\Valkey>valkey-cli -h Valkey_Cluster_Endpoint --tls -p 6379
set a "hello"          // Set key "a" with a string value and no expiration
OK
get a                  // Get value for key "a"
"hello"
```

Sekarang Anda dapat melanjutkan ke (Opsional) Bersihkan.

(Opsional) Bersihkan

Jika Anda tidak lagi membutuhkan ElastiCache cache Amazon yang Anda buat, Anda dapat menghapusnya. Langkah ini membantu memastikan bahwa Anda tidak akan dikenai biaya untuk sumber daya yang tidak Anda gunakan. Anda dapat menggunakan ElastiCache konsol, file AWS CLI, atau ElastiCache API untuk menghapus cache Anda.

AWS Management Console

Untuk menghapus cache Anda menggunakan konsol:

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi di sisi kiri konsol, pilih Valkey Caches.
3. Pilih tombol radio di samping cache yang ingin Anda hapus.
4. Pilih Tindakan di kanan atas lalu pilih Hapus.
5. Anda dapat memilih untuk mengambil snapshot terakhir sebelum menghapus cache Anda.
6. Pada layar konfirmasi Hapus, masukkan kembali nama cache dan pilih Hapus untuk menghapus klaster, atau pilih Batal untuk mempertahankan klaster.

Setelah cache Anda masuk ke status DELETING, Anda tidak lagi dikenai biaya untuk itu.

AWS CLI

AWS CLI Contoh berikut menghapus cache menggunakan delete-serverless-cache perintah.

Linux

```
aws elasticache delete-serverless-cache \
  --serverless-cache-name CacheName
```

Windows

```
aws elasticache delete-serverless-cache ^
  --serverless-cache-name CacheName
```

Perhatikan bahwa nilai bidang Status diatur ke DELETING.

Sekarang Anda dapat melanjutkan ke Langkah Berikutnya.

Langkah Berikutnya

Untuk informasi lebih lanjut tentang ElastiCache lihat halaman berikut:

- Bekerja dengan ElastiCache
- Penskalaan ElastiCache
- Pencatatan dan pemantauan di Amazon ElastiCache
- ElastiCache praktik terbaik dan strategi caching
- Melakukan snapshot dan pemulihan
- Pemantauan acara Amazon SNS ElastiCache

Buat cache tanpa server Redis OSS

Pada langkah ini, Anda membuat cache baru di Amazon ElastiCache.

AWS Management Console

Untuk membuat cache baru menggunakan ElastiCache konsol:

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi di sisi kiri konsol, pilih cache Redis OSS.
3. Di sisi kanan konsol, pilih Buat cache Redis OSS
4. Pada Pengaturan cache, masukkan Nama. Anda juga dapat memasukkan deskripsi untuk cache.
5. Biarkan pengaturan default dipilih.
6. Klik Buat untuk membuat cache.

7. Setelah cache berada dalam status "AKTIF", Anda dapat mulai melakukan operasi baca dan tulis ke cache.

AWS CLI

AWS CLI Contoh berikut membuat cache baru menggunakan create-serverless-cache.

Linux

```
aws elasticache create-serverless-cache \
    --serverless-cache-name CacheName \
    --engine redis
```

Windows

```
aws elasticache create-serverless-cache ^
    --serverless-cache-name CacheName ^
    --engine redis
```

Perhatikan bahwa nilai bidang Status diatur ke `CREATING`.

Untuk memverifikasi bahwa ElastiCache telah selesai membuat cache, gunakan `describe-serverless-caches` perintah.

Linux

```
aws elasticache describe-serverless-caches --serverless-cache-name CacheName
```

Windows

```
aws elasticache describe-serverless-caches --serverless-cache-name CacheName
```

Setelah membuat cache baru, lanjutkan ke Membaca dan menulis data ke cache.

Membaca dan menulis data ke cache

Bagian ini mengasumsikan bahwa Anda telah membuat EC2 instance Amazon dan dapat terhubung dengannya. Untuk petunjuk tentang cara melakukannya, lihat Panduan EC2 Memulai Amazon.

Bagian ini juga mengasumsikan bahwa Anda memiliki pengaturan akses VPC dan pengaturan grup keamanan untuk EC2 instance dari mana Anda terhubung ke cache Anda, dan mengatur valkey-cli pada instance Anda. EC2 Untuk informasi selengkapnya tentang langkah tersebut lihat Menyiapkan ElastiCache.

Menemukan titik akhir cache Anda

AWS Management Console

Untuk menemukan titik akhir cache Anda menggunakan ElastiCache konsol:

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi di sisi kiri konsol, pilih Valkey cache cache Redis OSS.
3. Di sisi kanan konsol, klik nama cache yang baru saja Anda buat.
4. Dalam Detail cache, cari dan salin titik akhir cache.

AWS CLI

AWS CLI Contoh berikut menunjukkan untuk menemukan titik akhir untuk cache baru Anda menggunakan describe-serverless-caches perintah. Setelah Anda menjalankan perintah, cari bidang "Titik Akhir".

Linux

```
aws elasticache describe-serverless-caches \
  --serverless-cache-name CacheName
```

Windows

```
aws elasticache describe-serverless-caches ^
  --serverless-cache-name CacheName
```

Connect ke Valkey atau Redis OSS Cache (Linux)

Sekarang setelah Anda memiliki titik akhir yang Anda butuhkan, Anda dapat masuk ke EC2 instance Anda dan terhubung ke cache. Dalam contoh berikut, Anda menggunakan utilitas valkey-cli untuk terhubung ke cluster. Perintah berikut menghubungkan ke cache (catatan: ganti cache-endpoint dengan titik akhir yang Anda ambil pada langkah sebelumnya).

```
src/valkey-cli -h cache-endpoint --tls -p 6379
set a "hello"          // Set key "a" with a string value and no expiration
OK
get a                  // Get value for key "a"
"hello"
```

Connect ke Valkey atau Redis OSS Cache (Windows)

Sekarang setelah Anda memiliki titik akhir yang Anda butuhkan, Anda dapat masuk ke EC2 instance Anda dan terhubung ke cache. Dalam contoh berikut, Anda menggunakan utilitas valkey-cli untuk terhubung ke cluster. Perintah berikut terhubung ke cache. Buka Command Prompt dan ubah ke direktori Valkey dan jalankan perintah (catatan: ganti Cache_Endpoint dengan titik akhir yang Anda ambil pada langkah sebelumnya).

```
c:\Redis>valkey-cli -h Redis_Cluster_Endpoint --tls -p 6379
set a "hello"          // Set key "a" with a string value and no expiration
OK
get a                  // Get value for key "a"
"hello"
```

Sekarang Anda dapat melanjutkan ke (Opsional) Bersihkan.

(Opsional) Bersihkan

Jika Anda tidak lagi membutuhkan ElastiCache cache Amazon yang Anda buat, Anda dapat menghapusnya. Langkah ini membantu memastikan bahwa Anda tidak akan dikenai biaya untuk sumber daya yang tidak Anda gunakan. Anda dapat menggunakan ElastiCache konsol, file AWS CLI, atau ElastiCache API untuk menghapus cache Anda.

AWS Management Console

Untuk menghapus cache Anda menggunakan konsol:

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi di sisi kiri konsol, pilih Valkey atau Redis OSS Cache.
3. Pilih tombol radio di samping cache yang ingin Anda hapus.
4. Pilih Tindakan di kanan atas lalu pilih Hapus.
5. Anda dapat memilih untuk mengambil snapshot terakhir sebelum menghapus cache Anda.

6. Pada layar konfirmasi Hapus, masukkan kembali nama cache dan pilih Hapus untuk menghapus klaster, atau pilih Batal untuk mempertahankan klaster.

Setelah cache Anda masuk ke status DELETING, Anda tidak lagi dikenai biaya untuk itu.

AWS CLI

AWS CLI Contoh berikut menghapus cache menggunakan delete-serverless-cache perintah.

Linux

```
aws elasticache delete-serverless-cache \
  --serverless-cache-name CacheName
```

Windows

```
aws elasticache delete-serverless-cache ^
  --serverless-cache-name CacheName
```

Perhatikan bahwa nilai bidang Status diatur ke DELETING.

Sekarang Anda dapat melanjutkan ke Langkah Berikutnya.

Langkah Berikutnya

Untuk informasi lebih lanjut tentang ElastiCache lihat halaman berikut:

- Bekerja dengan ElastiCache
- Penskalaan ElastiCache
- Pencatatan dan pemantauan di Amazon ElastiCache
- ElastiCache praktik terbaik dan strategi caching
- Melakukan snapshot dan pemulihan
- Pemantauan acara Amazon SNS ElastiCache

Buat cache tanpa server Memcached

AWS Management Console

Untuk membuat cache tanpa server Memcached baru menggunakan konsol: ElastiCache

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi di sisi kiri konsol, pilih Cache Memcached.
3. Di bagian kanan konsol, pilih Buat cache Memcached.
4. Pada Pengaturan cache, masukkan Nama. Anda juga dapat memasukkan deskripsi untuk cache.
5. Biarkan pengaturan default dipilih.
6. Klik Buat untuk membuat cache.
7. Setelah cache berada dalam status "AKTIF", Anda dapat mulai melakukan operasi pembacaan dan penulisan ke cache.

Untuk membuat cache baru menggunakan AWS CLI

AWS CLI Contoh berikut membuat cache baru menggunakan create-serverless-cache.

Linux

```
aws elasticache create-serverless-cache \
  --serverless-cache-name CacheName \
  --engine memcached
```

Windows

```
aws elasticache create-serverless-cache ^
  --serverless-cache-name CacheName ^
  --engine memcached
```

Perhatikan bahwa nilai bidang Status diatur ke `CREATING`.

Untuk memverifikasi bahwa ElastiCache telah selesai membuat cache, gunakan `describe-serverless-caches` perintah.

Linux

```
aws elasticache describe-serverless-caches --serverless-cache-name CacheName
```

Windows

```
aws elasticache describe-serverless-caches --serverless-cache-name CacheName
```

Setelah membuat cache baru, lanjutkan ke Membaca dan menulis data ke cache.

Membaca dan menulis data ke cache

Bagian ini mengasumsikan bahwa Anda telah membuat EC2 instance Amazon dan dapat terhubung dengannya. Untuk petunjuk tentang cara melakukannya, lihat Panduan EC2 Memulai Amazon.

Secara default, ElastiCache buat cache di VPC default Anda. Pastikan EC2 instance Anda juga dibuat di VPC default, sehingga dapat terhubung ke cache.

Menemukan titik akhir cache Anda

AWS Management Console

Untuk menemukan titik akhir cache Anda menggunakan ElastiCache konsol:

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi di sisi kiri konsol, pilih Cache Memcached.
3. Di sisi kanan konsol, klik nama cache yang baru saja Anda buat.
4. Dalam Detail cache, cari dan salin titik akhir cache.

AWS CLI

AWS CLI Contoh berikut menunjukkan untuk menemukan titik akhir untuk cache baru Anda menggunakan describe-serverless-caches perintah. Setelah Anda menjalankan perintah, cari bidang "Titik Akhir".

Linux

```
aws elasticache describe-serverless-caches \
  --serverless-cache-name CacheName
```

Windows

```
aws elasticache describe-serverless-caches ^
  --serverless-cache-name CacheName
```

Menghubungkan menggunakan OpenSSL

Untuk informasi tentang cara menghubungkan menggunakan OpenSSL, lihat ElastiCache enkripsi dalam transit (TLS)

Menghubungkan menggunakan klien Java Memcached

```
import java.security.KeyStore;
import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManagerFactory;
import net.spy.memcached.AddrUtil;
import net.spy.memcached.ConnectionFactoryBuilder;
import net.spy.memcached.FailureMode;
import net.spy.memcached.MemcachedClient;

public class TLSDemo {
    public static void main(String[] args) throws Exception {
        ConnectionFactoryBuilder connectionFactoryBuilder = new
 ConnectionFactoryBuilder();
        // Build SSLContext
        TrustManagerFactory tmf =
 TrustManagerFactory.getInstance(TrustManagerFactory.getDefaultAlgorithm());
        tmf.init((KeyStore) null);
        SSLContext sslContext = SSLContext.getInstance("TLS");
        sslContext.init(null, tmf.getTrustManagers(), null);
        // Create the client in TLS mode
        connectionFactoryBuilder.setSSLContext(sslContext);
        // Set Failure Mode to Retry
        connectionFactoryBuilder.setFailureMode(FailureMode.Retry);
        MemcachedClient client = new MemcachedClient(connectionFactoryBuilder.build(),
 AddrUtil.getAddresses("mycluster-fnjyzo.serverless.use1.cache.amazonaws.com:11211"));

        // Store a data item for an hour.
        client.set("theKey", 3600, "This is the data value");
    }
}
```

Menghubungkan menggunakan klien PHP Memcached

```
<?php
$cluster_endpoint = "mycluster.serverless.use1.cache.amazonaws.com";
$server_port = 11211;
```

```
/* Initialize a persistent Memcached client in TLS mode */
$tls_client = new Memcached('persistent-id');
$tls_client->addServer($cluster_endpoint, $server_port);
if(!$tls_client->setOption(Memcached::OPT_USE_TLS, 1)) {
    echo $tls_client->getLastErrorMessage(), "\n";
    exit(1);
}
$tls_config = new MemcachedTLSContextConfig();
$tls_config->hostname = '*.serverless.use1.cache.amazonaws.com';
$tls_config->skip_cert_verify = false;
$tls_config->skip_hostname_verify = false;
$tls_client->createAndSetTLSContext((array)$tls_config);

 /* store the data for 60 seconds in the cluster */
$tls_client->set('key', 'value', 60);
?>
```

Menghubungkan menggunakan klien Python Memcached (Pymemcache)

Lihat https://pymemcache.readthedocs. io/en/latest/getting_started.html

```
import ssl
from pymemcache.client.base import Client

context = ssl.create_default_context()
cluster_endpoint = <To be taken from the AWS CLI / console>
target_port = 11211
memcached_client = Client(("{cluster_endpoint}", target_port), tls_context=context)
memcached_client.set("key", "value", expire=500, noreply=False)
assert self.memcached_client.get("key").decode() == "value"
```

Connect menggunakan NodeJS/TS klien Memcached (Electrode-io memcache)

Lihat https://github.com/electrode-io/memcache dan https://www.npmjs.com/package/ memcache-client

Instal melalui `npm i memcache-client`

Dalam aplikasi, buat klien TLS memcache sebagai berikut:

```
var memcache = require("memcache-client");
const client = new memcache.MemcacheClient({server: "{cluster_endpoint}:11211", tls:
 {}});
```

```
client.set("key", "value");
```

Menghubungkan menggunakan klien Rust Memcached (rust-memcache)

Lihat https://crates. io/crates/memcachedan https://github.com/aisk/rust-memcache.

```
// create connection with to memcached server node:
let client = memcache::connect("memcache+tls://<cluster_endpoint>:11211?
verify_mode=none").unwrap();

// set a string value
client.set("foo", "bar", 0).unwrap();
```

Menghubungkan menggunakan klien Go Memcached (Gomemcache)

Lihat https://github.com/bradfitz/gomemcache

```
c := New(net.JoinHostPort("{cluster_endpoint}", strconv.Itoa(port)))
c.DialContext = func(ctx context.Context, network, addr string) (net.Conn, error) {
var td tls.Dialer
td.Config = &tls.Config{}
return td.DialContext(ctx, network, addr)
}
foo := &Item{Key: "foo", Value: []byte("fooval"), Flags: 123}
err := c.Set(foo)
```

Menghubungkan menggunakan klien Ruby Memcached (Dalli)

Lihat https://github.com/petergoldstein/dalli

```
require 'dalli'
ssl_context = OpenSSL::SSL::SSLContext.new
ssl_context.ssl_version = :SSLv23
ssl_context.verify_hostname = true
ssl_context.verify_mode = OpenSSL::SSL::VERIFY_PEER
client = Dalli::Client.new("<cluster_endpoint>:11211", :ssl_context => ssl_context);
client.get("abc")
```

Connect menggunakan Memcached .NET client () EnyimMemcachedCore

Lihat https://github.com/cnblogs/EnyimMemcachedCore

```
"MemcachedClient": {
"Servers": [
{
"Address": "{cluster_endpoint}",
"Port": 11211
}
],
"UseSslStream":  true
}
```

Sekarang Anda dapat melanjutkan ke (Opsional) Bersihkan.

(Opsional) Bersihkan

Menggunakan AWS Management Console

Prosedur berikut menghapus satu cache dari deployment Anda. Untuk menghapus beberapa cache, ulangi prosedur untuk setiap cache yang ingin dihapus. Anda tidak perlu menunggu satu cache selesai dihapus untuk memulai prosedur penghapusan cache lainnya.

Untuk menghapus cache

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di dasbor ElastiCache konsol, pilih mesin yang berjalan pada cache yang ingin Anda hapus. Daftar semua cache yang menjalankan mesin tersebut akan muncul.
3. Untuk memilih cache yang akan dihapus, pilih nama cache dari daftar cache.

   > **Important**
   >
   > Anda hanya dapat menghapus satu cache pada satu waktu dari ElastiCache konsol. Memilih beberapa cache akan menonaktifkan operasi hapus.

4. Untuk Tindakan, pilih Hapus.
5. Pada layar konfirmasi Hapus Cache, pilih Hapus untuk menghapus cache, atau pilih Batal untuk mempertahankan klaster.
6. Jika Anda memilih Hapus, status cache berubah menjadi menghapus.

Setelah cache Anda masuk ke status DELETING, Anda tidak lagi dikenai biaya untuk itu.

Menggunakan AWS CLI

Kode berikut menghapus cache my-cache.

```
aws elasticache delete-serverless-cache --serverless-cache-name my-cache
```

Tindakan delete-serverless-cache CLI hanya menghapus satu cache tanpa server. Untuk menghapus beberapa cache, panggil delete-serverless-cache setiap cache tanpa server yang ingin Anda hapus. Anda tidak perlu menunggu satu cache nirserver selesai dihapus untuk menghapus cache yang lain.

Untuk Linux, macOS, atau Unix:

```
aws elasticache delete-serverless-cache \
  --serverless-cache-name my-cache
```

Untuk Windows:

```
aws elasticache delete-serverless-cache ^
  --serverless-cache-name my-cache
```

Untuk informasi selengkapnya, lihat ElastiCache topik AWS CLI untuk delete-serverless-cache.

Sekarang Anda dapat melanjutkan ke Langkah Berikutnya.

Langkah Berikutnya

Untuk informasi lebih lanjut tentang ElastiCache lihat:

- Bekerja dengan ElastiCache
- Penskalaan ElastiCache
- Kuota untuk ElastiCache
- ElastiCache praktik terbaik dan strategi caching
- Melihat ElastiCache acara

Tutorial: Memulai dengan Python dan ElastiCache

Bagian ini berisi tutorial langsung untuk membantu Anda mempelajari tentang Valkey dan ElastiCache Redis OSS. Kami mendorong Anda untuk mengikuti salah satu tutorial dengan bahasa yang sesuai.

> **Note**
>
> AWS SDKs tersedia untuk berbagai bahasa. Untuk daftar yang lengkapnya, lihat Alat untuk Amazon Web Services.

Topik

- Python dan ElastiCache

Python dan ElastiCache

Dalam tutorial ini, Anda menggunakan AWS SDK for Python (Boto3) untuk menulis program sederhana untuk melakukan operasi berikut: ElastiCache

- Buat ElastiCache untuk cluster Redis OSS (mode cluster diaktifkan dan mode cluster dinonaktifkan)
- Periksa apakah pengguna atau grup pengguna ada, jika tidak, buat mereka. (Fitur ini tersedia dengan Valkey 7.2 dan seterusnya, dan dengan Redis OSS 6.0 hingga 7.1.)
- Connect ke ElastiCache
- Lakukan operasi seperti pengaturan dan mendapatkan string, membaca dari dan menulis ke steam dan menerbitkan dan berlangganan dari saluran. Pub/Sub

Saat Anda mengerjakan tutorial ini, Anda dapat merujuk ke dokumentasi AWS SDK untuk Python (Boto). Bagian berikut khusus untuk ElastiCache: klien ElastiCache tingkat rendah

Prasyarat Tutorial

- Siapkan kunci AWS akses untuk menggunakan file AWS SDKs. Untuk informasi selengkapnya, lihat Menyiapkan ElastiCache.
- Instal Python 3.0 atau yang lebih baru. Untuk informasi selengkapnya, lihat https://www.python.org/downloads. Untuk petunjuk, lihat Quickstart dalam dokumentasi Boto 3.

Topik

- Tutorial: Membuat ElastiCache cluster dan pengguna
- Tutorial: Menghubungkan ke ElastiCache
- Contoh penggunaan

Tutorial: Membuat ElastiCache cluster dan pengguna

Contoh berikut menggunakan boto3 SDK ElastiCache untuk operasi manajemen Redis OSS (kluster atau pembuatan pengguna) dan redis-py/ untuk penanganan data. redis-py-cluster

Topik

- Buat klaster dengan mode klaster dinonaktifkan

- Membuat klaster dengan mode klaster dinonaktifkan dengan TLS dan RBAC
- Membuat klaster dengan mode klaster diaktifkan
- Membuat klaster dengan mode klaster diaktifkan dengan TLS dan RBAC
- Periksa apakah users/usergroup ada, jika tidak, buat

Buat klaster dengan mode klaster dinonaktifkan

Salin program berikut dan tempel ke file bernama CreateClusterModeDisabledCluster.py.

```
import boto3
import logging

logging.basicConfig(level=logging.INFO)
client = boto3.client('elasticache')

def
 create_cluster_mode_disabled(CacheNodeType='cache.t3.small',EngineVersion='6.0',NumCacheCluste
 cache cluster',ReplicationGroupId=None):
    """Creates an ElastiCache Cluster with cluster mode disabled

    Returns a dictionary with the API response

    :param CacheNodeType: Node type used on the cluster. If not specified,
 cache.t3.small will be used
    Refer to https://docs.aws.amazon.com/AmazonElastiCache/latest/dg/
CacheNodes.SupportedTypes.html for supported node types
    :param EngineVersion: Engine version to be used. If not specified, latest will be
 used.
    :param NumCacheClusters: Number of nodes in the cluster. Minimum 1 (just a primary
 node) and maximun 6 (1 primary and 5 replicas).
    If not specified, cluster will be created with 1 primary and 1 replica.
    :param ReplicationGroupDescription: Description for the cluster.
    :param ReplicationGroupId: Name for the cluster
    :return: dictionary with the API results

    """
    if not ReplicationGroupId:
        return 'ReplicationGroupId parameter is required'

    response = client.create_replication_group(
        AutomaticFailoverEnabled=True,
        CacheNodeType=CacheNodeType,
```

```
        Engine='valkey',
        EngineVersion=EngineVersion,
        NumCacheClusters=NumCacheClusters,
        ReplicationGroupDescription=ReplicationGroupDescription,
        ReplicationGroupId=ReplicationGroupId,
        SnapshotRetentionLimit=30,
    )
    return response


if __name__ == '__main__':

    # Creates an ElastiCache Cluster mode disabled cluster, based on cache.m6g.large
 nodes, Valkey 8.0, one primary and two replicas
    elasticacheResponse = create_cluster_mode_disabled(
        #CacheNodeType='cache.m6g.large',
        EngineVersion='8.0',
        NumCacheClusters=3,
        ReplicationGroupDescription='Valkey cluster mode disabled with replicas',
        ReplicationGroupId='valkey202104053'
        )

    logging.info(elasticacheResponse)
```

Untuk menjalankan program ini, masukkan perintah berikut:

```
python CreateClusterModeDisabledCluster.py
```

Untuk informasi selengkapnya, lihat Mengelola cluster di ElastiCache.

Membuat klaster dengan mode klaster dinonaktifkan dengan TLS dan RBAC

Untuk memastikan keamanan, Anda dapat menggunakan Keamanan Lapisan Pengangkutan (TLS) dan Kontrol Akses Berbasis Peran (RBAC) saat membuat klaster dengan mode klaster dinonaktifkan. Tidak seperti Valkey atau Redis OSS AUTH, di mana semua klien yang diautentikasi memiliki akses grup replikasi penuh jika token mereka diautentikasi, RBAC memungkinkan Anda untuk mengontrol akses cluster melalui grup pengguna. Grup pengguna ini dirancang sebagai cara untuk mengatur akses ke grup replikasi. Untuk informasi selengkapnya, lihat Kontrol Akses Berbasis Peran (RBAC).

Salin program berikut dan tempel ke file bernama ClusterModeDisabledWithRBAC.py.

```
import boto3
import logging
```

```
logging.basicConfig(level=logging.INFO)
client = boto3.client('elasticache')

def
 create_cluster_mode_disabled_rbac(CacheNodeType='cache.t3.small',EngineVersion='6.0',NumCacheC
 cache cluster',ReplicationGroupId=None, UserGroupIds=None,
 SecurityGroupIds=None,CacheSubnetGroupName=None):
    """Creates an ElastiCache Cluster with cluster mode disabled and RBAC

    Returns a dictionary with the API response

    :param CacheNodeType: Node type used on the cluster. If not specified,
 cache.t3.small will be used
    Refer to https://docs.aws.amazon.com/AmazonElastiCache/latest/dg/
CacheNodes.SupportedTypes.html for supported node types
    :param EngineVersion: Engine version to be used. If not specified, latest will be
 used.
    :param NumCacheClusters: Number of nodes in the cluster. Minimum 1 (just a primary
 node) and maximun 6 (1 primary and 5 replicas).
    If not specified, cluster will be created with 1 primary and 1 replica.
    :param ReplicationGroupDescription: Description for the cluster.
    :param ReplicationGroupId: Mandatory name for the cluster.
    :param UserGroupIds: The ID of the user group to be assigned to the cluster.
    :param SecurityGroupIds: List of security groups to be assigned. If not defined,
 default will be used
    :param CacheSubnetGroupName: subnet group where the cluster will be placed. If not
 defined, default will be used.
    :return: dictionary with the API results

    """
    if not ReplicationGroupId:
        return {'Error': 'ReplicationGroupId parameter is required'}
    elif not isinstance(UserGroupIds,(list)):
        return {'Error': 'UserGroupIds parameter is required and must be a list'}

    params={'AutomaticFailoverEnabled': True,
            'CacheNodeType': CacheNodeType,
            'Engine': 'valkey',
            'EngineVersion': EngineVersion,
            'NumCacheClusters': NumCacheClusters,
            'ReplicationGroupDescription': ReplicationGroupDescription,
            'ReplicationGroupId': ReplicationGroupId,
            'SnapshotRetentionLimit': 30,
```

```
            'TransitEncryptionEnabled': True,
            'UserGroupIds':UserGroupIds
        }

    # defaults will be used if CacheSubnetGroupName or SecurityGroups are not explicit.
    if isinstance(SecurityGroupIds,(list)):
        params.update({'SecurityGroupIds':SecurityGroupIds})
    if CacheSubnetGroupName:
        params.update({'CacheSubnetGroupName':CacheSubnetGroupName})

    response = client.create_replication_group(**params)
    return response

if __name__ == '__main__':

    # Creates an ElastiCache Cluster mode disabled cluster, based on cache.m6g.large
 nodes, Valkey 8.0, one primary and two replicas.
    # Assigns the existent user group "mygroup" for RBAC authentication

    response=create_cluster_mode_disabled_rbac(
        CacheNodeType='cache.m6g.large',
        EngineVersion='8.0',
        NumCacheClusters=3,
        ReplicationGroupDescription='Valkey cluster mode disabled with replicas',
        ReplicationGroupId='valkey202104',
        UserGroupIds=[
            'mygroup'
        ],
        SecurityGroupIds=[
            'sg-7cc73803'
        ],
        CacheSubnetGroupName='default'
    )

    logging.info(response)
```

Untuk menjalankan program ini, masukkan perintah berikut:

```
python ClusterModeDisabledWithRBAC.py
```

Untuk informasi selengkapnya, lihat Mengelola cluster di ElastiCache.

Membuat klaster dengan mode klaster diaktifkan

Salin program berikut dan tempel ke file bernama ClusterModeEnabled.py.

```
import boto3
import logging

logging.basicConfig(level=logging.INFO)
client = boto3.client('elasticache')

def
 create_cluster_mode_enabled(CacheNodeType='cache.t3.small',EngineVersion='6.0',NumNodeGroups=1
 ReplicationGroupDescription='Sample cache with cluster mode
 enabled',ReplicationGroupId=None):
    """Creates an ElastiCache Cluster with cluster mode enabled

    Returns a dictionary with the API response

    :param CacheNodeType: Node type used on the cluster. If not specified,
 cache.t3.small will be used
    Refer to https://docs.aws.amazon.com/AmazonElastiCache/latest/dg/
CacheNodes.SupportedTypes.html for supported node types
    :param EngineVersion: Engine version to be used. If not specified, latest will be
 used.
    :param NumNodeGroups: Number of shards in the cluster. Minimum 1 and maximun 90.
    If not specified, cluster will be created with 1 shard.
    :param ReplicasPerNodeGroup: Number of replicas per shard. If not specified 1
 replica per shard will be created.
    :param ReplicationGroupDescription: Description for the cluster.
    :param ReplicationGroupId: Name for the cluster
    :return: dictionary with the API results

    """
    if not ReplicationGroupId:
        return 'ReplicationGroupId parameter is required'

    response = client.create_replication_group(
        AutomaticFailoverEnabled=True,
        CacheNodeType=CacheNodeType,
        Engine='valkey',
        EngineVersion=EngineVersion,
        ReplicationGroupDescription=ReplicationGroupDescription,
        ReplicationGroupId=ReplicationGroupId,
```

```
    #   Creates a cluster mode enabled cluster with 1 shard(NumNodeGroups), 1 primary
 node (implicit) and 2 replicas (replicasPerNodeGroup)
        NumNodeGroups=NumNodeGroups,
        ReplicasPerNodeGroup=ReplicasPerNodeGroup,
        CacheParameterGroupName='default.valkey7.2.cluster.on'
    )

    return response


# Creates a cluster mode enabled
response = create_cluster_mode_enabled(
    CacheNodeType='cache.m6g.large',
    EngineVersion='6.0',
    ReplicationGroupDescription='Valkey cluster mode enabled with replicas',
    ReplicationGroupId='valkey20210',
#   Creates a cluster mode enabled cluster with 1 shard(NumNodeGroups), 1 primary
 (implicit) and 2 replicas (replicasPerNodeGroup)
    NumNodeGroups=2,
    ReplicasPerNodeGroup=1,
)

logging.info(response)
```

Untuk menjalankan program ini, masukkan perintah berikut:

```
python ClusterModeEnabled.py
```

Untuk informasi selengkapnya, lihat Mengelola cluster di ElastiCache.

Membuat klaster dengan mode klaster diaktifkan dengan TLS dan RBAC

Untuk memastikan keamanan, Anda dapat menggunakan Keamanan Lapisan Pengangkutan (TLS) dan Kontrol Akses Berbasis Peran (RBAC) saat membuat klaster dengan mode klaster diaktifkan. Tidak seperti Valkey atau Redis OSS AUTH, di mana semua klien yang diautentikasi memiliki akses grup replikasi penuh jika token mereka diautentikasi, RBAC memungkinkan Anda untuk mengontrol akses cluster melalui grup pengguna. Grup pengguna ini dirancang sebagai cara untuk mengatur akses ke grup replikasi. Untuk informasi selengkapnya, lihat Kontrol Akses Berbasis Peran (RBAC).

Salin program berikut dan tempel ke file bernama ClusterModeEnabledWithRBAC.py.

```
import boto3
import logging
```

```
logging.basicConfig(level=logging.INFO)
client = boto3.client('elasticache')

def
 create_cluster_mode_enabled(CacheNodeType='cache.t3.small',EngineVersion='6.0',NumNodeGroups=1
 ReplicationGroupDescription='Sample cache with cluster
 mode enabled',ReplicationGroupId=None,UserGroupIds=None,
 SecurityGroupIds=None,CacheSubnetGroupName=None,CacheParameterGroupName='default.valkey7.2.clu
    """Creates an ElastiCache Cluster with cluster mode enabled and RBAC

    Returns a dictionary with the API response

    :param CacheNodeType: Node type used on the cluster. If not specified,
 cache.t3.small will be used
    Refer to https://docs.aws.amazon.com/AmazonElastiCache/latest/dg/
CacheNodes.SupportedTypes.html for supported node types
    :param EngineVersion: Engine version to be used. If not specified, latest will be
 used.
    :param NumNodeGroups: Number of shards in the cluster. Minimum 1 and maximun 90.
    If not specified, cluster will be created with 1 shard.
    :param ReplicasPerNodeGroup: Number of replicas per shard. If not specified 1
 replica per shard will be created.
    :param ReplicationGroupDescription: Description for the cluster.
    :param ReplicationGroupId: Name for the cluster.
    :param CacheParameterGroupName: Parameter group to be used. Must be compatible with
 the engine version and cluster mode enabled.
    :return: dictionary with the API results

    """
    if not ReplicationGroupId:
        return 'ReplicationGroupId parameter is required'
    elif not isinstance(UserGroupIds,(list)):
        return {'Error': 'UserGroupIds parameter is required and must be a list'}

    params={'AutomaticFailoverEnabled': True,
            'CacheNodeType': CacheNodeType,
            'Engine': 'valkey',
            'EngineVersion': EngineVersion,
            'ReplicationGroupDescription': ReplicationGroupDescription,
            'ReplicationGroupId': ReplicationGroupId,
            'SnapshotRetentionLimit': 30,
            'TransitEncryptionEnabled': True,
            'UserGroupIds':UserGroupIds,
```

```
            'NumNodeGroups': NumNodeGroups,
            'ReplicasPerNodeGroup': ReplicasPerNodeGroup,
            'CacheParameterGroupName': CacheParameterGroupName
        }

    # defaults will be used if CacheSubnetGroupName or SecurityGroups are not explicit.
    if isinstance(SecurityGroupIds,(list)):
        params.update({'SecurityGroupIds':SecurityGroupIds})
    if CacheSubnetGroupName:
        params.update({'CacheSubnetGroupName':CacheSubnetGroupName})

    response = client.create_replication_group(**params)
    return response

if __name__ == '__main__':
    # Creates a cluster mode enabled cluster
    response = create_cluster_mode_enabled(
        CacheNodeType='cache.m6g.large',
        EngineVersion='7.2',
        ReplicationGroupDescription='Valkey cluster mode enabled with replicas',
        ReplicationGroupId='valkey2021',
    #   Creates a cluster mode enabled cluster with 1 shard(NumNodeGroups), 1 primary
 (implicit) and 2 replicas (replicasPerNodeGroup)
        NumNodeGroups=2,
        ReplicasPerNodeGroup=1,
        UserGroupIds=[
            'mygroup'
        ],
        SecurityGroupIds=[
            'sg-7cc73803'
        ],
        CacheSubnetGroupName='default'

    )

    logging.info(response)
```

Untuk menjalankan program ini, masukkan perintah berikut:

```
python ClusterModeEnabledWithRBAC.py
```

Untuk informasi selengkapnya, lihat Mengelola cluster di ElastiCache.

Periksa apakah users/usergroup ada, jika tidak, buat

Dengan RBAC, Anda membuat pengguna dan memberi mereka izin tertentu menggunakan string akses. Anda menetapkan pengguna ke grup pengguna yang selaras dengan peran tertentu (administrator, sumber daya manusia) yang kemudian disebarkan ke satu atau lebih ElastiCache untuk grup replikasi Redis OSS. Dengan melakukan ini, Anda dapat menetapkan batas keamanan antara klien menggunakan grup atau grup replikasi Valkey atau Redis OSS yang sama dan mencegah klien mengakses data satu sama lain. Untuk informasi selengkapnya, lihat Kontrol Akses Berbasis Peran (RBAC).

Salin program berikut dan tempel ke file bernama UserAndUserGroups.py. Perbarui mekanisme untuk memberikan kredensial. Kredensial dalam contoh ini ditampilkan sebagai dapat diganti dan diberikan item yang tidak dideklarasikan. Hindari kredensial hard-coding.

Contoh ini menggunakan string akses dengan izin untuk pengguna. Untuk informasi selengkapnya tentang string akses, lihatMenentukan Izin Menggunakan String Akses.

```
import boto3
import logging

logging.basicConfig(level=logging.INFO)
client = boto3.client('elasticache')

def check_user_exists(UserId):
    """Checks if UserId exists

    Returns True if UserId exists, otherwise False
    :param UserId: ElastiCache User ID
    :return: True|False
    """
    try:
        response = client.describe_users(
            UserId=UserId,
        )
        if response['Users'][0]['UserId'].lower() == UserId.lower():
            return True
    except Exception as e:
        if e.response['Error']['Code'] == 'UserNotFound':
            logging.info(e.response['Error'])
            return False
        else:
            raise
```

```
def check_group_exists(UserGroupId):
    """Checks if UserGroupID exists

    Returns True if Group ID exists, otherwise False
    :param UserGroupId: ElastiCache User ID
    :return: True|False
    """

    try:
        response = client.describe_user_groups(
            UserGroupId=UserGroupId
        )
        if response['UserGroups'][0]['UserGroupId'].lower() == UserGroupId.lower():
            return True
    except Exception as e:
        if e.response['Error']['Code'] == 'UserGroupNotFound':
            logging.info(e.response['Error'])
            return False
        else:
            raise

def create_user(UserId=None,UserName=None,Password=None,AccessString=None):
    """Creates a new user

    Returns the ARN for the newly created user or the error message
    :param UserId: ElastiCache user ID. User IDs must be unique
    :param UserName: ElastiCache user name. ElastiCache allows multiple users with the
 same name as long as the associated user ID is unique.
    :param Password: Password for user. Must have at least 16 chars.
    :param AccessString: Access string with the permissions for the user.
    :return: user ARN
    """
    try:
        response = client.create_user(
            UserId=UserId,
            UserName=UserName,
            Engine='Redis',
            Passwords=[Password],
            AccessString=AccessString,
            NoPasswordRequired=False
        )
        return response['ARN']
    except Exception as e:
```

```
        logging.info(e.response['Error'])
        return e.response['Error']

def create_group(UserGroupId=None, UserIds=None):
    """Creates a new group.
    A default user is required (mandatory) and should be specified in the UserIds list

    Return: Group ARN
    :param UserIds: List with user IDs to be associated with the new group. A default
 user is required
    :param UserGroupId: The ID (name) for the group
    :return: Group ARN
    """
    try:
        response = client.create_user_group(
            UserGroupId=UserGroupId,
            Engine='Redis',
            UserIds=UserIds
        )
        return response['ARN']
    except Exception as e:
        logging.info(e.response['Error'])


if __name__ == '__main__':

    groupName='mygroup2'
    userName = 'myuser2'
    userId=groupName+'-'+userName

    # Creates a new user if the user ID does not exist.
    for tmpUserId,tmpUserName in [ (userId,userName), (groupName+'-
default','default')]:
        if not check_user_exists(tmpUserId):
            response=create_user(UserId=tmpUserId,
 UserName=EXAMPLE,Password=EXAMPLE,AccessString='on ~* +@all')
            logging.info(response)
        # assigns the new user ID to the user group
    if not check_group_exists(groupName):
        UserIds = [ userId , groupName+'-default']
        response=create_group(UserGroupId=groupName,UserIds=UserIds)
        logging.info(response)
```

Untuk menjalankan program ini, masukkan perintah berikut:

```
python UserAndUserGroups.py
```

Tutorial: Menghubungkan ke ElastiCache

Contoh berikut menggunakan klien Valkey atau Redis OSS untuk terhubung ke. ElastiCache

Topik

- Menghubungkan ke klaster yang menonaktifkan mode klaster
- Menghubungkan ke klaster yang mengaktifkan mode klaster

Menghubungkan ke klaster yang menonaktifkan mode klaster

Salin program berikut dan tempel ke file bernama ConnectClusterModeDisabled.py. Perbarui mekanisme untuk memberikan kredensial. Kredensial dalam contoh ini ditampilkan sebagai dapat diganti dan diberikan item yang tidak dideklarasikan. Hindari kredensial hard-coding.

```
from redis import Redis
import logging

logging.basicConfig(level=logging.INFO)
redis = Redis(host='primary.xxx.yyyyyy.zzz1.cache.amazonaws.com', port=6379,
 decode_responses=True, ssl=True, username=example, password=EXAMPLE)

if redis.ping():
    logging.info("Connected to Redis")
```

Untuk menjalankan program ini, masukkan perintah berikut:

```
python ConnectClusterModeDisabled.py
```

Menghubungkan ke klaster yang mengaktifkan mode klaster

Salin program berikut dan tempel ke file bernama ConnectClusterModeEnabled.py.

```
from rediscluster import RedisCluster
import logging

logging.basicConfig(level=logging.INFO)
```

```
redis = RedisCluster(startup_nodes=[{"host":
 "xxx.yyy.clustercfg.zzz1.cache.amazonaws.com","port": "6379"}],
 decode_responses=True,skip_full_coverage_check=True)

if redis.ping():
    logging.info("Connected to Redis")
```

Untuk menjalankan program ini, masukkan perintah berikut:

```
python ConnectClusterModeEnabled.py
```

Contoh penggunaan

Contoh berikut menggunakan boto3 SDK ElastiCache untuk bekerja dengan ElastiCache Redis OSS.

Topik

- Menetapkan dan Mendapatkan string
- Tentukan dan Dapatkan hash dengan beberapa item
- Publikasikan (tulis) dan berlangganan (baca) dari Pub/Sub saluran
- Tulis dan baca dari aliran

Menetapkan dan Mendapatkan string

Salin program berikut dan tempel ke file bernama SetAndGetStrings.py.

```
import time
import logging
logging.basicConfig(level=logging.INFO,format='%(asctime)s: %(message)s')

keyName='mykey'
currTime=time.ctime(time.time())

# Set the key 'mykey' with the current date and time as value.
# The Key will expire and removed from cache in 60 seconds.
redis.set(keyName, currTime, ex=60)

# Sleep just for better illustration of TTL (expiration) value
time.sleep(5)

# Retrieve the key value and current TTL
```

```
keyValue=redis.get(keyName)
keyTTL=redis.ttl(keyName)

logging.info("Key {} was set at {} and has {} seconds until expired".format(keyName,
 keyValue, keyTTL))
```

Untuk menjalankan program ini, masukkan perintah berikut:

```
python SetAndGetStrings.py
```

Tentukan dan Dapatkan hash dengan beberapa item

Salin program berikut dan tempel ke file bernama SetAndGetHash.py.

```
import logging
import time

logging.basicConfig(level=logging.INFO,format='%(asctime)s: %(message)s')

keyName='mykey'
keyValues={'datetime': time.ctime(time.time()), 'epochtime': time.time()}

# Set the hash 'mykey' with the current date and time in human readable format
 (datetime field) and epoch number (epochtime field).
redis.hset(keyName, mapping=keyValues)

# Set the key to expire and removed from cache in 60 seconds.
redis.expire(keyName, 60)

# Sleep just for better illustration of TTL (expiration) value
time.sleep(5)

# Retrieves all the fields and current TTL
keyValues=redis.hgetall(keyName)
keyTTL=redis.ttl(keyName)

logging.info("Key {} was set at {} and has {} seconds until expired".format(keyName,
 keyValues, keyTTL))
```

Untuk menjalankan program ini, masukkan perintah berikut:

```
python SetAndGetHash.py
```

Publikasikan (tulis) dan berlangganan (baca) dari Pub/Sub saluran

Salin program berikut dan tempel ke file bernama PubAndSub.py.

```
import logging
import time

def handlerFunction(message):
    """Prints message got from PubSub channel to the log output

    Return None
    :param message: message to log
    """
    logging.info(message)

logging.basicConfig(level=logging.INFO)
redis = Redis(host="redis202104053.tihewd.ng.0001.use1.cache.amazonaws.com", port=6379,
 decode_responses=True)


# Creates the subscriber connection on "mychannel"
subscriber = redis.pubsub()
subscriber.subscribe(**{'mychannel': handlerFunction})

# Creates a new thread to watch for messages while the main process continues with its
 routines
thread = subscriber.run_in_thread(sleep_time=0.01)

# Creates publisher connection on "mychannel"
redis.publish('mychannel', 'My message')

# Publishes several messages. Subscriber thread will read and print on log.
while True:
    redis.publish('mychannel',time.ctime(time.time()))
    time.sleep(1)
```

Untuk menjalankan program ini, masukkan perintah berikut:

```
python PubAndSub.py
```

Tulis dan baca dari aliran

Salin program berikut dan tempel ke file bernama ReadWriteStream.py.

```
from redis import Redis
import redis.exceptions as exceptions
import logging
import time
import threading

logging.basicConfig(level=logging.INFO)

def writeMessage(streamName):
    """Starts a loop writting the current time and thread name to 'streamName'

    :param streamName: Stream (key) name to write messages.
    """
    fieldsDict={'writerId':threading.currentThread().getName(),'myvalue':None}
    while True:
        fieldsDict['myvalue'] = time.ctime(time.time())
        redis.xadd(streamName,fieldsDict)
        time.sleep(1)

def readMessage(groupName=None,streamName=None):
    """Starts a loop reading from 'streamName'
    Multiple threads will read from the same stream consumer group. Consumer group is
 used to coordinate data distribution.
    Once a thread acknowleges the message, it won't be provided again. If message
 wasn't acknowledged, it can be served to another thread.

    :param groupName: stream group were multiple threads will read.
    :param streamName: Stream (key) name where messages will be read.
    """

    readerID=threading.currentThread().getName()
    while True:
        try:
            # Check if the stream has any message
            if redis.xlen(streamName)>0:
                # Check if if the messages are new (not acknowledged) or not (already
 processed)
                streamData=redis.xreadgroup(groupName,readerID,
{streamName:'>'},count=1)
                if len(streamData) > 0:
                    msgId,message = streamData[0][1][0]
                    logging.info("{}: Got {} from ID
 {}".format(readerID,message,msgId))
```

```
                    #Do some processing here. If the message has been processed
 sucessfuly, acknowledge it and (optional) delete the message.
                    redis.xack(streamName,groupName,msgId)
                    logging.info("Stream message ID {} read and processed successfuly
 by {}".format(msgId,readerID))
                    redis.xdel(streamName,msgId)
            else:
                pass
        except:
            raise

        time.sleep(0.5)

# Creates the stream 'mystream' and consumer group 'myworkergroup' where multiple
 threads will write/read.
try:
    redis.xgroup_create('mystream','myworkergroup',mkstream=True)
except exceptions.ResponseError as e:
    logging.info("Consumer group already exists. Will continue despite the error:
 {}".format(e))
except:
    raise

# Starts 5 writer threads.
for writer_no in range(5):
    writerThread = threading.Thread(target=writeMessage, name='writer-'+str(writer_no),
 args=('mystream',),daemon=True)
    writerThread.start()

# Starts 10 reader threads
for reader_no in range(10):
    readerThread = threading.Thread(target=readMessage, name='reader-'+str(reader_no),
 args=('myworkergroup','mystream',),daemon=True)
    readerThread.daemon = True
    readerThread.start()

# Keep the code running for 30 seconds
time.sleep(30)
```

Untuk menjalankan program ini, masukkan perintah berikut:

```
python ReadWriteStream.py
```

Tutorial: Mengkonfigurasi Lambda untuk ElastiCache mengakses di VPC

Dalam tutorial ini Anda dapat mempelajari cara membuat cache ElastiCache tanpa server, membuat fungsi Lambda, lalu menguji fungsi Lambda, dan secara opsional membersihkan setelahnya.

Topik

Langkah 1: Membuat cache ElastiCache tanpa server.

Untuk membuat cache tanpa server, ikuti langkah-langkah ini.

Langkah 1.1: Buat cache tanpa server

Pada langkah ini, Anda membuat cache tanpa server di VPC Amazon default di wilayah us-east-1 di akun Anda menggunakan (CLI). AWS Command Line Interface Untuk informasi tentang membuat cache tanpa server menggunakan ElastiCache konsol atau API, lihat. Buat cache tanpa server Redis OSS

```
aws elasticache create-serverless-cache \
  --serverless-cache-name cache-01  \
--description "ElastiCache IAM auth application" \
--engine valkey
```

Perhatikan bahwa nilai bidang Status diatur ke `CREATING`. Diperlukan waktu satu menit ElastiCache untuk menyelesaikan pembuatan cache Anda.

Langkah 1.2: Salin titik akhir cache tanpa server

Verifikasi bahwa ElastiCache untuk Redis OSS telah selesai membuat cache dengan perintah. `describe-serverless-caches`

```
aws elasticache describe-serverless-caches \
--serverless-cache-name cache-01
```

Salin Alamat Titik Akhir yang ditunjukkan pada output. Anda akan memerlukan alamat ini saat membuat paket deployment untuk fungsi Lambda Anda.

Langkah 1.3: Buat Peran IAM

1. Buat dokumen kebijakan kepercayaan IAM, seperti yang ditunjukkan di bawah ini, untuk peran Anda yang memungkinkan akun Anda mengambil peran baru. Simpan kebijakan ini ke file bernama trust-policy.json.

   JSON

   ```
   {
   "Version": "2012-10-17",
       "Statement": [{
   "Effect": "Allow",
           "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
           "Action": "sts:AssumeRole"
       },
       {
         "Effect": "Allow",
         "Principal": {
           "Service": "lambda.amazonaws.com"
         },
         "Action": "sts:AssumeRole"
       }]
   }
   ```

2. Buat dokumen kebijakan IAM, seperti yang ditunjukkan di bawah ini. Simpan kebijakan ke file bernama policy.json.

   JSON

   ```
   {
   "Version": "2012-10-17",
     "Statement": [
       {
   "Effect" : "Allow",
         "Action" : [
           "elasticache:Connect"
         ],
   ```

```
      "Resource" : [
        "arn:aws:elasticache:us-
east-1:123456789012:serverlesscache:cache-01",
        "arn:aws:elasticache:us-east-1:123456789012:user:iam-user-01"
      ]
    }
  ]
}
```

3. Buat peran IAM.

```
aws iam create-role \
--role-name "elasticache-iam-auth-app" \
--assume-role-policy-document file://trust-policy.json
```

4. Buat kebijakan IAM.

```
aws iam create-policy \
  --policy-name "elasticache-allow-all" \
  --policy-document file://policy.json
```

5. Lampirkan kebijakan IAM ke peran tersebut.

```
aws iam attach-role-policy \
 --role-name "elasticache-iam-auth-app" \
 --policy-arn "arn:aws:iam::123456789012:policy/elasticache-allow-all"
```

Langkah 1.4: Buat pengguna default

1. Buat pengguna default baru.

```
aws elasticache create-user \
  --user-name default \
--user-id default-user-disabled \
--engine redis \
--authentication-mode Type=no-password-required \
--access-string "off +get ~keys*"
```

2. Buat pengguna baru yang didukung IAM.

```
aws elasticache create-user \
```

```
  --user-name iam-user-01 \
--user-id iam-user-01 \
--authentication-mode Type=iam \
--engine redis \
--access-string "on ~* +@all"
```

3. Buat grup pengguna dan lampirkan pengguna.

```
aws elasticache create-user-group \
  --user-group-id iam-user-group-01 \
--engine redis \
--user-ids default-user-disabled iam-user-01

aws elasticache modify-serverless-cache \
  --serverless-cache-name cache-01  \
--user-group-id iam-user-group-01
```

Langkah 2: Buat fungsi Lambda untuk ElastiCache

Untuk membuat fungsi Lambda untuk mengakses ElastiCache cache, lakukan langkah-langkah ini.

Langkah 2.1: Buat fungsi Lambda

Dalam tutorial ini, kami memberikan contoh kode dalam Python untuk fungsi Lambda Anda.

Python

Contoh kode Python berikut membaca dan menulis item ke cache Anda ElastiCache .
Salin kode tersebut dan simpan ke dalam file bernama `app.py`. Pastikan untuk mengganti `elasticache_endpoint` nilai dalam kode dengan alamat titik akhir yang Anda salin pada langkah sebelumnya.

```
from typing import Tuple, Union
from urllib.parse import ParseResult, urlencode, urlunparse

import botocore.session
import redis
from botocore.model import ServiceId
from botocore.signers import RequestSigner
from cachetools import TTLCache, cached
import uuid
```

```
class ElastiCacheIAMProvider(redis.CredentialProvider):
    def __init__(self, user, cache_name, is_serverless=False, region="us-east-1"):
        self.user = user
        self.cache_name = cache_name
        self.is_serverless = is_serverless
        self.region = region

        session = botocore.session.get_session()
        self.request_signer = RequestSigner(
            ServiceId("elasticache"),
            self.region,
            "elasticache",
            "v4",
            session.get_credentials(),
            session.get_component("event_emitter"),
        )

    # Generated IAM tokens are valid for 15 minutes
    @cached(cache=TTLCache(maxsize=128, ttl=900))
    def get_credentials(self) -> Union[Tuple[str], Tuple[str, str]]:
        query_params = {"Action": "connect", "User": self.user}
        if self.is_serverless:
            query_params["ResourceType"] = "ServerlessCache"
        url = urlunparse(
            ParseResult(
                scheme="https",
                netloc=self.cache_name,
                path="/",
                query=urlencode(query_params),
                params="",
                fragment="",
            )
        )
        signed_url = self.request_signer.generate_presigned_url(
            {"method": "GET", "url": url, "body": {}, "headers": {}, "context": {}},
            operation_name="connect",
            expires_in=900,
            region_name=self.region,
        )
        # RequestSigner only seems to work if the URL has a protocol, but
        # Elasticache only accepts the URL without a protocol
        # So strip it off the signed URL before returning
        return (self.user, signed_url.removeprefix("https://"))
```

```
def lambda_handler(event, context):
    username = "iam-user-01" # replace with your user id
    cache_name = "cache-01" # replace with your cache name
    elasticache_endpoint = "cache-01-xxxxx.serverless.use1.cache.amazonaws.com" #
 replace with your cache endpoint
    creds_provider = ElastiCacheIAMProvider(user=username, cache_name=cache_name,
 is_serverless=True)
    redis_client = redis.Redis(host=elasticache_endpoint, port=6379,
 credential_provider=creds_provider, ssl=True, ssl_cert_reqs="none")

    key='uuid'
    # create a random UUID - this will be the sample element we add to the cache
    uuid_in = uuid.uuid4().hex
    redis_client.set(key, uuid_in)
    result = redis_client.get(key)
    decoded_result = result.decode("utf-8")
    # check the retrieved item matches the item added to the cache and print
    # the results
    if decoded_result == uuid_in:
        print(f"Success: Inserted {uuid_in}. Fetched {decoded_result} from Valkey.")
    else:
        raise Exception(f"Bad value retrieved. Expected {uuid_in}, got
 {decoded_result}")

    return "Fetched value from Valkey"
```

Kode ini menggunakan pustaka redis-py Python untuk memasukkan item ke dalam cache Anda dan mengambilnya. Kode ini menggunakan cachetools untuk menyimpan token Auth IAM yang dihasilkan selama 15 menit. Untuk membuat paket penerapan yang berisi redis-py dan cachetools, lakukan langkah-langkah berikut.

Di direktori proyek Anda yang berisi file kode sumber app.py, buat paket folder untuk menginstal pustaka redis-py dan cachetools ke dalamnya.

```
mkdir package
```

Instal redis-py, cachetools menggunakan pip.

```
pip install --target ./package redis
pip install --target ./package cachetools
```

Buat file.zip yang berisi pustaka redis-py dan cachetools. Di Linux atau macOS, jalankan perintah CLI berikut. Di Windows, gunakan utilitas zip pilihan Anda untuk membuat file.zip dengan pustaka redis-py dan cachetools di root.

```
cd package
zip -r ../my_deployment_package.zip .
```

Tambahkan kode fungsi Anda ke file .zip. Di Linux atau macOS, jalankan perintah CLI berikut. Di Windows, gunakan utilitas zip pilihan Anda untuk menambahkan app.py ke root file.zip Anda.

```
cd ..
zip my_deployment_package.zip app.py
```

Langkah 2.2: Buat peran IAM (peran eksekusi)

Lampirkan kebijakan AWS terkelola yang diberi nama `AWSLambdaVPCAccessExecutionRole` ke peran.

```
aws iam attach-role-policy \
 --role-name "elasticache-iam-auth-app" \
 --policy-arn "arn:aws:iam::aws:policy/service-role/AWSLambdaVPCAccessExecutionRole"
```

Langkah 2.3: Unggah paket deployment (buat fungsi Lambda)

Pada langkah ini, Anda membuat fungsi Lambda (AccessValkey) menggunakan perintah AWS CLI create-function.

Dari direktori proyek yang berisi file paket deployment Anda .zip, jalankan perintah Lambda CLI berikut. `create-function`

Untuk opsi peran, gunakan ARN dari peran eksekusi yang Anda buat di langkah sebelumnya. Untuk vpc-config masukkan daftar yang dipisahkan koma dari subnet VPC default Anda dan ID grup keamanan VPC default Anda. Anda dapat menemukan nilai-nilai ini di konsol Amazon VPC. Untuk menemukan subnet VPC default Anda, pilih Your VPCs, lalu pilih VPC default AWS akun Anda. Untuk menemukan grup keamanan untuk VPC ini, buka Keamanan dan pilih Grup keamanan. Pastikan Anda memilih wilayah us-east-1.

```
aws lambda create-function \
```

```
--function-name AccessValkey  \
--region us-east-1 \
--zip-file fileb://my_deployment_package.zip \
--role arn:aws:iam::123456789012:role/elasticache-iam-auth-app \
--handler app.lambda_handler \
--runtime python3.12  \
--timeout 30 \
--vpc-config SubnetIds=comma-separated-vpc-subnet-ids,SecurityGroupIds=default-
security-group-id
```

Langkah 3: Uji fungsi Lambda dengan ElastiCache

Pada langkah ini, Anda menjalankan fungsi Lambda secara manual menggunakan perintah pemanggilan. Ketika fungsi Lambda dijalankan, ia menghasilkan UUID dan menulisnya ke ElastiCache cache yang Anda tentukan dalam kode Lambda Anda. Fungsi Lambda kemudian mengambil item dari cache.

1. Memanggil fungsi Lambda AccessValkey () menggunakan perintah pemanggilan AWS Lambda .

   ```
   aws lambda invoke \
   --function-name AccessValkey  \
   --region us-east-1 \
   output.txt
   ```

2. Verifikasikan bahwa fungsi Lambda berhasil dijalankan sebagai berikut:

   - Tinjau file output.txt.
   - Verifikasi hasil di CloudWatch Log dengan membuka CloudWatch konsol dan memilih grup log untuk fungsi Anda (/aws/lambda/AccessValkey). Log stream akan berisi output seperti yang berikut ini:

     ```
     Success: Inserted 826e70c5f4d2478c8c18027125a3e01e. Fetched
      826e70c5f4d2478c8c18027125a3e01e from Valkey.
     ```

   - Tinjau hasilnya di AWS Lambda konsol.

Langkah 4: Bersihkan (Opsional)

Untuk membersihkannya, ambil langkah-langkah ini.

Langkah 4.1: Hapus fungsi Lambda

```
aws lambda delete-function \
 --function-name AccessValkey
```

Langkah 4.2: Hapus cache Tanpa Server

Hapus cache.

```
aws elasticache delete-serverless-cache \
 --serverless-cache-name cache-01
```

Hapus pengguna dan grup pengguna.

```
aws elasticache delete-user \
 --user-id default-user-disabled

aws elasticache delete-user \
 --user-id iam-user-01

aws elasticache delete-user-group \
 --user-group-id iam-user-group-01
```

Langkah 4.3: Hapus Peran dan kebijakan IAM

```
aws iam detach-role-policy \
 --role-name "elasticache-iam-auth-app" \
 --policy-arn "arn:aws:iam::123456789012:policy/elasticache-allow-all"

aws iam detach-role-policy \
--role-name "elasticache-iam-auth-app" \
--policy-arn "arn:aws:iam::aws:policy/service-role/AWSLambdaVPCAccessExecutionRole"

aws iam delete-role \
 --role-name "elasticache-iam-auth-app"

 aws iam delete-policy \
  --policy-arn "arn:aws:iam::123456789012:policy/elasticache-allow-all"
```

Merancang dan mengelola ElastiCache cluster Anda sendiri

Jika Anda memerlukan kontrol halus atas cluster Anda, Anda dapat memilih untuk mendesain ElastiCache cluster Anda sendiri. ElastiCache memungkinkan Anda untuk mengoperasikan cluster berbasis node dengan memilih tipe node, jumlah node, dan penempatan node di seluruh AWS Availability Zones untuk cluster Anda. Karena ElastiCache merupakan layanan yang dikelola sepenuhnya, ia secara otomatis mengelola penyediaan perangkat keras, pemantauan, penggantian node, dan penambalan perangkat lunak untuk cluster Anda.

Untuk informasi tentang cara menyiapkannya, lihat Menyiapkan ElastiCache. Untuk detail tentang cara mengelola, memperbarui, atau menghapus simpul atau klaster, lihat Mengelola node di ElastiCache. Untuk ikhtisar komponen utama ElastiCache penerapan Amazon saat Anda mendesain ElastiCache klaster Anda sendiri, lihat konsep-konsep utama ini.

Topik

- ElastiCache komponen dan fitur
- ElastiCache terminologi
- Tutorial: Cara mendesain cluster Anda sendiri
- Menghapus klaster
- ElastiCache Tutorial dan video lainnya
- Mengelola node di ElastiCache
- Mengelola cluster di ElastiCache
- Membandingkan cache yang dirancang sendiri Valkey, Memcached, dan Redis OSS
- Migrasi online untuk Valkey atau Redis OSS
- Memilih wilayah dan zona ketersediaan untuk ElastiCache

ElastiCache komponen dan fitur

Berikut ini, Anda dapat menemukan ikhtisar komponen utama ElastiCache penyebaran Amazon.

Topik

- ElastiCache simpul
- ElastiCache pecahan

- ElastiCache kluster
- ElastiCache replikasi
- ElastiCache titik akhir
- ElastiCache kelompok parameter
- ElastiCache keamanan
- ElastiCache kelompok subnet
- ElastiCache cadangan
- ElastiCache acara

ElastiCache simpul

Node adalah blok bangunan terkecil dari sebuah ElastiCache deployment. Simpul dapat berdiri sendiri dari atau terkait dengan simpul lainnya.

Simpul adalah potongan RAM berukuran tetap yang terhubung ke jaringan secara aman. Setiap simpul menjalankan sebuah instans mesin dan versi yang dipilih pada saat Anda membuat klaster Anda. Jika diperlukan, Anda dapat menaikkan atau menurunkan skala simpul dalam klaster ke jenis instans yang berbeda. Untuk informasi selengkapnya, lihat Penskalaan ElastiCache.

Setiap simpul dalam klaster adalah jenis instans yang sama dan menjalankan mesin cache yang sama. Setiap simpul cache mempunyai nama dan port Layanan Nama Domain (DNS) sendiri. Beberapa jenis simpul cache didukung, masing-masing dengan jumlah yang bervariasi dari memori yang terkait. Untuk daftar jenis instans simpul yang didukung, lihat Jenis simpul yang didukung.

Anda dapat membeli node pay-as-you-go berdasarkan, di mana Anda hanya membayar untuk penggunaan node. Anda juga dapat membeli simpul terpesan dengan tarif per jam yang jauh lebih murah. Jika tingkat penggunaan Anda tinggi, pembelian simpul direservasi dapat menghemat uang Anda. Misalkan klaster Anda hampir setiap saat digunakan, dan Anda terkadang menambahkan simpul untuk menangani lonjakan penggunaan. Dalam kasus ini, Anda dapat membeli sejumlah simpul terpesan untuk bekerja pada hampir semua kesempatan. Anda kemudian dapat membeli pay-as-you-go node untuk saat-saat Anda sesekali perlu menambahkan node. Untuk informasi lain tentang simpul direservasi, lihat Simpul terpesan.

Untuk informasi lain tentang simpul, lihat Mengelola node di ElastiCache.

ElastiCache pecahan

Shard Valkey atau Redis OSS (disebut grup node di API dan CLI) adalah pengelompokan dari satu hingga enam node terkait. Cluster Valkey atau Redis OSS dengan mode cluster diaktifkan selalu memiliki setidaknya satu pecahan.

Sharding adalah metode partisi database yang memisahkan database besar menjadi bagian yang lebih kecil, lebih cepat, dan lebih mudah dikelola yang disebut pecahan data. Hal ini dapat meningkatkan efisiensi database dengan mendistribusikan operasi di beberapa bagian terpisah. Menggunakan pecahan dapat menawarkan banyak manfaat termasuk peningkatan kinerja, skalabilitas, dan efisiensi biaya.

Cluster Valkey dan Redis OSS dengan mode cluster diaktifkan dapat memiliki hingga 500 pecahan, dengan data Anda dipartisi di seluruh pecahan. Batas node atau shard dapat ditingkatkan hingga maksimum 500 per cluster jika versi mesin Valkey atau Redis OSS adalah 5.0.6 atau lebih tinggi. Sebagai contoh, Anda dapat memilih untuk mengonfigurasi sebuah klaster dengan 500 simpul yang berkisar antara 83 serpihan (satu primer dan 5 replika per serpihan) dan 500 serpihan (satu primer dan tanpa replika). Pastikan alamat IP yang tersedia mencukupi untuk mengakomodasi peningkatan tersebut. Kesalahan umumnya termasuk subnet dalam grup subnet memiliki rentang CIDR yang terlalu kecil atau subnet dibagikan dan banyak digunakan oleh klaster lainnya. Untuk informasi selengkapnya, lihat Membuat grup subnet. Untuk versi di bawah 5.0.6, batasnya adalah 250 per klaster.

Untuk meminta penambahan batas, lihat Batas Layanan AWS dan pilih jenis batas Simpul per klaster per jenis instans.

Serpihan beberapa simpul mengimplementasikan replikasi dengan memiliki satu simpul primer baca/tulis dan 1–5 simpul replika. Untuk informasi selengkapnya, lihat Ketersediaan tinggi menggunakan grup replikasi.

Untuk informasi selengkapnya tentang serpihan, lihat Bekerja dengan pecahan di ElastiCache.

ElastiCache kluster

Cluster adalah pengelompokan logis dari satu atau lebih node. Data dipartisi di seluruh node dalam cluster Memcached, dan melintasi pecahan dalam cluster Valkey atau Redis OSS yang mengaktifkan mode cluster.

Banyak ElastiCache operasi ditargetkan pada cluster:

- Membuat klaster
- Mengubah klaster
- Mengambil snapshot klaster (semua versi Redis)
- Menghapus klaster
- Melihat elemen di klaster
- Menambahkan atau menghapus tag alokasi biaya ke dan dari klaster

Untuk informasi selengkapnya, lihat topik terkait berikut:

- Mengelola cluster di ElastiCache dan Mengelola node di ElastiCache

  Informasi tentang klaster, simpul, dan operasi terkait.
- AWS batas layanan: Amazon ElastiCache

  Informasi tentang ElastiCache batas, seperti jumlah maksimum node atau cluster. Untuk melampaui batas tertentu, Anda dapat membuat permintaan menggunakan formulir permintaan node ElastiCache cache Amazon.
- Mitigasi Kegagalan

  Informasi tentang meningkatkan toleransi kesalahan cluster Anda dan grup replikasi Valkey atau Redis OSS.

Konfigurasi klaster umum

Berikut adalah konfigurasi klaster umum.

Cluster Valkey atau Redis OSS

Cluster Valkey atau Redis OSS dengan mode cluster dinonaktifkan selalu berisi hanya satu pecahan (di API dan CLI, satu grup node). Pecahan Valkey atau Redis OSS berisi satu hingga enam node. Jika terdapat lebih dari satu simpul dalam sebuah serpihan, serpihan tersebut mendukung replikasi. Dalam hal ini, satu node adalah node read/write utama dan yang lainnya adalah node replika read-only.

Untuk meningkatkan toleransi kesalahan, kami sarankan memiliki setidaknya dua node di cluster Valkey atau Redis OSS dan mengaktifkan Multi-AZ. Untuk informasi selengkapnya, lihat Mitigasi Kegagalan.

Saat permintaan klaster Valkey atau Redis OSS Anda berubah, Anda dapat meningkatkan atau menurunkan skala. Untuk melakukan ini, pindahkan cluster Anda ke jenis instance node yang berbeda. Jika aplikasi Anda dibaca intensif, sebaiknya tambahkan replika hanya-baca ke cluster. Dengan melakukan ini, Anda dapat menyebarkan pembacaan ke jumlah simpul yang lebih tepat.

Anda juga dapat menggunakan tingkatan data. Data yang lebih sering diakses disimpan dalam memori dan data yang lebih jarang diakses disimpan di disk. Keuntungan menggunakan tingkatan data adalah mengurangi kebutuhan memori. Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.

ElastiCache mendukung perubahan tipe node Valkey atau Redis OSS cluster ke tipe node yang lebih besar secara dinamis. Untuk informasi tentang kenaikan atau penurunan skala, lihat Penskalaan cluster simpul tunggal untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) atau Menskalakan node replika untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan).

Konfigurasi cluster khas untuk Memcached

Memcached mendukung hingga 300 node per pelanggan untuk setiap AWS Wilayah dengan setiap cluster memiliki 1-60 node. Data dipartisi di seluruh simpul di klaster Memcached.

Saat Anda menjalankan mesin Memcached, cluster dapat terdiri dari 1-60 node. Basis data dipartisi di seluruh simpul. Aplikasi Anda membaca dan menulis ke titik akhir setiap simpul. Untuk informasi selengkapnya, lihat Penemuan Otomatis.

Untuk meningkatkan toleransi kesalahan, temukan node Memcached Anda di berbagai Availability Zones (AZs) dalam Region klaster. AWS Dengan cara ini, kegagalan dalam satu AZ akan berdampak minimal pada keseluruhan klaster dan aplikasi. Untuk informasi selengkapnya, lihat Mitigasi Kegagalan.

Seiring perubahan permintaan atas klaster Memcached, Anda dapat menskalakan ke luar atau ke dalam dengan menambahkan atau menghapus simpul. Tindakan ini akan mempartisi ulang data Anda ke sejumlah simpul baru tersebut. Saat mempartisi data Anda, sebaiknya gunakan hashing yang konsisten. Untuk informasi selengkapnya tentang hashing yang konsisten, lihat Mengkonfigurasi ElastiCache klien Anda untuk penyeimbangan beban yang efisien (Memcached).

ElastiCache replikasi

Untuk Valkey dan Redis OSS, replikasi diimplementasikan dengan mengelompokkan dari dua hingga enam node dalam pecahan (dalam API dan CLI, disebut grup node). Salah satu simpul ini adalah

simpul primer baca/tulis. Semua simpul lain adalah simpul replika baca-saja. Replikasi hanya tersedia ElastiCache untuk Valkey dan Redis OSS, dan bukan untuk Memcached. ElastiCache

Setiap replika baca berisi salinan data dari simpul primer. Simpul replika menggunakan mekanisme replikasi asinkron untuk tetap sinkron dengan simpul primer. Aplikasi dapat membaca dari simpul mana pun dalam klaster, tetapi hanya dapat menulis ke simpul primer. Replika baca meningkatkan skalabilitas dengan menyebarkan proses baca ke beberapa titik akhir. Replika baca juga meningkatkan toleransi kesalahan dengan mempertahankan beberapa salinan data. Menemukan replika baca di beberapa Zona Ketersediaan dapat meningkatkan toleransi kesalahan lebih baik lagi. Untuk informasi selengkapnya tentang toleransi kesalahan, lihat Mitigasi Kegagalan.

Cluster Valkey atau Redis OSS mendukung satu pecahan (dalam API dan CLI, disebut grup node).

Replikasi dari perspektif API dan CLI menggunakan terminologi yang berbeda untuk mempertahankan kompatibilitas dengan versi sebelumnya, tetapi hasilnya sama. Tabel berikut menunjukkan istilah API dan CLI untuk menerapkan replikasi.

Membandingkan Replikasi: Valkey atau Redis OSS (mode cluster dinonaktifkan) dan Valkey atau Redis OSS (mode cluster diaktifkan) -> Valkey atau Redis OSS cluster dengan mode cluster diaktifkan vs Valkey atau Redis OSS cluster dengan mode cluster dinonaktifkan

Dalam tabel berikut, Anda dapat menemukan perbandingan fitur Valkey atau Redis OSS (mode cluster dinonaktifkan) dan grup replikasi Valkey atau Redis OSS (mode cluster diaktifkan).

	Valkey atau Redis OSS cluster dengan mode cluster dinonaktifkan	Valkey atau Redis OSS cluster dengan mode cluster diaktifkan
Serpihan (grup simpul)	1	1–500
Replika untuk setiap serpihan (grup simpul)	0–5	0–5
Pembuatan partisi data	Tidak	Ya
Tambah/Hapus replika	Ya	Ya
Tambah/Hapus grup simpul	Tidak	Ya
Mendukung kenaikan skala	Ya	Ya

	Valkey atau Redis OSS cluster dengan mode cluster dinonaktifkan	Valkey atau Redis OSS cluster dengan mode cluster diaktifkan
Mendukung peningkatan mesin	Ya	Ya
Naikkan replika menjadi primer	Ya	Otomatis
Multi-AZ	Opsional	Wajib
Cadangkan/Pulihkan	Ya	Ya
Catatan:		
Jika primer tidak memiliki replika dan gagal, semua data primer akan hilang.		
Anda dapat menggunakan backup dan restore untuk bermigrasi ke Valkey atau Redis OSS (mode cluster diaktifkan).		
Anda dapat menggunakan backup dan restore untuk mengubah ukuran cluster Valkey atau Redis OSS (mode cluster enabled) Anda.		

Semua serpihan (dalam API dan CLI, grup simpul) dan simpul harus berada di Wilayah AWS yang sama. Namun, Anda dapat menyediakan node individual di beberapa Availability Zone dalam AWS Region tersebut.

Replika baca melindungi dari potensi kehilangan data karena data Anda direplikasi pada dua simpul atau lebih—primer dan satu atau beberapa replika baca. Untuk meningkatkan keandalan dan mempercepat pemulihan, kami sarankan Anda membuat satu atau beberapa replika baca di Zona Ketersediaan yang berbeda.

Anda juga dapat memanfaatkan penyimpanan data Global. Dengan menggunakan fitur Global Datastore for Redis OSS, Anda dapat bekerja dengan replikasi yang dikelola sepenuhnya, cepat, andal, dan aman di seluruh Wilayah. AWS Dengan menggunakan fitur ini, Anda dapat membuat klaster replika baca lintas wilayah ElastiCache untuk mengaktifkan pembacaan latensi rendah dan pemulihan bencana di seluruh Wilayah. AWS Untuk informasi selengkapnya, lihat Replikasi lintas AWS Wilayah menggunakan datastores global.

Replikasi: Batas dan pengecualian

- Multi-AZ tidak didukung pada jenis simpul T1.

ElastiCache titik akhir

Endpoint adalah alamat unik yang digunakan aplikasi Anda untuk terhubung ke ElastiCache node atau cluster.

Titik akhir node tunggal untuk Valkey atau Redis OSS dengan mode cluster dinonaktifkan

Titik akhir untuk satu node Valkey atau Redis OSS cluster digunakan untuk terhubung ke cluster untuk membaca dan menulis.

Titik akhir multi-node untuk Valkey atau Redis OSS dengan mode cluster dinonaktifkan

Beberapa node Valkey atau Redis OSS cluster dengan mode cluster dinonaktifkan memiliki dua jenis endpoint. Titik akhir primer selalu tersambung ke simpul primer dalam klaster, bahkan jika simpul tertentu dalam peran primer berubah. Gunakan titik akhir primer untuk semua penulisan ke klaster.

Gunakan Titik Akhir Pembaca untuk membagi koneksi masuk ke titik akhir secara merata di antara semua replika baca. Gunakan Endpoint Node individual untuk operasi baca (Dalam API/CLI ini disebut sebagai Read Endpoints).

Titik akhir Valkey atau Redis OSS (Mode Cluster Diaktifkan)

Cluster Valkey atau Redis OSS dengan mode cluster diaktifkan memiliki titik akhir konfigurasi tunggal. Dengan menyambung ke titik akhir konfigurasi, aplikasi Anda mampu menemukan titik akhir primer dan baca untuk setiap serpihan di klaster.

Untuk informasi selengkapnya, lihat Menemukan titik akhir koneksi di ElastiCache.

ElastiCache untuk titik akhir Memcached

Setiap simpul di klaster Memcached memiliki titik akhirnya sendiri. Klaster juga memiliki titik akhir yang disebut titik akhir konfigurasi. Jika Anda mengaktifkan Penemuan Otomatis dan terhubung ke titik akhir konfigurasi, aplikasi Anda secara otomatis mengetahui setiap titik akhir simpul, bahkan setelah menambahkan atau menghapus simpul dari klaster tersebut. Untuk informasi selengkapnya, lihat Penemuan Otomatis.

Untuk informasi selengkapnya, lihat Menemukan titik akhir koneksi di ElastiCache.

ElastiCache kelompok parameter

Grup parameter cache adalah cara mudah untuk mengelola pengaturan runtime untuk perangkat lunak mesin yang didukung. Parameter digunakan untuk mengontrol penggunaan memori, kebijakan pengosongan, ukuran item, dan lainnya. Grup ElastiCache parameter adalah kumpulan bernama parameter khusus mesin yang dapat Anda terapkan ke cluster. Dengan melakukan ini, Anda memastikan bahwa semua simpul dalam klaster dikonfigurasi dengan cara yang pasti sama.

Untuk daftar parameter yang didukung, nilai defaultnya, dan parameter mana yang dapat dimodifikasi, lihat DescribeEngineDefaultParameters (CLI: describe-engine-default-parameters).

Untuk informasi lebih rinci tentang grup ElastiCache parameter, lihatMengkonfigurasi parameter mesin menggunakan grup ElastiCache parameter.

ElastiCache keamanan

Untuk keamanan yang ditingkatkan, akses ElastiCache node dibatasi untuk aplikasi yang berjalan di EC2 instans Amazon yang Anda izinkan. Anda dapat mengontrol EC2 instans Amazon yang dapat mengakses klaster Anda menggunakan grup keamanan.

Secara default, semua ElastiCache cluster baru diluncurkan di lingkungan Amazon Virtual Private Cloud (Amazon VPC). Anda dapat menggunakan grup subnet untuk memberikan akses klaster dari EC2 instans Amazon yang berjalan pada subnet tertentu.

Selain membatasi akses node, ElastiCache mendukung TLS dan enkripsi di tempat untuk node yang menjalankan versi tertentu. ElastiCache Untuk informasi selengkapnya, lihat berikut ini:

- Keamanan data di Amazon ElastiCache
- Mengautentikasi dengan perintah Valkey dan Redis OSS AUTH

ElastiCache kelompok subnet

Grup subnet adalah kumpulan subnet (biasanya privat) yang dapat Anda tetapkan untuk klaster Anda yang berjalan di lingkungan Amazon VPC.

Jika Anda membuat klaster di Amazon VPC, Anda harus menentukan grup subnet cache. ElastiCache menggunakan grup subnet cache tersebut untuk memilih subnet dan alamat IP di dalam subnet itu untuk dikaitkan dengan simpul cache Anda.

Untuk informasi selengkapnya tentang penggunaan grup subnet cache di lingkungan Amazon VPC, lihat hal berikut.

- Amazon VPCs dan ElastiCache keamanan
- Langkah 3. Otorisasi akses ke cluster
- Subnet dan grup subnet

ElastiCache cadangan

Cadangan adalah point-in-time salinan cluster Valkey atau Redis OSS atau cache tanpa server, atau cache tanpa server Memcached. Cadangan dapat digunakan untuk memulihkan klaster yang ada atau menyemai klaster baru. Cadangan terdiri dari semua data dalam klaster ditambah beberapa metadata.

Bergantung pada versi Valkey atau Redis OSS yang berjalan di cluster Anda, proses pencadangan memerlukan jumlah memori cadangan yang berbeda agar berhasil. Untuk informasi selengkapnya, lihat berikut ini:

- Melakukan snapshot dan pemulihan
- Cara penerapan sinkronisasi dan pencadangan
- Dampak performa pencadangan klaster yang dirancang sendiri
- Memastikan Anda memiliki cukup memori untuk membuat snapshot Valkey atau Redis OSS

ElastiCache acara

Saat peristiwa penting terjadi di cluster cache, ElastiCache kirimkan pemberitahuan ke topik Amazon SNS tertentu. Peristiwa penting dapat mencakup hal seperti kegagalan atau keberhasilan penambahan simpul, perubahan grup keamanan, dan lainnya. Dengan memantau peristiwa penting, Anda dapat mengetahui status klaster terbaru Anda dan dalam banyak kasus dapat mengambil tindakan korektif.

Untuk informasi lebih lanjut tentang ElastiCache acara, lihatPemantauan acara Amazon SNS ElastiCache .

ElastiCache terminologi

Pada Oktober 2016, Amazon ElastiCache meluncurkan dukungan untuk Redis OSS 3.2. Pada saat itu, kami menambahkan dukungan untuk mempartisi data Anda hingga 500 pecahan (disebut grup node di ElastiCache API dan). AWS CLI Untuk menjaga kompatibilitas dengan versi sebelumnya, kami memperluas operasi API versi 2015-02-02 untuk menyertakan fungsionalitas Redis OSS yang baru.

Pada saat yang sama, kami mulai menggunakan terminologi di ElastiCache konsol yang digunakan dalam fungsi baru ini dan umum di seluruh industri. Perubahan ini berarti bahwa pada beberapa tahap, terminologi yang digunakan di API dan CLI mungkin berbeda dari terminologi yang digunakan di konsol. Daftar berikut mengidentifikasi istilah yang mungkin berbeda antara API, CLI, dan konsol.

Klaster cache atau simpul vs. simpul

Ada one-to-one hubungan antara node dan cluster cache ketika tidak ada node replika. Dengan demikian, ElastiCache konsol sering menggunakan istilah secara bergantian. Konsol sekarang menggunakan istilah simpul secara menyeluruh. Satu-satunya pengecualian adalah tombol Buat Klaster, yang memulai proses untuk membuat klaster dengan atau tanpa simpul replika.

ElastiCache API dan AWS CLI terus menggunakan istilah seperti yang mereka miliki di masa lalu.

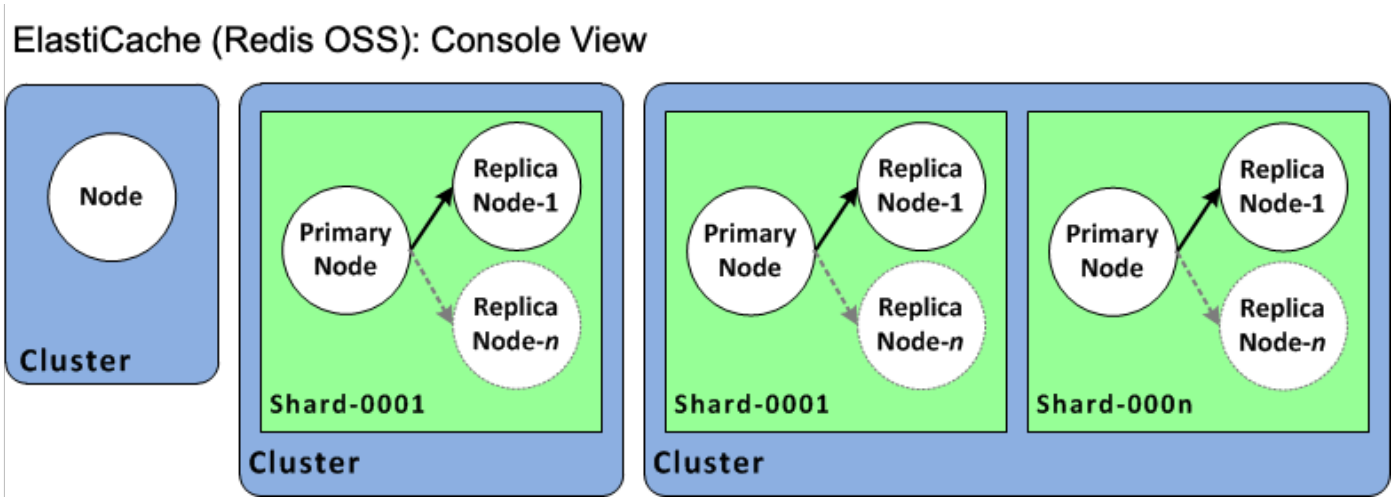
Kelompok replikasi Cluster vs Valkey atau Redis OSS

Konsol sekarang menggunakan istilah cluster untuk semua ElastiCache untuk cluster Redis OSS. Konsol menggunakan istilah klaster dalam semua kondisi berikut:

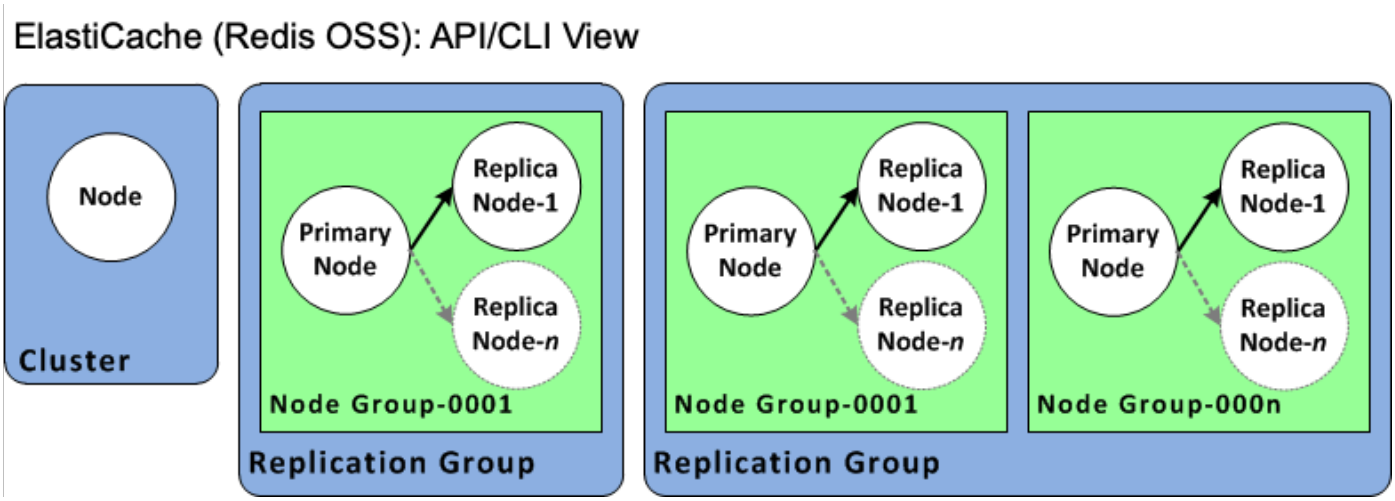
- Ketika cluster adalah node tunggal Valkey atau Redis OSS cluster.
- Ketika cluster adalah cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) yang mendukung replikasi dalam pecahan tunggal (dalam API dan CLI, disebut grup node).
- Ketika cluster adalah cluster Valkey atau Redis OSS (mode cluster enabled) yang mendukung replikasi dalam 1-90 pecahan atau hingga 500 dengan permintaan peningkatan batas. Untuk meminta penambahan batas, lihat Batas layanan AWS dan pilih jenis batas Simpul per klaster per jenis instans.

Untuk informasi lebih lanjut tentang grup replikasi Valkey atau Redis OSS, lihat. Ketersediaan tinggi menggunakan grup replikasi

Diagram berikut menggambarkan berbagai topologi ElastiCache untuk cluster Redis OSS dari perspektif konsol.



ElastiCache API dan AWS CLI operasi masih membedakan node tunggal ElastiCache untuk kluster Redis OSS dari grup replikasi Valkey atau Redis OSS multi-node. Diagram berikut menggambarkan berbagai ElastiCache topologi Redis OSS dari API dan perspektif. ElastiCache AWS CLI



Grup Replikasi Valkey atau Redis OSS vs. datastore global

Datastore global adalah kumpulan dari satu atau lebih cluster yang mereplikasi satu sama lain di seluruh Wilayah, sedangkan grup replikasi Valkey atau Redis OSS mereplikasi data di seluruh cluster yang diaktifkan mode cluster dengan beberapa pecahan. Penyimpanan data global mencakup klaster berikut:

- Klaster primer (aktif) – Klaster primer menerima operasi tulis yang direplikasi ke semua klaster dalam penyimpanan data global. Klaster primer juga menerima permintaan baca.

- Klaster sekunder (pasif) – Klaster sekunder hanya menerima permintaan baca dan mereplikasi pembaruan data dari klaster primer. Cluster sekunder harus berada di AWS Wilayah yang berbeda dari cluster primer.

Untuk informasi tentang penyimpanan data global, lihat Replikasi lintas AWS Wilayah menggunakan datastores global.

Tutorial: Cara mendesain cluster Anda sendiri

Berikut adalah cara merancang cluster Anda sendiri untuk Valkey, Memcached dan Redis OSS.

Topik

- Merancang cluster ElastiCache (Valkey) Anda sendiri
- Merancang cluster ElastiCache Redis OSS Anda sendiri

Merancang cluster ElastiCache (Valkey) Anda sendiri

Berikut ini adalah tindakan satu kali yang harus Anda ambil untuk mulai merancang cluster ElastiCache (Valkey) Anda.

Langkah 1: Membuat grup subnet

Sebelum Anda membuat cluster ElastiCache (Valkey), Anda terlebih dahulu membuat grup subnet. Grup subnet cache adalah kumpulan subnet yang dapat ditetapkan untuk klaster cache Anda dalam VPC. Saat meluncurkan klaster cache di VPC, Anda harus memilih grup subnet cache. Kemudian ElastiCache menggunakan kelompok subnet cache itu untuk menetapkan alamat IP dalam subnet itu ke setiap node cache di cluster.

Saat Anda membuat grup subnet baru, perhatikan jumlah alamat IP yang tersedia. Jika subnet memiliki sangat sedikit alamat IP yang bebas, Anda akan dibatasi dalam hal jumlah simpul yang dapat ditambahkan ke klaster. Untuk mengatasi masalah ini, Anda dapat menetapkan satu atau beberapa subnet ke grup subnet sehingga Anda memiliki jumlah alamat IP yang cukup dalam Zona Ketersediaan dari klaster Anda. Setelah itu, Anda dapat menambahkan lebih banyak simpul ke klaster Anda.

Untuk informasi lebih lanjut tentang pengaturan, ElastiCache lihatMenyiapkan ElastiCache.

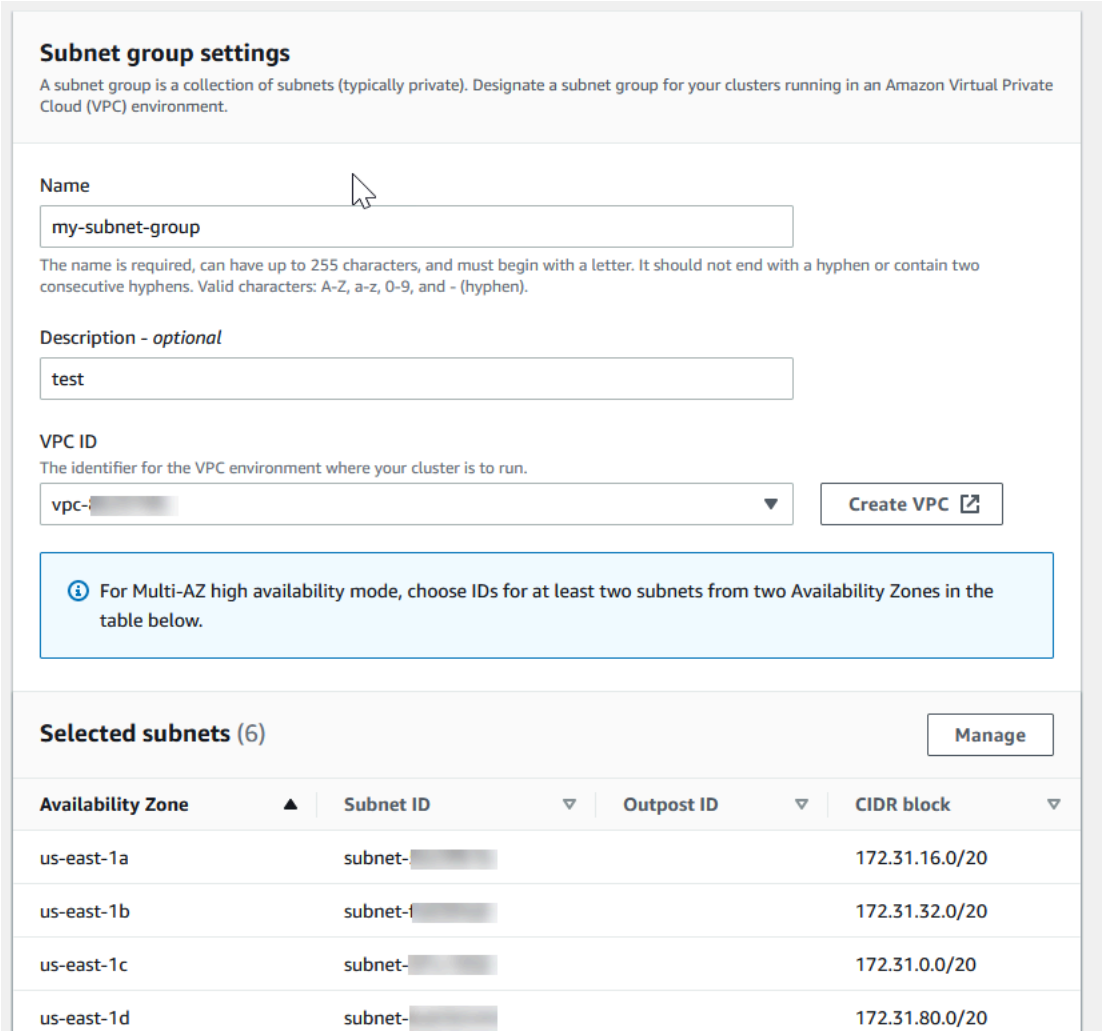
Prosedur berikut menunjukkan cara membuat grup subnet yang disebut `mysubnetgroup` (konsol), dan AWS CLI.

Membuat grup subnet (Konsol)

Prosedur berikut menunjukkan cara membuat grup subnet (konsol).

Untuk membuat grup subnet (Konsol)

1. Masuk ke Konsol AWS Manajemen, dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di daftar navigasi, pilih Grup subnet.
3. Pilih Buat Grup Subnet.
4. Pada wizard Buat Grup Subnet, lakukan hal berikut. Jika semua pengaturan sudah sesuai keinginan Anda, pilih Ya, Buat.
    a. Pada kotak Nama, ketik nama grup subnet Anda.
    b. Di kotak Deskripsi, ketik deskripsi untuk grup subnet Anda.
    c. Pada kotak ID VPC, pilih Amazon VPC yang Anda buat.
    d. Di daftar Availability Zone dan Subnet ID, pilih Availability Zone atau Menggunakan zona lokal dengan ElastiCache dan ID subnet pribadi Anda, lalu pilih Tambah.



5. Pada pesan konfirmasi yang muncul, pilih Tutup.

Grup subnet baru Anda muncul di daftar Grup Subnet konsol. ElastiCache Di bagian bawah jendela, Anda dapat memilih grup subnet untuk melihat detail, misalnya semua subnet yang terkait dengan grup ini.

Membuat grup subnet (AWS CLI)

Pada prompt perintah, gunakan perintah `create-cache-subnet-group` untuk membuat grup subnet.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-cache-subnet-group \
    --cache-subnet-group-name mysubnetgroup \
    --cache-subnet-group-description "Testing" \
```

```
    --subnet-ids subnet-53df9c3a
```

Untuk Windows:

```
aws elasticache create-cache-subnet-group ^
    --cache-subnet-group-name mysubnetgroup ^
    --cache-subnet-group-description "Testing" ^
    --subnet-ids subnet-53df9c3a
```

Perintah ini seharusnya menghasilkan output yang serupa dengan yang berikut:

```
{
    "CacheSubnetGroup": {
        "VpcId": "vpc-37c3cd17",
        "CacheSubnetGroupDescription": "Testing",
        "Subnets": [
            {
                "SubnetIdentifier": "subnet-53df9c3a",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2a"
                }
            }
        ],
        "CacheSubnetGroupName": "mysubnetgroup"
    }
}
```

Untuk informasi lebih lanjut, lihat AWS CLI topiknyacreate-cache-subnet-group.

Langkah 2: Buat klaster

Sebelum membuat klaster untuk tujuan produksi, Anda tentu perlu mempertimbangkan pengaturan konfigurasi klaster untuk memenuhi kebutuhan bisnis Anda. Masalah terkait itu dibahas di bagian Mempersiapkan cluster di ElastiCache. Untuk tujuan latihan Memulai ini, Anda akan membuat klaster dengan mode klaster dinonaktifkan dan Anda dapat menggunakan nilai konfigurasi default jika sesuai.

Klaster yang Anda buat akan berjalan langsung, dan tidak berjalan di sandbox. Anda akan dikenakan biaya ElastiCache penggunaan standar untuk instans sampai Anda menghapusnya. Jumlah biayanya cukup kecil (biasanya kurang dari satu dolar) jika Anda menyelesaikan latihan yang dijelaskan di sini

dalam satu sesi dan menghapus klaster itu ketika Anda sudah selesai. Untuk informasi selengkapnya tentang tarif ElastiCache penggunaan, lihat Amazon ElastiCache.

Klaster Anda diluncurkan dalam cloud privat virtual (VPC) berdasarkan layanan Amazon VPC.

Membuat cluster Valkey (mode cluster dinonaktifkan) (Konsol)

Untuk membuat cluster Valkey (mode cluster dinonaktifkan) menggunakan konsol ElastiCache

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Dari daftar di sudut kanan atas, pilih AWS Wilayah tempat Anda ingin meluncurkan cluster ini.
3. Pilih Mulai di panel navigasi.
4. Pilih Buat VPC dan ikuti langkah-langkah yang telah dijelaskan dalam Membuat Cloud Privat Virtual (VPC).
5. Pada halaman ElastiCache dashboard, pilih Valkey cache atau Redis OSS cache dan kemudian pilih Create Valkey cache or Create Redis OSS cache.
6. Di bagian Pengaturan klaster, lakukan hal berikut:

    a. Pilih Konfigurasi dan buat klaster baru.
    b. Untuk Mode klaster, pilih Dinonaktifkan.
    c. Untuk Info klaster masukkan nilai untuk Nama.
    d. (Opsional) Masukkan nilai untuk Deskripsi.
7. Di bagian Lokasi:

    AWS Cloud

    1. Untuk AWS Cloud, sebaiknya terima pengaturan default untuk Multi-AZ dan Failover otomatis. Untuk informasi selengkapnya, lihat Meminimalkan waktu henti ElastiCache untuk Redis OSS dengan Multi-AZ.
    2. Pada Pengaturan klaster

        a. Untuk Versi mesin, pilih versi yang tersedia.
        b. Untuk Port, gunakan port default, 6379. Jika Anda memiliki alasan untuk menggunakan port lain, masukkan nomor port tersebut.
        c. Untuk Grup parameter, pilih grup parameter atau buat yang baru. Grup parameter mengontrol parameter runtime dari klaster Anda. Untuk informasi selengkapnya

tentang grup parameter, lihat Parameter Valkey dan Redis OSS dan Membuat grup ElastiCache parameter.

> **Note**
>
> Saat Anda memilih grup parameter untuk menetapkan nilai konfigurasi mesin, grup parameter tersebut diterapkan ke semua klaster di penyimpanan data global. Pada halaman Grup Parameter, atribut Global ya/tidak menunjukkan apakah grup parameter adalah bagian dari penyimpanan data global.

d. Untuk Jenis simpul, pilih panah bawah ( ).
Pada kotak dialog Ubah jenis simpul, pilih nilai untuk Keluarga instans untuk jenis simpul yang Anda inginkan. Kemudian pilih jenis simpul yang ingin Anda gunakan untuk klaster ini, lalu pilih Simpan.

Untuk informasi selengkapnya, lihat Memilih ukuran simpul Anda.

Jika Anda memilih jenis simpul r6gd, tingkatan data akan diaktifkan secara otomatis. Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.

e. Untuk Jumlah replika, pilih jumlah replika baca yang Anda inginkan. Jika Anda mengaktifkan Multi-AZ, jumlahnya harus antara 1-5.

3. Pada Konektivitas

a. Untuk Jenis jaringan, pilih versi IP yang akan didukung oleh klaster ini.

b. Untuk grup Subnet, pilih subnet yang ingin Anda terapkan ke cluster ini. ElastiCache menggunakan grup subnet itu untuk memilih subnet dan alamat IP dalam subnet itu untuk dikaitkan dengan node Anda. ElastiCache cluster memerlukan subnet dual-stack dengan keduanya IPv4 dan IPv6 alamat yang ditetapkan untuk beroperasi dalam mode dual-stack dan subnet -only untuk beroperasi sebagai IPv6 -only. IPv6

Saat membuat grup subnet baru, masukkan ID VPC yang menaungi grup subnet tersebut.

Untuk informasi selengkapnya, lihat:

- Memilih jenis jaringan di ElastiCache.

- Membuat subnet di VPC Anda.

  Jika Anda adalah Menggunakan zona lokal dengan ElastiCache, Anda harus membuat atau memilih subnet yang berada di zona lokal.

  Untuk informasi selengkapnya, lihat Subnet dan grup subnet.

4. Untuk Penempatan zona ketersediaan, Anda memiliki dua opsi:

   - Tidak ada preferensi — ElastiCache memilih Availability Zone.
   - Tentukan zona ketersediaan – Anda menentukan Zona Ketersediaan untuk setiap klaster.

     Jika Anda memilih untuk menentukan Zona Ketersediaan, untuk setiap klaster di setiap serpihan, pilih Zona Ketersediaan dari daftar.

   Untuk informasi selengkapnya, lihat Memilih wilayah dan zona ketersediaan untuk ElastiCache.

5. Pilih Berikutnya
6. Di bawah pengaturan Advanced Valkey atau Redis OSS

   - Untuk Keamanan:

     i. Untuk mengenkripsi data Anda, Anda memiliki opsi berikut:

     - Enkripsi diam – Mengaktifkan enkripsi pada data yang disimpan di disk. Untuk informasi selengkapnya, lihat Enkripsi Diam.

       > **Note**
       >
       > Anda memiliki opsi untuk menyediakan kunci enkripsi yang berbeda dengan memilih kunci AWS KMS yang Dikelola Pelanggan dan memilih kunci. Untuk informasi selengkapnya, lihat Menggunakan kunci yang dikelola pelanggan dari AWS KMS.

     - Enkripsi bergerak – Mengaktifkan enkripsi data selama pengiriman. Untuk informasi selengkapnya, lihat Enkripsi bergerak. Untuk Valkey dan untuk Redis OSS 6.0 dan di atasnya, jika Anda mengaktifkan Enkripsi dalam

perjalanan, Anda akan diminta untuk menentukan salah satu opsi Kontrol Akses berikut:

- Tanpa Kontrol Akses – Ini adalah pengaturan default. Opsi ini menunjukkan bahwa tidak ada batasan akses pengguna ke klaster.
- Daftar Kontrol Akses Grup Pengguna – Pilih grup pengguna dengan kumpulan pengguna tertentu yang dapat mengakses klaster. Untuk informasi selengkapnya, lihat Mengelola Grup Pengguna dengan Konsol dan CLI.
- AUTH Default User - Mekanisme otentikasi untuk server Redis OSS. Untuk informasi lebih lanjut, lihat AUTH.

- AUTH — Mekanisme otentikasi untuk server Redis OSS. Untuk informasi lebih lanjut, lihat AUTH.

> **Note**
>
> Untuk Valkey dan untuk versi Redis OSS antara 3.2.6 dan seterusnya, tidak termasuk versi 3.2.10, Redis OSS AUTH adalah satu-satunya pilihan.

ii. Untuk Grup keamanan, pilih grup keamanan yang Anda inginkan untuk klaster ini. Grup keamanan bertindak sebagai firewall untuk mengontrol akses jaringan ke klaster Anda. Anda dapat menggunakan grup keamanan default untuk VPC Anda atau membuat yang baru.

Untuk informasi selengkapnya tentang grup keamanan, lihat Grup Keamanan untuk VPC Anda dalam Panduan Pengguna Amazon VPC.

7. Untuk pencadangan otomatis terjadwal secara berkala, pilih Aktifkan pencadangan otomatis, lalu masukkan jumlah hari yang diinginkan untuk mempertahankan cadangan otomatis sebelum dihapus secara otomatis. Jika Anda tidak ingin melakukan pencadangan otomatis terjadwal secara berkala, hapus kotak centang Aktifkan pencadangan otomatis. Apa pun pilihannya, Anda dapat membuat pencadangan secara manual kapan saja.

   Untuk informasi lebih lanjut tentang cadangan dan pemulihan Redis OSS, lihat. Melakukan snapshot dan pemulihan

8. (Opsional) Tentukan periode pemeliharaan. Jendela pemeliharaan adalah waktu yang biasanya satu jam setiap minggu saat ElastiCache menjadwalkan pemeliharaan sistem untuk klaster Anda. Anda dapat mengizinkan ElastiCache untuk memilih hari dan waktu untuk jendela pemeliharaan Anda (Tidak ada preferensi), atau Anda dapat memilih hari, waktu, dan durasi sendiri (Tentukan jendela pemeliharaan). Jika Anda memilih Tentukan periode pemeliharaan dari daftar, pilih Hari mulai, Waktu mulai, dan Durasi (dalam jam) untuk periode pemeliharaan. Semua waktu menggunakan zona waktu UTC.

   Untuk informasi selengkapnya, lihat Mengelola pemeliharaan ElastiCache cluster.

9. (Opsional) Untuk Log:

   - Di bagian Format log, pilih Teks atau JSON.
   - Di bawah Jenis Tujuan, pilih CloudWatch Log atau Kinesis Firehose.
   - Di bawah Tujuan log, pilih Buat baru dan masukkan nama grup CloudWatch log Log atau nama aliran Firehose Anda, atau pilih Pilih yang ada, lalu pilih nama grup CloudWatch log Log atau nama aliran Firehose Anda,

10. Untuk Tag, untuk membantu mengelola cluster dan ElastiCache sumber daya lainnya, Anda dapat menetapkan metadata Anda sendiri ke setiap sumber daya dalam bentuk tag. Untuk informasi selengkapnya, lihat Menandai sumber daya Anda ElastiCache .
11. Pilih Berikutnya.
12. Tinjau semua entri dan pilihan Anda, lalu lakukan koreksi yang diperlukan. Saat Anda siap, pilih Buat.

On premises

1. Untuk On-premise, sebaiknya Anda membiarkan Failover otomatis tetap aktif. Untuk informasi selengkapnya, lihat Meminimalkan waktu henti ElastiCache untuk Redis OSS dengan Multi-AZ
2. Untuk menyelesaikan pembuatan klaster, ikuti langkah-langkah di Menggunakan Outposts.

Segera setelah status klaster Anda tersedia, Anda dapat memberikan Amazon EC2 akses ke sana, menghubungkannya, dan mulai menggunakannya. Untuk informasi selengkapnya, lihat Langkah 3. Otorisasi akses ke cluster dan Langkah 4. Connect ke node cluster.

> **Important**
>
> Setelah klaster Anda tersedia, Anda akan ditagih untuk setiap jam atau durasi saat klaster aktif, meskipun Anda tidak sedang aktif menggunakannya. Untuk menghentikan tagihan biaya untuk klaster ini, Anda harus menghapusnya. Lihat Menghapus cluster di ElastiCache.

Untuk menangani mode klaster diaktifkan, lihat topik berikut:

- Untuk menggunakan konsol, lihat Membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) (Konsol).
- Untuk menggunakan AWS CLI, lihatMembuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) ()AWS CLI.

Langkah 3. Otorisasi akses ke cluster

Bagian ini mengasumsikan bahwa Anda terbiasa meluncurkan dan menghubungkan ke EC2 instans Amazon. Untuk informasi selengkapnya, lihat Panduan EC2 Memulai Amazon.

Semua ElastiCache cluster dirancang untuk diakses dari EC2 instans Amazon. Skenario yang paling umum adalah mengakses ElastiCache cluster dari EC2 instance Amazon di Amazon Virtual Private Cloud (Amazon VPC) yang sama, yang akan menjadi kasus untuk latihan ini.

Secara default, akses jaringan ke klaster Anda dibatasi untuk akun yang digunakan untuk membuatnya. Sebelum Anda dapat terhubung ke cluster dari sebuah EC2 instance, Anda harus mengotorisasi EC2 instance untuk mengakses cluster.

Kasus penggunaan yang paling umum adalah ketika aplikasi yang digunakan pada sebuah EC2 instance perlu terhubung ke cluster di VPC yang sama. Cara termudah untuk mengelola akses antara EC2 instance dan cluster di VPC yang sama adalah dengan melakukan hal berikut:

1. Buat grup keamanan VPC untuk klaster Anda. Grup keamanan ini dapat digunakan untuk membatasi akses ke instans klaster. Sebagai contoh, Anda dapat membuat aturan kustom untuk grup keamanan ini yang mengizinkan akses TCP menggunakan port yang Anda tetapkan untuk klaster saat Anda membuatnya dan alamat IP yang Anda gunakan untuk mengakses klaster tersebut.

   Port default untuk cluster Valkey atau Redis OSS dan grup replikasi adalah. 6379

   > **Important**
   >
   > Grup ElastiCache keamanan Amazon hanya berlaku untuk cluster yang tidak berjalan di lingkungan Amazon Virtual Private Cloud (VPC). Jika Anda menjalankannya di Amazon Virtual Private Cloud, Grup Keamanan tidak tersedia pada panel navigasi konsol.
   > Jika Anda menjalankan ElastiCache node di VPC Amazon, Anda mengontrol akses ke cluster Anda dengan grup keamanan Amazon VPC, yang berbeda dari grup keamanan. ElastiCache Untuk informasi selengkapnya tentang penggunaan ElastiCache di VPC Amazon, lihat Amazon VPCs dan ElastiCache keamanan

2. Buat grup keamanan VPC untuk EC2 instance Anda (server web dan aplikasi). Grup keamanan ini dapat, jika diperlukan, mengizinkan akses ke EC2 instance dari Internet melalui tabel routing VPC. Misalnya, Anda dapat menetapkan aturan pada grup keamanan ini untuk mengizinkan akses TCP ke EC2 instance melalui port 22.

3. Buat aturan kustom di grup keamanan untuk Cluster Anda yang memungkinkan koneksi dari grup keamanan yang Anda buat untuk EC2 instans Anda. Hal ini akan mengizinkan semua anggota grup keamanan untuk mengakses klaster.

> **Note**
>
> Jika Anda berencana untuk menggunakan Zona Lokal, pastikan Anda telah mengaktifkannya. Saat Anda membuat grup subnet di zona lokal, VPC Anda diperluas ke Zona Lokal tersebut dan VPC Anda akan memperlakukan subnet itu seperti subnet lain di Zona Ketersediaan lainnya. Semua gateway dan tabel rute yang berkaitan akan disesuaikan secara otomatis.

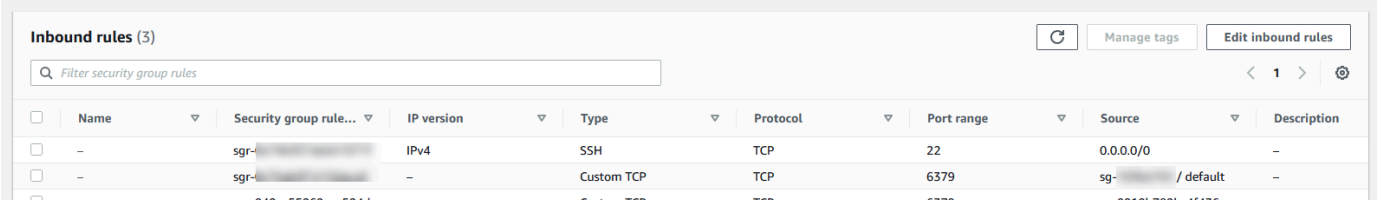
Untuk membuat aturan dalam grup keamanan VPC yang memungkinkan koneksi dari grup keamanan lain

1. Masuk ke Konsol AWS Manajemen dan buka konsol VPC Amazon di https://console.aws.amazon.com /vpc.
2. Pada panel navigasi, pilih Grup Keamanan.
3. Pilih atau buat grup keamanan yang akan Anda gunakan untuk instans Klaster Anda. Pada Aturan Masuk, pilih Edit Aturan Masuk lalu pilih Tambahkan Aturan. Grup keamanan ini akan mengizinkan akses bagi anggota dari grup keamanan lain.
4. Dari Jenis, pilih Aturan TCP Kustom.

   a. Untuk Rentang Port, tentukan port yang Anda gunakan saat membuat klaster.

      Port default untuk cluster Valkey atau Redis OSS dan grup replikasi adalah. 6379

   b. Pada kotak Sumber, masukkan ID dari grup keamanan. Dari daftar pilih grup keamanan yang akan Anda gunakan untuk EC2 instans Amazon Anda.
5. Pilih Simpan jika selesai.



Setelah Anda mengaktifkan akses, Anda sekarang siap untuk terhubung ke simpul, yang dibahas pada bagian berikutnya.

Untuk informasi tentang mengakses ElastiCache klaster Anda dari VPC Amazon yang berbeda, Wilayah yang AWS berbeda, atau bahkan jaringan perusahaan Anda, lihat berikut ini:

- Pola Akses untuk Mengakses ElastiCache Cache di VPC Amazon
- Mengakses ElastiCache sumber daya dari luar AWS

Langkah 4. Connect ke node cluster

Sebelum melanjutkan, selesaikan Langkah 3. Otorisasi akses ke cluster.

Bagian ini mengasumsikan bahwa Anda telah membuat EC2 instance Amazon dan dapat terhubung dengannya. Untuk petunjuk tentang cara melakukannya, lihat Panduan EC2 Memulai Amazon.

EC2 Instance Amazon dapat terhubung ke node cluster hanya jika Anda telah mengotorisasi untuk melakukannya.

Temukan titik akhir simpul Anda

Ketika klaster Anda dalam status tersedia dan Anda telah mengotorisasi akses ke sana, Anda dapat masuk ke EC2 instans Amazon dan terhubung ke cluster. Untuk melakukan itu, Anda perlu menentukan titik akhir terlebih dahulu.

Menemukan titik akhir klaster Valkey (mode cluster dinonaktifkan) (Konsol)

Jika cluster Valkey (mode cluster dinonaktifkan) hanya memiliki satu node, titik akhir node digunakan untuk membaca dan menulis. Jika klaster memiliki beberapa simpul, terdapat tiga jenis titik akhir; titik akhir primer, titik akhir pembaca dan titik akhir simpul.

Titik akhir primer adalah nama DNS yang selalu diresolusi ke simpul primer di klaster. Titik akhir primer tidak terpengaruh oleh perubahan klaster Anda, seperti promosi replika baca ke peran primer. Untuk aktivitas tulis, sebaiknya aplikasi Anda terhubung ke titik akhir primer.

Titik akhir pembaca akan membagi koneksi masuk secara merata ke titik akhir antara semua replika baca dalam sebuah cluster. ElastiCache Faktor lain seperti saat aplikasi membuat koneksi atau cara aplikasi menggunakan atau menggunakan ulang koneksi akan menentukan distribusi lalu lintas. Titik akhir pembaca tetap mengikuti perubahan klaster dalam waktu nyata saat replika ditambahkan atau dihapus. Anda dapat menempatkan beberapa replika baca ElastiCache klaster Anda di AWS Availability Zones (AZ) yang berbeda untuk memastikan ketersediaan titik akhir pembaca yang tinggi.

> **Note**
>
> Titik akhir pembaca bukan penyeimbang beban. Ini adalah catatan DNS yang akan diresolusi sebagai alamat IP dari salah satu simpul replika dengan metode round robin.

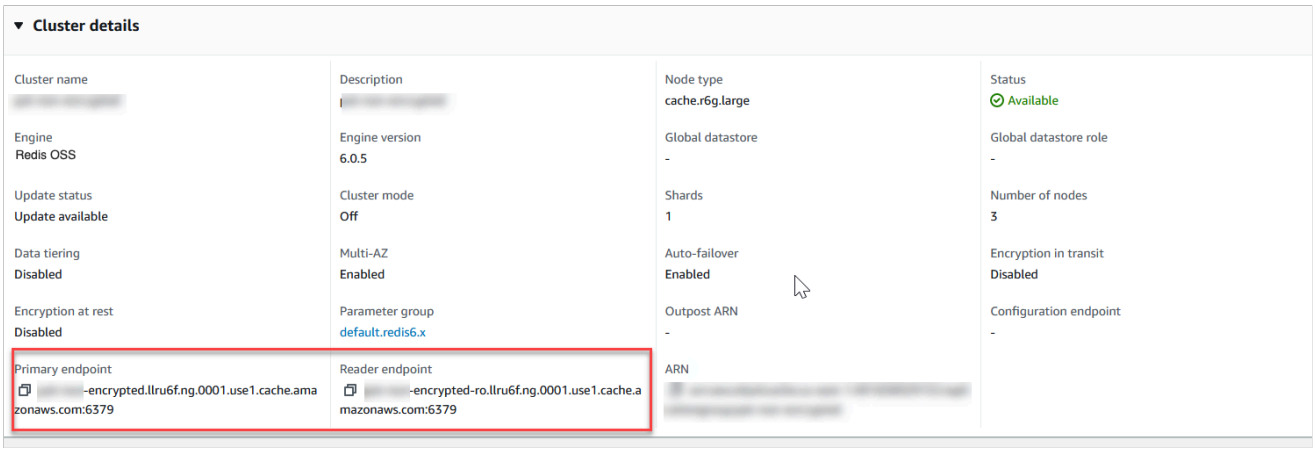
Untuk aktivitas baca, aplikasi juga dapat menghubungkan ke simpul mana pun di klaster. Tidak seperti titik akhir primer, titik akhir simpul diresolusi ke titik akhir tertentu. Jika Anda membuat

perubahan dalam klaster Anda, seperti menambahkan atau menghapus replika, Anda harus memperbarui titik akhir simpul di aplikasi Anda.

Untuk menemukan titik akhir klaster Valkey (mode cluster dinonaktifkan)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cache Valkey atau cache Redis OSS.

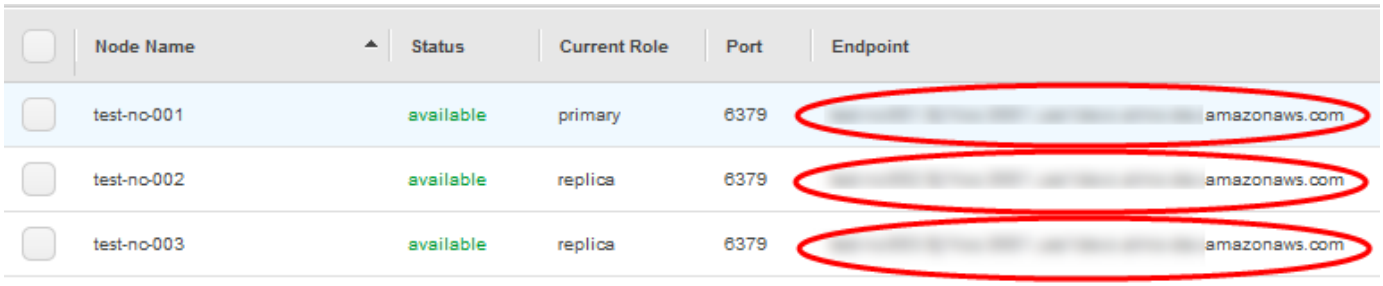
   Layar cluster akan muncul dengan daftar yang akan mencakup cache tanpa server Valkey atau Redis OSS yang ada, Valkey (mode cluster dinonaktifkan) dan Valkey (mode cluster diaktifkan) cluster. Pilih klaster yang Anda buat di bagian Membuat cluster Valkey (mode cluster dinonaktifkan) (Konsol).
3. Untuk menemukan titik akhir and/or Pembaca Utama cluster, pilih nama cluster (bukan tombol radio).



   Titik akhir Primer dan Pembaca untuk klaster Valkey (mode cluster dinonaktifkan)

   Jika hanya ada satu simpul dalam klaster, berarti tidak ada titik akhir primer dan Anda dapat melanjutkan ke langkah berikutnya.
4. Jika klaster Valkey (mode cluster dinonaktifkan) memiliki node replika, Anda dapat menemukan titik akhir node replika cluster dengan memilih nama cluster dan kemudian memilih tab Nodes.

   Layar simpul muncul dengan setiap simpul yang ada di klaster, primer dan replika, yang tercantum dengan titik akhirnya.



Titik akhir node untuk cluster Valkey (mode cluster dinonaktifkan)

5. Untuk menyalin titik akhir ke clipboard Anda:

   a. Temukan satu per satu titik akhir yang ingin Anda salin.

   b. Pilih ikon salin langsung di depan titik akhir.

   Titik akhir sekarang disalin ke clipboard Anda. Untuk informasi tentang menggunakan titik akhir agar terhubung ke simpul, lihat Menghubungkan ke simpul.

Titik akhir utama Valkey (mode cluster dinonaktifkan) terlihat seperti berikut ini. Ada perbedaan yang tergantung pada apakah enkripsi Bergerak aktif atau tidak.

Enkripsi bergerak tidak diaktifkan

```
clusterName.xxxxxx.nodeId.regionAndAz.cache.amazonaws.com:port

redis-01.7abc2d.0001.usw2.cache.amazonaws.com:6379
```

Enkripsi bergerak diaktifkan

```
master.clusterName.xxxxxx.regionAndAz.cache.amazonaws.com:port

master.ncit.ameaqx.use1.cache.amazonaws.com:6379
```

Untuk mengetahui lebih banyak cara menemukan titik akhir Anda, lihat topik yang relevan untuk jenis klaster dan mesin yang Anda jalankan.

- Menemukan titik akhir koneksi di ElastiCache
- Menemukan Titik Akhir untuk Cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan) (Konsol)— Anda memerlukan titik akhir Konfigurasi dari klaster.

- Menemukan Titik Akhir (AWS CLI)
- Menemukan Titik Akhir (ElastiCache API)

Connect ke cluster Valkey atau Redis OSS atau grup replikasi (Linux)

Sekarang Anda memiliki titik akhir yang Anda butuhkan, Anda dapat masuk ke sebuah EC2 instance dan terhubung ke cluster atau grup replikasi. Dalam contoh berikut, Anda menggunakan utilitas valkey-cli untuk terhubung ke cluster. Versi terbaru dari valkey-cli juga mendukung SSL/TLS untuk menghubungkan cluster yang diaktifkan. encryption/authentication

Contoh berikut menggunakan EC2 instans Amazon yang menjalankan Amazon Linux dan Amazon Linux 2. Untuk detail tentang menginstal dan mengkompilasi valkey-cli dengan distribusi Linux lainnya, lihat dokumentasi untuk sistem operasi spesifik Anda..

> **Note**
>
> Proses ini mencakup pengujian koneksi menggunakan utilitas valkey-cli hanya untuk penggunaan yang tidak direncanakan. Untuk daftar klien Valkey dan Redis OSS yang didukung, lihat dokumentasi Valkey. Untuk contoh menggunakan AWS SDKs with ElastiCache, lihatTutorial: Memulai dengan Python dan ElastiCache.

Menghubungkan ke klaster tanpa enkripsi dengan mode klaster dinonaktifkan

1. Jalankan perintah berikut untuk terhubung ke cluster *primary-endpoint* dan ganti dan *port number* dengan titik akhir cluster dan nomor port Anda. (Port default untuk Valkey atau Redis OSS adalah 6379.)

   ```
   src/valkey-cli -h primary-endpoint -p port number
   ```

   Hasil dalam prompt perintah Valkey atau Redis OSS terlihat mirip dengan yang berikut ini:

   ```
   primary-endpoint:port number
   ```

2. Anda sekarang dapat menjalankan perintah Valkey atau Redis OSS.

   ```
   set x Hello
   OK
   ```

```
get x
"Hello"
```

Menghubungkan ke klaster tanpa enkripsi dengan mode klaster diaktifkan

1. Jalankan perintah berikut untuk terhubung ke cluster *configuration-endpoint* dan ganti dan *port number* dengan titik akhir cluster dan nomor port Anda. (Port default untuk Valkey atau Redis OSS adalah 6379.)

   ```
   src/valkey-cli -h configuration-endpoint -c -p port number
   ```

   > **Note**
   >
   > Pada perintah sebelumnya, opsi -c memungkinkan mode klaster mengikuti pengalihan -ASK dan -MOVED.

   Hasil dalam prompt perintah Valkey atau Redis OSS terlihat mirip dengan yang berikut ini:

   ```
   configuration-endpoint:port number
   ```

2. Anda sekarang dapat menjalankan perintah Valkey atau Redis OSS. Perhatikan bahwa pengalihan terjadi karena Anda mengaktifkannya menggunakan opsi -c. Jika pengalihan tidak diaktifkan, perintah akan menghasilkan kesalahan MOVED. Untuk informasi selengkapnya tentang kesalahan MOVED, lihat spesifikasi cluster Redis OSS.

   ```
   set x Hi
   -> Redirected to slot [16287] located at 172.31.28.122:6379
   OK
   set y Hello
   OK
   get y
   "Hello"
   set z Bye
   -> Redirected to slot [8157] located at 172.31.9.201:6379
   OK
   get z
   "Bye"
   get x
   ```

```
-> Redirected to slot [16287] located at 172.31.28.122:6379
"Hi"
```

Menghubungkan ke cluster yang Encryption/Authentication diaktifkan

Secara default, valkey-cli menggunakan koneksi TCP yang tidak terenkripsi saat menghubungkan ke Valkey atau Redis OSS. Opsi ini `BUILD_TLS=yes` memungkinkan SSL/TLS pada saat kompilasi valkey-cli seperti yang ditunjukkan pada bagian sebelumnya. Unduh dan atur akses baris perintah Mengaktifkan AUTH bersifat opsional. Namun, Anda harus mengaktifkan enkripsi bergerak untuk mengaktifkan AUTH. Untuk detail selengkapnya tentang ElastiCache enkripsi dan otentikasi, lihatElastiCache enkripsi dalam transit (TLS).

> **Note**
>
> Anda dapat menggunakan opsi `--tls` dengan valkey-cli untuk terhubung ke mode cluster yang diaktifkan dan cluster terenkripsi yang dinonaktifkan. Jika token AUTH pada klaster telah diatur, maka Anda dapat menggunakan opsi `-a` untuk menyediakan kata sandi AUTH.

Dalam contoh berikut, pastikan untuk mengganti *`cluster-endpoint`* dan *`port number`* dengan titik akhir cluster dan nomor port Anda. (Port default untuk Valkey atau Redis OSS adalah 6379.)

Menghubungkan ke klaster terenkripsi dengan mode klaster dinonaktifkan

Contoh berikut menghubungkan ke klaster yang mengaktifkan enkripsi dan autentikasi:

```
src/valkey-cli -h cluster-endpoint --tls -a your-password -p port number
```

Contoh berikut terhubung ke klaster yang hanya mengaktifkan enkripsi:

```
src/valkey-cli -h cluster-endpoint --tls -p port number
```

Menghubungkan ke klaster terenkripsi dengan mode klaster diaktifkan

Contoh berikut menghubungkan ke klaster yang mengaktifkan enkripsi dan autentikasi:

```
src/valkey-cli -c -h cluster-endpoint --tls -a your-password -p port number
```

Contoh berikut terhubung ke klaster yang hanya mengaktifkan enkripsi:

```
src/valkey-cli -c -h cluster-endpoint --tls -p port number
```

Setelah Anda terhubung ke cluster, Anda dapat menjalankan perintah Valkey atau Redis OSS seperti yang ditunjukkan pada contoh sebelumnya untuk cluster yang tidak terenkripsi.

valkey-cli alternatif

Jika cluster tidak diaktifkan mode cluster dan Anda perlu membuat koneksi ke cluster untuk pengujian singkat tetapi tanpa melalui kompilasi valkey-cli, Anda dapat menggunakan telnet atau openssl. Dalam contoh perintah berikut, pastikan untuk mengganti *cluster-endpoint* dan *port number* dengan titik akhir cluster dan nomor port Anda. (Port default untuk Valkey atau Redis OSS adalah 6379.)

Contoh berikut menghubungkan ke and/or otentikasi enkripsi diaktifkan mode cluster cluster dinonaktifkan:

```
openssl s_client -connect cluster-endpoint:port number
```

Jika kata sandi klaster telah ditetapkan, hubungkan ke klaster terlebih dahulu. Setelah terhubung, lakukan autentikasi pada klaster menggunakan perintah berikut, lalu tekan tombol `Enter`. Dalam contoh berikut, ganti *your-password* dengan kata sandi untuk cluster Anda.

```
Auth your-password
```

Contoh berikut menghubungkan ke klaster (mode klaster dinonaktifkan) yang tidak memiliki enkripsi atau autentikasi aktif:

```
telnet cluster-endpoint port number
```

Connect ke cluster Valkey atau Redis OSS atau grup replikasi (Windows)

Untuk terhubung ke cluster Valkey atau Redis OSS dari instance EC2 Windows menggunakan Valkey CLI atau Redis OSS CLI, Anda harus mengunduh paket valkey-cli dan menggunakan valkey-cli.exe untuk terhubung ke cluster Valkey atau Redis OSS dari instance Windows. EC2

Dalam contoh berikut, Anda menggunakan utilitas valkey-cli untuk terhubung ke cluster yang tidak mengaktifkan enkripsi dan menjalankan Valkey atau Redis OSS. Untuk informasi lebih lanjut tentang

Valkey atau Redis OSS dan perintah yang tersedia, lihat perintah Valkey dan Redis OSS di situs web Valkey.

Untuk terhubung ke cluster Valkey atau Redis OSS yang tidak diaktifkan enkripsi menggunakan valkey-cli

1. Connect ke EC2 instans Amazon Anda menggunakan utilitas koneksi pilihan Anda. Untuk petunjuk tentang cara menyambung ke EC2 instans Amazon, lihat Panduan EC2 Memulai Amazon.
2. Salin dan tempel tautan https://github.com/microsoftarchive/redis/releases/download/win-3.0.504/Redis-x64-3.0.504.zip di browser Internet untuk mengunduh file zip untuk klien Valkey dari rilis yang tersedia di GitHub https://github.com/microsoftarchive/redis/releases/tag/win-3.0.504

   Ekstrak file zip tersebut ke folder/jalur yang Anda inginkan.

   Buka Command Prompt dan ubah ke direktori Valkey dan jalankan perintah`c:\Valkey>valkey-cli -h` *`Redis_Cluster_Endpoint`* `-p 6379`.

   Misalnya:

   ```
   c:\Valkey>valkey-cli -h cmd.xxxxxxx.ng.0001.usw2.cache.amazonaws.com -p 6379
   ```
3. Jalankan perintah Valkey atau Redis OSS.

   Anda sekarang terhubung ke cluster dan dapat menjalankan perintah Valkey atau Redis OSS seperti berikut ini.

   ```
   set a "hello"          // Set key "a" with a string value and no expiration
   OK
   get a                  // Get value for key "a"
   "hello"
   get b                  // Get value for key "b" results in miss
   (nil)
   set b "Good-bye" EX 5  // Set key "b" with a string value and a 5 second expiration
   "Good-bye"
   get b                  // Get value for key "b"
   "Good-bye"
                          // wait >= 5 seconds
   get b
   (nil)                  // key has expired, nothing returned
   quit                   // Exit from valkey-cli
   ```

Apa yang saya lakukan selanjutnya?

Sekarang setelah Anda mencoba latihan Memulai, Anda dapat menjelajahi bagian berikut untuk mempelajari lebih lanjut ElastiCache dan alat yang tersedia:

- Memulai dengan AWS
- Alat untuk Amazon Web Services
- AWS Antarmuka Baris Perintah
- Referensi ElastiCache API Amazon

Setelah Anda menyelesaikan latihan Memulai, Anda dapat membaca bagian ini untuk mempelajari lebih lanjut tentang ElastiCache administrasi:

- Memilih ukuran simpul Anda

  Anda menginginkan cache Anda cukup besar untuk mengakomodasi semua data yang ingin dijadikan cache. Pada saat yang sama, Anda tidak ingin mengeluarkan biaya lebih dari kebutuhan cache Anda. Gunakan topik ini untuk membantu Anda memilih ukuran simpul yang terbaik.
- ElastiCache praktik terbaik dan strategi caching

  Identifikasi dan atasi masalah yang dapat memengaruhi efisiensi klaster Anda.

Merancang cluster ElastiCache Redis OSS Anda sendiri

Berikut ini adalah tindakan satu kali yang harus Anda ambil untuk merancang cluster ElastiCache Redis OSS Anda sendiri.

Untuk informasi lebih lanjut tentang pengaturan ElastiCache lihatMenyiapkan ElastiCache.

Topik

- Langkah 1: Membuat grup subnet
- Langkah 2: Buat klaster
- Langkah 3: Mengizinkan akses ke klaster
- Langkah 4: Menghubungkan ke simpul klaster

Langkah 1: Membuat grup subnet

Sebelum membuat klaster, grup subnet dibuat terlebih dahulu. Grup subnet cache adalah kumpulan subnet yang dapat ditetapkan untuk klaster cache Anda dalam VPC. Saat meluncurkan klaster cache di VPC, Anda harus memilih grup subnet cache. Kemudian ElastiCache menggunakan kelompok subnet cache itu untuk menetapkan alamat IP dalam subnet itu ke setiap node cache di cluster.

Saat Anda membuat grup subnet baru, perhatikan jumlah alamat IP yang tersedia. Jika subnet memiliki sangat sedikit alamat IP yang bebas, Anda akan dibatasi dalam hal jumlah simpul yang dapat ditambahkan ke klaster. Untuk mengatasi masalah ini, Anda dapat menetapkan satu atau beberapa subnet ke grup subnet sehingga Anda memiliki jumlah alamat IP yang cukup dalam Zona Ketersediaan dari klaster Anda. Setelah itu, Anda dapat menambahkan lebih banyak simpul ke klaster Anda.

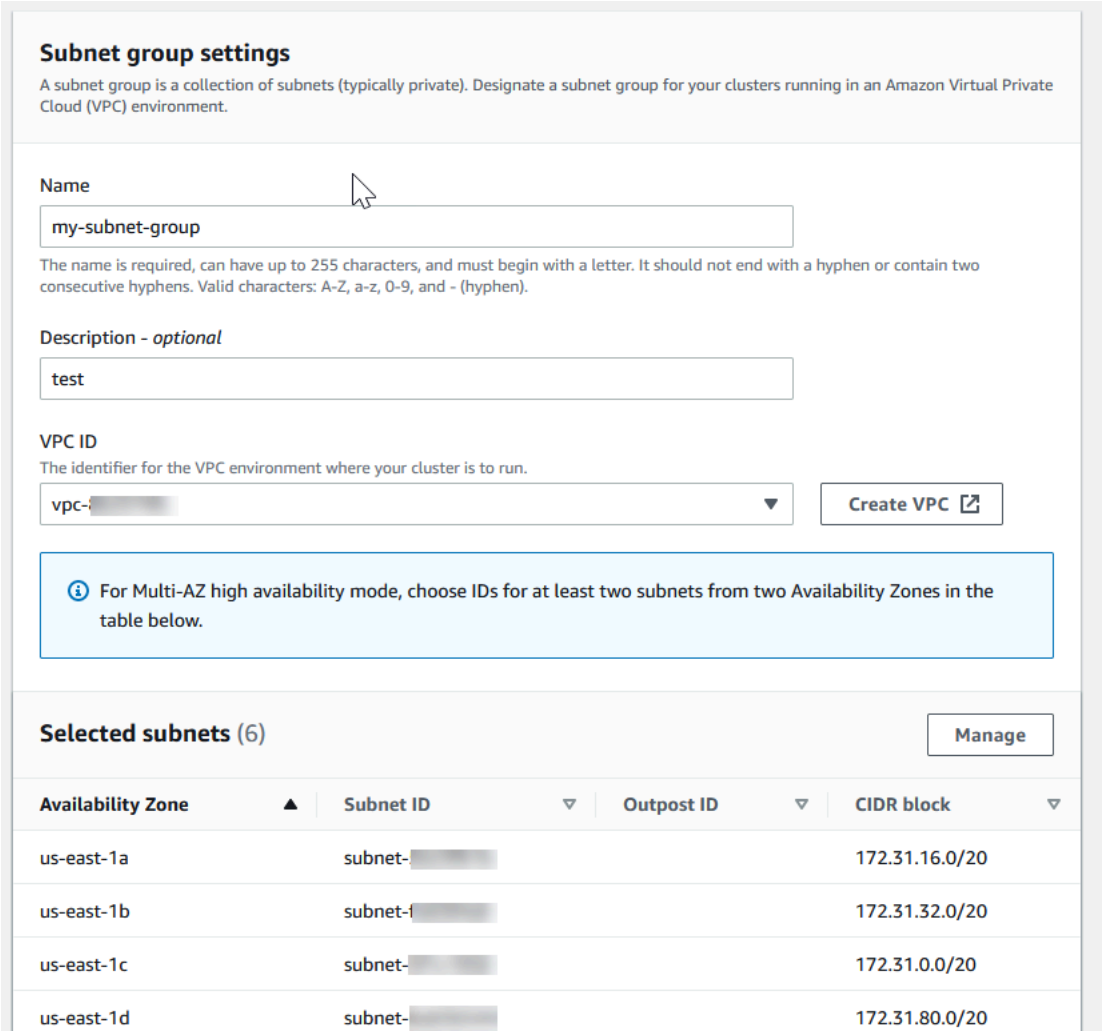
Prosedur berikut menunjukkan cara membuat grup subnet yang disebut `mysubnetgroup` (konsol), dan AWS CLI.

Membuat grup subnet (Konsol)

Prosedur berikut menunjukkan cara membuat grup subnet (konsol).

Untuk membuat grup subnet (Konsol)

1. Masuk ke Konsol AWS Manajemen, dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di daftar navigasi, pilih Grup subnet.
3. Pilih Buat Grup Subnet.
4. Pada wizard Buat Grup Subnet, lakukan hal berikut. Jika semua pengaturan sudah sesuai keinginan Anda, pilih Ya, Buat.
    a. Pada kotak Nama, ketik nama grup subnet Anda.
    b. Di kotak Deskripsi, ketik deskripsi untuk grup subnet Anda.
    c. Pada kotak ID VPC, pilih Amazon VPC yang Anda buat.
    d. Pada daftar Zona Ketersediaan dan ID Subnet, pilih Zona Ketersediaan atau Zona Lokal dan ID subnet privat Anda, lalu pilih Tambahkan.



5. Pada pesan konfirmasi yang muncul, pilih Tutup.

Grup subnet baru Anda muncul di daftar Grup Subnet konsol. ElastiCache Di bagian bawah jendela, Anda dapat memilih grup subnet untuk melihat detail, misalnya semua subnet yang terkait dengan grup ini.

Membuat grup subnet (AWS CLI)

Pada prompt perintah, gunakan perintah `create-cache-subnet-group` untuk membuat grup subnet.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-cache-subnet-group \
    --cache-subnet-group-name mysubnetgroup \
    --cache-subnet-group-description "Testing" \
```

```
    --subnet-ids subnet-53df9c3a
```

Untuk Windows:

```
aws elasticache create-cache-subnet-group ^
    --cache-subnet-group-name mysubnetgroup ^
    --cache-subnet-group-description "Testing" ^
    --subnet-ids subnet-53df9c3a
```

Perintah ini seharusnya menghasilkan output yang serupa dengan yang berikut:

```
{
    "CacheSubnetGroup": {
        "VpcId": "vpc-37c3cd17",
        "CacheSubnetGroupDescription": "Testing",
        "Subnets": [
            {
                "SubnetIdentifier": "subnet-53df9c3a",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2a"
                }
            }
        ],
        "CacheSubnetGroupName": "mysubnetgroup"
    }
}
```

Untuk informasi lebih lanjut, lihat AWS CLI topiknyacreate-cache-subnet-group.

Langkah 2: Buat klaster

Sebelum membuat klaster untuk tujuan produksi, Anda tentu perlu mempertimbangkan pengaturan konfigurasi klaster untuk memenuhi kebutuhan bisnis Anda. Masalah terkait itu dibahas di bagian Mempersiapkan cluster di ElastiCache. Untuk tujuan latihan Memulai ini, Anda akan membuat klaster dengan mode klaster dinonaktifkan dan Anda dapat menggunakan nilai konfigurasi default jika sesuai.

Klaster yang Anda buat akan berjalan langsung, dan tidak berjalan di sandbox. Anda akan dikenakan biaya ElastiCache penggunaan standar untuk instans sampai Anda menghapusnya. Jumlah biayanya cukup kecil (biasanya kurang dari satu dolar) jika Anda menyelesaikan latihan yang dijelaskan di sini

dalam satu sesi dan menghapus klaster itu ketika Anda sudah selesai. Untuk informasi selengkapnya tentang tarif ElastiCache penggunaan, lihat Amazon ElastiCache.

Klaster Anda diluncurkan dalam cloud privat virtual (VPC) berdasarkan layanan Amazon VPC.

Membuat cluster Redis OSS (mode cluster dinonaktifkan) (Konsol)

Untuk membuat cluster Redis OSS (mode cluster dinonaktifkan) menggunakan konsol ElastiCache

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Dari daftar di sudut kanan atas, pilih AWS Wilayah tempat Anda ingin meluncurkan cluster ini.
3. Pilih Mulai di panel navigasi.
4. Pilih Buat VPC dan ikuti langkah-langkah yang telah dijelaskan dalam Membuat Cloud Privat Virtual (VPC).
5. Di halaman ElastiCache dasbor, pilih cache Valkey atau cache Redis OSS. Untuk latihan ini kita akan memilih cache Redis OSS, dan kemudian pilih Buat cache Redis OSS.
6. Di bagian Pengaturan klaster, lakukan hal berikut:

    a. Pilih Konfigurasi dan buat klaster baru.
    b. Untuk Mode klaster, pilih Dinonaktifkan.
    c. Untuk Info klaster masukkan nilai untuk Nama.
    d. (Opsional) Masukkan nilai untuk Deskripsi.
7. Di bagian Lokasi:

    AWS Cloud

    1. Untuk AWS Cloud, sebaiknya terima pengaturan default untuk Multi-AZ dan Failover otomatis. Untuk informasi selengkapnya, lihat Meminimalkan waktu henti ElastiCache untuk Redis OSS dengan Multi-AZ.
    2. Pada Pengaturan klaster

        a. Untuk Versi mesin, pilih versi yang tersedia.
        b. Untuk Port, gunakan port default, 6379. Jika Anda memiliki alasan untuk menggunakan port lain, masukkan nomor port tersebut.
        c. Untuk Grup parameter, pilih grup parameter atau buat yang baru. Grup parameter mengontrol parameter runtime dari klaster Anda. Untuk informasi selengkapnya

tentang grup parameter, lihat Parameter Valkey dan Redis OSS dan Membuat grup ElastiCache parameter.

> **Note**
>
> Saat Anda memilih grup parameter untuk menetapkan nilai konfigurasi mesin, grup parameter tersebut diterapkan ke semua klaster di penyimpanan data global. Pada halaman Grup Parameter, atribut Global ya/tidak menunjukkan apakah grup parameter adalah bagian dari penyimpanan data global.

d. Untuk Jenis simpul, pilih panah bawah
( ).
Pada kotak dialog Ubah jenis simpul, pilih nilai untuk Keluarga instans untuk jenis simpul yang Anda inginkan. Kemudian pilih jenis simpul yang ingin Anda gunakan untuk klaster ini, lalu pilih Simpan.

Untuk informasi selengkapnya, lihat Memilih ukuran simpul Anda.

Jika Anda memilih jenis simpul r6gd, tingkatan data akan diaktifkan secara otomatis. Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.

e. Untuk Jumlah replika, pilih jumlah replika baca yang Anda inginkan. Jika Anda mengaktifkan Multi-AZ, jumlahnya harus antara 1-5.

3. Pada Konektivitas

a. Untuk Jenis jaringan, pilih versi IP yang akan didukung oleh klaster ini.

b. Untuk grup Subnet, pilih subnet yang ingin Anda terapkan ke cluster ini. ElastiCache menggunakan grup subnet itu untuk memilih subnet dan alamat IP dalam subnet itu untuk dikaitkan dengan node Anda. ElastiCache cluster memerlukan subnet dual-stack dengan keduanya IPv4 dan IPv6 alamat yang ditetapkan untuk beroperasi dalam mode dual-stack dan subnet -only untuk beroperasi sebagai IPv6 -only. IPv6

Saat membuat grup subnet baru, masukkan ID VPC yang menaungi grup subnet tersebut.

Untuk informasi selengkapnya, lihat:

- Memilih jenis jaringan di ElastiCache.

- Membuat subnet di VPC Anda.

  Jika Anda adalah Menggunakan zona lokal dengan ElastiCache, Anda harus membuat atau memilih subnet yang berada di zona lokal.

  Untuk informasi selengkapnya, lihat Subnet dan grup subnet.

4. Untuk Penempatan zona ketersediaan, Anda memiliki dua opsi:

   - Tidak ada preferensi — ElastiCache memilih Availability Zone.
   - Tentukan zona ketersediaan – Anda menentukan Zona Ketersediaan untuk setiap klaster.

     Jika Anda memilih untuk menentukan Zona Ketersediaan, untuk setiap klaster di setiap serpihan, pilih Zona Ketersediaan dari daftar.

   Untuk informasi selengkapnya, lihat Memilih wilayah dan zona ketersediaan untuk ElastiCache.

5. Pilih Berikutnya
6. Di bawah pengaturan Advanced Redis OSS

   - Untuk Keamanan:

     i. Untuk mengenkripsi data Anda, Anda memiliki opsi berikut:

     - Enkripsi diam – Mengaktifkan enkripsi pada data yang disimpan di disk. Untuk informasi selengkapnya, lihat Enkripsi Diam.

       > **Note**
       >
       > Anda memiliki opsi untuk menyediakan kunci enkripsi yang berbeda dengan memilih kunci AWS KMS yang Dikelola Pelanggan dan memilih kunci. Untuk informasi selengkapnya, lihat Menggunakan kunci yang dikelola pelanggan dari AWS KMS.

     - Enkripsi bergerak – Mengaktifkan enkripsi data selama pengiriman. Untuk informasi selengkapnya, lihat Enkripsi bergerak. Untuk ElastiCache engine versi 6.0 untuk Redis OSS dan di atasnya, jika Anda mengaktifkan enkripsi

dalam perjalanan, Anda akan diminta untuk menentukan salah satu opsi Kontrol Akses berikut:

- Tanpa Kontrol Akses – Ini adalah pengaturan default. Opsi ini menunjukkan bahwa tidak ada batasan akses pengguna ke klaster.
- Daftar Kontrol Akses Grup Pengguna – Pilih grup pengguna dengan kumpulan pengguna tertentu yang dapat mengakses klaster. Untuk informasi selengkapnya, lihat Mengelola Grup Pengguna dengan Konsol dan CLI.
- AUTH Default User - Mekanisme otentikasi untuk server Valkey dan Redis OSS. Untuk informasi lebih lanjut, lihat AUTH.

- AUTH — Mekanisme otentikasi untuk server Redis OSS. Untuk informasi lebih lanjut, lihat AUTH.

> **Note**
>
> Untuk versi Redis OSS antara 3.2.6 dan seterusnya, tidak termasuk versi 3.2.10, Redis OSS AUTH adalah satu-satunya pilihan.

ii. Untuk Grup keamanan, pilih grup keamanan yang Anda inginkan untuk klaster ini. Grup keamanan bertindak sebagai firewall untuk mengontrol akses jaringan ke klaster Anda. Anda dapat menggunakan grup keamanan default untuk VPC Anda atau membuat yang baru.

Untuk informasi selengkapnya tentang grup keamanan, lihat Grup Keamanan untuk VPC Anda dalam Panduan Pengguna Amazon VPC.

7. Untuk pencadangan otomatis terjadwal secara berkala, pilih Aktifkan pencadangan otomatis, lalu masukkan jumlah hari yang diinginkan untuk mempertahankan cadangan otomatis sebelum dihapus secara otomatis. Jika Anda tidak ingin melakukan pencadangan otomatis terjadwal secara berkala, hapus kotak centang Aktifkan pencadangan otomatis. Apa pun pilihannya, Anda dapat membuat pencadangan secara manual kapan saja.

   Untuk informasi selengkapnya tentang pencadangan dan pemulihan, lihatMelakukan snapshot dan pemulihan.

8. (Opsional) Tentukan periode pemeliharaan. Jendela pemeliharaan adalah waktu yang biasanya satu jam setiap minggu saat ElastiCache menjadwalkan pemeliharaan sistem

untuk klaster Anda. Anda dapat mengizinkan ElastiCache untuk memilih hari dan waktu untuk jendela pemeliharaan Anda (Tidak ada preferensi), atau Anda dapat memilih hari, waktu, dan durasi sendiri (Tentukan jendela pemeliharaan). Jika Anda memilih Tentukan periode pemeliharaan dari daftar, pilih Hari mulai, Waktu mulai, dan Durasi (dalam jam) untuk periode pemeliharaan. Semua waktu menggunakan zona waktu UTC.

Untuk informasi selengkapnya, lihat Mengelola pemeliharaan ElastiCache cluster.

9. (Opsional) Untuk Log:
   - Di bagian Format log, pilih Teks atau JSON.
   - Di bawah Jenis Tujuan, pilih CloudWatch Log atau Kinesis Firehose.
   - Di bawah Tujuan log, pilih Buat baru dan masukkan nama grup CloudWatch log Log atau nama aliran Firehose Anda, atau pilih Pilih yang ada, lalu pilih nama grup CloudWatch log Log atau nama aliran Firehose Anda,
10. Untuk Tag, untuk membantu mengelola cluster dan ElastiCache sumber daya lainnya, Anda dapat menetapkan metadata Anda sendiri ke setiap sumber daya dalam bentuk tag. Untuk informasi selengkapnya, lihat Menandai sumber daya Anda ElastiCache .
11. Pilih Berikutnya.
12. Tinjau semua entri dan pilihan Anda, lalu lakukan koreksi yang diperlukan. Saat Anda siap, pilih Buat.

On premises

1. Untuk On-premise, sebaiknya Anda membiarkan Failover otomatis tetap aktif. Untuk informasi selengkapnya, lihat Meminimalkan waktu henti ElastiCache untuk Redis OSS dengan Multi-AZ
2. Untuk menyelesaikan pembuatan klaster, ikuti langkah-langkah di Menggunakan Outposts.

Segera setelah status klaster Anda tersedia, Anda dapat memberikan Amazon EC2 akses ke sana, terhubung ke sana, dan mulai menggunakannya. Untuk informasi selengkapnya, lihat Langkah 3. Otorisasi akses ke cluster dan Langkah 4. Connect ke node cluster.

> **Important**
>
> Setelah klaster Anda tersedia, Anda akan ditagih untuk setiap jam atau durasi saat klaster aktif, meskipun Anda tidak sedang aktif menggunakannya. Untuk menghentikan tagihan biaya untuk klaster ini, Anda harus menghapusnya. Lihat Menghapus cluster di ElastiCache.

Membuat Redis OSS (mode cluster dinonaktifkan) cluster ()AWS CLI

Example

Kode CLI berikut membuat cluster cache Redis OSS (mode cluster dinonaktifkan) tanpa replika.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-cache-cluster \
--cache-cluster-id my-cluster \
--cache-node-type cache.r4.large \
--engine redis \
--num-cache-nodes 1 \
--snapshot-arns arn:aws:s3:::my_bucket/snapshot.rdb
```

Untuk Windows:

```
aws elasticache create-cache-cluster ^
--cache-cluster-id my-cluster ^
--cache-node-type cache.r4.large ^
--engine redis ^
--num-cache-nodes 1 ^
--snapshot-arns arn:aws:s3:::my_bucket/snapshot.rdb
```

Untuk menangani mode klaster diaktifkan, lihat topik berikut:

- Untuk menggunakan konsol, lihat Membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) (Konsol).
- Untuk menggunakan AWS CLI, lihatMembuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) ()AWS CLI.

Langkah 3: Mengizinkan akses ke klaster

Bagian ini mengasumsikan bahwa Anda terbiasa meluncurkan dan menghubungkan ke EC2 instans Amazon. Untuk informasi selengkapnya, lihat Panduan EC2 Memulai Amazon.

Semua ElastiCache cluster dirancang untuk diakses dari EC2 instance Amazon. Skenario yang paling umum adalah mengakses ElastiCache cluster dari EC2 instance Amazon di Amazon Virtual Private Cloud (Amazon VPC) yang sama, yang akan menjadi kasus untuk latihan ini.

Secara default, akses jaringan ke klaster Anda dibatasi untuk akun yang digunakan untuk membuatnya. Sebelum Anda dapat terhubung ke cluster dari sebuah EC2 instance, Anda harus mengotorisasi EC2 instance untuk mengakses cluster. Langkah-langkah yang diperlukan tergantung pada apakah Anda meluncurkan cluster Anda ke EC2 -VPC atau EC2 -Classic.

Kasus penggunaan yang paling umum adalah ketika aplikasi yang digunakan pada sebuah EC2 instance perlu terhubung ke cluster di VPC yang sama. Cara termudah untuk mengelola akses antara EC2 instance dan cluster di VPC yang sama adalah dengan melakukan hal berikut:

1. Buat grup keamanan VPC untuk klaster Anda. Grup keamanan ini dapat digunakan untuk membatasi akses ke instans klaster. Sebagai contoh, Anda dapat membuat aturan kustom untuk grup keamanan ini yang mengizinkan akses TCP menggunakan port yang Anda tetapkan untuk klaster saat Anda membuatnya dan alamat IP yang Anda gunakan untuk mengakses klaster tersebut.

   Port default untuk cluster Redis OSS dan grup replikasi adalah. 6379

   > **Important**
   >
   > Grup ElastiCache keamanan Amazon hanya berlaku untuk cluster yang tidak berjalan di lingkungan Amazon Virtual Private Cloud (VPC). Jika Anda menjalankannya di Amazon Virtual Private Cloud, Grup Keamanan tidak tersedia pada panel navigasi konsol.
   > Jika Anda menjalankan ElastiCache node di VPC Amazon, Anda mengontrol akses ke cluster Anda dengan grup keamanan Amazon VPC, yang berbeda dari grup keamanan. ElastiCache Untuk informasi selengkapnya tentang penggunaan ElastiCache di VPC Amazon, lihat Amazon VPCs dan ElastiCache keamanan

2. Buat grup keamanan VPC untuk EC2 instance Anda (server web dan aplikasi). Grup keamanan ini dapat, jika diperlukan, mengizinkan akses ke EC2 instance dari Internet melalui tabel routing

VPC. Misalnya, Anda dapat menetapkan aturan pada grup keamanan ini untuk mengizinkan akses TCP ke EC2 instance melalui port 22.

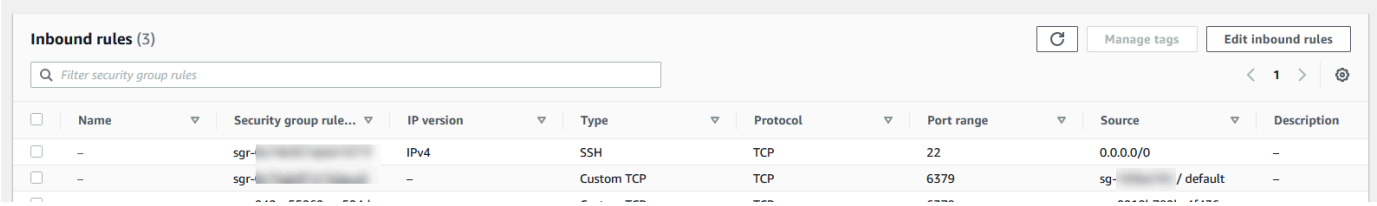
3. Buat aturan kustom di grup keamanan untuk Cluster Anda yang memungkinkan koneksi dari grup keamanan yang Anda buat untuk EC2 instans Anda. Hal ini akan mengizinkan semua anggota grup keamanan untuk mengakses klaster.

> **Note**
>
> Jika Anda berencana untuk menggunakannyaMenggunakan zona lokal dengan ElastiCache, pastikan Anda telah mengaktifkannya. Saat Anda membuat grup subnet di zona lokal, VPC Anda diperluas ke Zona Lokal tersebut dan VPC Anda akan memperlakukan subnet itu seperti subnet lain di Zona Ketersediaan lainnya. Semua gateway dan tabel rute yang berkaitan akan disesuaikan secara otomatis.

Untuk membuat aturan dalam grup keamanan VPC yang memungkinkan koneksi dari grup keamanan lain

1. Masuk ke Konsol AWS Manajemen dan buka konsol VPC Amazon di https://console.aws.amazon.com /vpc.
2. Pada panel navigasi, pilih Grup Keamanan.
3. Pilih atau buat grup keamanan yang akan Anda gunakan untuk instans Klaster Anda. Pada Aturan Masuk, pilih Edit Aturan Masuk lalu pilih Tambahkan Aturan. Grup keamanan ini akan mengizinkan akses bagi anggota dari grup keamanan lain.
4. Dari Jenis, pilih Aturan TCP Kustom.
   a. Untuk Rentang Port, tentukan port yang Anda gunakan saat membuat klaster.

      Port default untuk cluster Redis OSS dan grup replikasi adalah. 6379
   b. Pada kotak Sumber, masukkan ID dari grup keamanan. Dari daftar pilih grup keamanan yang akan Anda gunakan untuk EC2 instans Amazon Anda.
5. Pilih Simpan jika selesai.



Setelah Anda mengaktifkan akses, Anda sekarang siap untuk terhubung ke simpul, yang dibahas pada bagian berikutnya.

Untuk informasi tentang mengakses ElastiCache klaster Anda dari VPC Amazon yang berbeda, Wilayah yang AWS berbeda, atau bahkan jaringan perusahaan Anda, lihat berikut ini:

- Pola Akses untuk Mengakses ElastiCache Cache di VPC Amazon
- Mengakses ElastiCache sumber daya dari luar AWS

Langkah 4: Menghubungkan ke simpul klaster

Sebelum melanjutkan, selesaikan Langkah 3: Mengizinkan akses ke klaster.

Bagian ini mengasumsikan bahwa Anda telah membuat EC2 instance Amazon dan dapat terhubung dengannya. Untuk petunjuk tentang cara melakukannya, lihat Panduan EC2 Memulai Amazon.

EC2 Instance Amazon dapat terhubung ke node cluster hanya jika Anda telah mengotorisasi untuk melakukannya.

Temukan titik akhir simpul Anda

Ketika klaster Anda dalam status tersedia dan Anda telah mengotorisasi akses ke sana, Anda dapat masuk ke EC2 instans Amazon dan terhubung ke cluster. Untuk melakukan itu, Anda perlu menentukan titik akhir terlebih dahulu.

Menemukan Titik Akhir Cluster Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol)

Jika cluster Redis OSS (mode cluster dinonaktifkan) hanya memiliki satu node, titik akhir node digunakan untuk membaca dan menulis. Jika klaster memiliki beberapa simpul, terdapat tiga jenis titik akhir; titik akhir primer, titik akhir pembaca dan titik akhir simpul.

Titik akhir primer adalah nama DNS yang selalu diresolusi ke simpul primer di klaster. Titik akhir primer tidak terpengaruh oleh perubahan klaster Anda, seperti promosi replika baca ke peran primer. Untuk aktivitas tulis, sebaiknya aplikasi Anda terhubung ke titik akhir primer.

Titik akhir pembaca akan membagi koneksi masuk secara merata ke titik akhir antara semua replika baca di cluster ElastiCache for Redis OSS. Faktor lain seperti saat aplikasi membuat koneksi atau cara aplikasi menggunakan atau menggunakan ulang koneksi akan menentukan distribusi lalu lintas. Titik akhir pembaca tetap mengikuti perubahan klaster dalam waktu nyata saat replika ditambahkan atau dihapus. Anda dapat menempatkan beberapa replika baca klaster Redis OSS Anda ElastiCache di AWS Availability Zones (AZ) yang berbeda untuk memastikan ketersediaan titik akhir pembaca yang tinggi.

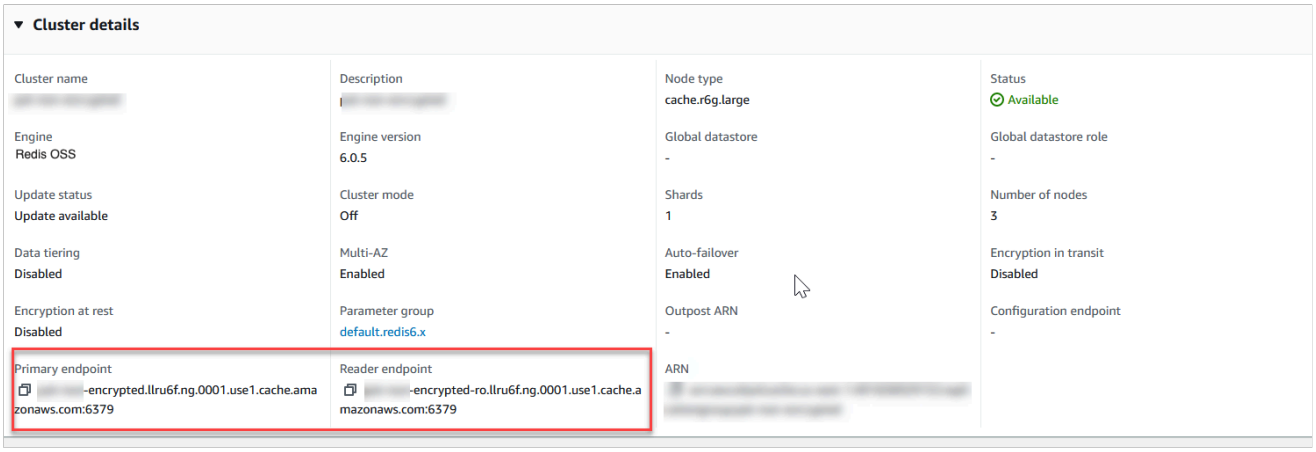
> **Note**
>
> Titik akhir pembaca bukan penyeimbang beban. Ini adalah catatan DNS yang akan diresolusi sebagai alamat IP dari salah satu simpul replika dengan metode round robin.

Untuk aktivitas baca, aplikasi juga dapat menghubungkan ke simpul mana pun di klaster. Tidak seperti titik akhir primer, titik akhir simpul diresolusi ke titik akhir tertentu. Jika Anda membuat perubahan dalam klaster Anda, seperti menambahkan atau menghapus replika, Anda harus memperbarui titik akhir simpul di aplikasi Anda.

Untuk menemukan titik akhir cluster Redis OSS (mode cluster dinonaktifkan)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cache Redis OSS.

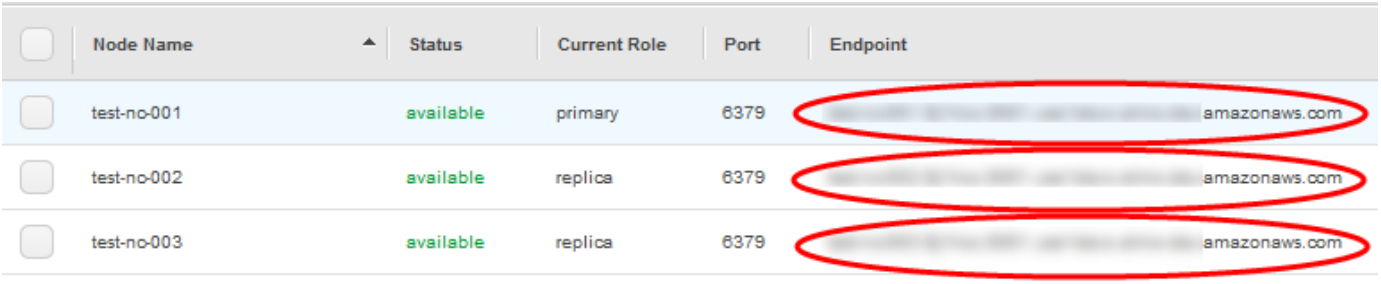
   Layar cluster akan muncul dengan daftar yang akan mencakup cache tanpa server Valkey atau Redis OSS yang ada, cluster Redis OSS (mode cluster dinonaktifkan) dan kluster Redis OSS (mode cluster diaktifkan). Pilih klaster yang Anda buat di bagian Membuat cluster Redis OSS (mode cluster dinonaktifkan) (Konsol).
3. Untuk menemukan titik akhir and/or Pembaca Utama klaster, pilih nama cluster (bukan tombol radio).



   Titik akhir Primer dan Pembaca untuk cluster Redis OSS (mode cluster dinonaktifkan)

   Jika hanya ada satu simpul dalam klaster, berarti tidak ada titik akhir primer dan Anda dapat melanjutkan ke langkah berikutnya.
4. Jika cluster Redis OSS (mode cluster dinonaktifkan) memiliki node replika, Anda dapat menemukan titik akhir node replika cluster dengan memilih nama cluster dan kemudian memilih tab Nodes.

Layar simpul muncul dengan setiap simpul yang ada di klaster, primer dan replika, yang tercantum dengan titik akhirnya.



Titik akhir node untuk cluster Redis OSS (mode cluster dinonaktifkan)

5. Untuk menyalin titik akhir ke clipboard Anda:

   a. Temukan satu per satu titik akhir yang ingin Anda salin.

   b. Pilih ikon salin langsung di depan titik akhir.

   Titik akhir sekarang disalin ke clipboard Anda. Untuk informasi tentang menggunakan titik akhir agar terhubung ke simpul, lihat Menghubungkan ke simpul.

Titik akhir utama Redis OSS (mode cluster dinonaktifkan) terlihat seperti berikut ini. Ada perbedaan yang tergantung pada apakah enkripsi Bergerak aktif atau tidak.

Enkripsi bergerak tidak diaktifkan

```
clusterName.xxxxxx.nodeId.regionAndAz.cache.amazonaws.com:port

redis-01.7abc2d.0001.usw2.cache.amazonaws.com:6379
```

Enkripsi bergerak diaktifkan

```
master.clusterName.xxxxxx.regionAndAz.cache.amazonaws.com:port

master.ncit.ameaqx.use1.cache.amazonaws.com:6379
```

Untuk mengetahui lebih banyak cara menemukan titik akhir Anda, lihat topik yang relevan untuk jenis klaster dan mesin yang Anda jalankan.

- Menemukan titik akhir koneksi di ElastiCache

- Menemukan Titik Akhir untuk Cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan) (Konsol)— Anda memerlukan titik akhir Konfigurasi dari klaster.
- Menemukan Titik Akhir (AWS CLI)
- Menemukan Titik Akhir (ElastiCache API)

Connect ke cluster Valkey atau Redis OSS atau grup replikasi (Linux)

Sekarang Anda memiliki titik akhir yang Anda butuhkan, Anda dapat masuk ke sebuah EC2 instance dan terhubung ke cluster atau grup replikasi. Dalam contoh berikut, Anda menggunakan utilitas valkey-cli untuk terhubung ke cluster. Versi terbaru dari valkey-cli juga mendukung SSL/TLS untuk menghubungkan cluster yang diaktifkan. encryption/authentication

Contoh berikut menggunakan EC2 instans Amazon yang menjalankan Amazon Linux dan Amazon Linux 2. Untuk detail tentang menginstal dan mengkompilasi valkey-cli dengan distribusi Linux lainnya, lihat dokumentasi untuk sistem operasi spesifik Anda..

> **Note**
>
> Proses ini mencakup pengujian koneksi menggunakan utilitas valkey-cli hanya untuk penggunaan yang tidak direncanakan. Untuk daftar klien yang didukung, lihat dokumentasi Valkey. Untuk contoh menggunakan AWS SDKs with ElastiCache, lihatTutorial: Memulai dengan Python dan ElastiCache.

Menghubungkan ke klaster tanpa enkripsi dengan mode klaster dinonaktifkan

1. Jalankan perintah berikut untuk terhubung ke cluster *`primary-endpoint`* dan ganti dan *`port number`* dengan titik akhir cluster dan nomor port Anda. (Port default untuk Valkey dan Redis OSS adalah 6379.)

   ```
   src/valkey-cli -h primary-endpoint -p port number
   ```

   Hasil dalam command prompt terlihat mirip dengan berikut ini:

   ```
   primary-endpoint:port number
   ```

2. Anda sekarang dapat menjalankan perintah Valkey dan Redis OSS.

```
set x Hello
OK

get x
"Hello"
```

Menghubungkan ke klaster tanpa enkripsi dengan mode klaster diaktifkan

1. Jalankan perintah berikut untuk terhubung ke cluster *configuration-endpoint* dan ganti dan *port number* dengan titik akhir cluster dan nomor port Anda. (Port default untuk Valkey dan Redis OSS adalah 6379.)

   ```
   src/valkey-cli -h configuration-endpoint -c -p port number
   ```

   > **Note**
   >
   > Pada perintah sebelumnya, opsi -c memungkinkan mode klaster mengikuti pengalihan -ASK dan -MOVED.

   Hasil dalam command prompt terlihat mirip dengan berikut ini:

   ```
   configuration-endpoint:port number
   ```

2. Anda sekarang dapat menjalankan perintah Valkey dan Redis OSS. Perhatikan bahwa pengalihan terjadi karena Anda mengaktifkannya menggunakan opsi -c. Jika pengalihan tidak diaktifkan, perintah akan menghasilkan kesalahan MOVED. Untuk informasi selengkapnya tentang kesalahan MOVED, lihat spesifikasi cluster.

   ```
   set x Hi
   -> Redirected to slot [16287] located at 172.31.28.122:6379
   OK
   set y Hello
   OK
   get y
   "Hello"
   set z Bye
   -> Redirected to slot [8157] located at 172.31.9.201:6379
   ```

```
OK
get z
"Bye"
get x
-> Redirected to slot [16287] located at 172.31.28.122:6379
"Hi"
```

Menghubungkan ke cluster yang Encryption/Authentication diaktifkan

Secara default, valkey-cli menggunakan koneksi TCP yang tidak terenkripsi saat menghubungkan ke Valkey dan Redis OSS. Opsi ini `BUILD_TLS=yes` memungkinkan SSL/TLS pada saat kompilasi valkey-cli seperti yang ditunjukkan pada bagian sebelumnya. Unduh dan atur akses baris perintah Mengaktifkan AUTH bersifat opsional. Namun, Anda harus mengaktifkan enkripsi bergerak untuk mengaktifkan AUTH. Untuk detail selengkapnya tentang ElastiCache enkripsi dan otentikasi, lihatElastiCache enkripsi dalam transit (TLS).

> **Note**
>
> Anda dapat menggunakan opsi `--tls` dengan valkey-cli untuk terhubung ke mode cluster yang diaktifkan dan cluster terenkripsi yang dinonaktifkan. Jika token AUTH pada klaster telah diatur, maka Anda dapat menggunakan opsi `-a` untuk menyediakan kata sandi AUTH.

Dalam contoh berikut, pastikan untuk mengganti *`cluster-endpoint`* dan *`port number`* dengan titik akhir cluster dan nomor port Anda. (Port default untuk Redis OSS adalah 6379.)

Menghubungkan ke klaster terenkripsi dengan mode klaster dinonaktifkan

Contoh berikut menghubungkan ke klaster yang mengaktifkan enkripsi dan autentikasi:

```
src/valkey-cli -h cluster-endpoint --tls -a your-password -p port number
```

Contoh berikut terhubung ke klaster yang hanya mengaktifkan enkripsi:

```
src/valkey-cli -h cluster-endpoint --tls -p port number
```

Menghubungkan ke klaster terenkripsi dengan mode klaster diaktifkan

Contoh berikut menghubungkan ke klaster yang mengaktifkan enkripsi dan autentikasi:

```
src/valkey-cli -c -h cluster-endpoint --tls -a your-password -p port number
```

Contoh berikut terhubung ke klaster yang hanya mengaktifkan enkripsi:

```
src/valkey-cli -c -h cluster-endpoint --tls -p port number
```

Setelah Anda terhubung ke cluster, Anda dapat menjalankan perintah Valkey atau Redis OSS seperti yang ditunjukkan pada contoh sebelumnya untuk cluster yang tidak terenkripsi.

Alternatif untuk valkey-cli atau Redis-cli

Jika cluster tidak diaktifkan mode cluster dan Anda perlu membuat koneksi ke cluster untuk pengujian singkat tetapi tanpa melalui kompilasi valkey-cli atau redis-cli, Anda dapat menggunakan telnet atau openssl. Dalam contoh perintah berikut, pastikan untuk mengganti *cluster-endpoint* dan *port number* dengan titik akhir cluster dan nomor port Anda. (Port default untuk Redis OSS adalah 6379.)

Contoh berikut menghubungkan ke and/or otentikasi enkripsi diaktifkan mode cluster cluster dinonaktifkan:

```
openssl s_client -connect cluster-endpoint:port number
```

Jika kata sandi klaster telah ditetapkan, hubungkan ke klaster terlebih dahulu. Setelah terhubung, lakukan autentikasi pada klaster menggunakan perintah berikut, lalu tekan tombol `Enter`. Dalam contoh berikut, ganti *your-password* dengan kata sandi untuk cluster Anda.

```
Auth your-password
```

Contoh berikut menghubungkan ke klaster (mode klaster dinonaktifkan) yang tidak memiliki enkripsi atau autentikasi aktif:

```
telnet cluster-endpoint port number
```

Connect ke cluster Valkey atau Redis OSS atau grup replikasi (Windows)

Untuk terhubung ke cluster dari instance EC2 Windows menggunakan Valkey atau Redis OSS CLI, Anda harus mengunduh paket valkey-cli dan menggunakan valkey-cli.exe untuk terhubung ke cluster Valkey atau Redis OSS dari instance Windows. EC2

Dalam contoh berikut, Anda menggunakan utilitas valkey-cli untuk terhubung ke cluster yang tidak mengaktifkan enkripsi dan menjalankan Valkey atau Redis OSS. Untuk informasi lebih lanjut tentang Valkey dan perintah yang tersedia, lihat perintah Valkey di situs web Valkey.

Untuk terhubung ke cluster Valkey atau Redis OSS yang tidak diaktifkan enkripsi menggunakan valkey-cli

1. Connect ke EC2 instans Amazon Anda menggunakan utilitas koneksi pilihan Anda. Untuk petunjuk tentang cara menyambung ke EC2 instans Amazon, lihat Panduan EC2 Memulai Amazon.
2. Salin dan tempel tautan https://github.com/microsoftarchive/redis/releases/download/win-3.0.504/Redis-x64-3.0.504.zip di browser Internet untuk mengunduh file zip untuk klien Redis OSS dari rilis yang tersedia di GitHub https://github.com/microsoftarchive/redis/releases/tag/win-3.0.504

   Ekstrak file zip tersebut ke folder/jalur yang Anda inginkan.

   Buka Command Prompt dan ubah ke direktori Valkey dan jalankan perintah`c:\Valkey>valkey-cli -h Valkey_Cluster_Endpoint -p 6379`.

   Misalnya:

   ```
   c:\Valkey>valkey-cli -h cmd.xxxxxxx.ng.0001.usw2.cache.amazonaws.com -p 6379
   ```
3. Jalankan perintah Valkey atau Redis OSS.

   Anda sekarang terhubung ke cluster dan dapat menjalankan perintah Valkey atau Redis OSS seperti berikut ini.

   ```
   set a "hello"          // Set key "a" with a string value and no expiration
   OK
   get a                  // Get value for key "a"
   "hello"
   get b                  // Get value for key "b" results in miss
   (nil)
   set b "Good-bye" EX 5  // Set key "b" with a string value and a 5 second expiration
   "Good-bye"
   get b                  // Get value for key "b"
   "Good-bye"
                          // wait >= 5 seconds
   get b
   (nil)                  // key has expired, nothing returned
   ```

```
quit                    // Exit from valkey-cli
```

Menghapus klaster

Selama klaster dalam status tersedia, Anda akan dikenai biaya, terlepas dari apakah Anda secara aktif menggunakannya atau tidak. Untuk menghentikan biaya, hapus klaster tersebut.

⚠ Warning

- Saat Anda menghapus ElastiCache klaster, snapshot manual Anda dipertahankan. Anda juga dapat membuat snapshot akhir sebelum klaster dihapus. Snapshot cache otomatis tidak dipertahankan. Untuk informasi selengkapnya, lihat Melakukan snapshot dan pemulihan.
- `CreateSnapshot`izin diperlukan untuk membuat snapshot akhir. Tanpa izin ini, panggilan API akan gagal dengan `Access Denied` pengecualian.

Menggunakan AWS Management Console

Prosedur berikut menghapus satu klaster dari deployment Anda. Untuk menghapus beberapa klaster, ulangi prosedur untuk setiap klaster yang ingin dihapus. Anda tidak perlu menunggu satu klaster selesai dihapus sebelum memulai prosedur untuk menghapus klaster lain.

Untuk menghapus klaster

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di dasbor ElastiCache mesin, pilih Valkey atau Redis OSS.

   Daftar semua cache yang berjalan di mesin itu muncul.
3. Untuk memilih klaster yang akan dihapus, pilih nama klaster tersebut dari daftar klaster. Dalam kasus ini, nama klaster yang Anda buat di Langkah 2: Buat klaster.

> **Important**
>
> Anda hanya dapat menghapus satu cluster pada satu waktu dari ElastiCache konsol. Memilih beberapa klaster akan menonaktifkan operasi penghapusan.

4. Untuk Tindakan, pilih Hapus.
5. Di layar Konfirmasi Hapus Klaster, ketikkan nama klaster dan pilih Cadangan Akhir. Kemudian pilih Hapus untuk menghapus klaster, atau pilih Batal untuk mempertahankan klaster.

   Jika Anda memilih Hapus, status klaster berubah menjadi menghapus.

Segera setelah klaster Anda tidak lagi tercantum dalam daftar klaster, Anda berhenti dikenai biaya untuk klaster tersebut.

Menggunakan AWS CLI

Kode berikut menghapus klaster cache `my-cluster`. Dalam kasus ini, ganti `my-cluster` dengan nama klaster yang Anda buat di Langkah 2: Buat klaster.

```
aws elasticache delete-cache-cluster --cache-cluster-id my-cluster
```

Tindakan CLI `delete-cache-cluster` hanya menghapus satu klaster cache. Untuk menghapus beberapa klaster cache, panggil `delete-cache-cluster` untuk setiap klaster cache yang ingin dihapus. Anda tidak perlu menunggu satu klaster cache selesai dihapus untuk dapat menghapus yang lain.

Untuk Linux, macOS, atau Unix:

```
aws elasticache delete-cache-cluster \
    --cache-cluster-id my-cluster \
    --region us-east-2
```

Untuk Windows:

```
aws elasticache delete-cache-cluster ^
    --cache-cluster-id my-cluster ^
    --region us-east-2
```

Untuk informasi selengkapnya, lihat ElastiCache topik AWS CLI untuk `delete-cache-cluster`.

ElastiCache Tutorial dan video lainnya

Tutorial berikut membahas tugas-tugas yang menarik bagi ElastiCache pengguna Amazon.

- ElastiCache Video
- Tutorial: Mengonfigurasi Fungsi Lambda untuk Mengakses Amazon di ElastiCache VPC Amazon

ElastiCache Video

Berikut adalah video untuk membantu Anda mempelajari ElastiCache konsep Amazon dasar dan lanjutan. Untuk informasi tentang AWS Pelatihan, lihat AWS Pelatihan & Sertifikasi.

Topik

- Video Pengantar
- Video Lanjutan

Video Pengantar

Video berikut memperkenalkan Anda ke Amazon ElastiCache.

Topik

- AWS Re:invent 2020: Apa yang baru di Amazon ElastiCache
- AWS Re:invent 2019: Apa yang baru di Amazon ElastiCache
- AWS Re:invent 2017: Apa yang baru di Amazon ElastiCache
- DAT204—Membangun Aplikasi yang Dapat Diskalakan pada Layanan NoSQL (Re:Invent AWS 2015)
- DAT207 — Mempercepat Kinerja Aplikasi dengan Amazon ElastiCache (re: Invent 2013)AWS

AWS Re:invent 2020: Apa yang baru di Amazon ElastiCache

AWS Re:invent 2020: Apa yang baru di Amazon ElastiCache

AWS Re:invent 2019: Apa yang baru di Amazon ElastiCache

AWS Re:invent 2019: Apa yang baru di Amazon ElastiCache

AWS Re:invent 2017: Apa yang baru di Amazon ElastiCache

AWS Re:invent 2017: Apa yang baru di Amazon ElastiCache

DAT204—Membangun Aplikasi yang Dapat Diskalakan pada Layanan NoSQL (Re:Invent AWS 2015)

Dalam sesi ini, kami membahas manfaat database NoSQL dan mengikuti tur layanan NoSQL utama yang ditawarkan oleh —Amazon DynamoDB dan Amazon. AWS ElastiCache Kemudian,

kami mendengar dari dua pelanggan terkemuka, Expedia dan Mapbox, tentang kasus penggunaan dan tantangan arsitektur mereka, dan bagaimana mereka mengatasinya menggunakan layanan NoSQL, termasuk AWS pola desain dan praktik terbaik. Setelah melalui sesi ini, Anda akan memiliki pemahaman yang lebih baik tentang NoSQL dan kemampuan NoSQL yang andal untuk bersiap mengatasi tantangan basis data Anda dengan percaya diri.

DAT204—Membangun Aplikasi yang Dapat Diskalakan pada Layanan NoSQL (Re:Invent AWS 2015)

DAT207 — Mempercepat Kinerja Aplikasi dengan Amazon ElastiCache (re: Invent 2013)AWS

Dalam video ini, pelajari bagaimana Anda dapat menggunakan Amazon ElastiCache untuk dengan mudah menerapkan sistem caching dalam memori untuk mempercepat kinerja aplikasi Anda. Kami menunjukkan kepada Anda cara menggunakan Amazon ElastiCache untuk meningkatkan latensi aplikasi Anda dan mengurangi beban pada server database Anda. Kami juga akan menunjukkan cara membuat sebuah lapisan caching yang mudah dikelola dan diskalakan seiring pertumbuhan aplikasi Anda. Selama sesi ini, kami membahas berbagai skenario dan kasus penggunaan yang dapat memperoleh manfaat dengan mengaktifkan caching, dan mendiskusikan fitur yang disediakan oleh Amazon. ElastiCache

DAT207 - Mempercepat Kinerja Aplikasi dengan Amazon ElastiCache (re: Invent 2013)

Video Lanjutan

Video berikut mencakup ElastiCache topik Amazon yang lebih canggih.

Topik

- Desain untuk sukses dengan praktik ElastiCache terbaik Amazon (re: Invent 2020)
- Tingkatkan aplikasi real-time Anda dengan Amazon ElastiCache (re:Invent 2019)
- Praktik terbaik: memigrasi kluster Redis OSS dari EC2 Amazon ElastiCache ke (re:Invent 2019)
- Menskalakan Platform Olahraga Fantasi dengan Amazon ElastiCache & Amazon STP11 Aurora (re: Invent 2018)
- Redis OSS yang Andal & Dapat Diskalakan di Cloud dengan Amazon ElastiCache (re: Invent 2018)
- ElastiCache Deep Dive: Pola Desain untuk Penyimpanan Data Dalam Memori (re: Invent 2018)
- DAT305—Amazon ElastiCache Deep Dive (Re:Invent 2017)
- DAT306—Amazon ElastiCache Deep Dive (re:Invent 2016)
- DAT317—Bagaimana IFTTT menggunakan Redis OSS ElastiCache untuk Memprediksi Acara (re: Invent 2016)

- DAT407—Amazon ElastiCache Deep Dive (re:Invent 2015)
- SDD402—Amazon ElastiCache Deep Dive (Re:Invent 2014)
- DAT307—Menyelam Jauh ke dalam ElastiCache Arsitektur dan Pola Desain Amazon (re: Invent 2013)

Desain untuk sukses dengan praktik ElastiCache terbaik Amazon (re: Invent 2020)

Dengan pertumbuhan eksplosif aplikasi bisnis yang kritis dan real-time yang dibangun di atas Redis OSS, ketersediaan, skalabilitas, dan keamanan telah menjadi pertimbangan utama. Pelajari praktik terbaik untuk menyiapkan Amazon ElastiCache agar berhasil dengan penskalaan online, ketersediaan tinggi di seluruh penerapan multi-AZ, dan konfigurasi keamanan.

Desain untuk sukses dengan praktik ElastiCache terbaik Amazon (re: Invent 2020)

Tingkatkan aplikasi real-time Anda dengan Amazon ElastiCache (re:Invent 2019)

Dengan pertumbuhan pesat dalam adopsi cloud dan skenario baru yang diberdayakannya, aplikasi memerlukan latensi mikrodetik dan throughput yang tinggi untuk mendukung jutaan permintaan per detik. Developer biasanya mengandalkan perangkat keras dan solusi khusus, seperti basis data berbasis disk yang dikombinasikan dengan teknik pengurangan data, untuk mengelola data untuk aplikasi waktu nyata. Pendekatan ini dapat menghabiskan banyak biaya dan tidak dapat diskalakan. Pelajari cara meningkatkan kinerja aplikasi real-time dengan menggunakan Amazon dalam memori yang dikelola sepenuhnya ElastiCache untuk kinerja ekstrem, skalabilitas tinggi, ketersediaan, dan keamanan.

Tingkatkan aplikasi real-time Anda dengan Amazon ElastiCache (re:Invent 2019:)

Praktik terbaik: memigrasi kluster Redis OSS dari EC2 Amazon ElastiCache ke (re:Invent 2019)

Mengelola cluster Redis OSS sendiri bisa jadi sulit. Anda harus menyediakan perangkat keras, patch perangkat lunak, melakukan cadangan data, dan memantau beban kerja secara konstan. Dengan fitur Migrasi Online yang baru dirilis untuk Amazon ElastiCache, kini Anda dapat dengan mudah memindahkan data Anda dari Redis OSS yang dihosting sendiri di Amazon EC2 ke Amazon yang dikelola sepenuhnya ElastiCache, dengan mode cluster dinonaktifkan. Dalam sesi ini, Anda mempelajari tentang alat Migrasi Online yang baru, melihat demo, dan, yang lebih penting, mempelajari praktik terbaik langsung untuk kelancaran migrasi ke Amazon. ElastiCache

Praktik terbaik: memigrasi kluster Redis OSS dari EC2 Amazon ElastiCache ke (re:Invent 2019)

Menskalakan Platform Olahraga Fantasi dengan Amazon ElastiCache & Amazon STP11 Aurora (re: Invent 2018)

Dream11 adalah perusahaan rintisan teknologi olahraga terkemuka di India. Perusahaan ini memiliki basis pengguna yang berkembang dari 40 juta lebih yang bermain beberapa olahraga, termasuk kriket, sepak bola, dan bola basket fantasi, dan saat ini melayani satu juta pengguna secara serempak, yang menghasilkan tiga juta permintaan per menit dengan waktu respon di bawah 50 milidetik. Dalam pembicaraan ini, CTO Dream11 Amit Sharma menjelaskan bagaimana perusahaan menggunakan Amazon Aurora dan Amazon ElastiCache untuk menangani lalu lintas flash, yang dapat tiga kali lipat dalam jendela respons 30 detik. Sharma juga berbicara tentang penskalaan transaksi tanpa penguncian, dan dia berbagi langkah-langkah untuk menangani lalu lintas flash - hingga melayani lima juta pengguna aktif setiap hari. Judul Lengkap: AWS re: Invent 2018: Menskalakan Platform Olahraga Fantasi dengan Amazon & Amazon ElastiCache Aurora () STP11

Menskalakan Platform Olahraga Fantasi dengan Amazon ElastiCache & Amazon STP11 Aurora (re: Invent 2018)

Redis OSS yang Andal & Dapat Diskalakan di Cloud dengan Amazon ElastiCache (re: Invent 2018)

Sesi ini mencakup fitur dan penyempurnaan dalam layanan kami yang kompatibel dengan Redis OSS, Amazon ElastiCache untuk Redis OSS. Kami mencakup fitur-fitur utama, seperti Redis OSS 5, peningkatan skalabilitas dan kinerja, keamanan dan kepatuhan, dan banyak lagi. Kami juga membahas fitur yang akan datang dan studi kasus pelanggan.

Redis OSS yang Andal & Dapat Diskalakan di Cloud dengan Amazon ElastiCache (re: Invent 2018)

ElastiCache Deep Dive: Pola Desain untuk Penyimpanan Data Dalam Memori (re: Invent 2018)

Dalam sesi ini, kami memberikan intip di balik layar untuk mempelajari tentang desain dan arsitektur Amazon ElastiCache. Lihat pola desain umum dengan penawaran Redis OSS dan Memcached kami dan bagaimana pelanggan menggunakannya untuk pemrosesan data dalam memori guna mengurangi latensi dan meningkatkan throughput aplikasi. Kami meninjau praktik ElastiCache terbaik, pola desain, dan anti-pola.

ElastiCache Deep Dive: Pola Desain untuk Penyimpanan Data Dalam Memori (re: Invent 2018)

DAT305—Amazon ElastiCache Deep Dive (Re:Invent 2017)

Lihat di balik layar untuk mempelajari tentang desain dan arsitektur Amazon ElastiCache. Lihat pola desain umum dengan penawaran Memcached dan Redis OSS kami dan bagaimana pelanggan

menggunakannya untuk operasi dalam memori guna mengurangi latensi dan meningkatkan throughput aplikasi. Selama video ini, kami meninjau praktik ElastiCache terbaik, pola desain, dan anti-pola.

Video ini memperkenalkan hal berikut:

- ElastiCache untuk resharding online Redis OSS
- ElastiCache keamanan dan enkripsi
- ElastiCache untuk Redis OSS versi 3.2.10

DAT305—Amazon ElastiCache Deep Dive (Re:Invent 2017)

DAT306—Amazon ElastiCache Deep Dive (re:Invent 2016)

Lihat di balik layar untuk mempelajari tentang desain dan arsitektur Amazon ElastiCache. Lihat pola desain umum dengan penawaran Memcached dan Redis OSS kami dan bagaimana pelanggan menggunakannya untuk operasi dalam memori guna mengurangi latensi dan meningkatkan throughput aplikasi. Selama sesi ini, kami meninjau praktik ElastiCache terbaik, pola desain, dan anti-pola.

DAT306—Amazon ElastiCache Deep Dive (re:Invent 2016)

DAT317—Bagaimana IFTTT menggunakan Redis OSS ElastiCache untuk Memprediksi Acara (re:Invent 2016)

IFTTT adalah layanan gratis yang memberdayakan orang untuk melakukan lebih banyak hal dengan layanan yang mereka sukai, mulai dari automasi tugas sederhana hingga mengubah cara seseorang berinteraksi dan mengendalikan rumah mereka. IFTTT menggunakan Redis OSS ElastiCache untuk menyimpan riwayat transaksi dan menjadwalkan prediksi serta indeks untuk dokumen log di Amazon S3. Lihat sesi ini untuk mempelajari bagaimana kekuatan skrip Lua dan tipe data Redis OSS memungkinkan orang untuk mencapai sesuatu yang tidak dapat mereka lakukan di tempat lain.

DAT317—Bagaimana IFTTT menggunakan Redis OSS ElastiCache untuk Memprediksi Acara (re:Invent 2016)

DAT407—Amazon ElastiCache Deep Dive (re:Invent 2015)

Intip di balik layar untuk belajar tentang desain dan arsitektur Amazon ElastiCache. Lihat pola desain umum dari penawaran Memcached dan Redis OSS kami dan bagaimana pelanggan menggunakannya untuk operasi dalam memori dan mencapai peningkatan latensi dan throughput

untuk aplikasi. Selama sesi ini, kami meninjau praktik terbaik, pola desain, dan anti-pola yang terkait dengan Amazon ElastiCache.

DAT407—Amazon ElastiCache Deep Dive (re:Invent 2015)

SDD402—Amazon ElastiCache Deep Dive (Re:Invent 2014)

Dalam video ini, kami memeriksa kasus penggunaan caching umum, untuk mesin Memcached dan ElastiCache ElastiCache untuk Redis OSS, pola yang membantu Anda menentukan mesin mana yang lebih baik untuk kebutuhan Anda, hashing yang konsisten, dan lainnya sebagai sarana untuk membangun aplikasi yang cepat dan dapat diskalakan. Frank Wiebe, Ilmuwan Utama di Adobe, merinci bagaimana Adobe menggunakan Amazon ElastiCache untuk meningkatkan pengalaman pelanggan dan meningkatkan skala bisnis mereka.

DAT402—Amazon ElastiCache Deep Dive (Re:Invent 2014)

DAT307—Menyelam Jauh ke dalam ElastiCache Arsitektur dan Pola Desain Amazon (re: Invent 2013)

Dalam video ini, kita memeriksa caching, strategi caching, penskalaan ke luar, dan pemantauan. Kami juga membandingkan ElastiCache untuk Memcached dan ElastiCache untuk mesin Redis OSS. Selama sesi ini, kami juga meninjau praktik terbaik dan pola desain yang terkait dengan Amazon ElastiCache.

DAT307 - Menyelam Jauh ke dalam ElastiCache Arsitektur dan Pola Desain Amazon (re AWS : Invent 2013).

Mengelola node di ElastiCache

Node adalah blok bangunan terkecil dari ElastiCache penyebaran Amazon. Simpul adalah potongan RAM berukuran tetap yang aman dan terpasang ke jaringan. Setiap simpul menjalankan mesin yang dipilih ketika klaster dibuat atau terakhir diubah. Setiap simpul mempunyai nama dan port Layanan Nama Domain (DNS) sendiri. Beberapa jenis ElastiCache node didukung, masing-masing dengan berbagai jumlah memori terkait dan daya komputasi.

Untuk pembahasan yang lebih terperinci tentang ukuran simpul yang akan digunakan, lihat Memilih ukuran simpul Anda.

Secara umum, karena dukungannya untuk sharding, penyebaran Valkey atau Redis OSS (mode cluster enabled) memiliki sejumlah node yang lebih kecil. Sebaliknya, penyebaran Valkey atau Redis

OSS (mode cluster dinonaktifkan) memiliki lebih sedikit node yang lebih besar dalam sebuah cluster. Untuk pembahasan yang lebih terperinci tentang ukuran simpul yang akan digunakan, lihat Memilih ukuran simpul Anda.

Topik

- Melihat Status ElastiCache Node
- Node dan pecahan Valkey atau Redis OSS
- Menghubungkan ke simpul
- Jenis simpul yang didukung
- Mem-boot ulang node
- Mengganti node (Valkey dan Redis OSS)
- Mengganti node (Memcached)
- Simpul terpesan
- Memigrasikan simpul generasi sebelumnya

Beberapa operasi penting yang berkaitan dengan simpul adalah sebagai berikut:

- Menambahkan node ke ElastiCache cluster
- Menghapus node dari ElastiCache cluster
- Penskalaan ElastiCache
- Menemukan titik akhir koneksi di ElastiCache
- Secara otomatis mengidentifikasi node di cluster Anda (Memcached)

Melihat Status ElastiCache Node

Menggunakan ElastiCache konsol, Anda dapat dengan cepat mengakses status ElastiCache node Anda. Status ElastiCache node menunjukkan kesehatan node. Anda dapat menggunakan prosedur berikut untuk melihat status ElastiCache node di ElastiCache konsol Amazon, AWS CLI perintah, atau operasi API.

Nilai status yang mungkin untuk ElastiCache node ada di tabel berikut. Tabel ini juga menunjukkan apakah Anda akan ditagih untuk ElastiCache node.

Jenis	Ditagih	Deskripsi
`available`	Ditagih	ElastiCache Node sehat dan tersedia.
`creating`	Tidak ditagih	ElastiCache Node sedang dibuat. Simpul tidak dapat diakses saat sedang dibuat.
`deleting`	Tidak ditagih	ElastiCache Node sedang dihapus.
`modifying`	Ditagih	ElastiCache Node sedang dimodifikasi karena permintaan pelanggan untuk memodifikasi node.
`updating`	Ditagih	Status Memperbarui menunjukkan satu atau beberapa hal berikut ini benar dari ElastiCache simpul Amazon: • ElastiCache Node sedang ditambal sebagai bagian dari udpate layanan. Untuk informasi selengkapnya tentang pembaruan layanan, lihat Halaman Bantuan Pemeliharaan ElastiCache Terkelola Amazon dan Pembaruan Layanan. • Grup keamanan VPC memperbarui untuk Cluster. ElastiCache

Jenis	Ditagih	Deskripsi
		• ElastiCache Cluster sedang ditingkatkan atau diperkecil. • Konfigurasi pengiriman log sedang dimodifikasi untuk ElastiCache Cluster. • Operasi penghapusan untuk ElastiCache node sedang tertunda. • Kata ElastiCache sandi sedang updated/rotated digunakan AWS Secrets Manager.
`rebooting cache cluster nodes`	Ditagih	ElastiCache Node sedang di-boot ulang karena permintaan pelanggan atau ElastiCache proses Amazon yang memerlukan reboot node.

Jenis	Ditagih	Deskripsi
`incompatible_parameters`	Tidak ditagih	Amazon tidak ElastiCache dapat memulai node karena parameter yang ditentukan dalam grup parameter node tidak kompatibel dengan node. Kembalikan perubahan parameter atau jadikan parameter ini kompatibel dengan simpul untuk mendapatkan akses ke simpul Anda. Untuk informasi selengkapnya tentang parameter yang tidak kompatibel, periksa daftar Peristiwa untuk ElastiCache node.
`incompatible_network`	Tidak ditagih	Status jaringan yang tidak kompatibel menunjukkan satu atau beberapa hal berikut berlaku untuk node Amazon: ElastiCache • Tidak ada alamat IP yang tersedia di subnet tempat ElastiCache node diluncurkan. • Subnet yang disebutkan dalam grup ElastiCache subnet tidak lagi ada di Amazon Virtual Private Cloud (Amazon VPC).

Jenis	Ditagih	Deskripsi
`restore_failed`	Tidak ditagih	Status gagal pemulihan menunjukkan salah satu dari berikut ini benar dari simpul Amazon: ElastiCache • Penggantian simpul gagal karena kapasitas instans tidak mencukupi berulang kali. Ini biasanya terjadi ketika menjalankan node generasi sebelumnya yang end-of-life. Namun, itu juga bisa terjadi dengan penggantian node generasi saat ini ketika AWS tidak memiliki kapasitas sesuai permintaan yang cukup untuk memenuhi permintaan Anda di Availability Zone yang ditentukan. Untuk informasi lebih lanjut tentang memperbaiki atau menghapus node ini, lihatMemigrasikan simpul generasi sebelumnya. • Snapshot RDB yang ditentukan gagal dipulihkan. • AWS Akun untuk ElastiCache cluster telah ditangguhkan. • Node gagal dan tidak dapat dipulihkan.

Jenis	Ditagih	Deskripsi
`snapshotting`	Ditagih	ElastiCache sedang membuat snapshot dari node Valkey atau Redis OSS.

Melihat Status ElastiCache Node dengan konsol

Untuk melihat status ElastiCache Node dengan konsol:

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Redis OSS Clusters atau Memcached Clusters. Halaman Cache muncul dengan daftar ElastiCache Node. Untuk setiap simpul, nilai status akan ditampilkan.
3. Anda kemudian dapat menavigasi ke tab Pembaruan Layanan untuk cache untuk menampilkan daftar Pembaruan layanan yang berlaku untuk cache.

Melihat Status ElastiCache Node dengan AWS CLI

Untuk melihat ElastiCache node dan informasi statusnya dengan menggunakan AWS CLI, gunakan `describe-cache-cluster` perintah. Misalnya, AWS CLI perintah berikut menampilkan setiap ElastiCache node.

```
aws elasticache describe-cache-clusters
```

Melihat Status ElastiCache Node melalui API

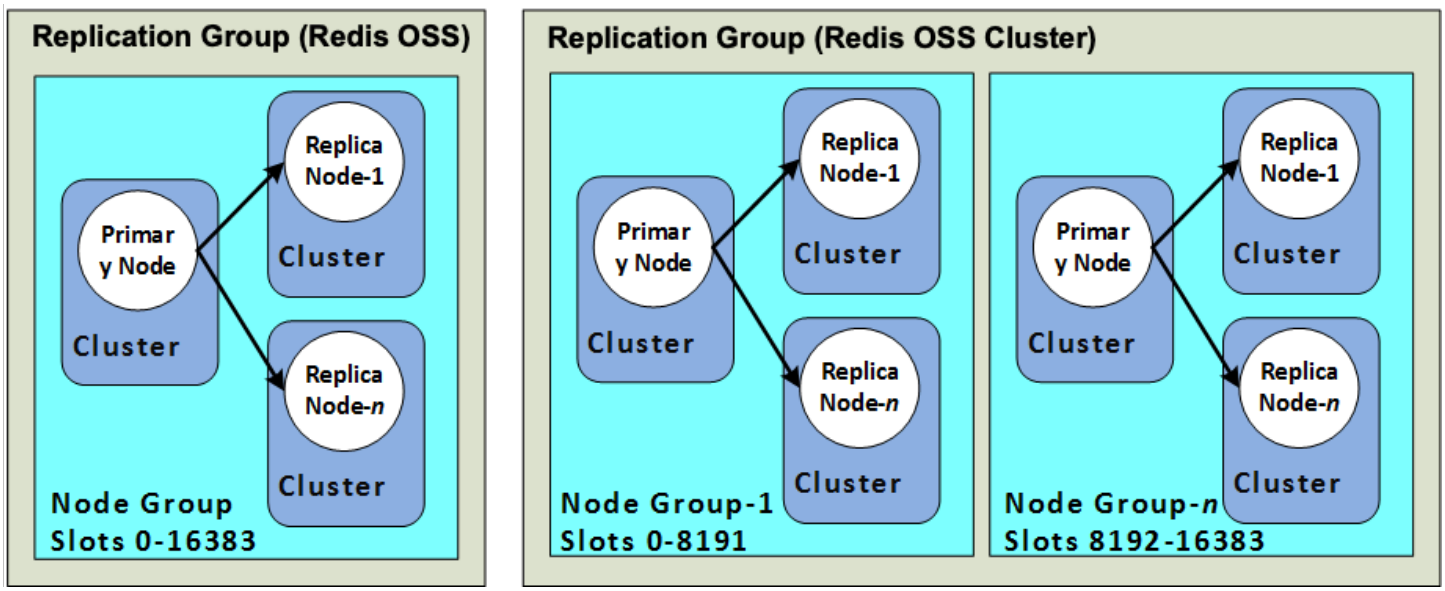
Untuk melihat status ElastiCache node menggunakan Amazon ElastiCache API, panggil `ShowCacheNodeInfo` flag `DescribeCacheClusteroperation` with untuk mengambil informasi tentang node cache individual.

Node dan pecahan Valkey atau Redis OSS

Serpihan (dalam API dan CLI, grup simpul) adalah susunan hierarkis simpul, yang masing-masing di-wrapping dalam klaster. Serpihan mendukung replikasi. Dalam pecahan, satu node berfungsi sebagai simpul read/write utama. Semua simpul lain dalam serpihan berfungsi sebagai replika hanya-baca dari simpul primer. Valkey, atau Redis OSS versi 3.2 dan yang lebih baru, mendukung beberapa

pecahan dalam sebuah cluster (di API dan CLI, grup replikasi). Dukungan ini memungkinkan partisi data Anda dalam cluster Valkey atau Redis OSS (mode cluster enabled).

Diagram berikut menggambarkan perbedaan antara cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dan cluster Valkey atau Redis OSS (mode cluster diaktifkan).



Cluster Valkey atau Redis OSS (mode cluster diaktifkan) mendukung replikasi melalui pecahan. Operasi API DescribeReplicationGroups(CLI: describe-replication-groups) mencantumkan grup node dengan node anggota, peran node dalam grup node, dan juga informasi lainnya.

Saat Anda membuat klaster Valkey atau Redis OSS, Anda menentukan apakah Anda ingin membuat cluster dengan pengelompokan diaktifkan. Cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) tidak pernah memiliki lebih dari satu pecahan, yang dapat diskalakan secara horizontal dengan menambahkan (hingga total lima) atau menghapus node replika baca. Untuk informasi selengkapnya, lihat Ketersediaan tinggi menggunakan grup replikasi, Menambahkan replika baca untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan), atau Menghapus replika baca untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan). Cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) juga dapat menskalakan secara vertikal dengan mengubah jenis node. Untuk informasi selengkapnya, lihat Menskalakan node replika untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan).

Batas node atau shard dapat ditingkatkan hingga maksimum 500 per cluster jika mesinnya Valkey atau Redis OSS versi 5.0.6 atau lebih tinggi. Sebagai contoh, Anda dapat memilih untuk mengonfigurasi sebuah klaster dengan 500 simpul yang berkisar antara 83 serpihan (satu primer dan 5 replika per serpihan) dan 500 serpihan (satu primer dan tanpa replika). Pastikan alamat IP yang tersedia mencukupi untuk mengakomodasi peningkatan tersebut. Kesalahan umumnya termasuk

subnet dalam grup subnet memiliki rentang CIDR yang terlalu kecil atau subnet dibagikan dan banyak digunakan oleh klaster lainnya. Untuk informasi selengkapnya, lihat Membuat grup subnet.

Untuk versi di bawah 5.0.6, batasnya adalah 250 per klaster.

Untuk meminta penambahan batas, lihat Batas Layanan AWS dan pilih jenis batas Simpul per klaster per jenis instans.

Setelah cluster Valkey atau Redis OSS (mode cluster enabled) dibuat, itu dapat diubah (diskalakan masuk atau keluar). Untuk informasi selengkapnya, lihat Penskalaan ElastiCache dan Mengganti node (Valkey dan Redis OSS).

Ketika Anda membuat klaster baru, Anda dapat melakukan seeding klaster tersebut dengan data dari klaster lama sehingga klaster baru tidak dimulai dari kosong. Pendekatan ini hanya berfungsi jika grup klaster memiliki jumlah serpihan yang sama dengan klaster lama. Melakukan hal ini dapat membantu jika Anda perlu mengubah jenis simpul atau versi mesin. Lihat informasi yang lebih lengkap di Membuat cadangan manual dan Melakukan pemulihan dari cadangan ke dalam cache baru.

Menghubungkan ke simpul

Menghubungkan ke node Valkey atau Redis OSS

Sebelum mencoba terhubung ke node Valkey atau Redis OSS di cluster Anda, Anda harus memiliki titik akhir untuk node. Untuk menemukan titik akhir, lihat yang berikut ini:

- Menemukan Titik Akhir Cluster Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol)
- Menemukan Titik Akhir untuk Cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan) (Konsol)
- Menemukan Titik Akhir (AWS CLI)
- Menemukan Titik Akhir (ElastiCache API)

Dalam contoh berikut, Anda menggunakan utilitas valkey-cli untuk terhubung ke cluster yang menjalankan Valkey atau Redis OSS.

> **Note**
>
> Untuk informasi selengkapnya tentang perintah yang tersedia, lihat halaman web Perintah.

Untuk terhubung ke cluster Valkey atau Redis OSS menggunakan valkey-cli

1. Connect ke EC2 instans Amazon Anda menggunakan utilitas koneksi pilihan Anda.

   > **Note**
   >
   > Untuk petunjuk tentang cara menyambung ke EC2 instans Amazon, lihat Panduan EC2 Memulai Amazon.

2. Untuk membangun `valkey-cli`, unduh dan instal GNU Compiler Collection (`gcc`). Pada prompt perintah EC2 instance Anda, masukkan perintah berikut dan masukkan `y` pada prompt konfirmasi.

   ```
   sudo yum install gcc
   ```

   Muncul output seperti yang berikut ini.

   ```
   Loaded plugins: priorities, security, update-motd, upgrade-helper
   ```

```
Setting up Install Process
Resolving Dependencies
--> Running transaction check

...(output omitted)...

Total download size: 27 M
Installed size: 53 M
Is this ok [y/N]: y
Downloading Packages:
(1/11): binutils-2.22.52.0.1-10.36.amzn1.x86_64.rpm      | 5.2 MB     00:00
(2/11): cpp46-4.6.3-2.67.amzn1.x86_64.rpm                | 4.8 MB     00:00
(3/11): gcc-4.6.3-3.10.amzn1.noarch.rpm                  | 2.8 kB     00:00

...(output omitted)...

Complete!
```

3. Unduh dan kompilasi utilitas valkey-cli. Utilitas ini termasuk dalam distribusi perangkat lunak Valkey. Pada prompt perintah EC2 instance Anda, ketik perintah berikut:

> **Note**
>
> Untuk sistem Ubuntu, sebelum menjalankan `make`, jalankan `make distclean`.

```
wget https://github.com/valkey-io/valkey/archive/refs/tags/8.0.0.tar.gz
tar xvzf valkey-8.0.0.tar.gz
cd valkey-8.0.0
make distclean      # ubuntu systems only
make
```

4. Pada prompt perintah EC2 instance Anda, ketik perintah berikut.

```
src/valkey-cli -c -h mycachecluster.eaogs8.0001.usw2.cache.amazonaws.com -p 6379
```

Prompt perintah Valkey atau Redis OSS yang mirip dengan yang berikut ini muncul.

```
redis mycachecluster.eaogs8.0001.usw2.cache.amazonaws.com 6379>
```

5. Uji koneksi dengan menjalankan perintah Valkey atau Redis OSS.

   Anda sekarang terhubung ke cluster dan dapat menjalankan perintah Valkey atau Redis OSS. Berikut ini adalah beberapa contoh perintah dengan tanggapan Valkey atau Redis OSS mereka.

```
set a "hello"          // Set key "a" with a string value and no expiration
OK
get a                  // Get value for key "a"
"hello"
get b                  // Get value for key "b" results in miss
(nil)
set b "Good-bye" EX 5  // Set key "b" with a string value and a 5 second expiration
get b
"Good-bye"
                   // wait 5 seconds
get b
(nil)                  // key has expired, nothing returned
quit                   // Exit from valkey-cli
```

Untuk terhubung ke simpul atau klaster yang memiliki enkripsi Lapisan Soket Aman (SSL) (enkripsi bergerak aktif), lihat ElastiCache enkripsi dalam transit (TLS).

Menghubungkan ke node Memcached

Sebelum mencoba terhubung ke klaster Memcached, Anda harus memiliki titik akhir untuk simpul. Untuk menemukan titik akhir, lihat yang berikut ini:

- Menemukan Titik Akhir Cluster (Konsol) (Memcached)
- Menemukan Titik Akhir (AWS CLI)
- Menemukan Titik Akhir (ElastiCache API)

Dalam contoh berikut, Anda menggunakan utilitas telnet untuk terhubung ke simpul yang menjalankan Memcached.

> **Note**
>
> Untuk informasi selengkapnya tentang Memcached dan perintah Memcached yang tersedia, lihat situs web Memcached.

Untuk terhubung ke simpul menggunakan telnet

1. Connect ke EC2 instans Amazon Anda dengan menggunakan utilitas koneksi pilihan Anda.

   > **Note**
   >
   > Untuk petunjuk tentang cara menyambung ke EC2 instans Amazon, lihat Panduan EC2 Memulai Amazon.

2. Unduh dan instal utilitas telnet di EC2 instans Amazon Anda. Pada prompt perintah EC2 instance Amazon Anda, ketik perintah berikut dan ketik y pada prompt perintah.

   ```
   sudo yum install telnet
   ```

   Muncul output seperti yang berikut ini.

   ```


   Loaded plugins: priorities, security, update-motd, upgrade-helper
   Setting up Install Process
   Resolving Dependencies
   --> Running transaction check

   ...(output omitted)...

   Total download size: 63 k
   Installed size: 109 k
   Is this ok [y/N]: y
   Downloading Packages:
   telnet-0.17-47.7.amzn1.x86_64.rpm                          |  63 kB     00:00

   ...(output omitted)...

   Complete!
   ```

3. Pada prompt perintah EC2 instance Amazon Anda, ketik perintah berikut, ganti titik akhir node Anda dengan yang ditunjukkan dalam contoh ini.

   ```
   telnet mycachecluster.eaogs8.0001.usw2.cache.amazonaws.com 11211
   ```

   Muncul output seperti yang berikut ini.

```
Trying 128.0.0.1...
Connected to mycachecluster.eaogs8.0001.usw2.cache.amazonaws.com.
Escape character is '^]'.
>
```

4. Uji koneksi dengan menjalankan perintah Memcached.

   Anda sekarang terhubung ke simpul, dan Anda dapat menjalankan perintah Memcached. Berikut adalah sebuah contoh.

```
set a 0 0 5      // Set key "a" with no expiration and 5 byte value
hello            // Set value as "hello"
STORED
get a            // Get value for key "a"
VALUE a 0 5
hello
END
get b            // Get value for key "b" results in miss
END
>
```

Jenis simpul yang didukung

ElastiCache mendukung jenis node berikut. Secara umum, jenis generasi saat ini memberikan lebih banyak memori dan daya komputasi dengan biaya lebih rendah dibandingkan dengan jenis generasi sebelumnya yang setara.

Untuk informasi selengkapnya tentang detail performa untuk setiap jenis node, lihat Jenis EC2 Instance Amazon.

> **Note**
>
> Jenis contoh berikut didukung di Wilayah AWS Asia Pasifik (Thailand) dan Meksiko (Tengah):
>
> - m7g/r7g: besar, xl, 2xl, 4xl, 8xl, 12xl, dan 16xl.
> - t3/t4g: mikro, kecil, dan sedang.

Untuk informasi tentang ukuran simpul yang akan digunakan, lihat Memilih ukuran simpul Anda.

Topik

Generasi Saat Ini (Memcached)

Tabel berikut menunjukkan bandwidth dasar dan burst untuk jenis contoh yang menggunakan mekanisme I/O kredit jaringan untuk meledak melampaui bandwidth dasar mereka.

> **Note**
>
> Jenis instans dengan kinerja jaringan yang dapat meledak menggunakan mekanisme I/O kredit jaringan untuk melampaui bandwidth dasar mereka dengan upaya terbaik.

Umum

Jenis instans	Versi Memcached minimum yang didukung	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.m7g.large		0,937	12,5
cache.m7g.xlarge		1.876	12,5
cache.m7g.2xlarge		3,75	15
cache.m7g.4xlarge		7.5	15
cache.m7g.8xlarge		15	N/A
cache.m7g.12xlarge		22.5	N/A
cache.m7g.16xlarge		30	N/A
cache.m6g.large	1.5.16	0,75	10.0
cache.m6g.xlarge	1.5.16	1,25	10.0
cache.m6g.2xlarge	1.5.16	2.5	10.0
cache.m6g.4xlarge	1.5.16	5.0	10.0
cache.m6g.8xlarge	1.5.16	12	N/A
cache.m6g.12xlarge	1.5.16	20	N/A
cache.m6g.16xlarge	1.5.16	25	N/A
cache.m5.large	1.5.16	0,75	10.0
cache.m5.xlarge	1.5.16	1,25	10.0
cache.m5.2xlarge	1.5.16	2.5	10.0
cache.m5.4xlarge	1.5.16	5.0	10.0
cache.m5.12xlarge	1.5.16	N/A	N/A

Jenis instans	Versi Memcached minimum yang didukung	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.m5.24xlarge	1.5.16	N/A	N/A
cache.m4.large	1.5.16	0,45	1.2
cache.m4.xlarge	1.5.16	0,75	2.8
cache.m4.2xlarge	1.5.16	1.0	10.0
cache.m4.4xlarge	1.5.16	2.0	10.0
cache.m4.10xlarge	1.5.16	5.0	10.0
cache.t4g.micro	1.5.16	0,064	5.0
cache.t4g.small	1.5.16	0,18	5.0
cache.t4g.medium	1.5.16	0,256	5.0
cache.t3.micro	1.5.16	0,064	5.0
cache.t3.small	1.5.16	0,18	5.0
cache.t3.medium	1.5.16	0,256	5.0
cache.t2.micro	1.5.16	0,064	1.024
cache.t2.small	1.5.16	0,18	1.024
cache.t2.medium	1.5.16	0,256	1.024

Memori dioptimalkan untuk Memcached

Jenis instans	Versi minimum yang didukung	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.r7g.large		0,937	12,5

Jenis instans	Versi minimum yang didukung	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.r7g.xlarge		1.876	12,5
cache.r7g.2xlarge		3,75	15
cache.r7g.4xlarge		7,5	15
cache.r7g.8xlarge		15	N/A
cache.r7g.12xlarge		22.5	N/A
cache.r7g.16xlarge		30	N/A
cache.r6g.large	1.5.16	0,75	10.0
cache.r6g.xlarge	1.5.16	1,25	10.0
cache.r6g.2xlarge	1.5.16	2.5	10.0
cache.r6g.4xlarge	1.5.16	5.0	10.0
cache.r6g.8xlarge	1.5.16	12	N/A
cache.r6g.12xlarge	1.5.16	20	N/A
cache.r6g.16xlarge	1.5.16	25	N/A
cache.r5.large	1.5.16	0,75	10.0
cache.r5.xlarge	1.5.16	1,25	10.0
cache.r5.2xlarge	1.5.16	2.5	10.0
cache.r5.4xlarge	1.5.16	5.0	10.0
cache.r5.12xlarge	1.5.16	20	N/A
cache.r5.24xlarge	1.5.16	25	N/A
cache.r4.large	1.5.16	0,75	10.0

Jenis instans	Versi minimum yang didukung	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.r4.xlarge	1.5.16	1,25	10.0
cache.r4.2xlarge	1.5.16	2.5	10.0
cache.r4.4xlarge	1.5.16	5.0	10.0
cache.r4.8xlarge	1.5.16	12	N/A
cache.r4.16xlarge	1.5.16	25	N/A

Jaringan dioptimalkan untuk Memcached

Jenis instans	Versi minimum yang didukung	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.c7gn.large	1.6.6	6.25	30
cache.c7gn.xlarge	1.6.6	12,5	40
cache.c7gn.2xlarge	1.6.6	25	50
cache.c7gn.4xlarge	1.6.6	50	N/A
cache.c7gn.8xlarge	1.6.6	100	N/A
cache.c7gn.12xlarge	1.6.6	150	N/A
cache.c7gn.16xlarge	1.6.6	200	N/A

Generasi Saat Ini (Valkey atau Redis OSS)

Untuk informasi selengkapnya tentang Generasi Sebelumnya, harap baca Simpul Generasi Sebelumnya.

> **Note**
>
> Jenis instans dengan kinerja jaringan yang dapat meledak menggunakan mekanisme I/O kredit jaringan untuk melampaui bandwidth dasar mereka dengan upaya terbaik.

Umum

Jenis instans	Versi Redis OSS minimum yang didukung	Ditingkatkan I/O dengan Redis OSS 5.0.6+	TLS Offloading dengan Redis OSS 6.2.5+	I/O Multiplexing yang disempurnakan dengan Redis OSS 7.0.4+	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.m7g.large	6.2	T	T	T	0,937	12,5
cache.m7g.xlarge	6.2	Y	Y	Y	1.876	12,5
cache.m7g.2xlarge	6.2	Y	Y	Y	3,75	15
cache.m7g.4xlarge	6.2	Y	Y	Y	7,5	15
cache.m7g.8xlarge	6.2	Y	Y	Y	15	N/A
cache.m7g.12xlarge	6.2	Y	Y	Y	22.5	N/A
cache.m7g.16xlarge	6.2	Y	Y	Y	30	N/A

Jenis instans	Versi Redis OSS minimum yang didukung	Ditingkatkan I/O dengan Redis OSS 5.0.6+	TLS Offloading dengan Redis OSS 6.2.5+	I/O Multiplexing yang disempurnakan dengan Redis OSS 7.0.4+	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.m6g.large	5.0.6	T	T	T	0,75	10.0
cache.m6g.xlarge	5.0.6	Y	Y	Y	1,25	10.0
cache.m6g.2xlarge	5.0.6	Y	Y	Y	2.5	10.0
cache.m6g.4xlarge	5.0.6	Y	Y	Y	5.0	10.0
cache.m6g.8xlarge	5.0.6	Y	Y	Y	12	N/A
cache.m6g.12xlarge	5.0.6	Y	Y	Y	20	N/A
cache.m6g.16xlarge	5.0.6	Y	Y	Y	25	N/A
cache.m5.large	3.2.4	T	T	T	0,75	10.0
cache.m5.xlarge	3.2.4	Y	T	T	1,25	10.0
cache.m5.2xlarge	3.2.4	Y	Y	Y	2.5	10.0

Jenis instans	Versi Redis OSS minimum yang didukung	Ditingkatkan I/O dengan Redis OSS 5.0.6+	TLS Offloading dengan Redis OSS 6.2.5+	I/O Multiplexing yang disempurnakan dengan Redis OSS 7.0.4+	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.m5.4xlarge	3.2.4	Y	Y	Y	5.0	10.0
cache.m5.12xlarge	3.2.4	Y	Y	Y	12	N/A
cache.m5.24xlarge	3.2.4	Y	Y	Y	25	N/A
cache.m4.large	3.2.4	T	T	T	0,45	1.2
cache.m4.xlarge	3.2.4	Y	T	T	0,75	2.8
cache.m4.2xlarge	3.2.4	Y	Y	Y	1.0	10.0
cache.m4.4xlarge	3.2.4	Y	Y	Y	2.0	10.0
cache.m4.10xlarge	3.2.4	Y	Y	Y	5.0	10.0
cache.t4g.micro	3.2.4	T	T	T	0,064	5.0
cache.t4g.small	5.0.6	T	T	T	0,18	5.0
cache.t4g.medium	5.0.6	T	T	T	0,256	5.0

Jenis instans	Versi Redis OSS minimum yang didukung	Ditingkatkan I/O dengan Redis OSS 5.0.6+	TLS Offloading dengan Redis OSS 6.2.5+	I/O Multiplexing yang disempurnakan dengan Redis OSS 7.0.4+	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.t3.micro	3.2.4	T	T	T	0,064	5.0
cache.t3.small	3.2.4	T	T	T	0,18	5.0
cache.t3.medium	3.2.4	T	T	T	0,256	5.0
cache.t2.micro	3.2.4	T	T	T	0,064	1.024
cache.t2.small	3.2.4	T	T	T	0,18	1.024
cache.t2.medium	3.2.4	T	T	T	0,256	1.024

Memori yang dioptimalkan

Jenis instans	Versi Redis OSS minimum yang didukung	Ditingkatkan I/O dengan Redis OSS 5.0.6+	TLS Offloading dengan Redis OSS 6.2.5+	I/O Multiplexing yang disempurnakan dengan Redis OSS 7.0.4+	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.r7g.large	6.2	T	T	T	0,937	12,5

Jenis instans	Versi Redis OSS minimum yang didukung	Ditingkatkan I/O dengan Redis OSS 5.0.6+	TLS Offloading dengan Redis OSS 6.2.5+	I/O Multiplexing yang disempurnakan dengan Redis OSS 7.0.4+	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.r7g.xlarge	6.2	Y	Y	Y	1.876	12,5
cache.r7g.2xlarge	6.2	Y	Y	Y	3,75	15
cache.r7g.4xlarge	6.2	Y	Y	Y	7,5	15
cache.r7g.8xlarge	6.2	Y	Y	Y	15	N/A
cache.r7g.12xlarge	6.2	Y	Y	Y	22.5	N/A
cache.r7g.16xlarge	6.2	Y	Y	Y	30	N/A
cache.r6g.large	5.0.6	T	T	T	0,75	10.0
cache.r6g.xlarge	5.0.6	Y	Y	Y	1,25	10.0
cache.r6g.2xlarge	5.0.6	Y	Y	Y	2.5	10.0
cache.r6g.4xlarge	5.0.6	Y	Y	Y	5.0	10.0

Jenis instans	Versi Redis OSS minimum yang didukung	Ditingkatkan I/O dengan Redis OSS 5.0.6+	TLS Offloading dengan Redis OSS 6.2.5+	I/O Multiplexing yang disempurnakan dengan Redis OSS 7.0.4+	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.r6g.8xlarge	5.0.6	Y	Y	Y	12	N/A
cache.r6g.12xlarge	5.0.6	Y	Y	Y	20	N/A
cache.r6g.16xlarge	5.0.6	Y	Y	Y	25	N/A
cache.r5.large	3.2.4	T	T	T	0,75	10.0
cache.r5.xlarge	3.2.4	Y	T	T	1,25	10.0
cache.r5.2xlarge	3.2.4	Y	Y	Y	2.5	10.0
cache.r5.4xlarge	3.2.4	Y	Y	Y	5.0	10.0
cache.r5.12xlarge	3.2.4	Y	Y	Y	12	N/A
cache.r5.24xlarge	3.2.4	Y	Y	Y	25	N/A
cache.r4.large	3.2.4	T	T	T	0,75	10.0
cache.r4.xlarge	3.2.4	Y	T	T	1,25	10.0
cache.r4.2xlarge	3.2.4	Y	Y	Y	2.5	10.0

Jenis instans	Versi Redis OSS minimum yang didukung	Ditingkatkan I/O dengan Redis OSS 5.0.6+	TLS Offloading dengan Redis OSS 6.2.5+	I/O Multiplexing yang disempurnakan dengan Redis OSS 7.0.4+	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.r4.4xlarge	3.2.4	Y	Y	Y	5.0	10.0
cache.r4.8xlarge	3.2.4	Y	Y	Y	12	N/A
cache.r4.16xlarge	3.2.4	Y	Y	Y	25	N/A

Memori yang dioptimalkan dengan tingkatan data

Jenis instans	Versi Redis OSS minimum yang didukung	Ditingkatkan I/O dengan Redis OSS 5.0.6+	TLS Offloading dengan Redis OSS 6.2.5+	I/O Multiplexing yang disempurnakan dengan Redis OSS 7.0.4+	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.r6gd.xlarge	6.2.0	Y	T	T	1,25	10
cache.r6gd.2xlarge	6.2.0	Y	Y	Y	2.5	10

Jenis instans	Versi Redis OSS minimum yang didukung	Ditingkatkan I/O dengan Redis OSS 5.0.6+	TLS Offloading dengan Redis OSS 6.2.5+	I/O Multiplexing yang disempurnakan dengan Redis OSS 7.0.4+	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.r6gd.4xlarge	6.2.0	Y	Y	Y	5.0	10
cache.r6gd.8xlarge	6.2.0	Y	Y	Y	12	N/A
cache.r6gd.12xlarge	6.2.0	Y	Y	Y	20	N/A
cache.r6gd.16xlarge	6.2.0	Y	Y	Y	25	N/A

Jaringan yang dioptimalkan

Jenis instans	Versi Redis OSS minimum yang didukung	Ditingkatkan I/O dengan Redis OSS 5.0.6+	TLS Offloading dengan Redis OSS 6.2.5+	I/O Multiplexing yang disempurnakan dengan Redis OSS 7.0.4+	Bandwidth dasar (Gbps)	Bandwidth lonjakan (Gbps)
cache.c7gn.large	6.2	T	T	T	6.25	30
cache.c7gn.xlarge	6.2	Y	Y	Y	12,5	40
cache.c7gn.2xlarge	6.2	Y	Y	Y	25	50
cache.c7gn.4xlarge	6.2	Y	Y	Y	50	N/A
cache.c7gn.8xlarge	6.2	Y	Y	Y	100	N/A
cache.c7gn.12xlarge	6.2	Y	Y	Y	150	N/A
cache.c7gn.16xlarge	6.2	Y	Y	Y	200	N/A

Jenis simpul yang didukung oleh Wilayah AWS

Jenis node yang didukung dapat bervariasi antar AWS Wilayah. Untuk detail selengkapnya, lihat ElastiCacheharga Amazon.

instans Performa yang Dapat Melonjak

Anda dapat meluncurkan node cache T4G, T3-Standar, dan T2-Standar burstable tujuan umum di Amazon. ElastiCache Simpul ini menyediakan tingkat performa CPU dasar dengan kemampuan untuk melakukan lonjakan penggunaan CPU kapan pun hingga kredit yang diakumulasi habis. Kredit CPU menyediakan performa inti CPU penuh selama satu menit.

Node ElastiCache T4G, T3, dan T2 Amazon dikonfigurasi sebagai standar dan cocok untuk beban kerja dengan pemanfaatan CPU rata-rata yang secara konsisten di bawah kinerja dasar instans. Untuk melonjak di atas batas dasar, simpul menggunakan kredit yang telah diakumulasikan dalam saldo kredit CPU. Jika simpul hampir kehabisan kredit yang diakumulasikan, performa secara bertahap diturunkan ke tingkat performa dasar. Penurunan bertahap ini memastikan simpul tidak mengalami penurunan performa yang tajam saat saldo kredit CPU yang diakumulasikan habis. Untuk informasi selengkapnya, lihat Kredit CPU dan Kinerja Dasar untuk Instans Kinerja Burstable di Panduan Pengguna Amazon. EC2

Tabel berikut mencantumkan jenis simpul performa yang dapat melonjak, dan laju perolehan kredit CPU per jam. Ini juga menunjukkan jumlah maksimum kredit CPU yang diperoleh yang dapat diperoleh node dan jumlah v CPUs per node. Selain itu, tabel ini menunjukkan tingkat performa dasar sebagai persentase dari performa inti penuh (menggunakan satu vCPU tunggal).

Kredit CPU dihasilkan per jam	Kredit maksimum yang diperoleh yang dapat terakumulasi*	v CPUs	Performa dasar per vCPU	Memori (GiB)	Performa jaringan
12	288	2	10%	0,5	Hingga 5 Gigabit
24	576	2	20%	1,37	Hingga 5 Gigabit
24	576	2	20%	3.09	Hingga 5 Gigabit

Kredit CPU dihasilkan per jam	Kredit maksimum yang diperoleh yang dapat terakumulasi*	v CPUs	Performa dasar per vCPU	Memori (GiB)	Performa jaringan
12	288	2	10%	0,5	Hingga 5 Gigabit
24	576	2	20%	1,37	Hingga 5 Gigabit
24	576	2	20%	3.09	Hingga 5 Gigabit
6	144	1	10%	0,5	Rendah hingga sedang
12	288	1	20%	1,55	Rendah hingga sedang
24	576	2	20%	3.22	Rendah hingga sedang

* Jumlah kredit yang dapat terakumulasi setara dengan jumlah kredit yang dapat diperoleh dalam periode 24 jam.

** Performa dasar dalam tabel adalah per vCPU. Beberapa ukuran simpul memiliki lebih dari satu vCPU. Untuk ini, hitung pemanfaatan CPU dasar untuk node dengan mengalikan persentase vCPU dengan jumlah v. CPUs

Metrik kredit CPU berikut tersedia untuk instans performa yang dapat melonjak T3 dan T4g:

> **Note**
>
> Metrik ini tidak tersedia untuk instans performa yang dapat melonjak T2.

- `CPUCreditUsage`
- `CPUCreditBalance`

Untuk informasi selengkapnya tentang metrik ini, lihat Metrik Kredit CPU.

Selain itu, perhatikan detail berikut:

- Semua jenis simpul generasi saat ini dibuat di cloud privat virtual (VPC) berdasarkan Amazon VPC secara default.
- Redis OSS append-only files (AOF) tidak didukung untuk instance T2. Redis OSS variabel konfigurasi `appendonly` dan `appendfsync` tidak didukung pada Redis OSS versi 2.8.22 dan yang lebih baru.

Informasi Terkait

- Fitur dan Detail ElastiCache Produk Amazon
- Parameter Spesifik Tipe Node Memcache untuk Memcached
- Parameter Valkey dan Redis OSS
- Enkripsi Bergerak (TLS)

Mem-boot ulang node

Beberapa perubahan mengharuskan cluster Valkey, Memcached, atau Redis OSS reboot untuk perubahan yang akan diterapkan. Misalnya, untuk beberapa parameter, perubahan nilai parameter dalam grup parameter hanya diterapkan setelah boot ulang.

Topik

- Mem-boot ulang node Redis OSS (mode cluster hanya dinonaktifkan)
- Mem-boot ulang cluster untuk Memcached

Mem-boot ulang node Redis OSS (mode cluster hanya dinonaktifkan)

Untuk cluster Valkey atau Redis OSS (mode cluster dinonaktifkan), parameter dalam grup parameter yang diterapkan hanya setelah reboot adalah:

- activerehashing
- databases

Node Redis hanya dapat diperbarui melalui ElastiCache konsol. Anda hanya dapat mem-boot ulang satu simpul dalam satu waktu. Untuk me-reboot beberapa node, Anda harus mengulangi proses untuk setiap node.

> **Perubahan parameter Valkey atau Redis OSS (Mode Cluster Diaktifkan)**
>
> Jika Anda membuat perubahan pada parameter berikut pada cluster Valkey atau Redis OSS (mode cluster enabled), ikuti langkah-langkah berikutnya.
>
> - activerehashing
> - databases
>
> 1. Buat cadangan manual klaster Anda. Lihat Membuat cadangan manual.
> 2. Hapus cluster Valkey atau Redis OSS (mode cluster enabled). Lihat Menghapus cluster di ElastiCache.
> 3. Pulihkan klaster menggunakan grup parameter dan cadangan yang sudah diubah untuk melakukan seeding klaster baru. Lihat Melakukan pemulihan dari cadangan ke dalam cache baru.
>
> Perubahan pada parameter lain tidak memerlukan tindakan ini.

Menggunakan AWS Management Console

Anda dapat me-reboot node menggunakan ElastiCache konsol.

Untuk mem-boot ulang simpul (konsol)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari daftar di sudut kanan atas, pilih AWS Wilayah yang berlaku.
3. Di panel navigasi kiri, pilih Redis OSS.

   Daftar cluster yang menjalankan Redis OSS muncul.
4. Pilih klaster di bagian Nama Klaster.
5. Pada Nama simpul, pilih tombol radio di samping simpul yang ingin Anda boot ulang.
6. Pilih Tindakan, lalu pilih Boot ulang simpul.

Untuk mem-boot ulang beberapa simpul, ulangi langkah 2 hingga 5 untuk setiap simpul yang ingin Anda boot ulang. Anda tidak perlu menunggu satu simpul selesai di-boot ulang untuk melakukan boot ulang simpul lain.

Mem-boot ulang cluster untuk Memcached

Saat Anda me-reboot cluster Memcached, cluster akan membersihkan semua datanya dan memulai ulang mesinnya. Selama proses ini, Anda tidak dapat mengakses klaster. Karena klaster melakukan flushing terhadap semua datanya, saat klaster tersedia lagi, Anda memulai dengan klaster yang kosong.

Anda dapat me-reboot cluster menggunakan ElastiCache konsol, the AWS CLI, atau ElastiCache API. Apakah Anda menggunakan ElastiCache konsol, AWS CLI atau ElastiCache API, Anda hanya dapat memulai reboot satu cluster. Untuk melakukan boot ulang beberapa klaster, Anda harus mengulangi proses atau operasinya.

Menggunakan AWS Management Console

Anda dapat me-reboot cluster menggunakan ElastiCache konsol.

Untuk melakukan boot ulang klaster (konsol)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari daftar di pojok kanan atas, pilih AWS Wilayah yang Anda minati.
3. Di panel navigasi, pilih mesin yang berjalan pada klaster yang ingin Anda boot ulang.

Daftar klaster yang menjalankan mesin yang dipilih akan muncul.

4. Pilih klaster yang di-boot ulang dengan memilih di kotak di sebelah kiri nama klaster.

   Tombol Boot ulang menjadi aktif.

   Jika Anda memilih lebih dari satu klaster, tombol Boot ulang menjadi tidak aktif.

5. Pilih Boot ulang.

   Layar konfirmasi klaster boot ulang akan muncul.

6. Untuk melakukan boot ulang klaster, pilih Boot ulang. Status klaster berubah menjadi simpul klaster sedang di-boot ulang.

   Untuk membatalkan boot ulang klaster, pilih Batalkan.

Untuk melakukan boot ulang beberapa klaster, ulangi langkah 2 hingga 5 untuk setiap klaster yang ingin di-boot ulang. Anda tidak perlu menunggu satu klaster selesai di-boot ulang agar dapat melakukan boot ulang yang lain.

Untuk melakukan boot ulang simpul tertentu, pilih simpul, lalu pilih Boot ulang.

Menggunakan AWS CLI

Untuk melakukan boot ulang klaster (AWS CLI), gunakan operasi CLI `reboot-cache-cluster`.

Untuk melakukan boot ulang simpul tertentu dalam klaster, gunakan `--cache-node-ids-to-reboot` untuk menampilkan daftar klaster tertentu yang akan di-boot ulang. Perintah berikut melakukan boot ulang simpul 0001, 0002, dan 0004 dari my-cluster.

Untuk Linux, macOS, atau Unix:

```
aws elasticache reboot-cache-cluster \
    --cache-cluster-id my-cluster \
    --cache-node-ids-to-reboot 0001 0002 0004
```

Untuk Windows:

```
aws elasticache reboot-cache-cluster ^
    --cache-cluster-id my-cluster ^
    --cache-node-ids-to-reboot 0001 0002 0004
```

Untuk melakukan boot ulang semua simpul di klaster, gunakan parameter `--cache-node-ids-to-reboot` dan tampilkan daftar semua ID simpul dari klaster. Untuk informasi selengkapnya, lihat reboot-cache-cluster.

Menggunakan ElastiCache API

Untuk me-reboot cluster menggunakan ElastiCache API, gunakan `RebootCacheCluster` tindakan.

Untuk melakukan boot ulang simpul tertentu dalam klaster, gunakan `CacheNodeIdsToReboot` untuk menampilkan daftar klaster tertentu yang akan di-boot ulang. Perintah berikut melakukan boot ulang simpul 0001, 0002, dan 0004 dari my-cluster.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=RebootCacheCluster
   &CacheClusterId=my-cluster
   &CacheNodeIdsToReboot.member.1=0001
   &CacheNodeIdsToReboot.member.2=0002
   &CacheNodeIdsToReboot.member.3=0004
   &Version=2015-02-02
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &X-Amz-Credential=<credential>
```

Untuk melakukan boot ulang semua simpul di klaster, gunakan parameter `CacheNodeIdsToReboot` dan tampilkan daftar semua ID simpul dari klaster. Lihat informasi yang lebih lengkap di RebootCacheCluster.

Mengganti node (Valkey dan Redis OSS)

Amazon ElastiCache sering meningkatkan armadanya dengan tambalan dan peningkatan yang diterapkan ke instance dengan mulus. Namun, dari waktu ke waktu kami perlu meluncurkan kembali ElastiCache node Anda untuk menerapkan pembaruan OS wajib ke host yang mendasarinya. Penggantian ini diperlukan untuk menerapkan peningkatan yang memperkuat keamanan, keandalan, dan performa operasional.

Anda memiliki opsi untuk mengelola penggantian ini sendiri setiap saat sebelum periode penggantian simpul yang terjadwal. Ketika Anda mengelola penggantian sendiri, instans Anda menerima pembaruan OS ketika Anda meluncurkan kembali simpul tersebut dan penggantian simpul terjadwal Anda dibatalkan. Anda mungkin akan terus menerima peringatan yang menunjukkan bahwa penggantian simpul harus dilakukan. Jika Anda telah mengurangi kebutuhan pemeliharaan secara manual, Anda dapat mengabaikan peringatan ini.

> **Note**
>
> Node cache pengganti yang dihasilkan secara otomatis oleh Amazon ElastiCache mungkin memiliki alamat IP yang berbeda. Anda bertanggung jawab untuk meninjau konfigurasi aplikasi Anda untuk memastikan bahwa simpul cache Anda terkait dengan alamat IP yang sesuai.

Daftar berikut mengidentifikasi tindakan yang dapat Anda ambil saat ElastiCache menjadwalkan salah satu node Valkey atau Redis OSS Anda untuk penggantian. Untuk mempercepat pencarian informasi yang Anda butuhkan untuk situasi Anda, pilih dari menu berikut.

- Do nothing— Biarkan Amazon ElastiCache mengganti node sesuai jadwal.
- Change your maintenance window – Mengubah periode pemeliharaan Anda ke waktu yang lebih baik.
- Konfigurasi Valkey atau Redis OSS (mode cluster diaktifkan)
  - Replace the only node in any Valkey or Redis OSS cluster— Prosedur untuk mengganti node di cluster Valkey atau Redis OSS menggunakan backup dan restore.
  - Replace a replica node in any Valkey or Redis OSS cluster— Prosedur untuk mengganti replika baca di cluster Valkey atau Redis OSS apa pun dengan menambah dan mengurangi jumlah replika tanpa downtime cluster.

  - Replace any node in a Valkey or Redis OSS (cluster mode enabled) shard— Prosedur dinamis tanpa downtime cluster untuk mengganti node di cluster Valkey atau Redis OSS (mode cluster enabled) dengan melakukan scaling out dan scaling in.
- Konfigurasi Valkey atau Redis OSS (mode cluster dinonaktifkan)
  - Replace the only node in any Valkey or Redis OSS cluster— Prosedur untuk mengganti node apa pun di cluster Valkey atau Redis OSS menggunakan backup dan restore.
  - Replace a replica node in any Valkey or Redis OSS cluster— Prosedur untuk mengganti replika baca di cluster Valkey atau Redis OSS apa pun dengan menambah dan mengurangi jumlah replika tanpa downtime cluster.
  - Replace a node in a Valkey or Redis OSS (cluster mode disabled) cluster— Prosedur untuk mengganti node di cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) menggunakan replikasi.
  - Replace a Valkey or Redis OSS (cluster mode disabled) read-replica— Prosedur untuk mengganti replika baca secara manual dalam grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan).
  - Replace a Valkey or Redis OSS (cluster mode disabled) primary node— Prosedur untuk mengganti node primer secara manual dalam grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan).

Opsi penggantian node Valkey dan Redis OSS

- Jangan lakukan apa-apa — Jika Anda tidak melakukan apa-apa, ElastiCache ganti node sesuai jadwal.

  Untuk konfigurasi non-cluster dengan autofailover diaktifkan, cluster pada Valkey 7.2 dan di atasnya dan Redis OSS 5.0.6 dan di atasnya penggantian lengkap sementara cluster terus tetap online dan melayani permintaan tulis yang masuk. Untuk kluster yang diaktifkan failover otomatis pada Redis OSS 4.0.10 atau di bawahnya, Anda mungkin melihat gangguan penulisan singkat hingga beberapa detik yang terkait dengan pembaruan DNS.

  Jika node adalah anggota dari kluster yang diaktifkan failover otomatis, ElastiCache untuk Valkey atau Redis OSS menyediakan ketersediaan yang lebih baik selama penambalan, pembaruan, dan penggantian node terkait pemeliharaan lainnya.

Untuk konfigurasi ElastiCache cluster yang diatur untuk digunakan ElastiCache untuk klien cluster Valkey atau Redis OSS, penggantian sekarang selesai sementara cluster melayani permintaan tulis masuk.

Untuk konfigurasi non-cluster dengan autofailover diaktifkan, cluster pada Valkey 7.2 dan di atasnya dan Redis OSS 5.0.6 dan di atasnya penggantian lengkap sementara cluster terus tetap online dan melayani permintaan tulis yang masuk. Untuk kluster yang diaktifkan failover otomatis pada Redis OSS 4.0.10 atau di bawahnya, Anda mungkin melihat gangguan penulisan singkat hingga beberapa detik yang terkait dengan pembaruan DNS.

Jika node berdiri sendiri, Amazon ElastiCache pertama-tama meluncurkan node pengganti dan kemudian menyinkronkan dari node yang ada. Simpul yang ada tidak tersedia untuk permintaan layanan selama waktu ini. Setelah sinkronisasi selesai, node yang ada dihentikan dan node baru mengambil tempatnya. ElastiCache melakukan upaya terbaik untuk menyimpan data Anda selama operasi ini.

- Ubah jendela pemeliharaan Anda — Untuk acara pemeliharaan terjadwal, Anda menerima email atau acara pemberitahuan dari ElastiCache. Dalam hal ini, jika Anda mengubah periode pemeliharaan Anda sebelum waktu penggantian terjadwal, simpul Anda sekarang akan diganti pada waktu yang baru. Untuk informasi selengkapnya, lihat berikut ini:
  - Memodifikasi cluster ElastiCache
  - Mengubah grup replikasi

> **Note**
>
> Kemampuan untuk mengubah jendela pengganti Anda dengan memindahkan jendela pemeliharaan Anda hanya tersedia ketika ElastiCache pemberitahuan menyertakan jendela pemeliharaan. Jika notifikasi tersebut tidak menyertakan periode pemeliharaan, Anda tidak dapat mengubah periode pengganti.

Misalnya, katakanlah pemeliharaan dilakukan pada Kamis, 9 November, pukul 15.00 dan periode pemeliharaan berikutnya adalah Jumat, 10 November, pukul 17.00. Berikut adalah tiga skenario dengan hasilnya:

- Anda mengubah periode pemeliharaan Anda ke Jumat pukul 16.00, setelah tanggal dan waktu saat ini dan sebelum periode pemeliharaan terjadwal berikutnya. Simpul diganti pada hari Jumat, 10 November, pukul 16.00.
- Anda mengubah periode pemeliharaan Anda ke Sabtu pukul 16.00, setelah tanggal dan waktu saat ini dan sebelum periode pemeliharaan terjadwal berikutnya. Simpul diganti pada hari Sabtu, 11 November, pukul 16.00.
- Anda mengubah periode pemeliharaan Anda menjadi hari Rabu pukul 16.00, di awal minggu dari tanggal dan waktu saat ini). Simpul akan diganti pada Rabu depan, 15 November, pukul 16.00.

Untuk petunjuk, lihat Mengelola pemeliharaan ElastiCache cluster.

- Ganti satu-satunya node di cluster Valkey atau Redis OSS apa pun - Jika cluster tidak memiliki replika baca, Anda dapat menggunakan prosedur berikut untuk mengganti node.

  Untuk mengganti satu-satunya simpul menggunakan pencadangan dan pemulihan

  1. Buat snapshot dari klaster simpul. Untuk petunjuk, lihat Membuat cadangan manual.
  2. Buat klaster baru dengan melakukan seeding dari snapshot. Untuk petunjuk, lihat Melakukan pemulihan dari cadangan ke dalam cache baru.
  3. Hapus klaster dengan simpul yang dijadwalkan akan diganti. Untuk petunjuk, lihat Menghapus cluster di ElastiCache.
  4. Di aplikasi Anda, ganti titik akhir dari simpul lama dengan titik akhir dari simpul baru.

- Ganti simpul replika di cluster Valkey atau Redis OSS mana pun — Untuk mengganti cluster replika, tingkatkan jumlah replika Anda. Untuk melakukannya, tambahkan replika kemudian kurangi jumlah replika dengan menghapus replika yang ingin Anda ganti. Proses ini bersifat dinamis dan tidak menimbulkan waktu henti klaster.

> **Note**
>
> Jika serpihan atau grup replikasi Anda sudah memiliki lima replika, balikkan langkah 1 dan 2.

Untuk mengganti replika di cluster Valkey atau Redis OSS

1. Tingkatkan jumlah replika dengan menambahkan replika ke serpihan atau grup replikasi. Untuk informasi selengkapnya, lihat Menambah jumlah replika dalam serpihan.
2. Hapus replika yang ingin Anda ganti. Untuk informasi selengkapnya, lihat Mengurangi jumlah replika dalam serpihan.
3. Perbarui titik akhir dalam aplikasi Anda.

- Ganti node apa pun di pecahan Valkey atau Redis OSS (mode cluster enabled) — Untuk mengganti node dalam cluster tanpa downtime, gunakan resharding online. Pertama, tambahkan serpihan dengan melakukan penskalaan ke luar, lalu hapus serpihan dengan simpul yang akan diganti dengan melakukan penskalaan ke dalam.

  Untuk mengganti node apa pun di cluster Valkey atau Redis OSS (mode cluster diaktifkan)

  1. Skalakan keluar: Tambahkan serpihan tambahan dengan konfigurasi yang sama seperti serpihan yang ada dengan simpul yang akan diganti. Untuk informasi selengkapnya, lihat Menambahkan serpihan dengan resharding online.
  2. Penskalaan ke dalam: Hapus serpihan dengan simpul yang akan diganti. Untuk informasi selengkapnya, lihat Menghapus serpihan dengan resharding online.
  3. Perbarui titik akhir dalam aplikasi Anda.

- Ganti node di cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) - Jika cluster adalah cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) tanpa replika baca, gunakan prosedur berikut untuk mengganti node.

Untuk mengganti simpul menggunakan replikasi (hanya mode klaster dinonaktifkan)

1. Tambahkan replikasi sebagai primer ke klaster yang memiliki simpul yang dijadwalkan akan diganti. Jangan mengaktifkan Multi-AZ di klaster ini. Untuk petunjuk, lihat Untuk menambahkan replikasi ke cluster Valkey atau Redis OSS tanpa pecahan.
2. Tambahkan replika baca ke klaster. Untuk petunjuk, lihat Untuk menambahkan node ke ElastiCache cluster (konsol).
3. Promosikan replika-baca yang baru dibuat menjadi primer. Untuk petunjuk, lihat Mempromosikan replika baca ke primer, untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan).
4. Hapus simpul yang dijadwalkan akan diganti. Untuk petunjuk, lihat Menghapus node dari ElastiCache cluster.
5. Di aplikasi Anda, ganti titik akhir dari simpul lama dengan titik akhir dari simpul baru.

- Ganti replika baca-Valkey atau Redis OSS (mode cluster dinonaktifkan) - Jika node adalah replika baca, ganti simpulnya.

  Jika klaster Anda memiliki hanya satu simpul replika dan Multi-AZ diaktifkan, Anda harus menonaktifkan Multi-AZ sebelum dapat menghapus replika. Untuk petunjuk, lihat Mengubah grup replikasi.

  Untuk mengganti Valkey atau Redis OSS (mode cluster dinonaktifkan) baca replika

  1. Hapus replika yang dijadwalkan akan diganti. Untuk petunjuk, lihat yang berikut ini:
     - Mengurangi jumlah replika dalam serpihan
     - Menghapus node dari ElastiCache cluster
  2. Tambahkan replika baru untuk menggantikan replika yang dijadwalkan akan diganti. Jika Anda menggunakan nama yang sama dengan replika yang baru saja dihapus, Anda dapat melewati langkah 3. Untuk petunjuk, lihat yang berikut ini:
     - Menambah jumlah replika dalam serpihan
     - Menambahkan replika baca untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)
  3. Di aplikasi Anda, ganti titik akhir dari replika lama dengan titik akhir dari replika baru.

4. Jika Anda menonaktifkan Multi-AZ di awal, aktifkan kembali sekarang. Untuk petunjuk, lihat Mengaktifkan Multi-AZ .

- Ganti simpul utama Valkey atau Redis OSS (mode cluster dinonaktifkan) - Jika node adalah simpul utama, pertama-tama promosikan replika baca ke primer. Kemudian hapus replika yang sebelumnya merupakan simpul primer.

  Jika klaster Anda memiliki hanya satu replika dan Multi-AZ diaktifkan, Anda harus menonaktifkan Multi-AZ sebelum dapat menghapus replika pada langkah 2. Untuk petunjuk, lihat Mengubah grup replikasi.

  Untuk mengganti simpul utama Valkey atau Redis OSS (mode cluster dinonaktifkan)

  1. Promosikan replika baca menjadi primer. Untuk petunjuk, lihat Mempromosikan replika baca ke primer, untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan).
  2. Hapus simpul yang dijadwalkan akan diganti (primer yang lama). Untuk petunjuk, lihat Menghapus node dari ElastiCache cluster.
  3. Tambahkan replika baru untuk menggantikan replika yang dijadwalkan akan diganti. Jika Anda menggunakan nama yang sama dengan simpul yang baru saja Anda hapus, Anda tidak perlu melakukan perubahan titik akhir dalam aplikasi Anda.

     Untuk petunjuk, lihat Menambahkan replika baca untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan).
  4. Di aplikasi Anda, ganti titik akhir dari simpul lama dengan titik akhir dari simpul baru.
  5. Jika Anda menonaktifkan Multi-AZ di awal, aktifkan kembali sekarang. Untuk petunjuk, lihat Mengaktifkan Multi-AZ .

Mengganti node (Memcached)

Amazon ElastiCache untuk Memcached sering meningkatkan armadanya dengan tambalan dan peningkatan yang diterapkan ke instans dengan mulus. Namun, dari waktu ke waktu kami perlu meluncurkan kembali node Anda ElastiCache untuk Memcached untuk menerapkan pembaruan OS wajib ke host yang mendasarinya. Penggantian ini diperlukan untuk menerapkan peningkatan yang memperkuat keamanan, keandalan, dan performa operasional.

Anda memiliki opsi untuk mengelola penggantian ini sendiri setiap saat sebelum periode penggantian simpul yang terjadwal. Ketika Anda mengelola penggantian sendiri, instans Anda menerima pembaruan OS ketika Anda meluncurkan kembali simpul tersebut dan penggantian simpul terjadwal Anda dibatalkan. Anda mungkin akan terus menerima peringatan yang menunjukkan bahwa penggantian simpul akan dilakukan. Jika Anda telah mengurangi kebutuhan pemeliharaan secara manual, Anda dapat mengabaikan peringatan ini.

> **Note**
>
> Node cache pengganti yang dihasilkan secara otomatis oleh Amazon ElastiCache mungkin memiliki alamat IP yang berbeda. Anda bertanggung jawab untuk meninjau konfigurasi aplikasi Anda untuk memastikan bahwa simpul cache Anda terkait dengan alamat IP yang sesuai.

Daftar berikut mengidentifikasi tindakan yang dapat Anda lakukan saat ElastiCache menjadwalkan salah satu node Memcached Anda untuk penggantian.

- Jangan lakukan apa-apa — Jika Anda tidak melakukan apa-apa, ElastiCache ganti node sesuai jadwal. Ketika ElastiCache secara otomatis mengganti node dengan node baru, node baru awalnya kosong.
- Ubah jendela pemeliharaan Anda — Untuk acara pemeliharaan terjadwal, Anda menerima email atau acara pemberitahuan dariElastiCache. Dalam hal ini, jika Anda mengubah periode pemeliharaan Anda sebelum waktu penggantian terjadwal, simpul Anda sekarang diganti pada waktu yang baru. Untuk informasi selengkapnya, lihat Memodifikasi cluster ElastiCache .

  > **Note**
  >
  > Kemampuan untuk mengubah jendela pengganti Anda dengan memindahkan jendela pemeliharaan Anda hanya tersedia ketika ElastiCache pemberitahuan menyertakan

jendela pemeliharaan. Jika notifikasi tersebut tidak menyertakan periode pemeliharaan, Anda tidak dapat mengubah periode pengganti.

Misalnya, katakanlah pemeliharaan dilakukan pada Kamis, 9 November, pukul 15.00 dan periode pemeliharaan berikutnya adalah Jumat, 10 November, pukul 17.00. Berikut adalah tiga skenario dengan hasilnya:

- Anda mengubah periode pemeliharaan Anda ke Jumat pukul 16.00, setelah tanggal dan waktu saat ini dan sebelum periode pemeliharaan terjadwal berikutnya. Simpul diganti pada hari Jumat, 10 November, pukul 16.00.
- Anda mengubah periode pemeliharaan Anda ke Sabtu pukul 16.00, setelah tanggal dan waktu saat ini dan sebelum periode pemeliharaan terjadwal berikutnya. Simpul diganti pada hari Sabtu, 11 November, pukul 16.00.
- Anda mengubah periode pemeliharaan Anda menjadi hari Rabu pukul 16.00, di awal minggu dari tanggal dan waktu saat ini). Simpul akan diganti pada Rabu depan, 15 November, pukul 16.00.

Untuk petunjuk, lihat Mengelola pemeliharaan ElastiCache cluster.

- Ganti simpul secara manual – Jika Anda perlu mengganti simpul sebelum periode pemeliharaan berikutnya, ganti simpul secara manual.

  Jika Anda mengganti simpul secara manual, kunci akan didistribusikan kembali. Distribusi kembali ini menyebabkan cache miss.

  Untuk secara manual mengganti simpul Memcached

  1. Hapus simpul yang dijadwalkan akan diganti. Untuk petunjuk, lihat Menghapus node dari ElastiCache cluster.
  2. Tambahkan simpul baru ke klaster. Untuk petunjuk, lihat Menambahkan node ke ElastiCache cluster.
  3. Jika Anda tidak menggunakan penemuan otomatis di klaster ini, lihat aplikasi Anda dan ganti setiap titik akhir simpul lama dengan titik akhir simpul baru.

Simpul terpesan

Memesan satu atau lebih ElastiCache node mungkin merupakan cara bagi Anda untuk mengurangi biaya. Simpul terpesan dikenai biaya di muka dan bergantung pada jenis simpul dan durasi pemesanan—satu atau tiga tahun.

Untuk melihat apakah simpul terpesan menghemat biaya kasus penggunaan Anda, pertama-tama tentukan dulu ukuran simpul dan jumlah simpul yang Anda butuhkan. Kemudian, estimasikan penggunaan simpul dan bandingkan total biaya jika Anda menggunakan simpul sesuai permintaan versus simpul terpesan. Anda dapat memadupadankan penggunaan simpul cadangan dan simpul sesuai permintaan dalam klaster Anda. Untuk informasi harga, lihat ElastiCache Harga Amazon.

Topik

Mengelola biaya dengan simpul terpesan

Memesan satu atau beberapa simpul dapat menjadi cara untuk mengurangi biaya. Simpul terpesan dikenai biaya di muka dan bergantung pada jenis simpul dan durasi pemesanan—satu atau tiga tahun. Biaya ini jauh lebih kecil daripada biaya penggunaan per jam yang dikenakan untuk simpul sesuai permintaan.

Untuk melihat apakah simpul terpesan menghemat biaya kasus penggunaan Anda, pertama-tama tentukan dulu ukuran simpul dan jumlah simpul yang Anda butuhkan. Kemudian, estimasikan penggunaan simpul dan bandingkan total biaya jika Anda menggunakan simpul sesuai permintaan versus dengan simpul terpesan. Anda dapat memadupadankan penggunaan simpul terpesan dan simpul sesuai permintaan dalam klaster Anda. Untuk informasi harga, lihat ElastiCache Harga Amazon.

AWS Wilayah, tipe simpul, dan panjang jangka waktu harus dipilih saat pembelian, dan tidak dapat diubah nanti.

Anda dapat menggunakan AWS Management Console, API AWS CLI, atau ElastiCache API untuk membuat daftar dan membeli penawaran node cadangan yang tersedia.

Untuk informasi selengkapnya tentang node yang dicadangkan, lihat Amazon ElastiCache Reserved Nodes.

Penawaran simpul terpesan standar

Saat membeli instance node cadangan (RI) di Amazon ElastiCache, Anda dapat membeli komitmen untuk mendapatkan tarif diskon pada jenis dan AWS Wilayah instance node tertentu selama durasi instance node yang dicadangkan. Untuk menggunakan instance node ElastiCache cadangan Amazon, Anda membuat instance ElastiCache node baru, seperti yang Anda lakukan untuk instance sesuai permintaan.

Jika spesifikasi instance node cadangan baru cocok dengan instance node reserved yang ada untuk akun Anda, Anda akan ditagih dengan tarif diskon yang ditawarkan untuk instance node reserved. Jika tidak, instans simpul ditagih dengan tarif sesuai permintaan. Standar ini RIs tersedia dari keluarga instans R5 dan M5 dan seterusnya.

> **Note**
>
> Semua jenis penawaran yang dibahas selanjutnya tersedia dalam jangka waktu satu tahun dan tiga tahun.

Jenis Penawaran

Tidak ada Upfront RI yang menyediakan akses ke ElastiCache instans yang dipesan tanpa memerlukan pembayaran di muka. ElastiCache Instans yang tidak dipesan di muka Anda menagih tarif per jam diskon untuk setiap jam dalam jangka waktu tersebut, terlepas dari penggunaannya.

Partial Upfront RI membutuhkan sebagian dari ElasticCache instans cadangan untuk dibayar di muka. Sisa jam dalam jangka waktu pemesanan akan ditagih dengan tarif per jam yang didiskon, terlepas dari penggunaannya. Opsi ini adalah pengganti opsi warisan Pemanfaatan Berat, yang dijelaskan di bagian berikutnya.

RI Semua Di Muka membutuhkan pembayaran penuh yang harus dilakukan di awal jangka waktu RI. Anda tidak dikenai biaya lain untuk sisa jangka waktu terlepas dari jumlah jam yang digunakan.

Ukuran node cadangan fleksibel

Semua node yang dicadangkan berukuran fleksibel. Ketika Anda membeli node reserved, satu hal yang Anda tentukan adalah tipe node, misalnya cache.r6g.xlarge. Untuk informasi selengkapnya, tentang jenis node, lihat ElastiCacheHarga Amazon.

Jika Anda memiliki node, dan Anda perlu menskalakannya ke kapasitas yang lebih besar, node cadangan Anda secara otomatis diterapkan ke node skala Anda. Artinya, node cadangan Anda secara otomatis diterapkan untuk penggunaan ukuran apa pun dalam keluarga simpul yang sama. Node cadangan fleksibel ukuran tersedia untuk node dengan Wilayah yang sama. AWS Node cadangan fleksibel ukuran hanya dapat menskalakan dalam keluarga simpulnya. Misalnya, node cadangan untuk cache.r6g.xlarge dapat diterapkan ke cache.r6g.2xlarge, tetapi tidak ke cache.r6gd.large, karena cache.r6g dan cache.r6gd adalah keluarga simpul yang berbeda.

Fleksibilitas ukuran berarti Anda dapat bergerak bebas di antara konfigurasi dalam keluarga simpul yang sama. Misalnya, Anda dapat berpindah dari node cadangan r6g.xlarge (8 unit dinormalisasi) ke dua r6g.large node cadangan (8 unit dinormalisasi) (2*4 = 8 unit dinormalisasi) di Wilayah yang sama tanpa biaya tambahan. AWS

Memutakhirkan node dari Redis OSS ke Valkey

Dengan peluncuran Valkey di ElastiCache, Anda sekarang dapat menerapkan diskon node reservasi Redis OSS Anda ke mesin cache Valkey. Anda dapat meningkatkan dari Redis OSS ke Valkey sambil tetap mendapatkan keuntungan dari kontrak dan pemesanan yang ada. Selain dapat menerapkan manfaat Anda dalam keluarga dan mesin node cache, Anda bahkan dapat menerima lebih banyak nilai tambahan. Valkey diberi harga diskon 20% relatif terhadap Redis OSS, dan dengan fleksibilitas node yang dicadangkan, Anda dapat menggunakan node cadangan Redis OSS Anda untuk menutupi 20% lebih banyak node Valkey yang berjalan.

Untuk menghitung tingkat diskon, setiap ElastiCache node dan kombinasi mesin memiliki faktor normalisasi yang diukur dalam satuan. Unit node cadangan dapat diterapkan ke node yang sedang berjalan di dalam keluarga instance node reserved untuk mesin tertentu. Node cadangan Redis OSS juga dapat diterapkan di seluruh mesin untuk menutupi node Valkey yang sedang berjalan. Karena Valkey diberi harga diskon relatif terhadap Redis OSS dan Memcached, unitnya untuk jenis instance tertentu lebih rendah, yang memungkinkan node cadangan Redis OSS untuk mencakup lebih banyak node Valkey.

Sebagai contoh, katakanlah Anda telah membeli node cadangan untuk cache.r7g.4xlarge untuk mesin Redis OSS (32 unit) dan menjalankan satu node Redis OSS cache.r7g.4xlarge (32 unit). Jika Anda memutakhirkan node ke Valkey, faktor normalisasi node yang sedang berjalan turun menjadi

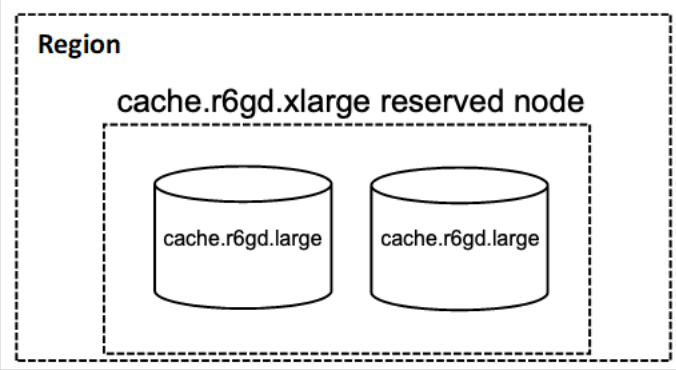
25,6 unit, dan node cadangan Anda yang ada memberi Anda 6,4 unit tambahan untuk digunakan terhadap node Valkey atau Redis OSS lain yang sedang berjalan dalam keluarga cache.r7g di Wilayah. Anda dapat menggunakan ini untuk menutupi 25% dari node Valkey cache.r7g.4xlarge lain di akun (25,6 unit), atau 100% dari node Valkey cache.r7g.xlarge (6,4 unit).

Membandingkan penggunaan dengan unit yang dinormalisasi

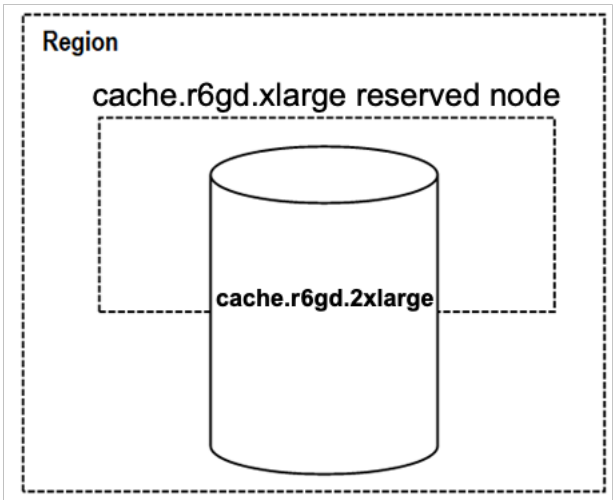
Anda dapat membandingkan penggunaan untuk ukuran node cadangan yang berbeda dengan menggunakan unit yang dinormalisasi. Misalnya, satu jam penggunaan pada dua node cache.r6g.4xlarge setara dengan 16 jam penggunaan pada satu cache.r6g.large. Tabel berikut menunjukkan jumlah unit dinormalisasi untuk setiap ukuran node:

Ukuran simpul	Unit dinormalisasi dengan Redis OSS atau Memcached	Unit dinormalisasi dengan Valkey
micro	0,5	0,4
small	1	.8
medium	2	1.6
large	4	3.2
xlarge	8	6.4
2xlarge	16	12.8
4xlarge	32	25.6
6xlarge	48	38.4
8xlarge	64	51.2
10xlarge	80	64
12xlarge	96	76.8
16xlarge	128	102.4
24xlarge	192	153,6

Misalnya, Anda membeli node cadangan cache.r6gd.xlarge, dan Anda memiliki dua node cadangan cache.r6gd.large yang sedang berjalan di akun Anda di Wilayah yang sama. AWS Dalam hal ini, manfaat penagihan diterapkan secara penuh ke kedua node.



Atau, jika Anda memiliki satu instance cache.r6gd.2xlarge yang berjalan di akun Anda di AWS Wilayah yang sama, manfaat penagihan diterapkan ke 50 persen dari penggunaan node yang dicadangkan.



Menghapus node yang dicadangkan

Persyaratan untuk node cadangan melibatkan komitmen satu tahun atau tiga tahun. Anda tidak dapat membatalkan node yang dicadangkan. Namun, Anda dapat menghapus node yang dicakup oleh discount node reserved. Proses untuk menghapus node yang dicakup oleh discount node reserved sama dengan node lainnya.

Jika Anda menghapus node yang dicakup oleh discount node reserved, Anda dapat meluncurkan node lain dengan spesifikasi yang kompatibel. Dalam hal ini, Anda tetap mendapatkan tarif diskon selama jangka waktu pemesanan (satu atau tiga tahun).

Penawaran simpul terpesan warisan

Ada tiga tingkat pemesanan simpul warisan—Pemanfaatan Berat, Pemanfaatan Sedang, dan Pemanfaatan Ringan. Simpul dapat dipesan pada tingkat pemanfaatan mana pun selama satu atau tiga tahun. Jenis simpul, tingkat pemanfaatan, dan jangka waktu pemesanan memengaruhi total biaya Anda. Periksa penghematan yang dapat disediakan simpul terpesan bagi bisnis Anda dengan membandingkan berbagai model sebelum Anda membeli simpul terpesan.

Simpul yang dibeli pada satu tingkat pemanfaatan atau jangka waktu tidak dapat dikonversi ke tingkat pemanfaatan atau jangka waktu yang berbeda.

Tingkat Pemanfaatan

Simpul terpesan Pemanfaatan Berat memungkinkan beban kerja yang memiliki dasar kapasitas yang konsisten atau menjalankan beban kerja dengan status yang stabil. Simpul terpesan Pemanfaatan Berat memerlukan komitmen di muka yang tinggi. Namun, jika Anda berencana menjalankan lebih dari 79 persen jangka waktu simpul terpesan, Anda bisa mendapatkan penghematan terbesar (hingga 70 persen dari harga Sesuai Permintaan). Dengan simpul terpesan Pemanfaatan Berat, Anda membayar biaya satu kali. Hal ini kemudian diikuti dengan biaya per jam yang lebih rendah selama durasi jangka waktu terlepas dari apakah simpul Anda berjalan.

Simpul terpesan Pemanfaatan Sedang adalah opsi terbaik jika Anda berencana untuk menggunakan simpul terpesan Anda dalam jumlah waktu yang banyak dan Anda ingin biaya satu kali yang lebih rendah atau ingin berhenti membayar simpul Anda ketika Anda menonaktifkannya. Simpul terpesan Pemanfaatan Sedang adalah pilihan yang lebih hemat biaya jika Anda berencana untuk menjalankan lebih dari 40 persen jangka waktu simpul terpesan. Opsi ini dapat menghemat hingga 64 persen dari harga Sesuai Permintaan. Dengan simpul terpesan Pemanfaatan Sedang, Anda membayar biaya satu kali sedikit lebih tinggi dibandingkan dengan simpul terpesan Pemanfaatan Ringan. Anda juga menerima tarif penggunaan per jam yang lebih rendah ketika Anda menjalankan simpul.

Simpul terpesan Pemanfaatan Ringan ideal untuk beban kerja berkala yang hanya berjalan beberapa jam sehari atau beberapa hari per minggu. Dengan simpul terpesan Pemanfaatan Ringan, Anda membayar biaya satu kali yang diikuti dengan biaya penggunaan per jam yang didiskon ketika simpul Anda berjalan. Anda dapat mulai menghemat ketika simpul Anda berjalan lebih dari 17 persen dari jangka waktu simpul terpesan. Anda dapat menghemat hingga 56 persen dari tarif Sesuai Permintaan di sepanjang jangka waktu simpul terpesan.

Penawaran simpul terpesan warisan

Penawaran	Biaya di muka	Biaya penggunaan	Keuntungan
Pemanfaatan Berat	Tertinggi	Biaya per jam terendah. Diterapkan pada seluruh jangka waktu, baik Anda menggunakan simpul terpesan maupun tidak.	Biaya keseluruhan terendah jika Anda berencana untuk menjalankan simpul terpesan Anda lebih dari 79 persen jangka waktu yang berdurasi tiga tahun.
Pemanfaatan Sedang	Sedang	Biaya penggunaan per jam dikenakan untuk setiap jam simpul berjalan. Tidak ada biaya per jam ketika simpul tidak berjalan.	Cocok untuk beban kerja elastis atau jika Anda memperkirakan penggunaan sedang, yaitu lebih dari 40 persen jangka waktu yang berdurasi tiga tahun.
Pemanfaatan Ringan	Terendah	Biaya penggunaan per jam dikenakan untuk setiap jam simpul berjalan. Tidak ada biaya per jam ketika simpul tidak berjalan. Biaya per jam tertinggi dari semua jenis penawaran, tetapi biaya hanya berlaku ketika simpul terpesan berjalan.	Biaya keseluruh an tertinggi jika Anda berencana untuk menjalank annya sepanjang waktu. Namun, biaya ini adalah biaya keseluruhan terendah jika Anda berencana untuk jarang menggunakan simpul terpesan, yaitu lebih dari sekitar 15 persen jangka waktu

Penawaran	Biaya di muka	Biaya penggunaan	Keuntungan
			yang berdurasi tiga tahun.
Penggunaan Sesuai Permintaan (Tanpa simpul terpesan)	Tidak ada	Biaya per jam tertinggi. Diterapkan setiap kali simpul berjalan.	Biaya per jam tertinggi.

Untuk informasi selengkapnya, lihat ElastiCache Harga Amazon.

Mendapatkan info tentang penawaran simpul terpesan

Sebelum Anda membeli simpul terpesan, Anda bisa mendapatkan informasi tentang penawaran simpul terpesan yang tersedia.

Contoh berikut menunjukkan cara mendapatkan harga dan informasi tentang penawaran node cadangan yang tersedia menggunakan AWS Management Console, AWS CLI, dan ElastiCache API.

Topik

- Mendapatkan info tentang penawaran simpul terpesan (Konsol)
- Mendapatkan info tentang penawaran simpul terpesan (AWS CLI)
- Mendapatkan info tentang penawaran node cadangan (API) ElastiCache

Mendapatkan info tentang penawaran simpul terpesan (Konsol)

Untuk mendapatkan harga dan informasi lain tentang penawaran klaster cadangan yang tersedia menggunakan AWS Management Console, gunakan prosedur berikut.

Untuk mendapatkan informasi tentang penawaran simpul terpesan yang tersedia

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Simpul Terpesan.
3. Pilih Beli Simpul terpesan.
4. Untuk Engine, pilih Valkey, Memcached, atau Redis OSS.
5. Untuk menentukan penawaran yang tersedia, tentukan opsi berikut:
   - Jenis Simpul
   - Jangka Waktu
   - Jenis Penawaran

   Setelah Anda menentukan pilihan ini, biaya per simpul dan total biaya pilihan Anda akan ditampilkan di Detail reservasi.
6. Pilih Batalkan untuk menghindari pembelian simpul dan dikenai biaya.

Mendapatkan info tentang penawaran simpul terpesan (AWS CLI)

Untuk mendapatkan harga dan informasi lain tentang penawaran node cadangan yang tersedia untuk Valkey atau Redis OSS, ketik perintah berikut pada prompt perintah:

```
aws elasticache describe-reserved-cache-nodes-offerings
```

Operasi ini menghasilkan output seperti yang berikut ini (format JSON):

```
{
          "ReservedCacheNodesOfferingId": "0xxxxxxx-xxeb-44ex-xx3c-xxxxxxxx072",
          "CacheNodeType": "cache.xxx.large",
          "Duration": 94608000,
          "FixedPrice": XXXX.X,
          "UsagePrice": X.X,
          "ProductDescription": "redis",
          "OfferingType": "All Upfront",
          "RecurringCharges": [
              {
                  "RecurringChargeAmount": X.X,
                  "RecurringChargeFrequency": "Hourly"
              }
          ]
 },
 {
          "ReservedCacheNodesOfferingId": "0xxxxxxx-xxeb-44ex-xx3c-xxxxxxxx072",
          "CacheNodeType": "cache.xxx.xlarge",
          "Duration": 94608000,
          "FixedPrice": XXXX.X,
          "UsagePrice": X.X,
          "ProductDescription": "redis",
          "OfferingType": "Partial Upfront",
          "RecurringCharges": [
              {
                  "RecurringChargeAmount": X.XXX,
                  "RecurringChargeFrequency": "Hourly"
              }
          ]
 },
 {
          "ReservedCacheNodesOfferingId": "0xxxxxxx-xxeb-44ex-xx3c-xxxxxxxx072",
          "CacheNodeType": "cache.xxx.large",
          "Duration": 31536000,
```

```
            "FixedPrice": X.X,
            "UsagePrice": X.X,
            "ProductDescription": "redis",
            "OfferingType": "No Upfront",
            "RecurringCharges": [
                {
                    "RecurringChargeAmount": X.XXX,
                    "RecurringChargeFrequency": "Hourly"
                }
            ]
}
```

Untuk mendapatkan harga dan informasi lain tentang penawaran node cadangan yang tersedia untuk Memcached, ketik perintah berikut pada prompt perintah:

```
 {
            "ReservedCacheNodesOfferingId": "0xxxxxxxx-xxeb-44ex-xx3c-xxxxxxxx072",
            "CacheNodeType": "cache.xxx.large",
            "Duration": 94608000,
            "FixedPrice": XXXX.X,
            "UsagePrice": X.X,
            "ProductDescription": "memcached",
            "OfferingType": "All Upfront",
            "RecurringCharges": [
                {
                    "RecurringChargeAmount": X.X,
                    "RecurringChargeFrequency": "Hourly"
                }
            ]
    },
 {
            "ReservedCacheNodesOfferingId": "0xxxxxxxx-xxeb-44ex-xx3c-xxxxxxxx072",
            "CacheNodeType": "cache.xxx.xlarge",
            "Duration": 94608000,
            "FixedPrice": XXXX.X,
            "UsagePrice": X.X,
            "ProductDescription": "memcached",
            "OfferingType": "Partial Upfront",
            "RecurringCharges": [
                {
                    "RecurringChargeAmount": X.XXXX,
                    "RecurringChargeFrequency": "Hourly"
                }
```

```
            ]
        },
        {
            "ReservedCacheNodesOfferingId": "0xxxxxxxx-xxeb-44ex-xx3c-xxxxxxxx072",
            "CacheNodeType": "cache.xx.12xlarge",
            "Duration": 31536000,
            "FixedPrice": X.X,
            "UsagePrice": X.X,
            "ProductDescription": "memcached",
            "OfferingType": "No Upfront",
            "RecurringCharges": [
                {
                    "RecurringChargeAmount": X.XXXX,
                    "RecurringChargeFrequency": "Hourly"
                }
            ]
}
```

Untuk informasi selengkapnya, lihat describe-reserved-cache-nodes-penawaran di Referensi. AWS CLI

Mendapatkan info tentang penawaran node cadangan (API) ElastiCache

Untuk mendapatkan harga dan informasi tentang penawaran simpul terpesan yang tersedia, panggil tindakan `DescribeReservedCacheNodesOfferings`.

Example

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=DescribeReservedCacheNodesOfferings
    &Version=2014-12-01
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &X-Amz-Algorithm
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20141201T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Untuk informasi selengkapnya, lihat DescribeReservedCacheNodesOfferingsdi Referensi ElastiCache API.

Membeli simpul terpesan

Contoh berikut menunjukkan cara membeli penawaran node cadangan menggunakan AWS Management Console, the AWS CLI, dan ElastiCache API.

> **Important**
>
> Mengikuti contoh di bagian ini menimbulkan biaya pada AWS akun Anda yang tidak dapat Anda balikkan.

Topik

Membeli simpul terpesan (Konsol)

Contoh ini menunjukkan pembelian penawaran simpul terpesan yang spesifik, 649fd0c8-cf6d-47a0-bfa6-060f8e75e95f, dengan ID simpul terpesan myreservationID.

Prosedur berikut menggunakan AWS Management Console untuk membeli penawaran node reserved dengan menawarkan id.

Untuk membeli simpul terpesan

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pada daftar navigasi, pilih tautan Simpul Terpesan.
3. Pilih tombol Beli simpul terpesan.
4. Untuk Engine, pilih Valkey, Memcached, atau Redis OSS.
5. Untuk menentukan penawaran yang tersedia, tentukan opsi berikut:
   - Jenis Simpul
   - Jangka Waktu
   - Jenis Penawaran
   - ID simpul terpesan opsional

Setelah Anda menentukan pilihan ini, biaya per simpul dan total biaya pilihan Anda akan ditampilkan di Detail reservasi.

6. Pilih Beli.

Membeli simpul terpesan AWS CLI

Contoh berikut menunjukkan pembelian penawaran klaster terpesan yang spesifik, 649fd0c8-cf6d-47a0-bfa6-060f8e75e95f, dengan ID simpul terpesan myreservationID.

Ketikkan perintah berikut pada prompt perintah:

Untuk Linux, macOS, atau Unix:

```
aws elasticache purchase-reserved-cache-nodes-offering \
    --reserved-cache-nodes-offering-id 649fd0c8-cf6d-47a0-bfa6-060f8e75e95f \
    --reserved-cache-node-id myreservationID
```

Untuk Windows:

```
aws elasticache purchase-reserved-cache-nodes-offering ^
    --reserved-cache-nodes-offering-id 649fd0c8-cf6d-47a0-bfa6-060f8e75e95f ^
    --reserved-cache-node-id myreservationID
```

Perintah tersebut mengembalikan output serupa dengan berikut ini:

```
RESERVATION  ReservationId      Class           Start Time                Duration
 Fixed Price  Usage Price  Count  State            Description      Offering Type
RESERVATION  myreservationid    cache.xx.small  2013-12-19T00:30:23.247Z  1y
 XXX.XX USD   X.XXX USD    1      payment-pending  memcached        Medium Utilization
```

Untuk informasi selengkapnya, lihat purchase-reserved-cache-nodes-offering di AWS CLI Referensi.

Membeli node cadangan (ElastiCache API)

Contoh berikut menunjukkan pembelian penawaran simpul terpesan yang spesifik, 649fd0c8-cf6d-47a0-bfa6-060f8e75e95f, dengan ID klaster terpesan myreservationID.

Panggil operasi `PurchaseReservedCacheNodesOffering` dengan parameter berikut ini:

- `ReservedCacheNodesOfferingId` = `649fd0c8-cf6d-47a0-bfa6-060f8e75e95f`
- `ReservedCacheNodeID` = `myreservationID`
- `CacheNodeCount` = `1`

Example

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=PurchaseReservedCacheNodesOffering
    &ReservedCacheNodesOfferingId=649fd0c8-cf6d-47a0-bfa6-060f8e75e95f
    &ReservedCacheNodeID=myreservationID
    &CacheNodeCount=1
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Date=20141201T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20141201T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Untuk informasi selengkapnya, lihat PurchaseReservedCacheNodesOfferingdi Referensi ElastiCache API.

Mendapatkan info tentang simpul terpesan Anda

Anda bisa mendapatkan informasi tentang node cadangan yang telah Anda beli menggunakan AWS Management Console, the AWS CLI, dan ElastiCache API.

Topik

Mendapatkan info tentang simpul terpesan Anda (Konsol)

Prosedur berikut menjelaskan cara menggunakan AWS Management Console untuk mendapatkan informasi tentang node cadangan yang Anda beli.

Untuk mendapatkan informasi tentang simpul terpesan yang dibeli

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pada daftar navigasi, pilih tautan Simpul terpesan.

   Simpul terpesan untuk akun Anda muncul di daftar Simpul terpesan. Anda dapat memilih salah satu dari simpul terpesan dalam daftar untuk melihat informasi mendetail tentang simpul terpesan di panel detail di bagian bawah konsol.

Mendapatkan info tentang simpul terpesan Anda (AWS CLI)

Untuk mendapatkan informasi tentang node cadangan untuk AWS akun Anda, ketik perintah berikut pada prompt perintah:

```
aws elasticache describe-reserved-cache-nodes
```

Operasi ini menghasilkan output seperti yang berikut ini (format JSON):

```
{
    "ReservedCacheNodeId": "myreservationid",
    "ReservedCacheNodesOfferingId": "649fd0c8-cf6d-47a0-bfa6-060f8e75e95f",
    "CacheNodeType": "cache.xx.small",
    "DataTiering": "disabled",
```

```
    "Duration": "31536000",
    "ProductDescription": "memcached",
    "OfferingType": "Medium Utilization",
    "MaxRecords": 0
}
```

Untuk informasi lebih lanjut, lihat describe-reserved-cache-nodesdi AWS CLI Referensi.

Mendapatkan info tentang node cadangan Anda (ElastiCache API)

Untuk mendapatkan informasi tentang node yang dicadangkan untuk AWS akun Anda, hubungi `DescribeReservedCacheNodes` operasi.

Example

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=DescribeReservedCacheNodes
    &Version=2014-12-01
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Date=20141201T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20141201T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Untuk informasi selengkapnya, lihat DescribeReservedCacheNodesdi Referensi ElastiCache API.

Memigrasikan simpul generasi sebelumnya

Simpul generasi sebelumnya adalah jenis simpul yang sedang dihentikan secara bertahap. Jika Anda tidak memiliki cluster yang menggunakan tipe node generasi sebelumnya, ElastiCache tidak mendukung pembuatan cluster baru dengan tipe node tersebut.

Karena jenis simpul generasi sebelumnya memiliki jumlah terbatas, kami tidak dapat menjamin penggantian simpul akan berhasil ketika simpul menjadi tidak berkondisi baik di klaster Anda. Dalam skenario seperti tersebut, ketersediaan klaster Anda mungkin berdampak negatif.

Sebaiknya migrasikan klaster Anda ke jenis simpul baru untuk ketersediaan dan performa yang lebih baik. Untuk jenis simpul yang direkomendasikan untuk dimigrasikan, lihat Jalur Peningkatan. Untuk

daftar lengkap tipe node yang didukung dan tipe node generasi sebelumnya ElastiCache, lihatJenis simpul yang didukung.

Migrasi node pada cluster Valkey atau Redis OSS

Prosedur berikut menjelaskan cara memigrasikan tipe node cluster Valkey atau Redis OSS Anda menggunakan Konsol. ElastiCache Selama proses ini, cluster Valkey atau Redis OSS Anda akan terus melayani permintaan dengan waktu henti minimal. Bergantung pada konfigurasi klaster Anda, Anda mungkin mengalami waktu henti berikut. Informasi berikut ini adalah perkiraan dan mungkin berbeda berdasarkan konfigurasi spesifik Anda:

- Mode klaster dinonaktifkan (simpul tunggal) mungkin mengalami waktu henti sekitar 60 detik, terutama karena propagasi DNS.
- Mode cluster dinonaktifkan (dengan node replika) dapat melihat sekitar 1 detik untuk cluster yang menjalankan Valkey 7.2 dan di atasnya atau Redis OSS 5.0.6 dan di atasnya. Semua versi yang lebih rendah dapat mengalami sekitar 10 detik.
- Mode klaster diaktifkan mungkin mengalami waktu henti sekitar 1 detik.

Untuk memodifikasi tipe node cluster Valkey atau Redis OSS menggunakan konsol:

1. Masuk ke Konsol dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cluster Valkey atau cluster Redis OSS.
3. Pada daftar klaster, pilih klaster yang ingin Anda migrasikan.
4. Pilih Tindakan, lalu pilih Ubah.
5. Pilih jenis simpul baru dari daftar jenis simpul.
6. Jika Anda ingin segera melakukan proses migrasi, pilih Terapkan segera. Jika Terapkan segera tidak dipilih, proses migrasi dilakukan selama periode pemeliharaan klaster berikutnya.
7. Pilih Ubah. Jika Anda memilih Terapkan segera pada langkah sebelumnya, status klaster berubah ke mengubah. Ketika status berubah ke tersedia, pengubahan selesai dan Anda dapat mulai menggunakan klaster baru tersebut.

Untuk memodifikasi tipe node cluster Valkey atau Redis OSS menggunakan: AWS CLI

Gunakan modify-replication-groupAPI seperti yang ditunjukkan berikut:

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group /
 --replication-group-id my-replication-group /
 --cache-node-type new-node-type /
 --apply-immediately
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
 --replication-group-id my-replication-group ^
 --cache-node-type new-node-type ^
 --apply-immediately
```

Dalam skenario ini, nilainya *new-node-type* adalah tipe node yang Anda migrasi. Dengan meneruskan parameter `--apply-immediately`, pembaruan akan diterapkan segera ketika grup replikasi bertransisi dari status mengubah menjadi tersedia. Jika Terapkan segera tidak dipilih, proses migrasi dilakukan selama periode pemeliharaan klaster berikutnya.

> **Note**
>
> Jika Anda tidak dapat mengubah klaster dengan `InvalidCacheClusterState`, Anda perlu menghapus simpul yang gagal dipulihkan terlebih dahulu.

Memperbaiki atau menghapus restore-failed-node

Prosedur berikut menjelaskan cara memperbaiki atau menghapus node yang gagal pemulihan dari cluster Valkey atau Redis OSS Anda. Untuk mempelajari lebih lanjut tentang bagaimana ElastiCache node masuk ke status restore-failed, lihat. Melihat Status ElastiCache Node Sebaiknya pertama-tama menghapus node apa pun dalam keadaan gagal pemulihan, kemudian memigrasikan node generasi sebelumnya yang tersisa di ElastiCache cluster ke tipe node generasi yang lebih baru, dan akhirnya menambahkan kembali jumlah node yang diperlukan.

Untuk menghapus simpul yang gagal dipulihkan (konsol):

1. Masuk ke Konsol dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cluster Valkey atau cluster Redis OSS.
3. Dari daftar klaster, pilih nama klaster yang ingin Anda hapus simpulnya.

4. Dari daftar serpihan, pilih nama serpihan yang ingin Anda hapus simpulnya. Lewati langkah ini jika mode klaster dinonaktifkan untuk klaster.
5. Dari daftar simpul, pilih simpul dengan status `restore-failed`.
6. Pilih Tindakan, lalu pilih Hapus simpul.

Setelah Anda menghapus node restore-failed dari ElastiCache cluster Anda, Anda sekarang dapat bermigrasi ke jenis generasi yang lebih baru. Untuk informasi selengkapnya, lihat di atas di Migrasi node pada cluster Valkey atau Redis OSS.

Untuk menambahkan kembali node ke ElastiCache cluster Anda, lihatMenambahkan node ke ElastiCache cluster.

Memigrasikan simpul di klaster Memcached

Untuk bermigrasi ElastiCache untuk Memcached ke jenis node yang berbeda, Anda harus membuat klaster baru, yang selalu dimulai kosong yang dapat diisi oleh aplikasi Anda.

Untuk memigrasikan ElastiCache tipe node cluster Memcached Anda menggunakan Konsol: ElastiCache

- Buat klaster baru dengan jenis simpul baru. Untuk informasi selengkapnya, lihat Membuat klaster Memcached (konsol).
- Dalam aplikasi Anda, perbarui titik akhir ke titik akhir klaster baru. Untuk informasi selengkapnya, lihat Menemukan Titik Akhir Cluster (Konsol) (Memcached)
- Hapus klaster lama. Untuk informasi selengkapnya, lihat Menghapus cluster di ElastiCache

Mengelola cluster di ElastiCache

Cluster adalah kumpulan dari satu atau lebih node cache, yang semuanya menjalankan instance perangkat lunak mesin Valkey, Memcached, dan Redis OSS. Saat membuat klaster, Anda menentukan mesin dan versi yang akan digunakan oleh semua simpul.

Cluster Valkey dan Redis OSS

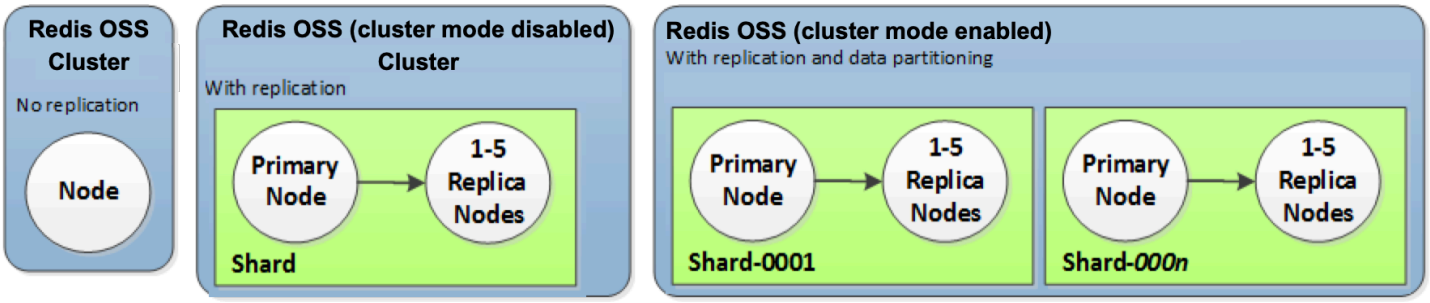
Diagram berikut menggambarkan cluster Valkey atau Redis OSS yang khas. Cluster ini dapat berisi satu node atau hingga enam node di dalam shard (API/CLI: grup node), cluster Valkey atau Redis OSS simpul tunggal (mode cluster dinonaktifkan) tidak memiliki pecahan, dan cluster Valkey multi-

node atau Redis OSS (mode cluster dinonaktifkan) memiliki pecahan tunggal. Cluster Valkey atau Redis OSS (mode cluster enabled) dapat memiliki hingga 500 pecahan, dengan data Anda dipartisi di seluruh pecahan. Batas node atau shard dapat ditingkatkan hingga maksimum 500 per cluster jika versi mesin Valkey 7.2 dan lebih tinggi atau Redis OSS 5.0.6 dan lebih tinggi. Sebagai contoh, Anda dapat memilih untuk mengonfigurasi sebuah klaster dengan 500 simpul yang berkisar antara 83 serpihan (satu primer dan 5 replika per serpihan) dan 500 serpihan (satu primer dan tanpa replika). Pastikan alamat IP yang tersedia mencukupi untuk mengakomodasi peningkatan tersebut. Kesalahan umumnya termasuk subnet dalam grup subnet memiliki rentang CIDR yang terlalu kecil atau subnet dibagikan dan banyak digunakan oleh klaster lainnya. Untuk informasi selengkapnya, lihat Membuat grup subnet. Untuk versi di bawah 5.0.6, batasnya adalah 250 per klaster.

Untuk meminta penambahan batas, lihat Batas Layanan AWS dan pilih jenis batas Simpul per klaster per jenis instans.

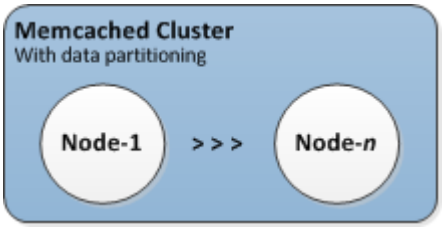
Ketika Anda memiliki beberapa node dalam pecahan Valkey atau Redis OSS, salah satu node adalah simpul utama. read/write Semua simpul lain dalam serpihan adalah replika hanya-baca.

Cluster Valkey atau Redis OSS yang khas terlihat sebagai berikut.



Cluster memcache

Cluster Memcached yang khas terlihat sebagai berikut. Cluster memcache berisi dari 1 hingga 60 node, di mana Anda secara horizontal mempartisi data Anda.



Operasi elasticache untuk Valkey, Memcached, dan Redis OSS

Sebagian besar ElastiCache operasi dilakukan di tingkat cluster. Anda dapat menyiapkan klaster dengan jumlah simpul tertentu dan grup parameter yang mengontrol properti untuk setiap simpul.

Semua simpul dalam klaster dirancang agar berupa jenis simpul yang sama serta memiliki parameter dan pengaturan grup keamanan yang sama.

Setiap klaster harus memiliki pengidentifikasi klaster. Pengidentifikasi klaster adalah nama yang diberikan pelanggan untuk klaster. Identifier ini menentukan cluster tertentu ketika berinteraksi dengan ElastiCache API dan perintah. AWS CLI Pengidentifikasi cluster harus unik untuk pelanggan tersebut di suatu AWS Wilayah.

ElastiCache mendukung beberapa versi mesin. Kecuali jika Anda memiliki alasan tertentu, kami menyarankan menggunakan versi terbaru.

ElastiCache cluster dirancang untuk diakses menggunakan EC2 instance Amazon. Jika Anda meluncurkan klaster Anda di cloud privat virtual (VPC) berdasarkan layanan Amazon VPC, Anda dapat mengaksesnya dari luar AWS. Untuk informasi selengkapnya, lihat Mengakses ElastiCache sumber daya dari luar AWS.

Untuk daftar versi yang didukung, lihatMesin dan versi yang didukung,Versi mesin Redis OSS yang didukung, danDidukung ElastiCache untuk versi Memcached.

Memilih jenis jaringan di ElastiCache

ElastiCache mendukung Internet Protocol versi 4 dan 6 (IPv4 dan IPv6), memungkinkan Anda untuk mengkonfigurasi cluster Anda untuk menerima:

- hanya IPv4 koneksi,
- hanya IPv6 koneksi,
- keduanya IPv4 dan IPv6 koneksi (dual-stack)

IPv6 didukung untuk beban kerja menggunakan Valkey 7.2 dan seterusnya, atau Redis OSS 6.2 dan seterusnya, pada semua instance yang dibangun di atas sistem Nitro. Tidak ada biaya tambahan untuk mengakses ElastiCache lebih IPv6.

> **Note**
>
> Migrasi cluster yang dibuat sebelum ketersediaan IPV6 /dual-stack tidak didukung. Beralih antarjenis jaringan pada klaster yang baru dibuat juga tidak didukung.

IPv6 didukung untuk beban kerja menggunakan Memcached 1.6.6 dan seterusnya pada semua instance yang dibangun di sistem Nitro. Tidak ada biaya tambahan untuk mengakses ElastiCache lebih IPv6.

Mengonfigurasi subnet untuk jenis jaringan

Jika Anda membuat cluster di VPC Amazon, Anda harus menentukan grup subnet. ElastiCache menggunakan grup subnet itu untuk memilih subnet dan alamat IP dalam subnet itu untuk dikaitkan dengan node Anda. ElastiCache cluster memerlukan subnet dual-stack dengan keduanya IPv4 dan IPv6 alamat yang ditetapkan untuk beroperasi dalam mode dual-stack dan subnet -only untuk beroperasi sebagai IPv6 -only. IPv6

Menggunakan tumpukan ganda

Saat menggunakan ElastiCache untuk Redis OSS dalam mode cluster diaktifkan, dari perspektif aplikasi, menghubungkan ke semua node cluster melalui titik akhir konfigurasi tidak berbeda dengan menghubungkan langsung ke node cache individu. Untuk mencapai hal ini, klien yang sadar klaster harus terlibat dalam proses penemuan klaster dan meminta informasi konfigurasi untuk semua simpul. Protokol penemuan Redis hanya mendukung satu IP per simpul.

Saat Anda membuat cluster cache dengan ElastiCache untuk Memcacheddan memilih dual-stack sebagai jenis jaringan, Anda kemudian perlu menunjuk jenis penemuan IP — salah satu atau. IPv4 IPv6 ElastiCache akan default jenis jaringan dan penemuan IP ke IPv6, tetapi itu dapat diubah. Jika Anda menggunakan Penemuan Otomatis, hanya alamat IP dari jenis IP yang Anda pilih yang dikembalikan ke klien Memcached. Untuk informasi selengkapnya, lihat Secara otomatis mengidentifikasi node di cluster Anda (Memcached).

Untuk mempertahankan kompatibilitas mundur dengan semua klien yang ada, penemuan IP diperkenalkan, yang memungkinkan Anda memilih jenis IP (yaitu, IPv4 atau IPv6) untuk beriklan di protokol penemuan. Meskipun ini membatasi penemuan otomatis hanya pada satu jenis IP, dual-stack masih bermanfaat untuk beban kerja yang diaktifkan mode cluster, karena memungkinkan migrasi (atau rollback) dari tipe IP IPv6 Discovery IPv4 ke tanpa downtime.

TLS mengaktifkan cluster tumpukan ElastiCache ganda

Ketika TLS diaktifkan untuk cluster fungsi penemuan cluster seperti`cluster slots`,`cluster shards`, dan `cluster nodes` dengan Valkey atau Redis OSS dan `config get cluster` dengan Memcached mengembalikan ElastiCache nama host alih-alih. IPs Nama host kemudian digunakan sebagai pengganti IPs untuk terhubung ke ElastiCache cluster dan melakukan jabat tangan TLS. Ini berarti bahwa klien tidak akan terpengaruh oleh parameter Penemuan IP. Untuk klaster dengan TLS diaktifkan, parameter Penemuan IP tidak berpengaruh pada protokol IP yang disukai. Sebagai gantinya, protokol IP yang digunakan akan ditentukan oleh protokol IP mana yang lebih dipilih klien saat meresolusi nama host DNS.

Untuk contoh tentang cara mengonfigurasi preferensi protokol IP saat meresolusi nama host DNS, lihat TLS mengaktifkan cluster tumpukan ElastiCache ganda.

Menggunakan AWS Management Console (Valkey dan Redis OSS)

Saat membuat cluster menggunakan AWS Management Console, di bawah Konektivitas, pilih jenis jaringan, baik IPv4, IPv6atau tumpukan ganda. Jika Anda membuat cluster Valkey atau Redis OSS (mode cluster enabled) dan memilih tumpukan ganda, Anda kemudian harus memilih jenis IP Discovery, salah satu atau. IPv6 IPv4

Untuk informasi selengkapnya, lihat Membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) (Konsol) atau Membuat Valkey atau Redis OSS (mode cluster dinonaktifkan) (Konsol).

Saat membuat grup replikasi menggunakan AWS Management Console, pilih jenis jaringan, baik IPv4, IPv6atau Dual stack. Jika Anda memilih tumpukan ganda, Anda harus memilih jenis IP Discovery, salah satu IPv6 atau IPv4.

Untuk informasi selengkapnya, lihat Membuat grup replikasi Valkey atau Redis OSS (Cluster Mode Disabled) dari awal atau Membuat grup replikasi di Valkey atau Redis OSS (Mode Cluster Diaktifkan) dari awal.

Menggunakan AWS Management Console (Memcached)

Saat membuat cluster cache menggunakan AWS Management Console, di bawah Konektivitas, pilih jenis jaringan, baik IPv4, IPv6atau tumpukan ganda. Jika Anda memilih tumpukan ganda, Anda harus memilih jenis IP Discovery, salah satu IPv6 atau IPv4.

Untuk informasi selengkapnya, lihat Membuat klaster Memcached (konsol).

Menggunakan CLI dengan Valkey, Memcached, atau Redis OSS.

Redis OSS

Saat membuat cluster cache dengan Valkey atau Redis OSS menggunakan CLI, Anda menggunakan create-cache-clusterperintah dan menentukan parameter dan: `NetworkType` `IPDiscovery`

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-cache-cluster \
    --cache-cluster-id "cluster-test" \
    --engine redis \
    --cache-node-type cache.m5.large \
    --num-cache-nodes 1 \
    --network-type dual_stack \
    --ip-discovery ipv4
```

Untuk Windows:

```
aws elasticache create-cache-cluster ^
    --cache-cluster-id "cluster-test" ^
    --engine redis ^
    --cache-node-type cache.m5.large ^
    --num-cache-nodes 1 ^
    --network-type dual_stack ^
    --ip-discovery ipv4
```

Saat membuat grup replikasi dengan mode cluster dinonaktifkan menggunakan CLI, Anda menggunakan create-replication-groupperintah dan menentukan `NetworkType` parameter dan: `IPDiscovery`

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
   --replication-group-id sample-repl-group \
   --replication-group-description "demo cluster with replicas" \
   --num-cache-clusters 3 \
   --primary-cluster-id redis01 \
   --network-type dual_stack \
   --ip-discovery ipv4
```

Untuk Windows:

```
aws elasticache create-replication-group ^
   --replication-group-id sample-repl-group ^
   --replication-group-description "demo cluster with replicas" ^
   --num-cache-clusters 3 ^
   --primary-cluster-id redis01 ^
   --network-type dual_stack ^
   --ip-discovery ipv4
```

Saat membuat grup replikasi dengan mode cluster diaktifkan dan digunakan IPv4 untuk penemuan IP menggunakan CLI, Anda menggunakan create-replication-groupperintah dan menentukan `NetworkType` parameter dan: `IPDiscovery`

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
  --replication-group-id demo-cluster \
  --replication-group-description "demo cluster" \
  --cache-node-type cache.m5.large \
  --num-node-groups 2 \
  --engine redis \
  --cache-subnet-group-name xyz \
  --network-type dual_stack \
  --ip-discovery ipv4 \
  --region us-east-1
```

Untuk Windows:

```
aws elasticache create-replication-group ^
  --replication-group-id demo-cluster ^
  --replication-group-description "demo cluster" ^
  --cache-node-type cache.m5.large ^
  --num-node-groups 2 ^
  --engine redis ^
  --cache-subnet-group-name xyz ^
  --network-type dual_stack ^
  --ip-discovery ipv4 ^
  --region us-east-1
```

Saat membuat grup replikasi dengan mode cluster diaktifkan dan digunakan IPv6 untuk penemuan IP menggunakan CLI, Anda menggunakan create-replication-groupperintah dan menentukan `NetworkType` parameter dan: `IPDiscovery`

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
  --replication-group-id demo-cluster \
  --replication-group-description "demo cluster" \
  --cache-node-type cache.m5.large \
  --num-node-groups 2 \
  --engine redis \
  --cache-subnet-group-name xyz \
  --network-type dual_stack \
  --ip-discovery ipv6 \
  --region us-east-1
```

Untuk Windows:

```
aws elasticache create-replication-group ^
  --replication-group-id demo-cluster ^
  --replication-group-description "demo cluster" ^
  --cache-node-type cache.m5.large ^
  --num-node-groups 2 ^
  --engine redis ^
  --cache-subnet-group-name xyz ^
  --network-type dual_stack ^
  --ip-discovery ipv6 ^
  --region us-east-1
```

Memcache

Saat membuat cluster cache dengan memcached menggunakan CLI, Anda menggunakan create-cache-clusterperintah dan menentukan parameter dan: `NetworkType` `IPDiscovery`

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-cache-cluster \
    --cache-cluster-id "cluster-test" \
    --engine memcached \
    --cache-node-type cache.m5.large \
    --num-cache-nodes 1 \
    --network-type dual_stack \
    --ip-discovery ipv4
```

Untuk Windows:

```
aws elasticache create-cache-cluster ^
    --cache-cluster-id "cluster-test" ^
    --engine memcached ^
    --cache-node-type cache.m5.large ^
    --num-cache-nodes 1 ^
    --network-type dual_stack ^
    --ip-discovery ipv4
```

Secara otomatis mengidentifikasi node di cluster Anda (Memcached)

Untuk cluster yang menjalankan mesin Memcached, ElastiCache mendukung Auto Discovery — kemampuan program klien untuk secara otomatis mengidentifikasi semua node dalam cluster cache, dan untuk memulai dan memelihara koneksi ke semua node ini.

> **Note**
>
> Penemuan Otomatis ditambahkan untuk cluster cache yang berjalan di Amazon ElastiCache Memcached. Auto Discovery tidak tersedia untuk mesin Valkey atau Redis OSS.

Dengan Penemuan Otomatis, aplikasi Anda tidak perlu secara manual terhubung ke tiap-tiap simpul cache; sebagai gantinya, aplikasi Anda terhubung ke satu simpul Memcached dan mengambil daftar

simpul. Dari daftar tersebut, aplikasi Anda akan mengetahui simpul yang tersisa di klaster dan dapat terhubung ke simpul yang mana pun. Anda tidak perlu melakukan hardcoding setiap titik akhir simpul cache ke dalam kode aplikasi Anda.

Jika Anda menggunakan tipe jaringan tumpukan ganda pada cluster Anda, Auto Discovery hanya akan mengembalikan IPv4 atau IPv6 alamat, tergantung pada mana yang Anda pilih. Untuk informasi selengkapnya, lihat Memilih jenis jaringan di ElastiCache.

Semua simpul cache di klaster memelihara daftar metadata tentang semua simpul lainnya. Metadata ini diperbarui setiap kali simpul ditambahkan atau dihapus dari klaster.

Topik

Manfaat Penemuan Otomatis dengan Memcached

Saat menggunakan Memcached, Auto Discovery menawarkan manfaat berikut:

- Ketika Anda menambah jumlah simpul dalam klaster cache, simpul yang baru akan mendaftarkan diri dengan titik akhir konfigurasi dan dengan semua simpul lainnya. Ketika Anda menghapus simpul dari klaster cache, simpul yang dihapus tersebut akan menghapus pendaftarannya sendiri. Dalam kedua kasus, semua simpul lain di klaster diperbarui dengan metadata simpul cache yang terbaru.
- Kegagalan simpul cache akan terdeteksi secara otomatis; simpul yang gagal diganti secara otomatis.

  > **Note**
  >
  > Hingga penggantian simpul selesai, simpul akan terus gagal.

- Program klien hanya perlu terhubung ke titik akhir konfigurasi. Setelah itu, pustaka Penemuan Otomatis terhubung ke semua simpul lain di klaster.
- Program klien melakukan polling klaster sekali setiap menit (interval ini dapat disesuaikan jika diperlukan). Jika ada perubahan pada konfigurasi klaster, seperti simpul yang baru atau dihapus, klien menerima daftar metadata yang diperbarui. Kemudian klien terhubung ke, atau memutuskan koneksi dari, simpul ini sesuai kebutuhan.

Auto Discovery diaktifkan pada semua cluster cache ElastiCache Memcached. Anda tidak perlu memulai ulang simpul cache Anda yang mana pun untuk menggunakan fitur ini.

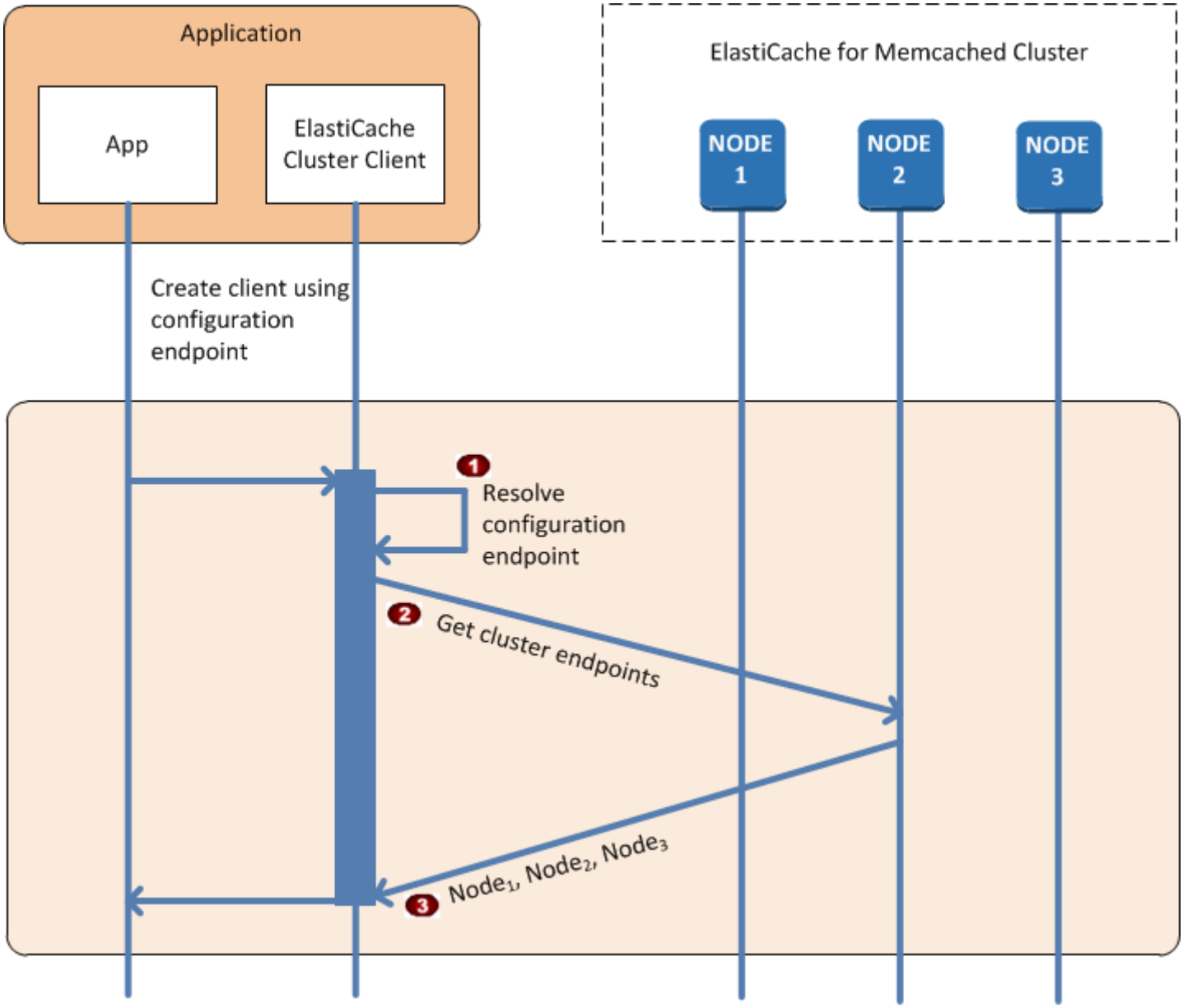
Cara Kerja Penemuan Otomatis

Topik

Bagian ini menjelaskan bagaimana aplikasi klien menggunakan Klien ElastiCache Cluster untuk mengelola koneksi node cache, dan berinteraksi dengan item data dalam cache.

Menyambung ke Simpul Cache

Dari sudut pandang aplikasi, terhubung ke titik akhir konfigurasi klaster tidak berbeda dengan terhubung secara langsung ke tiap-tiap simpul cache. Diagram urutan berikut menunjukkan proses menghubungkan ke simpul cache.



Proses Menghubungkan ke Simpul Cache

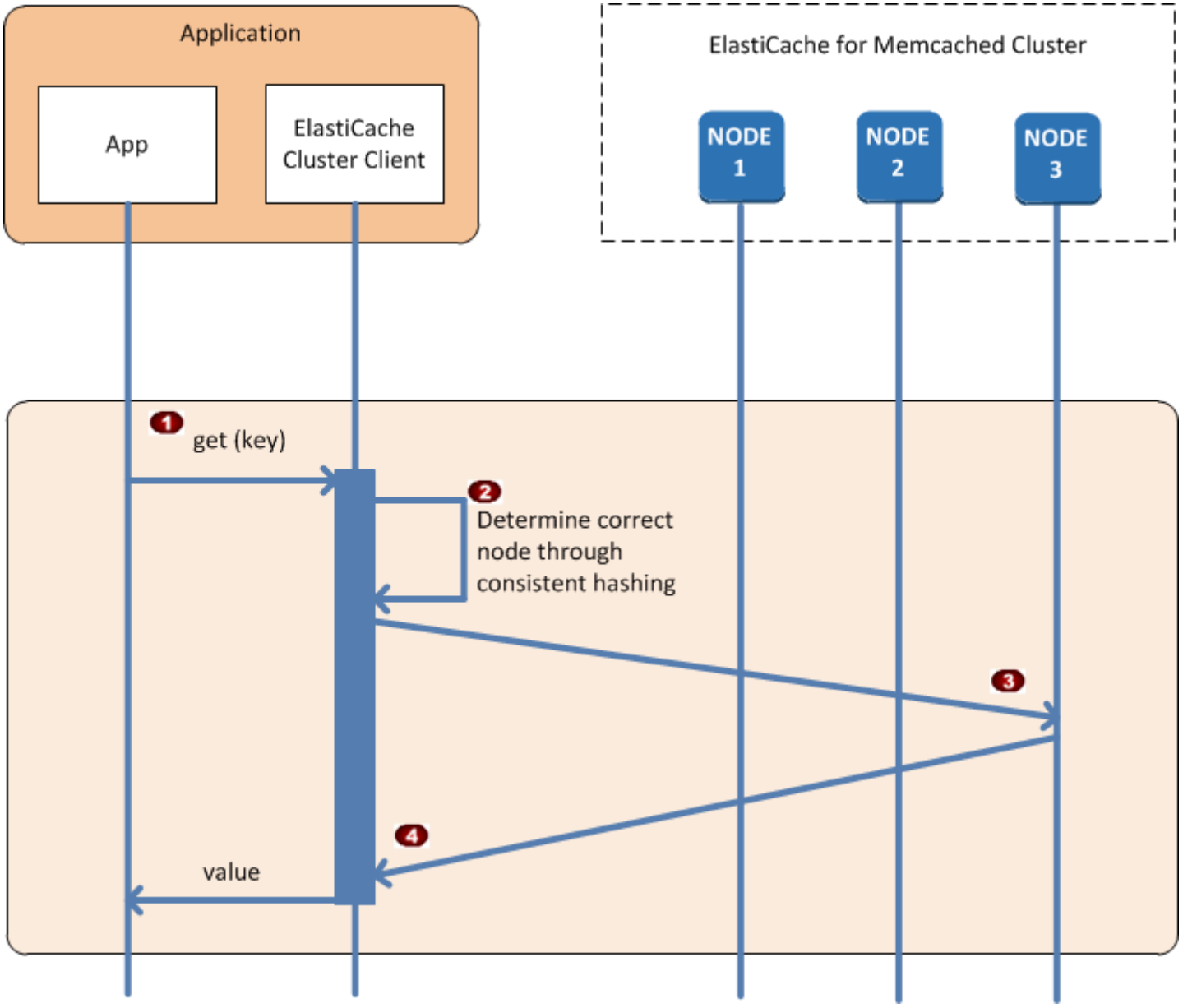
- Aplikasi meresolusi nama DNS dari titik akhir konfigurasi. Karena titik akhir konfigurasi memelihara entri CNAME untuk semua simpul cache, nama DNS diresolusi ke salah satu simpul; klien kemudian dapat terhubung ke simpul tersebut.
- Klien meminta informasi konfigurasi untuk semua simpul lainnya. Karena setiap simpul memelihara informasi konfigurasi untuk semua simpul di klaster, simpul mana pun dapat meneruskan informasi konfigurasi ke klien jika diminta.

- Klien menerima daftar terkini yang berisi nama host simpul cache dan alamat IP. Klien kemudian dapat terhubung ke semua simpul lain dalam klaster.

> **Note**
>
> Program klien menyegarkan daftar nama host simpul cache dan alamat IP sekali setiap menit. Interval polling ini dapat disesuaikan jika perlu.

Operasi Klaster Normal

Ketika aplikasi telah terhubung ke semua node cache, ElastiCache Cluster Client menentukan node mana yang harus menyimpan item data individual, dan node mana yang harus ditanyakan untuk item data tersebut nanti. Diagram urutan berikut menunjukkan proses operasi klaster normal.



Proses Operasi Klaster Normal

- Aplikasi ini mengeluarkan permintaan get untuk item data tertentu, yang teridentifikasi oleh kuncinya.
- Klien menggunakan algoritma hashing terhadap kunci untuk menentukan simpul cache yang berisi item data.
- Item data diminta dari simpul yang sesuai.
- Item data ditampilkan ke aplikasi.

Operasi Lainnya

Dalam beberapa situasi, Anda mungkin melakukan perubahan pada simpul klaster. Misalnya, Anda mungkin menambahkan simpul tambahan untuk mengakomodasi permintaan tambahan, atau menghapus simpul untuk menghemat uang selama periode berkurangnya permintaan. Atau Anda mungkin mengganti simpul karena kegagalan simpul atau hal lainnya.

Ketika ada perubahan dalam klaster yang memerlukan pembaruan metadata ke titik akhir klaster, perubahan tersebut dilakukan pada semua simpul secara bersamaan. Dengan demikian, metadata di simpul mana pun menjadi konsisten dengan metadata di semua simpul lain di klaster.

Dalam setiap kasus ini, metadata menjadi konsisten di antara semua simpul setiap saat karena metadata diperbarui pada saat yang sama untuk semua simpul dalam klaster. Anda harus selalu menggunakan titik akhir konfigurasi untuk memperoleh titik akhir dari berbagai simpul di klaster. Dengan titik akhir konfigurasi, Anda memastikan bahwa Anda tidak akan memperoleh data titik akhir dari simpul yang "menghilang".

Menambahkan Simpul

Pada saat simpul sedang dibuat, titik akhirnya tidak disertakan ke dalam metadata. Begitu sudah tersedia, simpul akan ditambahkan ke metadata dari setiap simpul klaster. Dalam skenario ini, metadata bersifat konsisten di antara semua simpul dan Anda akan dapat berinteraksi dengan simpul baru hanya setelah simpul itu tersedia. Sebelum simpul tersedia, Anda tidak akan mengetahuinya dan akan berinteraksi dengan simpul di klaster Anda seakan-akan simpul baru tersebut tidak ada.

Menghapus Simpul

Ketika simpul dihapus, pertama-tama titik akhirnya dihapus dari metadata lalu simpul tersebut dihapus dari klaster. Dalam skenario ini metadata di semua simpul bersifat konsisten dan tidak ada situasi saat metadata berisi titik akhir untuk simpul yang akan dihapus sementara simpul tersebut tidak tersedia. Selama waktu penghapusannya, simpul tidak dilaporkan dalam metadata dan aplikasi Anda hanya akan berinteraksi dengan simpul n-1 yang tersisa, seolah-olah simpul tersebut tidak ada.

Mengganti simpul

Jika sebuah node gagal ElastiCache , hapus node itu dan putar penggantinya. Proses penggantian membutuhkan waktu beberapa menit. Selama waktu itu, metadata di semua simpul masih menunjukkan titik akhir untuk simpul yang gagal, tetapi setiap percobaan untuk berinteraksi dengan simpul tersebut akan gagal. Oleh karena itu, logika Anda harus selalu menyertakan logika percobaan ulang.

Menggunakan Penemuan Otomatis

Untuk mulai menggunakan Auto Discovery with ElastiCache for Memcached, ikuti langkah-langkah berikut:

- Dapatkan Endpoint Konfigurasi
- Unduh Klien ElastiCache Cluster
- Ubah Program Aplikasi Anda

Dapatkan Endpoint Konfigurasi

Untuk terhubung ke klaster, program klien harus mengetahui titik akhir konfigurasi klaster. Lihat topik Menemukan Titik Akhir Cluster (Konsol) (Memcached).

Anda juga dapat menggunakan perintah `aws elasticache describe-cache-clusters` dengan parameter `--show-cache-node-info`:

Apa pun metode yang Anda gunakan untuk menemukan titik akhir klaster, titik akhir konfigurasi akan selalu memiliki .cfg pada alamatnya.

Example Menemukan titik akhir menggunakan for AWS CLI ElastiCache

Untuk Linux, macOS, atau Unix:

```
aws elasticache describe-cache-clusters \
    --cache-cluster-id mycluster \
    --show-cache-node-info
```

Untuk Windows:

```
aws elasticache describe-cache-clusters ^
    --cache-cluster-id mycluster ^
    --show-cache-node-info
```

Operasi ini menghasilkan output seperti yang berikut ini (format JSON):

```
{
    "CacheClusters": [
        {
            "Engine": "memcached",
            "CacheNodes": [
```

```
                {
                    "CacheNodeId": "0001",
                    "Endpoint": {
                        "Port": 11211,
                        "Address": "mycluster.fnjyzo.cfg.0001.use1.cache.amazonaws.com"
                    },
                    "CacheNodeStatus": "available",
                    "ParameterGroupStatus": "in-sync",
                    "CacheNodeCreateTime": "2016-10-12T21:39:28.001Z",
                    "CustomerAvailabilityZone": "us-east-1e"
                },
                {
                    "CacheNodeId": "0002",
                    "Endpoint": {
                        "Port": 11211,
                        "Address": "mycluster.fnjyzo.cfg.0002.use1.cache.amazonaws.com"
                    },
                    "CacheNodeStatus": "available",
                    "ParameterGroupStatus": "in-sync",
                    "CacheNodeCreateTime": "2016-10-12T21:39:28.001Z",
                    "CustomerAvailabilityZone": "us-east-1a"
                }
            ],
            "CacheParameterGroup": {
                "CacheNodeIdsToReboot": [],
                "CacheParameterGroupName": "default.memcached1.4",
                "ParameterApplyStatus": "in-sync"
            },
            "CacheClusterId": "mycluster",
            "PreferredAvailabilityZone": "Multiple",
            "ConfigurationEndpoint": {
                "Port": 11211,
                "Address": "mycluster.fnjyzo.cfg.use1.cache.amazonaws.com"
            },
            "CacheSecurityGroups": [],
            "CacheClusterCreateTime": "2016-10-12T21:39:28.001Z",
            "AutoMinorVersionUpgrade": true,
            "CacheClusterStatus": "available",
            "NumCacheNodes": 2,
            "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
            "CacheSubnetGroupName": "default",
            "EngineVersion": "1.4.24",
            "PendingModifiedValues": {},
```

```
            "PreferredMaintenanceWindow": "sat:06:00-sat:07:00",
            "CacheNodeType": "cache.r3.large"
        }
    ]
}
```

Unduh Klien ElastiCache Cluster

Untuk memanfaatkan Penemuan Otomatis, program klien harus menggunakan Klien Klaster ElastiCache. ElastiCache Cluster Client tersedia untuk Java, PHP, dan .NET dan berisi semua logika yang diperlukan untuk menemukan dan menghubungkan ke semua node cache Anda.

Untuk mengunduh Klien ElastiCache Cluster

1. Masuk ke Konsol AWS Manajemen dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari ElastiCache konsol, pilih ElastiCache Cluster Client lalu pilih Download.

Kode sumber untuk Klien ElastiCache Cluster untuk Java tersedia di https://github.com/amazonwebservices/aws-elasticache-cluster-client- memcached-for-java. Pustaka ini didasarkan pada klien Spymemcached yang populer. Klien ElastiCache Cluster dirilis di bawah Lisensi Perangkat Lunak https://aws.amazon.comAmazon/asl. Anda bebas mengubah kode sumber sesuai kebutuhan Anda. Anda bahkan dapat memasukkan kode ke pustaka Memcached sumber terbuka lainnya, atau ke kode klien Anda sendiri.

> **Note**
>
> Untuk menggunakan ElastiCache Cluster Client untuk PHP, pertama-tama Anda harus menginstalnya di EC2 instans Amazon Anda. Untuk informasi selengkapnya, lihat Menginstal klien ElastiCache cluster untuk PHP.
> Untuk klien yang didukung TLS, unduh biner dengan PHP versi 7.4 atau lebih tinggi.
> Untuk menggunakan ElastiCache Cluster Client untuk.NET, pertama-tama Anda harus menginstalnya di EC2 instans Amazon Anda. Untuk informasi selengkapnya, lihat Menginstal klien ElastiCache cluster untuk.NET.

Ubah Program Aplikasi Anda

Perubahan program aplikasi Anda sehingga menggunakan Penemuan Otomatis. Bagian berikut menunjukkan cara menggunakan ElastiCache Cluster Client untuk Java, PHP, dan .NET.

> **Important**
>
> Saat menentukan titik akhir konfigurasi klaster, pastikan bahwa titik akhir memiliki ".cfg" di alamatnya, seperti ditunjukkan di sini. Jangan menggunakan CNAME atau titik akhir tanpa ".cfg" dalamnya.
>
> ```
> "mycluster.fnjyzo.cfg.use1.cache.amazonaws.com";
> ```
>
> Kegagalan untuk secara eksplisit menentukan titik akhir konfigurasi klaster akan mengakibatkan konfigurasi yang dibuat ke simpul tertentu.

Menggunakan ElastiCache Cluster Client untuk Java

Program di bawah ini menunjukkan cara menggunakan ElastiCache Cluster Client untuk terhubung ke endpoint konfigurasi cluster dan menambahkan item data ke cache. Dengan Penemuan Otomatis, program ini terhubung ke semua simpul dalam klaster tanpa intervensi lebih lanjut.

```
package com.amazon.elasticache;

import java.io.IOException;
import java.net.InetSocketAddress;

// Import the &AWS;-provided library with Auto Discovery support
import net.spy.memcached.MemcachedClient;

public class AutoDiscoveryDemo {

    public static void main(String[] args) throws IOException {

        String configEndpoint = "mycluster.fnjyzo.cfg.use1.cache.amazonaws.com";
        Integer clusterPort = 11211;

        MemcachedClient client = new MemcachedClient(
                                 new InetSocketAddress(configEndpoint,
                                                       clusterPort));
```

```
        // The client will connect to the other cache nodes automatically.

        // Store a data item for an hour.
        // The client will decide which cache host will store this item.
        client.set("theKey", 3600, "This is the data value");
    }
}
```

Menggunakan ElastiCache Cluster Client untuk PHP

Program di bawah ini menunjukkan cara menggunakan ElastiCache Cluster Client untuk terhubung ke endpoint konfigurasi cluster dan menambahkan item data ke cache. Dengan Penemuan Otomatis, program ini akan terhubung ke semua simpul di klaster tanpa intervensi lebih lanjut.

Untuk menggunakan ElastiCache Cluster Client untuk PHP, pertama-tama Anda harus menginstalnya di EC2 instance Amazon Anda. Untuk informasi selengkapnya, lihat Menginstal klien ElastiCache cluster untuk PHP

```
<?php

 /**
  * Sample PHP code to show how to integrate with the Amazon ElastiCache
  * Auto Discovery feature.
  */

  /* Configuration endpoint to use to initialize memcached client.
   * This is only an example.  */
  $server_endpoint = "mycluster.fnjyzo.cfg.use1.cache.amazonaws.com";

  /* Port for connecting to the ElastiCache cluster.
   * This is only an example  */
  $server_port = 11211;

 /**
  * The following will initialize a Memcached client to utilize the Auto Discovery
 feature.
  *
  * By configuring the client with the Dynamic client mode with single endpoint, the
  * client will periodically use the configuration endpoint to retrieve the current
 cache
  * cluster configuration. This allows scaling the cache cluster up or down in number
 of nodes
  * without requiring any changes to the PHP application.
```

```
 *
 * By default the Memcached instances are destroyed at the end of the request.
 * To create an instance that persists between requests,
 *    use persistent_id to specify a unique ID for the instance.
 * All instances created with the same persistent_id will share the same connection.
 * See http://php.net/manual/en/memcached.construct.php for more information.
 */
 $dynamic_client = new Memcached('persistent-id');
 $dynamic_client->setOption(Memcached::OPT_CLIENT_MODE,
Memcached::DYNAMIC_CLIENT_MODE);
 $dynamic_client->addServer($server_endpoint, $server_port);

 /**
 * Store the data for 60 seconds in the cluster.
 * The client will decide which cache host will store this item.
 */
 $dynamic_client->set('key', 'value', 60);


/**
 * Configuring the client with Static client mode disables the usage of Auto Discovery
 * and the client operates as it did before the introduction of Auto Discovery.
 * The user can then add a list of server endpoints.
 */
 $static_client = new Memcached('persistent-id');
 $static_client->setOption(Memcached::OPT_CLIENT_MODE, Memcached::STATIC_CLIENT_MODE);
 $static_client->addServer($server_endpoint, $server_port);

/**
 * Store the data without expiration.
 * The client will decide which cache host will store this item.
 */
 $static_client->set('key', 'value');
 ?>
```

Untuk contoh tentang cara menggunakan Klien ElastiCache Cluster dengan TLS diaktifkan, lihat Menggunakan enkripsi transit dengan PHP dan Memcached.

Menggunakan ElastiCache Cluster Client untuk.NET

> **Note**
>
> Klien klaster ElastiCache .NET telah tidak digunakan lagi per Mei 2022.

Klien.NET untuk ElastiCache adalah open source di https://github.com/awslabs/elasticache-cluster-config-net.

Aplikasi .NET biasanya mendapatkan konfigurasi dari file config yang dimilikinya. Berikut ini adalah contoh file config aplikasi.

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <configSections>
        <section
            name="clusterclient"
            type="Amazon.ElastiCacheCluster.ClusterConfigSettings,
 Amazon.ElastiCacheCluster" />
    </configSections>

    <clusterclient>
        <!-- the hostname and port values are from step 1 above -->
        <endpoint hostname="mycluster.fnjyzo.cfg.use1.cache.amazonaws.com"
 port="11211" />
    </clusterclient>
</configuration>
```

Program C# di bawah ini menunjukkan cara menggunakan Klien ElastiCache Cluster untuk terhubung ke titik akhir konfigurasi cluster dan menambahkan item data ke cache. Dengan Penemuan Otomatis, program ini akan terhubung ke semua simpul di klaster tanpa intervensi lebih lanjut.

```
// ****************
// Sample C# code to show how to integrate with the Amazon ElastiCcache Auto Discovery
 feature.

using System;

using Amazon.ElastiCacheCluster;
```

```
using Enyim.Caching;
using Enyim.Caching.Memcached;

public class DotNetAutoDiscoveryDemo  {

    public static void Main(String[] args)  {

        // instantiate a new client.
        ElastiCacheClusterConfig config = new ElastiCacheClusterConfig();
        MemcachedClient memClient = new MemcachedClient(config);

        // Store the data for 3600 seconds (1hour) in the cluster.
        // The client will decide which cache host will store this item.
        memClient.Store(StoreMode.Set, 3600, "This is the data value.");

    }  // end Main

}  // end class DotNetAutoDiscoverDemo
```

Menghubungkan ke Memcached Cache Node Secara Manual

Jika program klien Anda tidak menggunakan Auto Discovery, program ini dapat terhubung secara manual ke masing-masing node cache Memcached. Ini adalah perilaku default untuk klien Memcached.

Anda dapat memperoleh daftar nama host dan nomor port simpul cache dari Konsol Manajemen AWS. Anda juga dapat menggunakan AWS CLI `aws elasticache describe-cache-clusters` perintah dengan `--show-cache-node-info` parameter.

Example

Cuplikan kode Java berikut menunjukkan cara terhubung ke semua simpul di klaster cache empat simpul:

```
...

ArrayList<String> cacheNodes = new ArrayList<String>(
 Arrays.asList(
     "mycachecluster.fnjyzo.0001.use1.cache.amazonaws.com:11211",
     "mycachecluster.fnjyzo.0002.use1.cache.amazonaws.com:11211",
     "mycachecluster.fnjyzo.0003.use1.cache.amazonaws.com:11211",
     "mycachecluster.fnjyzo.0004.use1.cache.amazonaws.com:11211"));

MemcachedClient cache = new MemcachedClient(AddrUtil.getAddresses(cacheNodes));

...
```

Important

Jika Anda menaikkan atau menurunkan skala klaster cache Anda dengan menambahkan atau menghapus simpul, Anda akan perlu memperbarui daftar simpul dalam kode klien.

Menambahkan Penemuan Otomatis ke pustaka klien Memcached Anda

Informasi konfigurasi untuk Auto Discovery disimpan secara berlebihan di setiap node cluster cache Memcached. Aplikasi klien dapat mengueri simpul cache apa pun dan memperoleh informasi konfigurasi untuk semua simpul di klaster tersebut.

Cara sebuah aplikasi melakukannya akan tergantung pada versi mesin cache:

- Jika versi mesin cache adalah 1.4.14 atau lebih tinggi, gunakan perintah `config`.
- Jika versi mesin cache lebih rendah dari 1.4.14, gunakan perintah `get AmazonElastiCache:cluster`.

Output dari kedua perintah ini identik, dan dijelaskan pada bagian Format Output di bawah ini.

Cache engine versi 1.4.14 atau lebih tinggi

Untuk versi mesin cache 1.4.14 atau lebih tinggi, gunakan perintah `config`. Perintah ini telah ditambahkan ke ASCII Memcached dan protokol biner oleh ElastiCache, dan diimplementasikan di Cluster Client. ElastiCache Jika Anda ingin menggunakan Penemuan Otomatis dengan pustaka klien lain, maka pustaka itu akan perlu diperluas untuk mendukung perintah `config`.

> **Note**
>
> Dokumentasi berikut berkaitan dengan protokol ASCII; namun, perintah `config` mendukung baik ASCII maupun biner. Jika Anda ingin menambahkan dukungan Auto Discovery menggunakan protokol biner, lihat kode sumber untuk Klien ElastiCache Cluster.

Sintaksis

```
config [sub-command] [key]
```

Opsi

Nama	Deskripsi	Wajib
`sub-command`	Sub-perintah yang digunakan untuk berinteraksi dengan simpul cache. Untuk Penemuan Otomatis, sub-perintah ini adalah `get`.	Ya

Nama	Deskripsi	Wajib
`key`	Kunci yang digunakan untuk menyimpan konfigurasi klaster. Untuk Penemuan Otomatis, kunci ini disebut `cluster`.	Ya

Untuk mendapatkan informasi konfigurasi klaster, gunakan perintah berikut:

```
config get cluster
```

Cache engine versi 1.4.14 atau lebih rendah

Untuk mendapatkan informasi konfigurasi klaster, gunakan perintah berikut:

```
get AmazonElastiCache:cluster
```

> **Note**
>
> Jangan mengutak-atik tombol ": clusterAmazonElastiCache", karena di sinilah informasi konfigurasi cluster berada. Jika Anda menimpa kunci ini, maka klien mungkin salah dikonfigurasi untuk jangka waktu singkat (tidak lebih dari 15 detik) sebelum memperbarui informasi konfigurasi ElastiCache secara otomatis dan benar.

Format Output

Tergantung pada Anda menggunakan `config get cluster` atau `get AmazonElastiCache:cluster`, balasannya terdiri dari dua baris:

- Nomor versi informasi konfigurasi. Setiap kali simpul ditambahkan atau dihapus dari klaster cache, nomor versi bertambah satu angka.
- Daftar simpul cache. Setiap simpul dalam daftar dinyatakan dengan grup hostname|ip-address|port, dan setiap simpul dibatasi dengan spasi.

Karakter carriage return dan linefeed (CR+LF) muncul di akhir setiap baris. Baris data mengandung karakter linefeed (LF) di bagian akhir, tempat CR + LF ditambahkan. Baris versi konfigurasi diakhiri dengan LF tanpa CR.

Klaster cache yang berisi tiga simpul akan dinyatakan sebagai berikut:

```
configversion\n
hostname|ip-address|port hostname|ip-address|port hostname|ip-address|port\n\r\n
```

Setiap simpul ditampilkan dengan CNAME dan alamat IP privat. CNAME akan selalu ada; jika alamat IP privat tidak tersedia, maka tidak akan ditampilkan; namun, karakter pipa "|" akan tetap dicetak.

Example

Berikut adalah contoh payload yang ditampilkan ketika Anda membuat kueri informasi konfigurasi:

```
CONFIG cluster 0 136\r\n
12\n
myCluster.pc4ldq.0001.use1.cache.amazonaws.com|10.82.235.120|11211
 myCluster.pc4ldq.0002.use1.cache.amazonaws.com|10.80.249.27|11211\n\r\n
END\r\n
```

> **Note**
>
> - Baris kedua menunjukkan bahwa informasi konfigurasi telah diubah dua belas kali sampai saat ini.
> - Pada baris ketiga, daftar simpul ditampilkan dalam urutan abjad berdasarkan nama host. Pengurutan ini mungkin berbeda dengan yang Anda gunakan saat ini pada aplikasi klien Anda.

ElastiCache klien dengan penemuan otomatis

Program klien cluster dapat secara otomatis mengidentifikasi dan terhubung ke semua node cluster cache yang menjalankan mesin Memcached.

Bagian ini membahas menginstal dan mengkonfigurasi klien ElastiCache PHP dan .NET untuk digunakan dengan auto discovery.

Topik

- Menginstal & mengompilasi klien klaster
- Mengkonfigurasi klien ElastiCache

Menginstal & mengompilasi klien klaster

Bagian ini mencakup penginstalan, konfigurasi, dan kompilasi klien cluster penemuan otomatis PHP dan .NET Amazon ElastiCache auto.

Topik

- Menginstal klien ElastiCache cluster untuk.NET
- Menginstal klien ElastiCache cluster untuk PHP
- Mengompilasi kode sumber untuk klien ElastiCache cluster untuk PHP

Menginstal klien ElastiCache cluster untuk.NET

Anda dapat menemukan kode ElastiCache Klien.NET Cluster sebagai open source di https://github.com/awslabs/elasticache-cluster-config-net.

Bagian ini menjelaskan cara menginstal, memperbarui, dan menghapus komponen.NET untuk Klien ElastiCache Cluster di EC2 instans Amazon. Untuk informasi lain mengenai penemuan otomatis, lihat Secara otomatis mengidentifikasi node di cluster Anda (Memcached). Untuk contoh kode .NET untuk menggunakan klien tersebut, lihat Menggunakan ElastiCache Cluster Client untuk.NET.

Topik

- Menginstal .NET
- Unduh ElastiCache klien.NET cluster untuk ElastiCache
- Instal AWS rakitan dengan NuGet

Menginstal .NET

Anda harus memiliki .NET 3.5 atau yang lebih baru diinstal untuk menggunakan AWS .NET SDK untuk ElastiCache. Jika Anda tidak memiliki .NET 3.5 atau yang lebih baru, Anda dapat mengunduh dan menginstal versi terbaru dari http://www.microsoft.com/net.

Unduh ElastiCache klien.NET cluster untuk ElastiCache

Untuk mengunduh klien klaster ElastiCache .NET

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.

2. Pada panel navigasi, klik Klien ElastiCache Cluster.
3. Dalam daftar Download ElastiCache Memcached Cluster Client, pilih .NET, dan kemudian klik Download.

Instal AWS rakitan dengan NuGet

NuGet adalah sistem manajemen paket untuk platform.NET. NuGet mengetahui dependensi perakitan dan menginstal semua file yang diperlukan secara otomatis. NuGet rakitan terinstal disimpan dengan solusi Anda, bukan di lokasi pusat seperti`Program Files`, sehingga Anda dapat menginstal versi khusus untuk aplikasi tanpa membuat masalah kompatibilitas.

Instalasi NuGet

NuGet dapat diinstal dari Galeri Instalasi di MSDN; lihat https://visualstudiogallery.msdn.microsoft.com /27077b70-9dad-4c64-adcf-c7cf6bc9970c. Jika Anda menggunakan Visual Studio 2010 atau yang lebih baru, NuGet secara otomatis diinstal.

Anda dapat menggunakan NuGet dari Solution Explorer atau Package Manager Console.

Menggunakan NuGet dari Solution Explorer

Untuk menggunakan NuGet dari Solution Explorer di Visual Studio 2010

1. Dari menu Tools, pilih Library Package Manager.
2. Klik Package Manager Console.

Untuk menggunakan NuGet dari Solution Explorer di Visual Studio 2012 atau Visual Studio 2013

1. Dari menu Tools, pilih NuGet Package Manager.
2. Klik Package Manager Console.

Dari baris perintah, Anda bisa menginstal kompilasi menggunakan `Install-Package`, seperti yang ditunjukkan berikut ini.

```
Install-Package Amazon.ElastiCacheCluster
```

Untuk melihat halaman untuk setiap paket yang tersedia NuGet, seperti rakitan AWS SDK dan AWS.Extensions, lihat situs web di NuGet http://www.nuget.org. Halaman untuk setiap paket

menyertakan contoh baris perintah untuk menginstal paket menggunakan konsol dan daftar versi paket sebelumnya yang tersedia melaluiNuGet.

Untuk informasi selengkapnya tentang perintah Package Manager Console, lihat http://nuget.codeplex.com/wikipage?title=Package%20Manager%20Console%20Command%20Reference%20%28v1.3%29.

Menginstal klien ElastiCache cluster untuk PHP

Bagian ini menjelaskan cara menginstal, memperbarui, dan menghapus komponen PHP untuk Klien ElastiCache Cluster di EC2 instance Amazon. Untuk informasi selengkapnya tentang Penemuan Otomatis, lihat Secara otomatis mengidentifikasi node di cluster Anda (Memcached). Untuk contoh kode PHP untuk menggunakan klien, lihat Menggunakan ElastiCache Cluster Client untuk PHP.

Topik

Mengunduh paket penginstalan

Untuk memastikan bahwa Anda menggunakan versi ElastiCache Cluster Client yang benar untuk PHP, Anda perlu mengetahui versi PHP apa yang diinstal pada EC2 instans Amazon Anda. Anda juga perlu mengetahui apakah EC2 instans Amazon Anda menjalankan Linux versi 64-bit atau 32-bit.

Untuk menentukan versi PHP yang diinstal pada EC2 instans Amazon Anda

- Pada jendela perintah, jalankan perintah berikut:

```
php -v
```

Versi PHP akan ditampilkan pada output, seperti pada contoh ini:

```
PHP 5.4.10 (cli) (built: Jan 11 2013 14:48:57)
Copyright (c) 1997-2012 The PHP Group
Zend Engine v2.4.0, Copyright (c) 1998-2012 Zend Technologies
```

> **Note**
>
> Jika versi PHP dan Memcached Anda tidak kompatibel, Anda akan mendapatkan pesan kesalahan seperti berikut:
>
> ```
> PHP Warning: PHP Startup: memcached: Unable to initialize module
> Module compiled with module API=20100525
> ```

```
PHP compiled with module API=20131226
These options need to match
in Unknown on line 0
```

Jika ini terjadi, Anda perlu mengompilasi modul dari kode sumber. Untuk informasi selengkapnya, lihat Mengompilasi kode sumber untuk klien ElastiCache cluster untuk PHP.

Untuk menentukan arsitektur Amazon EC2 AMI Anda (64-bit atau 32-bit)

1. Masuk ke AWS Management Console dan buka EC2 konsol Amazon di https://console.aws.amazon.com/ec2/.
2. Dalam daftar Instans, klik EC2 instans Amazon Anda.
3. Pada tab Deskripsi, cari bidang AMI:. Instans 64-bit seharusnya memiliki `x86_64` sebagai bagian dari deskripsinya; untuk instans 32-bit, cari `i386` atau `i686` pada bidang ini.

Anda sekarang siap untuk mengunduh ElastiCache Cluster Client.

Untuk mengunduh klien ElastiCache cluster untuk PHP

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari ElastiCache konsol, pilih ElastiCache Cluster Client.
3. Dari daftar Download ElastiCache Memcached Cluster Client, pilih ElastiCache Cluster Client yang cocok dengan versi PHP dan arsitektur AMI Anda, lalu pilih tombol Download.

Bagi pengguna yang sudah memiliki ekstensi php-memcached terinstal

Untuk memperbarui instalan **`php-memcached`**

1. Hapus instalasi sebelumnya dari ekstensi Memcached untuk PHP seperti yang dijelaskan pada topik Menghapus klien klaster PHP.
2. Instal ElastiCache `php-memcached` ekstensi baru seperti yang dijelaskan sebelumnya diLangkah penginstalan untuk pengguna baru.

Langkah penginstalan untuk pengguna baru

Topik

- Menginstal PHP 7.x untuk pengguna baru
- Menginstal PHP 5.x untuk pengguna baru

Menginstal PHP 7.x untuk pengguna baru

Topik

- Untuk menginstal PHP 7 pada server Ubuntu 14.04 LTS AMI (64-bit dan 32-bit)
- Untuk menginstal PHP 7 pada AMI Amazon Linux 201609
- Untuk menginstal PHP 7 pada AMI SUSE Linux

Untuk menginstal PHP 7 pada server Ubuntu 14.04 LTS AMI (64-bit dan 32-bit)

1. Luncurkan instans baru dari AMI.
2. Jalankan perintah berikut:

```
sudo apt-get update
sudo apt-get install gcc g++
```

3. Instal PHP.7.

```
sudo yum install php70
```

4. Unduh Klien ElastiCache Cluster Amazon.

```
wget https://elasticache-downloads.s3.amazonaws.com/ClusterClient/PHP-7.0/
latest-64bit
```

5. Ekstrak file `latest-64bit`.

```
tar -zxvf latest-64bit
```

6. Dengan izin root, salin file artefak `amazon-elasticache-cluster-client.so` yang diekstrak ke `/usr/lib/php/20151012`.

```
sudo mv artifact/amazon-elasticache-cluster-client.so /usr/lib/php/20151012
```

7. Sisipkan baris `extension=amazon-elasticache-cluster-client.so` ke file `/etc/php/7.0/cli/php.ini`.

```
echo "extension=amazon-elasticache-cluster-client.so" | sudo tee --append /etc/
php/7.0/cli/php.ini
```

8. Mulai atau mulai ulang server Apache Anda.

```
sudo /etc/init.d/httpd start
```

Untuk menginstal PHP 7 pada AMI Amazon Linux 201609

1. Luncurkan instans baru dari AMI tersebut.
2. Jalankan perintah berikut:

```
sudo yum install gcc-c++
```

3. Instal PHP.7.

```
sudo yum install php70
```

4. Unduh Klien ElastiCache Cluster Amazon.

```
wget https://elasticache-downloads.s3.amazonaws.com/ClusterClient/PHP-7.0/
latest-64bit
```

5. Ekstrak file `latest-64bit`.

```
tar -zxvf latest-64bit
```

6. Dengan izin root, salin file artefak `amazon-elasticache-cluster-client.so` yang diekstrak ke `/usr/lib64/php/7.0/modules/`.

```
sudo mv artifact/amazon-elasticache-cluster-client.so /usr/lib64/php/7.0/modules/
```

7. Buat file `50-memcached.ini`.

```
echo "extension=amazon-elasticache-cluster-client.so" | sudo tee --append /etc/
php-7.0.d/50-memcached.ini
```

8. Mulai atau mulai ulang server Apache Anda.

```
sudo /etc/init.d/httpd start
```

Untuk menginstal PHP 7 pada AMI SUSE Linux

1. Luncurkan instans baru dari AMI tersebut.
2. Jalankan perintah berikut:

```
sudo zypper install gcc
```

3. Instal PHP.7.

```
sudo yum install php70
```

4. Unduh Klien ElastiCache Cluster Amazon.

```
wget https://elasticache-downloads.s3.amazonaws.com/ClusterClient/PHP-7.0/
latest-64bit
```

5. Ekstrak file `latest-64bit`.

```
tar -zxvf latest-64bit
```

6. Dengan izin root, salin file artefak `amazon-elasticache-cluster-client.so` yang diekstrak ke `/usr/lib64/php7/extensions/`.

```
sudo mv artifact/amazon-elasticache-cluster-client.so /usr/lib64/php7/extensions/
```

7. Sisipkan baris `extension=amazon-elasticache-cluster-client.so` ke file `/etc/php7/cli/php.ini`.

```
echo "extension=amazon-elasticache-cluster-client.so" | sudo tee --append /etc/
php7/cli/php.ini
```

8. Mulai atau mulai ulang server Apache Anda.

```
sudo /etc/init.d/httpd start
```

Menginstal PHP 5.x untuk pengguna baru

Topik

- Untuk menginstal PHP 5 pada AMI Amazon Linux 2014.03 (64-bit and 32-bit)
- Untuk menginstal PHP 5 pada AMI Red Hat Enterprise Linux 7.0 (64-bit dan 32-bit)
- Untuk menginstal PHP 5 pada AMI server Ubuntu 14.04 LTS (64-bit dan 32-bit)
- Untuk menginstal PHP 5 untuk AMI SUSE Linux enterprise server 11 (64-bit o 32-bit)
- Distribusi Linux lainnya

Untuk menginstal PHP 5 pada AMI Amazon Linux 2014.03 (64-bit and 32-bit)

1. Luncurkan instans Amazon Linux (baik 64-bit maupun 32-bit) dan login ke dalamnya.
2. Instal dependensi PHP:

```
sudo yum install gcc-c++ php php-pear
```

3. Unduh `php-memcached` paket yang benar untuk EC2 instans Amazon dan versi PHP Anda. Untuk informasi selengkapnya, lihat Mengunduh paket penginstalan.
4. Instal `php-memcached`. URI tersebut seharusnya adalah jalur pengunduhan untuk paket penginstalan:

```
sudo pecl install <package download path>
```

   Berikut adalah contoh perintah penginstalan untuk PHP 5.4, 64-bit Linux. Dalam contoh ini, ganti *X.Y.Z* dengan nomor versi sebenarnya:

```
sudo pecl install /home/AmazonElastiCacheClusterClient-X.Y.Z-PHP54-64bit.tgz
```

> **Note**
>
> Pastikan untuk menggunakan versi terbaru artefak penginstalan.

5. Dengan root/sudo izin, tambahkan file baru bernama `memcached.ini` dalam `/etc/php.d` direktori, dan masukkan “amazon-elasticache-cluster-clientextension=.so” dalam file:

```
echo "extension=amazon-elasticache-cluster-client.so" | sudo tee --append /etc/
php.d/memcached.ini
```

6. Mulai atau mulai ulang server Apache Anda.

```
sudo /etc/init.d/httpd start
```

Untuk menginstal PHP 5 pada AMI Red Hat Enterprise Linux 7.0 (64-bit dan 32-bit)

1. Luncurkan instans Red Hat Enterprise Linux (baik 64-bit maupun 32-bit) dan login ke dalamnya.
2. Instal dependensi PHP:

```
sudo yum install gcc-c++ php php-pear
```

3. Unduh `php-memcached` paket yang benar untuk EC2 instans Amazon dan versi PHP Anda. Untuk informasi selengkapnya, lihat Mengunduh paket penginstalan.
4. Instal `php-memcached`. URI tersebut seharusnya adalah jalur pengunduhan untuk paket penginstalan:

```
sudo pecl install <package download path>
```

5. Dengan root/sudo izin, tambahkan file baru bernama `memcached.ini` dalam `/etc/php.d` direktori, dan masukkan `extension=amazon-elasticache-cluster-client.so` dalam file.

```
echo "extension=amazon-elasticache-cluster-client.so" | sudo tee --append /etc/
php.d/memcached.ini
```

6. Mulai atau mulai ulang server Apache Anda.

```
sudo /etc/init.d/httpd start
```

Untuk menginstal PHP 5 pada AMI server Ubuntu 14.04 LTS (64-bit dan 32-bit)

1. Luncurkan instans Ubuntu Linux (baik 64-bit maupun 32-bit) dan login ke dalamnya.
2. Instal dependensi PHP:

```
sudo apt-get update
sudo apt-get install gcc g++ php5 php-pear
```

3. Unduh `php-memcached` paket yang benar untuk EC2 instans Amazon dan versi PHP Anda. Untuk informasi selengkapnya, lihat Mengunduh paket penginstalan.
4. Instal `php-memcached`. URI tersebut seharusnya adalah jalur pengunduhan untuk paket penginstalan.

```
sudo pecl install <package download path>
```

> **Note**
>
> Langkah penginstalan ini menginstal artefak build `amazon-elasticache-cluster-client.so` ke dalam direktori `/usr/lib/php5/20121212*`. Verifikasikan jalur absolut artefak build karena Anda membutuhkannya pada langkah berikutnya.

Jika perintah sebelumnya tidak berfungsi, Anda perlu secara manual mengekstrak artefak klien PHP `amazon-elasticache-cluster-client.so` dari file `*.tgz` unduhan, dan menyalinnya ke direktori `/usr/lib/php5/20121212*`.

```
tar -xvf <package download path>
cp amazon-elasticache-cluster-client.so /usr/lib/php5/20121212/
```

5. Dengan root/sudo izin, tambahkan file baru bernama `memcached.ini` di `/etc/php5/cli/conf.d` direktori, dan masukkan “extension=<absolute path amazon-elasticache-cluster-client ke.so>” dalam file.

```
echo "extension=<absolute path to amazon-elasticache-cluster-client.so>" | sudo tee
 --append /etc/php5/cli/conf.d/memcached.ini
```

6. Mulai atau mulai ulang server Apache Anda.

```
sudo /etc/init.d/httpd start
```

Untuk menginstal PHP 5 untuk AMI SUSE Linux enterprise server 11 (64-bit o 32-bit)

1. Luncurkan instans SUSE Linux (baik 64-bit maupun 32-bit) dan login ke dalamnya.
2. Instal dependensi PHP:

```
sudo zypper install gcc php53-devel
```

3. Unduh `php-memcached` paket yang benar untuk EC2 instans Amazon dan versi PHP Anda. Untuk informasi selengkapnya, lihat Mengunduh paket penginstalan.
4. Instal `php-memcached`. URI tersebut seharusnya adalah jalur pengunduhan untuk paket penginstalan.

```
sudo pecl install <package download path>
```

5. Dengan root/sudo izin, tambahkan file baru bernama `memcached.ini` dalam `/etc/php5/conf.d` direktori, dan masukkan **`extension=amazon-elasticache-cluster-client.so`** dalam file.

```
echo "extension=amazon-elasticache-cluster-client.so" | sudo tee --append /etc/
php5/conf.d/memcached.ini
```

6. Mulai atau mulai ulang server Apache Anda.

```
sudo /etc/init.d/httpd start
```

> **Note**
>
> Jika Langkah 5 tidak berfungsi untuk platform apa pun sebelumnya, verifikasi jalur penginstalan untuk `amazon-elasticache-cluster-client.so`. Juga, tentukan jalur lengkap biner dalam ekstensi. Selain itu, verifikasikan bahwa PHP yang digunakan adalah versi yang didukung. Kami mendukung versi 5.3 hingga 5.5.

Distribusi Linux lainnya

Pada beberapa sistem, terutama Cent OS7 dan Red Hat Enterprise Linux (RHEL) 7.1, `libsasl2.so.3` telah diganti. `libsasl2.so.2` Pada sistem tersebut, ketika Anda memuat klien ElastiCache cluster, ia mencoba dan gagal menemukan dan memuat`libsasl2.so.2`. Untuk mengatasi masalah ini, buat tautan simbolis ke `libsasl2.so.3` sehingga ketika klien mencoba memuat libsasl2.so.2, sistem diarahkan ke `libsasl2.so.3`. Kode berikut membuat tautan simbolis ini.

```
cd /usr/lib64
sudo ln libsasl2.so.3 libsasl2.so.2
```

Menghapus klien klaster PHP

Topik

- Menghapus versi sebelumnya dari PHP 7
- Menghapus versi lama PHP 5

Menghapus versi sebelumnya dari PHP 7

Untuk menghapus versi sebelumnya dari PHP 7

1. Hapus file `amazon-elasticache-cluster-client.so` dari direktori pustaka PHP yang sesuai seperti yang ditunjukkan sebelumnya pada petunjuk instalasi. Lihat bagian untuk penginstalan Anda di Bagi pengguna yang sudah memiliki ekstensi php-memcached terinstal.
2. Hapus baris `extension=amazon-elasticache-cluster-client.so` dari file `php.ini`.
3. Mulai atau mulai ulang server Apache Anda.

```
sudo /etc/init.d/httpd start
```

Menghapus versi lama PHP 5

Untuk menghapus versi lama PHP 5

1. Hapus ekstensi `php-memcached`:

   ```
   sudo pecl uninstall __uri/AmazonElastiCacheClusterClient
   ```

2. Hapus file `memcached.ini` yang ditambahkan pada direktori yang sesuai seperti yang ditunjukkan pada langkah penginstalan sebelumnya.

Mengompilasi kode sumber untuk klien ElastiCache cluster untuk PHP

Bagian ini mencakup cara mendapatkan dan mengkompilasi kode sumber untuk ElastiCache Cluster Client untuk PHP.

Ada dua paket yang perlu Anda tarik GitHub dan kompilasi; aws-elasticache-cluster-client-libmemcached dan -. aws-elasticache-cluster-client memcached-for-php

Topik

- Mengompilasi pustaka libmemcached
- Mengompilasi klien penemuan otomatis ElastiCache Memcached untuk PHP

Mengompilasi pustaka libmemcached

Untuk mengkompilasi perpustakaan aws-elasticache-cluster-client -libmemcached

1. Luncurkan EC2 instance Amazon.
2. Instal pustaka dependensi.
   - Pada AMI Amazon Linux 201509

     ```
     sudo yum install gcc gcc-c++ autoconf libevent-devel
     ```

   - Pada AMI Ubuntu 14.04

     ```
     sudo apt-get update
     sudo apt-get install libevent-dev gcc g++ make autoconf libsasl2-dev
     ```

3. Tarik repositori dan kompilasikan kode.

   ```
   Download and install  https://github.com/awslabs/aws-elasticache-cluster-client-
   libmemcached/archive/v1.0.18.tar.gz
   ```

Mengompilasi klien penemuan otomatis ElastiCache Memcached untuk PHP

Bagian berikut menjelaskan cara mengkompilasi ElastiCache Memcached Auto Discovery Client

Topik

- Mengompilasi klien ElastiCache Memcached untuk PHP 7

- Mengompilasi klien ElastiCache Memcached untuk PHP 5

Mengompilasi klien ElastiCache Memcached untuk PHP 7

Jalankan kelompok perintah berikut di bawah direktori kode.

```
git clone https://github.com/awslabs/aws-elasticache-cluster-client-memcached-for-
php.git
cd aws-elasticache-cluster-client-memcached-for-php
git checkout php7
sudo yum install php70-devel
phpize
./configure --with-libmemcached-dir=<libmemcached-install-directory> --disable-
memcached-sasl
make
make install
```

> **Note**
>
> Anda secara statis dapat menghubungkan pustaka libmemcached ke biner PHP sehingga dapat diporting di berbagai platform Linux. Untuk melakukan hal itu, jalankan dahulu perintah berikut `make`:
>
> ```
> sed -i "s#-lmemcached#<libmemcached-install-directory>/lib/libmemcached.a -
> lcrypt -lpthread -lm -lstdc++ -lsasl2#" Makefile
> ```

Mengompilasi klien ElastiCache Memcached untuk PHP 5

Kompilasikan `aws-elasticache-cluster-client-memcached-for-php` dengan menjalankan perintah berikut dalam folder `aws-elasticache-cluster-client-memcached-for-php/`.

```
git clone https://github.com/awslabs/aws-elasticache-cluster-client-memcached-for-
php.git
cd aws-elasticache-cluster-client-memcached-for-php
sudo yum install zlib-devel
phpize
./configure --with-libmemcached-dir=<libmemcached-install-directory>
make
make install
```

Mengkonfigurasi klien ElastiCache

ElastiCache Cluster sesuai dengan protokol dengan Valkey, Memcached, dan Redis OSS. Kode, aplikasi, dan alat paling populer yang Anda gunakan saat ini dengan lingkungan yang ada akan bekerja dengan mulus dengan layanan ini.

Bagian ini membahas pertimbangan khusus untuk menyambung ke simpul cache di ElastiCache.

Topik

Perintah terbatas

Untuk memberikan pengalaman layanan terkelola, ElastiCache batasi akses ke perintah khusus mesin cache tertentu yang memerlukan hak istimewa lanjutan. Untuk cluster cache yang menjalankan Redis OSS, perintah berikut tidak tersedia:

- `bgrewriteaof`
- `bgsave`
- `config`
- `debug`
- `migrate`
- `replicaof`
- `save`
- `slaveof`
- `shutdown`
- `sync`

Menemukan titik akhir dan nomor port simpul

Untuk terhubung ke simpul cache, aplikasi Anda perlu mengetahui titik akhir dan nomor port untuk simpul tersebut.

Menemukan titik akhir dan nomor port simpul (Konsol)

Untuk menentukan titik akhir dan nomor port simpul

1. Masuk ke konsol ElastiCache manajemen Amazon dan pilih mesin yang berjalan di cluster Anda.

   Daftar klaster yang menjalankan mesin yang dipilih akan muncul.
2. Lanjutkan di bawah ini untuk mesin dan konfigurasi yang sedang Anda jalankan.
3. Pilih nama klaster yang diinginkan.
4. Temukan kolom Port dan Titik akhir untuk simpul yang diinginkan.

Menemukan titik akhir dan nomor port simpul cache (AWS CLI)

Untuk menentukan titik akhir dan nomor port simpul cache, gunakan perintah `describe-cache-clusters` dengan parameter `--show-cache-node-info`.

```
aws elasticache describe-cache-clusters --show-cache-node-info
```

Nama DNS dan nomor port yang sepenuhnya memenuhi syarat terdapat di bagian Titik Akhir dari output.

Menemukan titik akhir node cache dan nomor port (ElastiCache API)

Untuk menentukan titik akhir dan nomor port simpul cache, gunakan tindakan `DescribeCacheClusters` dengan parameter `ShowCacheNodeInfo=true`.

Example

```
https://elasticache.us-west-2.amazonaws.com /
    ?Action=DescribeCacheClusters
    &ShowCacheNodeInfo=true
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
```

```
    &Timestamp=20140421T220302Z
    &Version=2014-09-30
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Credential=<credential>
    &X-Amz-Date=20140421T220302Z
    &X-Amz-Expires=20140421T220302Z
    &X-Amz-Signature=<signature>
    &X-Amz-SignedHeaders=Host
```

Menghubungkan untuk menggunakan penemuan otomatis

Jika aplikasi Anda menggunakan Penemuan Otomatis, Anda hanya perlu mengetahui titik akhir konfigurasi untuk klaster, bukan titik akhir individual untuk setiap simpul cache. Untuk informasi selengkapnya, lihat Secara otomatis mengidentifikasi node di cluster Anda (Memcached).

> **Note**
>
> Pada saat ini, Penemuan Otomatis hanya tersedia untuk klaster cache yang menjalankan Memcached.

Menghubungkan ke node di cluster Valkey atau Redis OSS

> **Note**
>
> Pada saat ini, cluster (API/CLI: grup replikasi) yang mendukung replikasi dan replika baca hanya didukung untuk cluster yang menjalankan Valkey atau Redis OSS.

Untuk cluster, ElastiCache menyediakan antarmuka konsol, CLI, dan API untuk mendapatkan informasi koneksi untuk masing-masing node.

Untuk aktivitas hanya baca, aplikasi juga dapat terhubung ke simpul apa pun di klaster. Namun, untuk aktivitas tulis, kami menyarankan agar aplikasi Anda terhubung ke titik akhir utama (Valkey atau Redis OSS (mode cluster dinonaktifkan)) atau titik akhir konfigurasi (Valkey atau Redis OSS (mode cluster diaktifkan)) untuk cluster alih-alih menghubungkan langsung ke node. Hal ini akan memastikan bahwa aplikasi Anda selalu dapat menemukan simpul yang benar, bahkan jika Anda memutuskan untuk mengonfigurasi ulang klaster Anda dengan mempromosikan replika baca menjadi peran primer.

Menghubungkan ke klaster dalam grup replikasi (Konsol)

Untuk menentukan titik akhir dan nomor port

- Lihat topik Menemukan Titik Akhir Cluster Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol).

Menghubungkan ke klaster dalam grup replikasi AWS CLI

Untuk menentukan titik akhir dan nomor port simpul cache

Gunakan perintah `describe-replication-groups` dengan nama grup replikasi Anda:

```
aws elasticache describe-replication-groups redis2x2
```

Perintah ini seharusnya menghasilkan output yang serupa dengan yang berikut:

```
{
    "ReplicationGroups": [
        {
            "Status": "available",
            "Description": "2 shards, 2 nodes (1 + 1 replica)",
            "NodeGroups": [
                {
                    "Status": "available",
                    "Slots": "0-8191",
                    "NodeGroupId": "0001",
                    "NodeGroupMembers": [
                        {
                            "PreferredAvailabilityZone": "us-west-2c",
                            "CacheNodeId": "0001",
                            "CacheClusterId": "redis2x2-0001-001"
                        },
                        {
                            "PreferredAvailabilityZone": "us-west-2a",
                            "CacheNodeId": "0001",
                            "CacheClusterId": "redis2x2-0001-002"
                        }
                    ]
                },
                {
                    "Status": "available",
```

```
                    "Slots": "8192-16383",
                    "NodeGroupId": "0002",
                    "NodeGroupMembers": [
                        {
                            "PreferredAvailabilityZone": "us-west-2b",
                            "CacheNodeId": "0001",
                            "CacheClusterId": "redis2x2-0002-001"
                        },
                        {
                            "PreferredAvailabilityZone": "us-west-2a",
                            "CacheNodeId": "0001",
                            "CacheClusterId": "redis2x2-0002-002"
                        }
                    ]
                }
            ],
            "ConfigurationEndpoint": {
                "Port": 6379,
                "Address": "redis2x2.9dcv5r.clustercfg.usw2.cache.amazonaws.com"
            },
            "ClusterEnabled": true,
            "ReplicationGroupId": "redis2x2",
            "SnapshotRetentionLimit": 1,
            "AutomaticFailover": "enabled",
            "SnapshotWindow": "13:00-14:00",
            "MemberClusters": [
                "redis2x2-0001-001",
                "redis2x2-0001-002",
                "redis2x2-0002-001",
                "redis2x2-0002-002"
            ],
            "CacheNodeType": "cache.m3.medium",
            "PendingModifiedValues": {}
        }
    ]
}
```

Menghubungkan ke cluster dalam grup replikasi (API) ElastiCache

Untuk menentukan titik akhir dan nomor port simpul cache

Panggil `DescribeReplicationGroups` dengan parameter berikut:

`ReplicationGroupId` – nama grup replikasi.

Example

```
https://elasticache.us-west-2.amazonaws.com /
    ?Action=DescribeCacheClusters
    &ReplicationGroupId=repgroup01
    &Version=2014-09-30
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20140421T220302Z
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Date=20140421T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20140421T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Nama DNS dan IP yang mendasari

Klien memelihara daftar server yang berisi alamat dan port dari server yang menampung data cache. Saat menggunakan ElastiCache, DescribeCacheClusters API (atau utilitas baris describe-cache-clusters perintah) mengembalikan entri DNS dan nomor port yang sepenuhnya memenuhi syarat yang dapat digunakan untuk daftar server.

> **Important**
>
> Aplikasi klien harus dikonfigurasi untuk sering meresolusi nama DNS simpul cache saat aplikasi ini mencoba terhubung ke titik akhir simpul cache.

ElastiCache memastikan bahwa nama DNS node cache tetap sama ketika node cache dipulihkan jika terjadi kegagalan.

Kebanyakan pustaka klien mendukung koneksi simpul cache persisten secara default. Sebaiknya menggunakan koneksi simpul cache persisten saat menggunakan ElastiCache. Caching DNS sisi klien dapat terjadi di beberapa tempat, termasuk pustaka klien, runtime bahasa, atau sistem operasi klien. Anda harus meninjau konfigurasi aplikasi Anda di setiap lapisan untuk memastikan bahwa Anda sering meresolusi alamat IP untuk simpul cache Anda.

Tingkatan data di ElastiCache

ElastiCache untuk cluster Valkey atau Redis OSS yang terdiri dari grup replikasi dan menggunakan tipe node dari keluarga r6gd memiliki data mereka berjenjang antara memori dan penyimpanan SSD lokal (solid state drive). Tiering data menyediakan opsi harga-kinerja baru untuk beban kerja Valkey atau Redis OSS dengan memanfaatkan solid state drive (SSDs) berbiaya lebih rendah di setiap node cluster selain menyimpan data dalam memori. Hal ini sangat ideal untuk beban kerja yang mengakses hingga 20 persen dari keseluruhan set datanya secara rutin, dan untuk aplikasi yang dapat menoleransi latensi tambahan saat mengakses data di SSD.

Pada ElastiCache cluster dengan tiering data, ElastiCache memantau waktu akses terakhir dari setiap item yang disimpannya. Ketika memori yang tersedia (DRAM) sepenuhnya dikonsumsi, ElastiCache menggunakan algoritma yang paling tidak baru digunakan (LRU) untuk secara otomatis memindahkan item yang jarang diakses dari memori ke SSD. Ketika data pada SSD kemudian diakses, ElastiCache secara otomatis dan asinkron memindahkannya kembali ke memori sebelum memproses permintaan. Jika Anda memiliki beban kerja yang mengakses hanya subset dari datanya secara teratur, tingkatan data adalah cara optimal untuk menskalakan kapasitas Anda dengan hemat biaya.

Perhatikan bahwa saat menggunakan tingkatan data, kunci itu sendiri selalu tetap dalam memori, sedangkan LRU mengatur penempatan nilai pada memori vs disk. Secara umum, sebaiknya buat kunci Anda lebih kecil dari ukuran nilai Anda saat menggunakan tingkatan data.

Tingkatan data dirancang untuk memiliki dampak performa minimal pada beban kerja aplikasi. Misalnya, anggaplah ada nilai String 500-byte, Anda dapat mengalami 300 mikrodetik tambahan latensi rata-rata untuk permintaan terhadap data yang tersimpan di SSD dibandingkan dengan permintaan terhadap data dalam memori.

Dengan ukuran simpul tingkatan data terbesar (cache.r6gd.16xlarge), Anda dapat menyimpan hingga 1 petabyte dalam satu klaster 500 simpul (500 TB saat menggunakan 1 replika baca). Tiering data kompatibel dengan semua perintah Valkey atau Redis OSS dan struktur data yang didukung di. ElastiCache Anda tidak memerlukan perubahan sisi klien untuk menggunakan fitur ini.

Topik

- Menggunakan tingkatan data
- Memulihkan data dari cadangan ke dalam klaster dengan tingkatan data diaktifkan

Praktik terbaik

Kami merekomendasikan praktik terbaik berikut:

- Tingkatan data sangat ideal untuk beban kerja yang mengakses hingga 20 persen dari keseluruhan set datanya secara rutin, dan untuk aplikasi yang dapat menoleransi latensi tambahan saat mengakses data di SSD.
- Saat menggunakan kapasitas SSD yang tersedia pada simpul bertingkatan data, kami menyarankan agar ukuran nilai lebih besar dari ukuran kunci. Ketika item dipindahkan antara DRAM dan SSD, kunci akan selalu tetap dalam memori dan hanya nilai yang dipindahkan ke tingkat SSD.

Batasan

Tingkatan data memiliki batasan berikut:

- Anda hanya dapat menggunakan tingkatan data pada klaster yang merupakan bagian dari grup replikasi.
- Jenis simpul yang Anda gunakan harus berasal dari keluarga r6gd, yang tersedia di wilayah berikut: `us-east-2`, `us-east-1`, `us-west-2`, `us-west-1`, `eu-west-1`, `eu-central-1`, `eu-north-1`, `eu-west-3`, `ap-northeast-1`, `ap-southeast-1`, `ap-southeast-2`, `ap-south-1`, `ca-central-1`, dan `sa-east-1`.
- Anda harus menggunakan mesin yang Valkey 7.2 atau yang lebih baru, atau Redis OSS 6.2 atau yang lebih baru.
- Anda tidak dapat memulihkan cadangan klaster r6gd ke klaster lain kecuali klaster lain tersebut menggunakan r6gd juga.
- Anda tidak dapat mengekspor cadangan ke Amazon S3 untuk klaster tingkatan data.
- Migrasi online tidak didukung untuk klaster yang berjalan pada jenis simpul r6gd.
- Penskalaan tidak didukung dari klaster tingkatan data (misalnya, sebuah klaster yang menggunakan jenis simpul r6gd) ke klaster yang tidak menggunakan tingkatan data (misalnya, klaster yang menggunakan jenis simpul r6g). Untuk informasi selengkapnya, lihat Penskalaan ElastiCache.

- Penskalaan otomatis didukung pada cluster menggunakan tiering data untuk Valkey versi 7.2 dan yang lebih baru, dan Redis OSS versi 7.0.7 dan yang lebih baru. Untuk informasi selengkapnya, lihat Auto Scaling Valkey dan Redis OSS cluster
- Tingkatan data hanya mendukung kebijakan maxmemory `volatile-lru`, `allkeys-lru`, `volatile-lfu`, `allkeys-lfu`, dan `noeviction`.
- Penyimpanan tanpa garpu didukung untuk Valkey versi 7.2 dan yang lebih baru, dan Redis OSS versi 7.0.7 dan yang lebih baru. Untuk informasi selengkapnya, lihat Cara penerapan sinkronisasi dan pencadangan.
- Item yang lebih besar dari 128 MiB tidak dipindahkan ke SSD.
- Mulai dari Valley 8.1 dan yang lebih baru, item yang ukuran nilai kuncinya kurang dari 40 byte tidak akan dipindahkan ke SSD.

Harga

Simpul R6gd memiliki kapasitas total 4,8x lebih banyak (memori + SSD) dan dapat membantu Anda mencapai lebih dari 60 persen penghematan ketika berjalan pada pemanfaatan maksimum dibandingkan dengan simpul R6g (memori saja). Untuk informasi selengkapnya, lihat harga ElastiCache .

Pemantauan

ElastiCache menawarkan metrik yang dirancang khusus untuk memantau cluster kinerja yang menggunakan tiering data. Untuk memantau rasio item dalam DRAM dibandingkan dengan SSD, Anda dapat menggunakan `CurrItems` metrik di Metrik untuk Valkey dan Redis OSS. Anda dapat menghitung persentase sebagai: (CurrItems dengan Dimensi: Tingkat = Memori* 100)/(CurrItems tanpa filter dimensi).

Jika kebijakan penggusuran yang dikonfigurasi memungkinkan, maka ElastiCache akan mulai mengusir item ketika persentase item dalam memori berkurang di bawah 5 persen. Pada node yang dikonfigurasi dengan kebijakan noeviction, operasi tulis akan menerima kesalahan kehabisan memori.

Anda tetap disarankan untuk mempertimbangkan penskalaan untuk cluster yang Diaktifkan Mode Cluster atau penskalaan untuk cluster yang dinonaktifkan Mode Cluster ketika persentase item dalam memori berkurang di bawah 5 persen. Untuk informasi lebih lanjut tentang penskalaan, lihatPenskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan). Untuk informasi lebih

lanjut tentang metrik untuk klaster Valkey atau Redis OSS yang menggunakan tingkatan data, lihat. Metrik untuk Valkey dan Redis OSS

Menggunakan tingkatan data

Menggunakan data tiering menggunakan AWS Management Console

Ketika membuat sebuah klaster sebagai bagian dari grup replikasi, Anda menggunakan tingkatan data dengan memilih jenis simpul dari keluarga r6gd, seperti cache.r6gd.xlarge. Memilih jenis simpul tersebut secara otomatis mengaktifkan tingkatan data.

Untuk informasi tentang cara membuat klaster, lihat Membuat cluster untuk Valkey atau Redis OSS.

Mengaktifkan tiering data menggunakan AWS CLI

Saat membuat grup replikasi menggunakan AWS CLI, Anda menggunakan tiering data dengan memilih tipe node dari keluarga r6gd, seperti cache.r6gd.xlarge dan menyetel parameternya. `--data-tiering-enabled`

Anda tidak dapat membatalkan penggunaan tingkatan data ketika memilih jenis simpul dari keluarga r6gd. Jika Anda mengatur parameter `--no-data-tiering-enabled`, operasi akan gagal.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
   --replication-group-id redis-dt-cluster \
   --replication-group-description "Redis OSS cluster with data tiering" \
   --num-node-groups 1 \
   --replicas-per-node-group 1 \
   --cache-node-type cache.r6gd.xlarge \
   --engine redis \
   --cache-subnet-group-name default \
   --automatic-failover-enabled \
   --data-tiering-enabled
```

Untuk Windows:

```
aws elasticache create-replication-group ^
   --replication-group-id redis-dt-cluster ^
   --replication-group-description "Redis OSS cluster with data tiering" ^
   --num-node-groups 1 ^
```

```
    --replicas-per-node-group 1 ^
    --cache-node-type cache.r6gd.xlarge ^
    --engine redis ^
    --cache-subnet-group-name default ^
    --automatic-failover-enabled ^
    --data-tiering-enabled
```

Setelah menjalankan operasi ini, Anda akan melihat respons seperti yang berikut ini:

```
{
    "ReplicationGroup": {
        "ReplicationGroupId": "redis-dt-cluster",
        "Description": "Redis OSS cluster with data tiering",
        "Status": "creating",
        "PendingModifiedValues": {},
        "MemberClusters": [
            "redis-dt-cluster"
        ],
        "AutomaticFailover": "enabled",
        "DataTiering": "enabled",
        "SnapshotRetentionLimit": 0,
        "SnapshotWindow": "06:00-07:00",
        "ClusterEnabled": false,
        "CacheNodeType": "cache.r6gd.xlarge",
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

Memulihkan data dari cadangan ke dalam klaster dengan tingkatan data diaktifkan

Anda dapat memulihkan cadangan ke klaster baru dengan tiering data yang diaktifkan menggunakan (Console), (AWS CLI) atau (ElastiCache API). Ketika Anda membuat sebuah klaster menggunakan jenis simpul dalam keluarga r6gd, tingkatan data diaktifkan.

Memulihkan data dari cadangan ke dalam klaster dengan tingkatan data diaktifkan (konsol)

Untuk memulihkan cadangan ke klaster baru dengan tingkatan data diaktifkan (konsol)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih Cadangan.

3. Di daftar cadangan, pilih kotak di sebelah kiri nama cadangan yang ingin Anda pulihkan.
4. Pilih Pulihkan.
5. Lengkapi kotak dialog Pulihkan Klaster. Pastikan untuk melengkapi semua bidang Wajib dan semua bidang yang ingin Anda ubah dari default.

    1. ID Klaster – Wajib. Nama klaster baru.
    2. Mode cluster diaktifkan (skala keluar) - Pilih ini untuk cluster Valkey atau Redis OSS (mode cluster diaktifkan).
    3. Jenis Simpul – Tentukan cache.r6gd.xlarge atau jenis simpul lainnya dari keluarga r6gd.
    4. Jumlah Serpihan – Memilih jumlah serpihan yang Anda inginkan di klaster baru (API/CLI: grup simpul).
    5. Replika per Serpihan – Pilih jumlah simpul replika baca yang Anda inginkan di setiap serpihan.
    6. Slot dan ruang kunci – Pilih cara yang Anda inginkan untuk mendistribusikan kunci di antara serpihan. Jika Anda memilih untuk menentukan distribusi kunci, lengkapi tabel yang menentukan rentang kunci untuk setiap serpihan.
    7. Zona ketersediaan – Tentukan cara yang Anda inginkan untuk memilih Zona Ketersediaan dari klaster.
    8. Port – Ubah ini hanya jika Anda menginginkan klaster baru menggunakan port yang berbeda.
    9. Pilih VPC – Pilih VPC tempat klaster ini akan dibuat.
    10. Grup Parameter - Pilih grup parameter yang menyimpan memori yang cukup untuk overhead Valkey atau Redis OSS untuk jenis node yang Anda pilih.
6. Jika pengaturan sudah sesuai keinginan Anda, pilih Buat.

Untuk informasi tentang cara membuat klaster, lihat Membuat cluster untuk Valkey atau Redis OSS.

Memulihkan data dari cadangan ke dalam klaster dengan tingkatan data diaktifkan (AWS CLI)

Saat membuat grup replikasi menggunakan AWS CLI, tiering data secara default digunakan dengan memilih tipe node dari keluarga r6gd, seperti cache.r6gd.xlarge dan mengatur parameter. `--data-tiering-enabled`

Anda tidak dapat membatalkan penggunaan tingkatan data ketika memilih jenis simpul dari keluarga r6gd. Jika Anda mengatur parameter `--no-data-tiering-enabled`, operasi akan gagal.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
   --replication-group-id redis-dt-cluster \
   --replication-group-description "Redis OSS cluster with data tiering" \
   --num-node-groups 1 \
   --replicas-per-node-group 1 \
   --cache-node-type cache.r6gd.xlarge \
   --engine redis \
   --cache-subnet-group-name default \
   --automatic-failover-enabled \
   --data-tiering-enabled \
   --snapshot-name my-snapshot
```

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group ^
   --replication-group-id redis-dt-cluster ^
   --replication-group-description "Redis OSS cluster with data tiering" ^
   --num-node-groups 1 ^
   --replicas-per-node-group 1 ^
   --cache-node-type cache.r6gd.xlarge ^
   --engine redis ^
   --cache-subnet-group-name default ^
   --automatic-failover-enabled ^
   --data-tiering-enabled ^
   --snapshot-name my-snapshot
```

Setelah menjalankan operasi ini, Anda akan melihat respons seperti yang berikut ini:

```
{
    "ReplicationGroup": {
        "ReplicationGroupId": "redis-dt-cluster",
        "Description": "Redis OSS cluster with data tiering",
        "Status": "creating",
        "PendingModifiedValues": {},
        "MemberClusters": [
            "redis-dt-cluster"
        ],
        "AutomaticFailover": "enabled",
        "DataTiering": "enabled",
        "SnapshotRetentionLimit": 0,
        "SnapshotWindow": "06:00-07:00",
        "ClusterEnabled": false,
```

```
        "CacheNodeType": "cache.r6gd.xlarge",
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false
    }
}
```

Mempersiapkan cluster di ElastiCache

Berikut ini, Anda dapat menemukan petunjuk tentang membuat cluster menggunakan ElastiCache konsol, the AWS CLI, atau ElastiCache API.

Anda juga dapat membuat ElastiCache cluster menggunakan AWS CloudFormation. Untuk informasi selengkapnya, lihat AWS ElastiCache::: CacheCluster di Panduan Pengguna AWS Cloud Formation, yang mencakup panduan tentang cara menerapkan pendekatan tersebut.

Setiap kali Anda membuat klaster atau grup replikasi, sebaiknya lakukan pekerjaan persiapan tertentu agar Anda tidak perlu melakukan peningkatan atau melakukan perubahan.

Topik

- Menentukan persyaratan ElastiCache klaster Anda
- Memilih ukuran simpul Anda

Menentukan persyaratan ElastiCache klaster Anda

Persiapan

Mengetahui jawaban atas pertanyaan-pertanyaan berikut membantu membuat ElastiCache klaster Anda menjadi lebih lancar:

- Jenis instans simpul apa yang dibutuhkan?

  Untuk panduan terkait cara memilih jenis simpul instans, lihat Memilih ukuran simpul Anda.
- Apakah klaster Anda diluncurkan di cloud privat virtual (VPC) berdasarkan Amazon VPC?

> **Important**
>
> Jika Anda akan meluncurkan klaster di VPC, pastikan untuk membuat grup subnet di VPC yang sama sebelum Anda mulai membuat klaster. Untuk informasi selengkapnya, lihat Subnet dan grup subnet.

ElastiCache dirancang untuk diakses dari dalam AWS menggunakan Amazon EC2. Namun, jika Anda meluncurkannya di VPC berdasarkan Amazon VPC dan klaster Anda berada di sebuah VPC, Anda dapat menyediakan akses dari luar AWS. Untuk informasi selengkapnya, lihat Mengakses ElastiCache sumber daya dari luar AWS.

- Apakah Anda perlu menyesuaikan nilai parameter tertentu?

  Jika ya, buat grup parameter kustom. Untuk informasi selengkapnya, lihat Membuat grup ElastiCache parameter.

  Jika Anda menjalankan Valkey atau Redis OSS, pertimbangkan pengaturan atau. `reserved-memory` `reserved-memory-percent` Untuk informasi selengkapnya, lihat Mengelola memori cadangan untuk Valkey dan Redis OSS.
- Apakah Anda perlu membuat grup keamanan VPC Anda sendiri?

  Untuk informasi selengkapnya, lihat Keamanan di VPC Anda.
- Bagaimana Anda akan menerapkan toleransi kesalahan?

  Untuk informasi selengkapnya, lihat Mitigasi Kegagalan.

Topik

- ElastiCache persyaratan memori dan prosesor
- Konfigurasi klaster Memcached
- Konfigurasi cluster Valkey dan Redis OSS
- ElastiCache persyaratan penskalaan
- ElastiCache persyaratan akses
- Persyaratan Wilayah, Zona Ketersediaan, dan Zona Lokal untuk ElastiCache

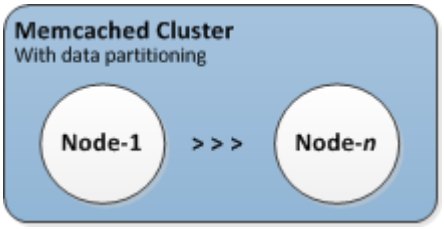
ElastiCache persyaratan memori dan prosesor

Blok bangunan dasar Amazon ElastiCache adalah simpul. Simpul dikonfigurasi secara tunggal atau dalam grup untuk membentuk klaster. Saat menentukan jenis simpul yang akan digunakan untuk klaster Anda, pertimbangkan konfigurasi simpul klaster dan jumlah data yang harus disimpan.

Mesin Memcached adalah multi-thread. Jadi, jumlah inti dari simpul akan berdampak pada daya komputasi yang tersedia untuk klaster.

Konfigurasi klaster Memcached

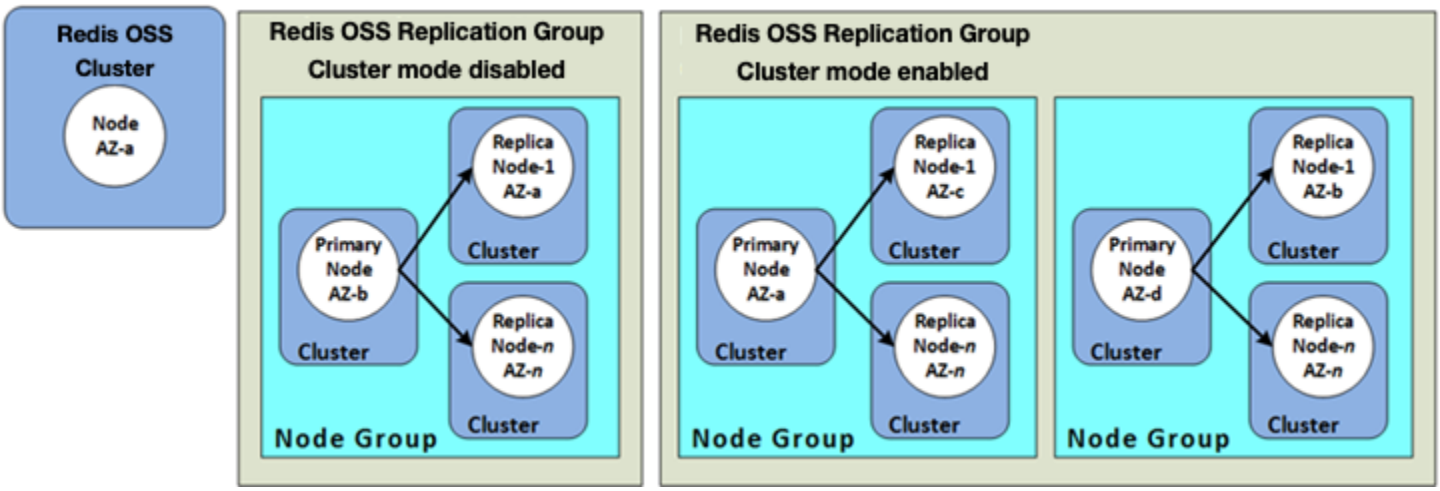
ElastiCache untuk cluster Memcached terdiri dari 1 hingga 60 node. Data dalam klaster Memcached dipartisi di seluruh simpul di klaster. Aplikasi Anda terhubung dengan klaster Memcached menggunakan alamat jaringan yang disebut Titik Akhir. Setiap simpul dalam klaster Memcached memiliki titik akhir sendiri yang digunakan oleh aplikasi Anda untuk membaca dari atau menulis ke simpul tertentu. Selain titik akhir simpul, klaster Memcached itu sendiri memiliki titik akhir yang disebut titik akhir konfigurasi. Aplikasi Anda dapat menggunakan titik akhir ini untuk membaca dari atau menulis ke klaster, dan membiarkan penemuan otomatis menentukan simpul yang akan melakukan operasi baca atau operasi tulis.



Untuk informasi selengkapnya, lihat Mengelola cluster di ElastiCache.

Konfigurasi cluster Valkey dan Redis OSS

ElastiCache untuk cluster Valkey dan Redis OSS terdiri dari 0 hingga 500 pecahan (juga disebut grup simpul). Data dalam cluster Valkey atau Redis OSS dipartisi di seluruh pecahan di cluster. Aplikasi Anda terhubung dengan cluster Valkey atau Redis OSS menggunakan alamat jaringan yang disebut Endpoint. Node dalam pecahan Valkey atau Redis OSS memenuhi salah satu dari dua peran: satu read/write primer dan semua node sekunder hanya-baca lainnya (juga disebut replika baca). Selain titik akhir node, cluster Valkey atau Redis OSS itu sendiri memiliki titik akhir yang disebut titik akhir konfigurasi. Aplikasi Anda dapat menggunakan endpoint ini untuk membaca dari atau menulis ke cluster, meninggalkan penentuan node mana yang akan dibaca atau ditulis hingga ElastiCache Redis OSS.



Untuk informasi selengkapnya, lihat Mengelola cluster di ElastiCache.

ElastiCache persyaratan penskalaan

Semua klaster dapat dinaikkan skalanya dengan membuat klaster baru yang memiliki jenis simpul baru yang lebih besar. Saat Anda meningkatkan kluster Memcached, cluster baru mulai kosong. Saat Anda meningkatkan klaster Valkey atau Redis OSS, Anda dapat menyemainya dari cadangan dan menghindari cluster baru mulai kosong.

Amazon ElastiCache untuk cluster Memcached dapat diskalakan atau masuk. Untuk menskalakan klaster Memcached ke luar atau ke dalam, Anda cukup menambahkan atau menghapus simpul dari klaster. Jika Anda telah mengaktifkan Penemuan Otomatis dan aplikasi Anda terhubung ke titik akhir konfigurasi dari klaster, Anda tidak perlu membuat perubahan apa pun dalam aplikasi saat Anda menambahkan atau menghapus simpul.

Untuk informasi selengkapnya, lihat Penskalaan ElastiCache dalam panduan ini.

ElastiCache persyaratan akses

Secara desain, ElastiCache kluster Amazon diakses dari EC2 instans Amazon. Akses jaringan ke ElastiCache cluster terbatas pada akun yang membuat cluster. Oleh karena itu, sebelum Anda dapat mengakses klaster dari EC2 instans Amazon, Anda harus mengotorisasi EC2 instans Amazon untuk mengakses klaster. Langkah-langkah untuk melakukan ini bervariasi, tergantung pada apakah Anda meluncurkan ke EC2 -VPC atau EC2 -Classic.

Jika Anda meluncurkan cluster Anda ke EC2 -VPC, Anda perlu memberikan masuknya jaringan ke cluster. Jika meluncurkan klaster ke EC2 -Classic, Anda harus memberikan grup keamanan Amazon Elastic Compute Cloud yang terkait dengan instans akses ke grup ElastiCache keamanan Anda. Untuk petunjuk yang lebih mendetail, lihat Langkah 3. Otorisasi akses ke cluster dalam panduan ini.

Persyaratan Wilayah, Zona Ketersediaan, dan Zona Lokal untuk ElastiCache

Amazon ElastiCache mendukung semua AWS wilayah. Dengan menempatkan ElastiCache kluster Anda di AWS Wilayah yang dekat dengan aplikasi Anda, Anda dapat mengurangi latensi. Jika klaster Anda memiliki beberapa simpul, menempatkan simpul Anda di berbagai Zona Ketersediaan atau Zona Lokal dapat mengurangi dampak kegagalan pada klaster Anda.

Untuk informasi selengkapnya, lihat berikut ini:

- Memilih wilayah dan zona ketersediaan untuk ElastiCache
- Menggunakan zona lokal dengan ElastiCache
- Mitigasi Kegagalan

Memilih ukuran simpul Anda

Ukuran node yang Anda pilih untuk ElastiCache klaster memengaruhi biaya, kinerja, dan toleransi kesalahan.

Ukuran simpul (Valkey dan Redis OSS)

Untuk informasi tentang manfaat prosesor Graviton, lihat Prosesor AWS Graviton.

Menjawab pertanyaan-pertanyaan berikut dapat membantu Anda menentukan jenis node minimum yang Anda butuhkan untuk implementasi Valkey atau Redis OSS Anda:

- Apakah Anda mengharapkan beban kerja terikat throughput dengan beberapa koneksi klien?

  Jika ini masalahnya dan Anda menjalankan Redis OSS versi 5.0.6 atau lebih tinggi, Anda bisa mendapatkan throughput dan latensi yang lebih baik dengan I/O fitur kami yang disempurnakan, jika CPUs tersedia digunakan untuk membongkar koneksi klien, atas nama mesin Redis OSS. Jika Anda menjalankan Redis OSS versi 7.0.4 atau lebih tinggi, di atas I/O, you will get additional acceleration with enhanced I/O multiplexing, where each dedicated network IO thread pipelines commands from multiple clients into the Redis OSS engine, taking advantage of Redis OSS' ability to efficiently process commands in batches. In ElastiCache for Redis OSS v7.1 and above, we extended the enhanced I/O threads functionality to also handle the presentation layer logic. By presentation layer, what we mean is that Enhanced I/O utas yang disempurnakan sekarang tidak hanya membaca input klien, tetapi juga mengurai input ke dalam format perintah biner Redis OSS, yang kemudian diteruskan ke utas utama untuk dieksekusi, memberikan peningkatan kinerja. Lihat postingan blog dan halaman versi yang didukung untuk detail tambahan.

- Apakah Anda memiliki beban kerja yang mengakses sebagian kecil dari datanya secara berkala?

  Jika ini masalahnya dan Anda menjalankan mesin Redis OSS versi 6.2 atau yang lebih baru, Anda dapat memanfaatkan tiering data dengan memilih tipe node r6gd. Dengan tingkatan data, data yang sudah lama tidak digunakan akan disimpan di SSD. Ketika data ini diambil, ada sedikit latensi, tetapi diimbangi dengan penghematan biaya. Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.

  Untuk informasi selengkapnya, lihat Jenis simpul yang didukung.

- Berapa banyak memori total yang dibutuhkan untuk data Anda?

  Untuk mendapatkan estimasi umum, cari tahu ukuran item yang ingin Anda simpan dalam cache. Kalikan ukuran ini dengan jumlah item yang ingin Anda simpan dalam cache pada waktu yang sama. Untuk mendapatkan estimasi yang wajar dari ukuran item, pertama-tama serialisasikan item cache Anda, kemudian hitung karakternya. Kemudian bagi jumlah karakter dengan jumlah serpihan dalam klaster Anda.

  Untuk informasi selengkapnya, lihat Jenis simpul yang didukung.

- Versi Redis OSS apa yang Anda jalankan?

  Versi Redis OSS sebelum 2.8.22 mengharuskan Anda untuk menyimpan lebih banyak memori untuk failover, snapshot, sinkronisasi, dan mempromosikan replika ke operasi utama. Persyaratan ini muncul karena Anda harus memiliki cukup memori tersedia untuk semua penulisan yang terjadi selama proses.

  Redis OSS versi 2.8.22 dan yang lebih baru menggunakan proses penyimpanan tanpa garpu yang membutuhkan lebih sedikit memori yang tersedia daripada proses sebelumnya.

  Untuk informasi selengkapnya, lihat berikut ini:

  - Cara penerapan sinkronisasi dan pencadangan
  - Memastikan Anda memiliki cukup memori untuk membuat snapshot Valkey atau Redis OSS

- Seberapa berat operasi tulis aplikasi Anda?

  Aplikasi dengan operasi tulis berat dapat membutuhkan ketersediaan memori yang jauh lebih banyak, memori yang tidak digunakan oleh data, saat mengambil snapshot atau failover. Setiap kali proses `BGSAVE` dilakukan, Anda harus memiliki cukup memori yang tidak digunakan oleh data untuk mengakomodasi semua operasi tulis yang berlangsung selama proses `BGSAVE`. Contohnya adalah ketika mengambil snapshot, ketika menyinkronkan klaster primer dengan

replika dalam klaster, dan ketika mengaktifkan fitur append-only file (AOF). Contoh lainnya adalah saat mempromosikan replika ke primer (jika Anda memiliki Multi-AZ yang aktif). Kemungkinan terburuknya adalah ketika semua data Anda ditulis ulang selama proses berlangsung. Dalam hal ini, Anda memerlukan ukuran instans simpul dengan memori dua kali lebih banyak dari yang diperlukan untuk data saja.

Untuk informasi selengkapnya, lihat Memastikan Anda memiliki cukup memori untuk membuat snapshot Valkey atau Redis OSS.

- Apakah implementasi Anda akan menjadi cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) mandiri, atau cluster Valkey atau Redis OSS (mode cluster enabled) dengan beberapa pecahan?

  Valkey atau Redis OSS (mode cluster dinonaktifkan) cluster

  Jika Anda menerapkan cluster Valkey atau Redis OSS (mode cluster disabled), tipe node Anda harus dapat mengakomodasi semua data Anda ditambah overhead yang diperlukan seperti yang dijelaskan dalam bullet sebelumnya.

  Sebagai contoh, misalnya Anda memperkirakan bahwa total ukuran semua item Anda adalah 12 GB. Dalam kasus ini, Anda dapat menggunakan simpul `cache.m3.xlarge` dengan memori 13,3 GB atau simpul `cache.r3.large` dengan memori 13,5 GB. Namun, Anda mungkin memerlukan lebih banyak memori untuk operasi `BGSAVE`. Jika aplikasi Anda memiliki operasi tulis berat, gandakan persyaratan memori ke setidaknya 24 GB. Dengan demikian, gunakan `cache.m3.2xlarge` dengan memori 27,9 GB atau `cache.r3.xlarge` dengan memori 30,5 GB.

  Valkey atau Redis OSS (mode cluster diaktifkan) dengan beberapa pecahan

  Jika Anda menerapkan cluster Valkey atau Redis OSS (mode cluster enabled) dengan beberapa pecahan, maka tipe node harus dapat mengakomodasi `bytes-for-data-and-overhead / number-of-shards` byte data.

  Sebagai contoh, misalnya Anda memperkirakan bahwa total ukuran semua item Anda adalah 12 GB dan Anda memiliki dua serpihan. Dalam kasus ini, Anda dapat menggunakan simpul `cache.m3.large` dengan memori 6,05 GB (12 GB / 2). Namun, Anda mungkin memerlukan lebih banyak memori untuk operasi `BGSAVE`. Jika aplikasi Anda memiliki operasi tulis berat, gandakan persyaratan memori ke setidaknya 12 GB per serpihan. Dengan demikian, gunakan `cache.m3.xlarge` dengan memori 13,3 GB atau `cache.r3.large` dengan memori 13,5 GB.

- Apakah Anda menggunakan Zona Lokal?

Local Zones memungkinkan Anda menempatkan sumber daya seperti ElastiCache klaster di beberapa lokasi yang dekat dengan pengguna Anda. Namun, ketika Anda memilih ukuran simpul Anda, perhatikan bahwa ukuran simpul yang tersedia saat ini terbatas seperti yang tertera berikut ini, terlepas dari persyaratan kapasitas:

- Generasi saat ini:

  Jenis simpul M5: `cache.m5.large`, `cache.m5.xlarge`, `cache.m5.2xlarge`, `cache.m5.4xlarge`, `cache.m5.12xlarge`, `cache.m5.24xlarge`

  Jenis simpul R5: `cache.r5.large`, `cache.r5.xlarge`, `cache.r5.2xlarge`, `cache.r5.4xlarge`, `cache.r5.12xlarge`, `cache.r5.24xlarge`

  Jenis simpul T3: `cache.t3.micro`, `cache.t3.small`, `cache.t3.medium`

Saat cluster Anda berjalan, Anda dapat memantau penggunaan memori, pemanfaatan prosesor, klik cache, dan metrik cache misses yang dipublikasikan. CloudWatch Anda mungkin memperhatikan bahwa klaster Anda tidak memiliki laju hit yang Anda inginkan atau bahwa kunci terlalu sering dikosongkan. Dalam kasus ini, Anda dapat memilih ukuran simpul yang berbeda dengan spesifikasi CPU dan memori yang lebih besar.

Saat memantau penggunaan CPU, ingat Valkey dan Redis OSS adalah single-threaded. Dengan demikian, kalikan penggunaan CPU yang dilaporkan dengan jumlah inti CPU untuk mengetahui penggunaan yang sebenarnya. Misalnya, CPU empat inti yang melaporkan tingkat penggunaan 20 persen sebenarnya adalah satu inti Redis OSS berjalan pada pemanfaatan 80 persen.

Ukuran node (Memcached)

Klaster Memcached berisi satu atau beberapa simpul dengan data klaster yang dipartisi di seluruh simpul. Oleh karena itu, kebutuhan memori klaster dan memori simpul berkaitan, tetapi tidak sama. Anda dapat mencapai kapasitas memori klaster yang Anda butuhkan dengan memiliki simpul yang berjumlah sedikit namun berukuran besar atau beberapa simpul yang lebih kecil. Seiring waktu, ketika kebutuhan Anda berubah, Anda dapat menambahkan simpul ke atau menghapus simpul dari klaster dan dengan demikian membayar hanya untuk apa yang Anda butuhkan.

Kapasitas memori total klaster Anda dihitung dengan mengalikan jumlah simpul dalam klaster dengan kapasitas RAM setiap simpul setelah dikurangi sistem overhead. Kapasitas setiap simpul bergantung pada jenis simpul.

```
cluster_capacity = number_of_nodes * (node_capacity - system_overhead)
```

Jumlah simpul dalam klaster merupakan faktor utama dalam ketersediaan klaster Anda yang menjalankan Memcached. Kegagalan dari satu simpul dapat berdampak pada ketersediaan aplikasi Anda dan beban pada basis data backend Anda. Dalam kasus seperti itu, ElastiCache menyediakan pengganti untuk simpul yang gagal dan akan direpopulasi. Untuk mengurangi dampak ketersediaan ini, sebarkan memori dan kapasitas komputasi Anda pada lebih banyak simpul dengan kapasitas yang lebih kecil, daripada menggunakan simpul berkapasitas tinggi yang lebih sedikit.

Dalam skenario di mana Anda ingin memiliki 35 GB memori cache, Anda dapat mengatur salah satu konfigurasi berikut:

- 11 simpul `cache.t2.medium` dengan memori 3,22 GB dengan masing-masing 2 thread = 35,42 GB dan 22 thread.
- 6 simpul `cache.m4.large` dengan memori 6,42 GB dengan masing-masing 2 thread = 38,52 GB dan 12 thread.
- 3 simpul `cache.r4.large` dengan memori 12,3 GB dengan masing-masing 2 thread = 36,90 GB dan 6 thread.
- 3 simpul `cache.m4.xlarge` dengan memori 14,28 GB dengan masing-masing 4 thread = 42,84 GB dan 12 thread.

Membandingkan opsi simpul

Jenis simpul	Memori (dalam GiB)	Inti	Biaya per jam*	Simpul yang diperlukan	Total memori (dalam GiB)	Total inti	Biaya bulanan
cache.t2.medium	3.22	2	$0.068	11	35,42	22	$538.56
cache.m4.large	6.42	2	$0.156	6	38.52	12	$673,92
cache.m4.xlarge	14.28	4	$0.311	3	42,84	12	$671.76

Jenis simpul	Memori (dalam GiB)	Inti	Biaya per jam*	Simpul yang diperlukan	Total memori (dalam GiB)	Total inti	Biaya bulanan
cache.m5.xlarge	12.93	4	$0.311	3	38.81	12	$671.76
cache.m6g.large	6,85	2	$0.147	6	41.1	12	$635
cache.r4.large	12.3	2	$0.228	3	36,9	6	$492.48
cache.r5.large	13.07	2	$0.216	3	39,22	6	$466.56
cache.r6g.large	13.07	2	$0.205	3	42.12	6	$442

* Biaya per jam per simpul mulai 8 Oktober 2020.

Biaya bulanan 100% penggunaan selama 30 hari (720 jam).

Opsi ini masing-masing menyediakan kapasitas memori yang sama namun kapasitas komputasi dan biaya yang berbeda. Untuk membandingkan biaya opsi spesifik Anda, lihat ElastiCache Harga Amazon.

Untuk klaster yang menjalankan Memcached, beberapa memori yang tersedia pada setiap simpul digunakan untuk overhead koneksi. Untuk informasi selengkapnya, lihat Overhead koneksi Memcached

Menggunakan beberapa simpul membutuhkan penyebaran kunci di seluruh simpul. Setiap simpul memiliki titik akhir sendiri. Untuk manajemen endpoint yang mudah, Anda dapat menggunakan fitur Auto Discovery, yang memungkinkan program klien untuk secara otomatis mengidentifikasi semua node dalam sebuah cluster. ElastiCache Untuk informasi selengkapnya, lihat Secara otomatis mengidentifikasi node di cluster Anda (Memcached).

Dalam beberapa kasus, Anda mungkin tidak yakin berapa banyak kapasitas yang Anda butuhkan. Jika demikian, untuk pengujian sebaiknya mulai dengan satu simpul `cache.m5.large`. Kemudian pantau penggunaan memori, pemanfaatan CPU, dan laju hit cache dengan ElastiCache metrik yang dipublikasikan ke Amazon. CloudWatch Untuk informasi selengkapnya tentang CloudWatch metrik ElastiCache, lihatPemantauan penggunaan dengan CloudWatch Metrik. Untuk produksi dan beban kerja yang lebih besar, simpul R5 memberikan performa dan nilai biaya RAM yang terbaik.

Jika klaster Anda tidak memiliki laju hit yang Anda inginkan, Anda dapat dengan mudah menambahkan lebih banyak simpul untuk meningkatkan total memori yang tersedia di klaster Anda.

Jika klaster Anda terikat oleh CPU tetapi memiliki laju hit yang mencukupi, atur klaster yang baru dengan jenis simpul yang memiliki daya komputasi lebih besar.

Membuat cluster untuk Valkey atau Redis OSS

Contoh berikut menunjukkan cara membuat cluster Valkey atau Redis OSS menggunakan AWS Management Console, AWS CLI dan API. ElastiCache

Membuat Valkey atau Redis OSS (mode cluster dinonaktifkan) (Konsol)

ElastiCache mendukung replikasi ketika Anda menggunakan mesin Valkey atau Redis OSS. Untuk memantau latensi antara saat data ditulis ke cluster utama Valkey atau Redis OSS read/write dan ketika disebarkan ke cluster sekunder hanya-baca, ElastiCache tambahkan ke cluster kunci khusus,. `ElastiCacheMasterReplicationTimestamp` Kunci ini adalah waktu saat ini dalam Waktu Terkoordinasi Universal (UTC). Karena cluster Valkey atau Redis OSS dapat ditambahkan ke grup replikasi di lain waktu, kunci ini termasuk dalam semua cluster Valkey atau Redis OSS, bahkan jika awalnya mereka bukan anggota grup replikasi. Untuk informasi selengkapnya tentang grup replikasi, lihat Ketersediaan tinggi menggunakan grup replikasi.

Untuk membuat cluster Valkey atau Redis OSS (mode cluster dinonaktifkan), ikuti langkah-langkah di. Membuat cluster Valkey (mode cluster dinonaktifkan) (Konsol)

Segera setelah status klaster Anda tersedia, Anda dapat memberikan Amazon EC2 akses ke sana, menghubungkannya, dan mulai menggunakannya. Untuk informasi selengkapnya, lihat Langkah 3. Otorisasi akses ke cluster dan Langkah 4. Connect ke node cluster.

> **Important**
>
> Setelah klaster Anda tersedia, Anda akan ditagih untuk setiap jam atau jam parsial saat klaster aktif, meskipun Anda tidak sedang aktif menggunakannya. Untuk menghentikan tagihan untuk klaster ini, Anda harus menghapusnya. Lihat Menghapus cluster di ElastiCache.

Membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) (Konsol)

Jika Anda menjalankan Redis OSS 3.2.4 atau yang lebih baru, Anda dapat membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan). Cluster Valkey atau Redis OSS (mode cluster enabled) mendukung partisi data Anda di 1 hingga 500 pecahan (API/CLI: grup node) tetapi dengan beberapa batasan. Untuk perbandingan Valkey atau Redis OSS (mode cluster dinonaktifkan) dan Valkey atau Redis OSS (mode cluster diaktifkan), lihat. Mesin dan versi yang didukung

Untuk membuat cluster Valkey atau Redis OSS (mode cluster enabled) menggunakan konsol ElastiCache

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Dari daftar di sudut kanan atas, pilih AWS Wilayah tempat Anda ingin meluncurkan cluster ini.
3. Pilih Mulai di panel navigasi.
4. Pilih Buat VPC dan ikuti langkah-langkah yang telah dijelaskan dalam Membuat Cloud Privat Virtual (VPC).
5. Pada halaman ElastiCache dashboard, pilih Create cluster dan kemudian pilih Create Valkey cluster atau Create Redis OSS cluster.
6. Di bagian Pengaturan klaster, lakukan hal berikut:

    a. Pilih Konfigurasi dan buat klaster baru.

    b. Untuk Mode klaster, pilih Diaktifkan.

    c. Untuk Info klaster, masukkan nilai untuk Nama.

    d. (Opsional) Masukkan nilai untuk Deskripsi.
7. Di bagian Lokasi:

    AWS Cloud

    1. Untuk AWS Cloud, sebaiknya terima pengaturan default untuk Multi-AZ dan Failover otomatis. Untuk informasi selengkapnya, lihat Meminimalkan waktu henti ElastiCache untuk Redis OSS dengan Multi-AZ.
    2. Pada Pengaturan klaster

        a. Untuk Versi mesin, pilih versi yang tersedia.

        b. Untuk Port, gunakan port default, 6379. Jika Anda memiliki alasan untuk menggunakan port lain, masukkan nomor port tersebut.

        c. Untuk Grup parameter, pilih grup parameter atau buat yang baru. Grup parameter mengontrol parameter runtime dari klaster Anda. Untuk informasi selengkapnya tentang grup parameter, lihat Parameter Valkey dan Redis OSS dan Membuat grup ElastiCache parameter.

> **Note**
>
> Saat Anda memilih grup parameter untuk menetapkan nilai konfigurasi mesin, grup parameter tersebut diterapkan ke semua klaster di penyimpanan data global. Pada halaman Grup Parameter, atribut Global ya/tidak menunjukkan apakah grup parameter adalah bagian dari penyimpanan data global.

d. Untuk Jenis simpul, pilih panah bawah
( ).
Pada kotak dialog Ubah jenis simpul, pilih nilai untuk Keluarga instans untuk jenis simpul yang Anda inginkan. Kemudian pilih jenis simpul yang ingin Anda gunakan untuk klaster ini, lalu pilih Simpan.

   Untuk informasi selengkapnya, lihat Memilih ukuran simpul Anda.

   Jika Anda memilih jenis simpul r6gd, tingkatan data akan diaktifkan secara otomatis. Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.

e. Untuk Jumlah pecahan, pilih jumlah pecahan (partisi/grup simpul) yang Anda inginkan untuk cluster Valkey atau Redis OSS (mode cluster diaktifkan) ini.

   Untuk beberapa versi Valkey atau Redis OSS (mode cluster diaktifkan), Anda dapat mengubah jumlah pecahan di cluster Anda secara dinamis:

   - Redis OSS 3.2.10 dan yang lebih baru - Jika cluster Anda menjalankan Redis OSS 3.2.10 atau versi yang lebih baru, Anda dapat mengubah jumlah pecahan di cluster Anda secara dinamis. Untuk informasi selengkapnya, lihat Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan).
   - Versi Redis OSS lainnya - Jika cluster Anda menjalankan versi Redis OSS sebelum versi 3.2.10, ada pendekatan lain. Untuk mengubah jumlah serpihan di klaster Anda, buat klaster baru dengan jumlah serpihan baru. Untuk informasi selengkapnya, lihat Melakukan pemulihan dari cadangan ke dalam cache baru.

f. Untuk Replika per serpihan, pilih jumlah simpul replika baca yang Anda inginkan dalam setiap serpihan.

   Pembatasan berikut ada untuk Valkey atau Redis OSS (mode cluster diaktifkan).

- Jika Multi-AZ diaktifkan, pastikan bahwa Anda memiliki setidaknya satu replika per serpihan.
- Replika akan berjumlah sama untuk setiap serpihan saat membuat klaster menggunakan konsol.
- Jumlah replika baca per serpihan bersifat tetap dan tidak dapat diubah. Jika Anda membutuhkan lebih banyak atau lebih sedikit replika per serpihan (API/CLI: grup simpul), Anda harus membuat klaster baru dengan jumlah replika yang baru. Untuk informasi selengkapnya, lihat Tutorial: Menyemai cluster baru yang dirancang sendiri dengan cadangan yang dibuat secara eksternal.

3. Di bagian Konektivitas

   a. Untuk Jenis jaringan, pilih versi IP yang akan didukung oleh klaster ini.

   b. Untuk grup Subnet, pilih subnet yang ingin Anda terapkan ke cluster ini. ElastiCache menggunakan grup subnet itu untuk memilih subnet dan alamat IP dalam subnet itu untuk dikaitkan dengan node Anda. ElastiCache cluster memerlukan subnet dual-stack dengan keduanya IPv4 dan IPv6 alamat yang ditetapkan untuk beroperasi dalam mode dual-stack dan subnet -only untuk beroperasi sebagai IPv6 -only. IPv6

      Saat membuat grup subnet baru, masukkan ID VPC yang menaungi grup subnet tersebut.

      Pilih Tipe IP Penemuan. Hanya alamat IP protokol yang Anda pilih yang dikembalikan.

      Untuk informasi selengkapnya, lihat:

      - Memilih jenis jaringan di ElastiCache.
      - Membuat subnet di VPC Anda.

      Jika Anda adalah Menggunakan zona lokal dengan ElastiCache, Anda harus membuat atau memilih subnet yang berada di zona lokal.

      Untuk informasi selengkapnya, lihat Subnet dan grup subnet.

4. Untuk Penempatan zona ketersediaan, Anda memiliki dua opsi:

   - Tidak ada preferensi — ElastiCache memilih Availability Zone.

- Tentukan zona ketersediaan – Anda menentukan Zona Ketersediaan untuk setiap klaster.

  Jika Anda memilih untuk menentukan Zona Ketersediaan, untuk setiap klaster di setiap serpihan, pilih Zona Ketersediaan dari daftar.

Untuk informasi selengkapnya, lihat Memilih wilayah dan zona ketersediaan untuk ElastiCache.

5. Pilih Berikutnya
6. Di bawah pengaturan Advanced Valkey atau pengaturan Advanced Redis OSS atau

   - Untuk Keamanan:

     i. Untuk mengenkripsi data Anda, Anda memiliki opsi berikut:

     - Enkripsi diam – Mengaktifkan enkripsi pada data yang disimpan di disk. Untuk informasi selengkapnya, lihat Enkripsi Diam.

       > **Note**
       >
       > Anda memiliki opsi untuk menyediakan kunci enkripsi yang berbeda dengan memilih kunci AWS KMS yang Dikelola Pelanggan dan memilih kunci. Untuk informasi selengkapnya, lihat Menggunakan kunci yang dikelola pelanggan dari AWS KMS.

     - Enkripsi bergerak – Mengaktifkan enkripsi data selama pengiriman. Untuk informasi selengkapnya, lihat Enkripsi bergerak. Untuk Valkey 7.2 dan di atasnya atau Redis OSS 6.0 ke atas, jika Anda mengaktifkan Enkripsi dalam perjalanan, Anda akan diminta untuk menentukan salah satu opsi Kontrol Akses berikut:

       - Tanpa Kontrol Akses – Ini adalah pengaturan default. Opsi ini menunjukkan bahwa tidak ada batasan akses pengguna ke klaster.
       - Daftar Kontrol Akses Grup Pengguna – Pilih grup pengguna dengan kumpulan pengguna tertentu yang dapat mengakses klaster. Untuk informasi selengkapnya, lihat Mengelola Grup Pengguna dengan Konsol dan CLI.

- AUTH Default User - Mekanisme otentikasi untuk server Valkey atau Redis OSS. Untuk informasi lebih lanjut, lihat AUTH.
- AUTH — Mekanisme otentikasi untuk server Valkey atau Redis OSS. Untuk informasi lebih lanjut, lihat AUTH.

> **Note**
>
> Untuk versi Redis OSS antara 3.2.6 dan seterusnya, tidak termasuk versi 3.2.10, AUTH adalah satu-satunya pilihan.

ii. Untuk Grup keamanan, pilih grup keamanan yang Anda inginkan untuk klaster ini. Grup keamanan bertindak sebagai firewall untuk mengontrol akses jaringan ke klaster Anda. Anda dapat menggunakan grup keamanan default untuk VPC Anda atau membuat yang baru.

Untuk informasi selengkapnya tentang grup keamanan, lihat Grup Keamanan untuk VPC Anda dalam Panduan Pengguna Amazon VPC.

7. Untuk pencadangan otomatis terjadwal secara berkala, pilih Aktifkan pencadangan otomatis, lalu masukkan jumlah hari yang diinginkan untuk mempertahankan cadangan otomatis sebelum dihapus secara otomatis. Jika Anda tidak ingin melakukan pencadangan otomatis terjadwal secara berkala, hapus kotak centang Aktifkan pencadangan otomatis. Apa pun pilihannya, Anda dapat membuat pencadangan secara manual kapan saja.

   Untuk informasi selengkapnya tentang pencadangan dan pemulihan, lihatMelakukan snapshot dan pemulihan.

8. (Opsional) Tentukan periode pemeliharaan. Jendela pemeliharaan adalah waktu yang biasanya satu jam setiap minggu saat ElastiCache menjadwalkan pemeliharaan sistem untuk klaster Anda. Anda dapat mengizinkan ElastiCache untuk memilih hari dan waktu untuk jendela pemeliharaan Anda (Tidak ada preferensi), atau Anda dapat memilih hari, waktu, dan durasi sendiri (Tentukan jendela pemeliharaan). Jika Anda memilih Tentukan periode pemeliharaan dari daftar, pilih Hari mulai, Waktu mulai, dan Durasi (dalam jam) untuk periode pemeliharaan. Semua waktu menggunakan zona waktu UTC.

   Untuk informasi selengkapnya, lihat Mengelola pemeliharaan ElastiCache cluster.

9. (Opsional) Untuk Log:

- Di bagian Format log, pilih Teks atau JSON.
- Di bawah Jenis Tujuan, pilih CloudWatch Log atau Kinesis Firehose.
- Di bawah Tujuan log, pilih Buat baru dan masukkan nama grup CloudWatch log Log atau nama aliran Firehose Anda, atau pilih Pilih yang ada, lalu pilih nama grup CloudWatch log Log atau nama aliran Firehose Anda,

10. Untuk Tag, untuk membantu mengelola cluster dan ElastiCache sumber daya lainnya, Anda dapat menetapkan metadata Anda sendiri ke setiap sumber daya dalam bentuk tag. Untuk informasi selengkapnya, lihat Menandai sumber daya Anda ElastiCache .
11. Pilih Berikutnya.
12. Tinjau semua entri dan pilihan Anda, lalu lakukan koreksi yang diperlukan. Saat Anda siap, pilih Buat.

On premises

1. Untuk On-premise, sebaiknya Anda membiarkan Failover otomatis tetap aktif. Untuk informasi selengkapnya, lihat Meminimalkan waktu henti ElastiCache untuk Redis OSS dengan Multi-AZ
2. Ikuti langkah-langkahnya dalam Menggunakan Outposts.

Untuk membuat ekuivalen menggunakan ElastiCache API atau AWS CLI bukan ElastiCache konsol, lihat yang berikut ini:

- API: CreateReplicationGroup
- CLI: create-replication-group

Segera setelah status klaster Anda tersedia, Anda dapat memberikan EC2 akses ke sana, terhubung ke sana, dan mulai menggunakannya. Untuk informasi selengkapnya, lihat Langkah 3. Otorisasi akses ke cluster dan Langkah 4. Connect ke node cluster.

> **Important**
>
> Setelah klaster Anda tersedia, Anda akan ditagih untuk setiap jam atau jam parsial saat klaster aktif, meskipun Anda tidak sedang aktif menggunakannya. Untuk menghentikan

tagihan untuk klaster ini, Anda harus menghapusnya. Lihat Menghapus cluster di ElastiCache.

Membuat klaster (AWS CLI)

Untuk membuat cluster menggunakan AWS CLI, gunakan `create-cache-cluster` perintah.

> **Important**
>
> Setelah klaster Anda tersedia, Anda akan ditagih untuk setiap jam atau jam parsial saat klaster aktif, meskipun Anda tidak sedang aktif menggunakannya. Untuk menghentikan tagihan untuk klaster ini, Anda harus menghapusnya. Lihat Menghapus cluster di ElastiCache.

Membuat cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) (CLI)

Example — Cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) tanpa replika baca

Kode CLI berikut membuat cluster cache Valkey atau Redis OSS (mode cluster dinonaktifkan) tanpa replika.

> **Note**
>
> Ketika membuat klaster menggunakan jenis simpul dari keluarga r6gd, Anda harus meneruskan parameter `data-tiering-enabled`.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-cache-cluster \
--cache-cluster-id my-cluster \
--cache-node-type cache.r4.large \
--engine redis \
--num-cache-nodes 1 \
--cache-parameter-group default.redis6.x \
--snapshot-arns arn:aws:s3:::amzn-s3-demo-bucket/snapshot.rdb
```

Untuk Windows:

```
aws elasticache create-cache-cluster ^
--cache-cluster-id my-cluster ^
--cache-node-type cache.r4.large ^
```

```
--engine redis ^
--num-cache-nodes 1 ^
--cache-parameter-group default.redis6.x ^
--snapshot-arns arn:aws:s3:::amzn-s3-demo-bucket/snapshot.rdb
```

Membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) ()AWS CLI

Cluster Valkey atau Redis OSS (mode cluster enabled) (API/CLI: grup replikasi) tidak dapat dibuat menggunakan operasi. `create-cache-cluster` Untuk membuat cluster Valkey atau Redis OSS (mode cluster enabled) (API/CLI: grup replikasi), lihat. Membuat grup replikasi Valkey atau Redis OSS (Cluster Mode Enabled) dari awal ()AWS CLI

Untuk informasi selengkapnya, lihat topik ElastiCache referensi AWS CLI untuk `create-replication-group`.

Membuat cluster untuk Valkey atau Redis OSS (API) ElastiCache

Untuk membuat cluster menggunakan ElastiCache API, gunakan `CreateCacheCluster` tindakan.

> **Important**
>
> Setelah klaster Anda tersedia, Anda akan ditagih untuk setiap jam atau jam parsial saat klaster aktif, meskipun Anda tidak menggunakannya. Untuk menghentikan tagihan untuk klaster ini, Anda harus menghapusnya. Lihat Menghapus cluster di ElastiCache.

Topik

- Membuat cluster cache Valkey atau Redis OSS (mode cluster dinonaktifkan) (API) ElastiCache
- Membuat cluster cache di Valkey atau Redis OSS (mode cluster diaktifkan) (API) ElastiCache

Membuat cluster cache Valkey atau Redis OSS (mode cluster dinonaktifkan) (API) ElastiCache

Kode berikut membuat Valkey atau Redis OSS (mode cluster dinonaktifkan) cache cluster (ElastiCache API).

Jeda baris ditambahkan agar dapat lebih mudah dibaca.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=CreateCacheCluster
    &CacheClusterId=my-cluster
```

```
    &CacheNodeType=cache.r4.large
    &CacheParameterGroup=default.redis3.2
    &Engine=redis
    &EngineVersion=3.2.4
    &NumCacheNodes=1
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &SnapshotArns.member.1=arn%3Aaws%3As3%3A%3A%3AmyS3Bucket%2Fdump.rdb
    &Timestamp=20150508T220302Z
    &Version=2015-02-02
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Credential=<credential>
    &X-Amz-Date=20150508T220302Z
    &X-Amz-Expires=20150508T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Signature=<signature>
```

Membuat cluster cache di Valkey atau Redis OSS (mode cluster diaktifkan) (API) ElastiCache

Cluster Valkey atau Redis OSS (mode cluster enabled) (API/CLI: grup replikasi) tidak dapat dibuat menggunakan operasi. `CreateCacheCluster` Untuk membuat cluster Valkey atau Redis OSS (mode cluster enabled) (API/CLI: grup replikasi), lihat. Membuat grup replikasi di Valkey atau Redis OSS (Mode Cluster Diaktifkan) dari awal (API) ElastiCache

Untuk informasi selengkapnya, lihat topik referensi ElastiCache API `CreateReplicationGroup`.

Membuat cluster untuk Memcached

Contoh berikut menunjukkan cara membuat cluster menggunakan AWS Management Console, AWS CLI dan ElastiCache API.

Membuat klaster Memcached (konsol)

Saat Anda menggunakan mesin Memcached, Amazon ElastiCache mendukung partisi data Anda secara horizontal melalui beberapa node. Memcached memungkinkan penemuan otomatis agar Anda tidak perlu melacak titik akhir untuk setiap simpul. Memcached melacak titik akhir setiap simpul, dengan memperbarui daftar titik akhir seiring penambahan dan penghapusan simpul. Aplikasi Anda hanya memerlukan titik akhir konfigurasi agar dapat berinteraksi dengan klaster.

Untuk membuat klaster Memcached melalui konsol, ikuti langkah-langkah di Membuat klaster Valkey (mode cluster dinonaktifkan) (Konsol). Saat Anda mencapai langkah lima, pilih Buat cache Memcached.

Segera setelah status klaster Anda tersedia, Anda dapat memberikan Amazon EC2 akses ke sana, terhubung ke sana, dan mulai menggunakannya. Untuk informasi selengkapnya, lihat langkah-langkah serupa Langkah 3. Otorisasi akses ke cluster danLangkah 4. Connect ke node cluster.

> **Important**
>
> Setelah klaster Anda tersedia, Anda akan ditagih untuk setiap jam atau jam parsial saat klaster aktif, meskipun Anda tidak sedang aktif menggunakannya. Untuk menghentikan tagihan untuk klaster ini, Anda harus menghapusnya. Lihat Menghapus cluster di ElastiCache.

Membuat klaster (AWS CLI)

Untuk membuat cluster menggunakan AWS CLI, gunakan `create-cache-cluster` perintah.

> **Important**
>
> Setelah klaster Anda tersedia, Anda akan ditagih untuk setiap jam atau jam parsial saat klaster aktif, meskipun Anda tidak sedang aktif menggunakannya. Untuk menghentikan tagihan untuk klaster ini, Anda harus menghapusnya. Lihat Menghapus cluster di ElastiCache.

Membuat Klaster Cache Memcached AWS CLI

Kode CLI berikut membuat klaster cache Memcached dengan 3 simpul.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-cache-cluster \
--cache-cluster-id my-cluster \
--cache-node-type cache.r4.large \
--engine memcached \
--engine-version 1.4.24 \
--cache-parameter-group default.memcached1.4 \
--num-cache-nodes 3
```

Untuk Windows:

```
aws elasticache create-cache-cluster ^
--cache-cluster-id my-cluster ^
--cache-node-type cache.r4.large ^
--engine memcached ^
--engine-version 1.4.24 ^
--cache-parameter-group default.memcached1.4 ^
--num-cache-nodes 3
```

Membuat cluster untuk Memcached (API) ElastiCache

Untuk membuat cluster menggunakan ElastiCache API, gunakan `CreateCacheCluster` tindakan.

> **Important**
>
> Setelah klaster Anda tersedia, Anda akan ditagih untuk setiap jam atau jam parsial saat klaster aktif, meskipun Anda tidak menggunakannya. Untuk menghentikan tagihan untuk klaster ini, Anda harus menghapusnya. Lihat Menghapus cluster di ElastiCache.

Topik

- Membuat cluster cache Memcached (API) ElastiCache

Membuat cluster cache Memcached (API) ElastiCache

Kode berikut membuat cluster Memcached dengan 3 node (ElastiCache API).

Jeda baris ditambahkan agar dapat lebih mudah dibaca.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=CreateCacheCluster
    &CacheClusterId=my-cluster
    &CacheNodeType=cache.r4.large
    &Engine=memcached
    &NumCacheNodes=3
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20150508T220302Z
    &Version=2015-02-02
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Credential=<credential>
```

```
    &X-Amz-Date=20150508T220302Z
    &X-Amz-Expires=20150508T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Signature=<signature>
```

Melihat detail ElastiCache cluster

Anda dapat melihat informasi detail tentang satu atau beberapa cluster menggunakan ElastiCache konsol, AWS CLI, atau ElastiCache API.

Melihat detail klaster Memcached (Konsol)

Anda dapat melihat detail cluster Memcached menggunakan ElastiCache konsol, AWS CLI for ElastiCache, atau API. ElastiCache

Prosedur berikut merinci cara melihat detail cluster Memcached menggunakan konsol. ElastiCache

Untuk melihat detail klaster Memcached

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Dari daftar di pojok kanan atas, pilih AWS Wilayah yang Anda minati.
3. Di dasbor ElastiCache mesin, pilih Memcached. Melakukan hal ini menampilkan daftar semua cluster Anda yang berjalan pada mesin Memcached.
4. Untuk melihat detail klaster, pilih kotak di sebelah kiri nama klaster.
5. Untuk melihat informasi simpul, pilih tab Simpul, yang menampilkan informasi tentang status simpul dan titik akhir.
6. Untuk melihat metrik, pilih tab Metrik, yang menampilkan metrik yang relevan untuk semua simpul di klaster. Untuk informasi selengkapnya, lihat Pemantauan penggunaan dengan CloudWatch Metrik
7. Pilih tab Jaringan dan keamanan untuk melihat detail tentang konektivitas jaringan klaster dan konfigurasi grup subnet dan grup keamanan VPC. Untuk informasi selengkapnya, lihat Subnet dan grup subnet.
8. Pilih tab Pemeliharaan untuk melihat detail di pengaturan pemeliharaan klaster. Untuk informasi selengkapnya, lihat Mengelola pemeliharaan ElastiCache cluster.
9. Pilih tab Tag untuk melihat detail pada tag apa pun yang diterapkan ke sumber daya klaster. Untuk informasi selengkapnya, lihat Menandai sumber daya Anda ElastiCache .

Melihat detail Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol)

Anda dapat melihat detail cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) menggunakan ElastiCache konsol, AWS CLI for ElastiCache, atau API. ElastiCache

Prosedur berikut merinci cara melihat detail cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) menggunakan konsol. ElastiCache

Untuk melihat detail cluster Valkey atau Redis OSS (mode cluster dinonaktifkan)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di dasbor ElastiCache mesin, pilih Valkey atau Redis OSS untuk menampilkan daftar semua cluster Anda yang berjalan di mesin itu.
3. Untuk melihat detail klaster, pilih kotak centang di sebelah kiri nama klaster. Pastikan Anda memilih cluster yang menjalankan mesin Valkey atau Redis OSS, bukan Clustered Valkey atau Clustered Redis OSS. Tindakan ini akan menampilkan detail tentang klaster, termasuk titik akhir primer dari klaster.
4. Untuk melihat informasi simpul:
    a. Pilih nama klaster.
    b. Pilih tab Serpihan dan Simpul. Tindakan ini akan menampilkan detail tentang setiap simpul, termasuk titik akhir simpul yang perlu digunakan untuk membaca dari klaster.
5. Untuk melihat metrik, pilih tab Metrik, yang menampilkan metrik yang relevan untuk semua simpul di klaster. Untuk informasi selengkapnya, lihat Pemantauan penggunaan dengan CloudWatch Metrik
6. Untuk melihat log, pilih tab Log, yang menunjukkan apakah klaster menggunakan log Lambat atau log Mesin dan memberikan detail yang relevan. Untuk informasi selengkapnya, lihat Pengiriman log.
7. Pilih tab Jaringan dan keamanan untuk melihat detail tentang konektivitas jaringan klaster dan konfigurasi grup subnet. Untuk informasi selengkapnya, lihat Subnet dan grup subnet.
8. Pilih tab Pemeliharaan untuk melihat detail di pengaturan pemeliharaan klaster. Untuk informasi selengkapnya, lihat Mengelola pemeliharaan ElastiCache cluster.
9. Pilih tab Pembaruan layanan untuk melihat detail tentang pembaruan layanan yang tersedia beserta tanggal penerapan yang direkomendasikan. Untuk informasi selengkapnya, lihat Pembaruan layanan di ElastiCache.
10. Pilih tab Tag untuk melihat detail pada tag apa pun yang diterapkan ke sumber daya klaster. Untuk informasi selengkapnya, lihat Menandai sumber daya Anda ElastiCache .

Melihat detail untuk cluster Valkey atau Redis OSS (Cluster Mode Enabled) (Konsol)

Anda dapat melihat detail cluster Valkey atau Redis OSS (mode cluster enabled) menggunakan ElastiCache konsol, AWS CLI for ElastiCache, atau API. ElastiCache

Prosedur berikut merinci cara melihat detail cluster Valkey atau Redis OSS (mode cluster enabled) menggunakan konsol. ElastiCache

Untuk melihat detail cluster Valkey atau Redis OSS (mode cluster enabled)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Dari daftar di pojok kanan atas, pilih AWS Wilayah yang Anda minati.
3. Di dasbor ElastiCache mesin, pilih Valkey atau Redis OSS untuk menampilkan daftar semua cluster Anda yang berjalan di mesin itu.
4. Untuk melihat detail cluster Valkey atau Redis OSS (mode cluster enabled), pilih kotak di sebelah kiri nama cluster. Pastikan Anda memilih cluster yang menjalankan mesin Valkey atau Clustered Redis OSS.

   Layar akan diperluas di bawah klaster dan menampilkan detail tentang klaster, termasuk titik akhir konfigurasi klaster.
5. Untuk melihat daftar serpihan klaster dan jumlah simpul di setiap serpihan, pilih tab Serpihan dan simpul.
6. Untuk melihat informasi spesifik tentang sebuah simpul:

   - Pilih ID serpihan.

     Tindakan ini akan menampilkan informasi tentang setiap simpul, termasuk titik akhir simpul yang perlu digunakan untuk membaca dari klaster.
7. Untuk melihat metrik, pilih tab Metrik, yang menampilkan metrik yang relevan untuk semua simpul di klaster. Untuk informasi selengkapnya, lihat Pemantauan penggunaan dengan CloudWatch Metrik
8. Untuk melihat log, pilih tab Log, yang menunjukkan apakah klaster menggunakan log Lambat atau log Mesin dan memberikan detail yang relevan. Untuk informasi selengkapnya, lihat Pengiriman log.
9. Pilih tab Jaringan dan keamanan untuk melihat detail tentang konektivitas jaringan klaster dan konfigurasi grup subnet, grup keamanan VPC dan metode enkripsi hal yang diaktifkan di klaster,

jika ada. Untuk informasi selengkapnya, lihat Subnet dan grup subnet dan Keamanan data di Amazon ElastiCache.

10. Pilih tab Pemeliharaan untuk melihat detail di pengaturan pemeliharaan klaster. Untuk informasi selengkapnya, lihat Mengelola pemeliharaan ElastiCache cluster.
11. Pilih tab Pembaruan layanan untuk melihat detail tentang pembaruan layanan yang tersedia beserta tanggal penerapan yang direkomendasikan. Untuk informasi selengkapnya, lihat Pembaruan layanan di ElastiCache.
12. Pilih tab Tag untuk melihat detail pada tag apa pun yang diterapkan ke sumber daya klaster. Untuk informasi selengkapnya, lihat Menandai sumber daya Anda ElastiCache .

Melihat detail ElastiCache cluster (AWS CLI)

Kode berikut mencantumkan rincian untuk*my-cluster*:

```
aws elasticache describe-cache-clusters --cache-cluster-id my-cluster
```

Ganti *my-cluster* dengan nama cluster Anda dalam kasus di mana cluster dibuat dengan 1 node cache dan 0 pecahan menggunakan `create-cache-cluster` perintah.

```
{
    "CacheClusters": [
        {
            "CacheClusterStatus": "available",
            "SecurityGroups": [
                {
                    "Status": "active",
                    "SecurityGroupId": "sg-dbe93fa2"
                }
            ],
            "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
            "Engine": "redis",
            "PreferredMaintenanceWindow": "wed:12:00-wed:13:00",
            "CacheSubnetGroupName": "default",
            "SnapshotWindow": "08:30-09:30",
            "TransitEncryptionEnabled": false,
            "AtRestEncryptionEnabled": false,
            "CacheClusterId": "my-cluster1",
            "CacheClusterCreateTime": "2018-02-26T21:06:43.420Z",
            "PreferredAvailabilityZone": "us-west-2c",
```

```
            "AuthTokenEnabled": false,
            "PendingModifiedValues": {},
            "CacheNodeType": "cache.r4.large",
           "DataTiering": "disabled",
            "CacheParameterGroup": {
                "CacheNodeIdsToReboot": [],
                "ParameterApplyStatus": "in-sync",
                "CacheParameterGroupName": "default.redis3.2"
            },
            "SnapshotRetentionLimit": 0,
            "AutoMinorVersionUpgrade": true,
            "EngineVersion": "3.2.10",
            "CacheSecurityGroups": [],
            "NumCacheNodes": 1
        }
```

```
{
    "CacheClusters": [
        {
            "SecurityGroups": [
                {
                    "Status": "active",
                    "SecurityGroupId": "sg-dbe93fa2"
                }
            ],
            "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
            "AuthTokenEnabled": false,
            "CacheSubnetGroupName": "default",
            "SnapshotWindow": "12:30-13:30",
            "AutoMinorVersionUpgrade": true,
            "CacheClusterCreateTime": "2018-02-26T21:13:24.250Z",
            "CacheClusterStatus": "available",
            "AtRestEncryptionEnabled": false,
            "PreferredAvailabilityZone": "us-west-2a",
            "TransitEncryptionEnabled": false,
            "ReplicationGroupId": "my-cluster2",
            "Engine": "redis",
            "PreferredMaintenanceWindow": "sun:08:30-sun:09:30",
            "CacheClusterId": "my-cluster2-001",
            "PendingModifiedValues": {},
            "CacheNodeType": "cache.r4.large",
            "DataTiering": "disabled",
```

```
            "CacheParameterGroup": {
                "CacheNodeIdsToReboot": [],
                "ParameterApplyStatus": "in-sync",
                "CacheParameterGroupName": "default.redis6.x"
            },
            "SnapshotRetentionLimit": 0,
            "EngineVersion": "6.0",
            "CacheSecurityGroups": [],
            "NumCacheNodes": 1
        },
        {
            "SecurityGroups": [
                {
                    "Status": "active",
                    "SecurityGroupId": "sg-dbe93fa2"
                }
            ],
            "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
            "AuthTokenEnabled": false,
            "CacheSubnetGroupName": "default",
            "SnapshotWindow": "12:30-13:30",
            "AutoMinorVersionUpgrade": true,
            "CacheClusterCreateTime": "2018-02-26T21:13:24.250Z",
            "CacheClusterStatus": "available",
            "AtRestEncryptionEnabled": false,
            "PreferredAvailabilityZone": "us-west-2b",
            "TransitEncryptionEnabled": false,
            "ReplicationGroupId": "my-cluster2",
            "Engine": "redis",
            "PreferredMaintenanceWindow": "sun:08:30-sun:09:30",
            "CacheClusterId": "my-cluster2-002",
            "PendingModifiedValues": {},
            "CacheNodeType": "cache.r4.large",
            "DataTiering": "disabled",
            "CacheParameterGroup": {
                "CacheNodeIdsToReboot": [],
                "ParameterApplyStatus": "in-sync",
                "CacheParameterGroupName": "default.redis6.x"
            },
            "SnapshotRetentionLimit": 0,
            "EngineVersion": "6.0",
            "CacheSecurityGroups": [],
            "NumCacheNodes": 1
```

```
        },
        {
            "SecurityGroups": [
                {
                    "Status": "active",
                    "SecurityGroupId": "sg-dbe93fa2"
                }
            ],
            "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
            "AuthTokenEnabled": false,
            "CacheSubnetGroupName": "default",
            "SnapshotWindow": "12:30-13:30",
            "AutoMinorVersionUpgrade": true,
            "CacheClusterCreateTime": "2018-02-26T21:13:24.250Z",
            "CacheClusterStatus": "available",
            "AtRestEncryptionEnabled": false,
            "PreferredAvailabilityZone": "us-west-2c",
            "TransitEncryptionEnabled": false,
            "ReplicationGroupId": "my-cluster2",
            "Engine": "redis",
            "PreferredMaintenanceWindow": "sun:08:30-sun:09:30",
            "CacheClusterId": "my-cluster2-003",
            "PendingModifiedValues": {},
            "CacheNodeType": "cache.r4.large",
            "DataTiering": "disabled",
            "CacheParameterGroup": {
                "CacheNodeIdsToReboot": [],
                "ParameterApplyStatus": "in-sync",
                "CacheParameterGroupName": "default.redis3.2"
            },
            "SnapshotRetentionLimit": 0,
            "EngineVersion": "3.2.10",
            "CacheSecurityGroups": [],
            "NumCacheNodes": 1
        }
```

```
{
    "CacheClusters": [
        {
            "SecurityGroups": [
                {
                    "Status": "active",
```

```
                    "SecurityGroupId": "sg-dbe93fa2"
                }
            ],
            "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
            "AuthTokenEnabled": true,
            "CacheSubnetGroupName": "default",
            "SnapshotWindow": "12:30-13:30",
            "AutoMinorVersionUpgrade": true,
            "CacheClusterCreateTime": "2018-02-26T21:17:01.439Z",
            "CacheClusterStatus": "available",
            "AtRestEncryptionEnabled": true,
            "PreferredAvailabilityZone": "us-west-2a",
            "TransitEncryptionEnabled": true,
            "ReplicationGroupId": "my-cluster3",
            "Engine": "redis",
            "PreferredMaintenanceWindow": "thu:11:00-thu:12:00",
            "CacheClusterId": "my-cluster3-0001-001",
            "PendingModifiedValues": {},
            "CacheNodeType": "cache.r4.large",
            "DataTiering": "disabled",
            "CacheParameterGroup": {
                "CacheNodeIdsToReboot": [],
                "ParameterApplyStatus": "in-sync",
                "CacheParameterGroupName": "default.redis6.x.cluster.on"
            },
            "SnapshotRetentionLimit": 0,
            "EngineVersion": "6.0",
            "CacheSecurityGroups": [],
            "NumCacheNodes": 1
        },
        {
            "SecurityGroups": [
                {
                    "Status": "active",
                    "SecurityGroupId": "sg-dbe93fa2"
                }
            ],
            "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
            "AuthTokenEnabled": true,
            "CacheSubnetGroupName": "default",
            "SnapshotWindow": "12:30-13:30",
            "AutoMinorVersionUpgrade": true,
```

```
            "CacheClusterCreateTime": "2018-02-26T21:17:01.439Z",
            "CacheClusterStatus": "available",
            "AtRestEncryptionEnabled": true,
            "PreferredAvailabilityZone": "us-west-2b",
            "TransitEncryptionEnabled": true,
            "ReplicationGroupId": "my-cluster3",
            "Engine": "redis",
            "PreferredMaintenanceWindow": "thu:11:00-thu:12:00",
            "CacheClusterId": "my-cluster3-0001-002",
            "PendingModifiedValues": {},
            "CacheNodeType": "cache.r4.large",
             "DataTiering": "disabled",
            "CacheParameterGroup": {
                "CacheNodeIdsToReboot": [],
                "ParameterApplyStatus": "in-sync",
                "CacheParameterGroupName": "default.redis3.2.cluster.on"
            },
            "SnapshotRetentionLimit": 0,
            "EngineVersion": "3.2.6",
            "CacheSecurityGroups": [],
            "NumCacheNodes": 1
        },
        {
            "SecurityGroups": [
                {
                    "Status": "active",
                    "SecurityGroupId": "sg-dbe93fa2"
                }
            ],
            "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
            "AuthTokenEnabled": true,
            "CacheSubnetGroupName": "default",
            "SnapshotWindow": "12:30-13:30",
            "AutoMinorVersionUpgrade": true,
            "CacheClusterCreateTime": "2018-02-26T21:17:01.439Z",
            "CacheClusterStatus": "available",
            "AtRestEncryptionEnabled": true,
            "PreferredAvailabilityZone": "us-west-2c",
            "TransitEncryptionEnabled": true,
            "ReplicationGroupId": "my-cluster3",
            "Engine": "redis",
            "PreferredMaintenanceWindow": "thu:11:00-thu:12:00",
            "CacheClusterId": "my-cluster3-0001-003",
```

```
            "PendingModifiedValues": {},
            "CacheNodeType": "cache.r4.large",
             "DataTiering": "disabled",
            "CacheParameterGroup": {
                "CacheNodeIdsToReboot": [],
                "ParameterApplyStatus": "in-sync",
                "CacheParameterGroupName": "default.redis6.x.cluster.on"
            },
            "SnapshotRetentionLimit": 0,
            "EngineVersion": "6.0",
            "CacheSecurityGroups": [],
            "NumCacheNodes": 1
        },
        {
            "SecurityGroups": [
                {
                    "Status": "active",
                    "SecurityGroupId": "sg-dbe93fa2"
                }
            ],
            "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
            "AuthTokenEnabled": true,
            "CacheSubnetGroupName": "default",
            "SnapshotWindow": "12:30-13:30",
            "AutoMinorVersionUpgrade": true,
            "CacheClusterCreateTime": "2018-02-26T21:17:01.439Z",
            "CacheClusterStatus": "available",
            "AtRestEncryptionEnabled": true,
            "PreferredAvailabilityZone": "us-west-2b",
            "TransitEncryptionEnabled": true,
            "ReplicationGroupId": "my-cluster3",
            "Engine": "redis",
            "PreferredMaintenanceWindow": "thu:11:00-thu:12:00",
            "CacheClusterId": "my-cluster3-0002-001",
            "PendingModifiedValues": {},
            "CacheNodeType": "cache.r4.large",
             "DataTiering": "disabled",
            "CacheParameterGroup": {
                "CacheNodeIdsToReboot": [],
                "ParameterApplyStatus": "in-sync",
                "CacheParameterGroupName": "default.redis6.x.cluster.on"
            },
            "SnapshotRetentionLimit": 0,
```

```
            "EngineVersion": "6.0",
            "CacheSecurityGroups": [],
            "NumCacheNodes": 1
        },
        {
            "SecurityGroups": [
                {
                    "Status": "active",
                    "SecurityGroupId": "sg-dbe93fa2"
                }
            ],
            "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
            "AuthTokenEnabled": true,
            "CacheSubnetGroupName": "default",
            "SnapshotWindow": "12:30-13:30",
            "AutoMinorVersionUpgrade": true,
            "CacheClusterCreateTime": "2018-02-26T21:17:01.439Z",
            "CacheClusterStatus": "available",
            "AtRestEncryptionEnabled": true,
            "PreferredAvailabilityZone": "us-west-2c",
            "TransitEncryptionEnabled": true,
            "ReplicationGroupId": "my-cluster3",
            "Engine": "redis",
            "PreferredMaintenanceWindow": "thu:11:00-thu:12:00",
            "CacheClusterId": "my-cluster3-0002-002",
            "PendingModifiedValues": {},
            "CacheNodeType": "cache.r4.large",
             "DataTiering": "disabled",
            "CacheParameterGroup": {
                "CacheNodeIdsToReboot": [],
                "ParameterApplyStatus": "in-sync",
                "CacheParameterGroupName": "default.redis3.2.cluster.on"
            },
            "SnapshotRetentionLimit": 0,
            "EngineVersion": "3.2.6",
            "CacheSecurityGroups": [],
            "NumCacheNodes": 1
        },
        {
            "SecurityGroups": [
                {
                    "Status": "active",
                    "SecurityGroupId": "sg-dbe93fa2"
```

```
                }
            ],
            "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
            "AuthTokenEnabled": true,
            "CacheSubnetGroupName": "default",
            "SnapshotWindow": "12:30-13:30",
            "AutoMinorVersionUpgrade": true,
            "CacheClusterCreateTime": "2018-02-26T21:17:01.439Z",
            "CacheClusterStatus": "available",
            "AtRestEncryptionEnabled": true,
            "PreferredAvailabilityZone": "us-west-2a",
            "TransitEncryptionEnabled": true,
            "ReplicationGroupId": "my-cluster3",
            "Engine": "redis",
            "PreferredMaintenanceWindow": "thu:11:00-thu:12:00",
            "CacheClusterId": "my-cluster3-0002-003",
            "PendingModifiedValues": {},
            "CacheNodeType": "cache.r4.large",
             "DataTiering": "disabled",
            "CacheParameterGroup": {
                "CacheNodeIdsToReboot": [],
                "ParameterApplyStatus": "in-sync",
                "CacheParameterGroupName": "default.redis6.x.cluster.on"
            },
            "SnapshotRetentionLimit": 0,
            "EngineVersion": "6.0",
            "CacheSecurityGroups": [],
            "NumCacheNodes": 1
        }
    ]
}
```

Dalam kasus di mana cluster dibuat menggunakan AWS Management Console (node cluster diaktifkan atau dinonaktifkan dengan 1 atau lebih pecahan), gunakan perintah berikut untuk menjelaskan detail cluster (ganti *my-cluster* dengan nama grup replikasi (nama cluster Anda):

```
aws elasticache describe-replication-groups --replication-group-id my-cluster
```

Untuk informasi lebih lanjut, lihat ElastiCache topik AWS CLI untuk describe-cache-clusters.

Melihat detail ElastiCache klaster (ElastiCache API)

Anda dapat melihat detail untuk klaster menggunakan `DescribeCacheClusters` tindakan ElastiCache API. Jika parameter `CacheClusterId` disertakan, detail untuk klaster yang ditentukan akan ditampilkan. Jika parameter `CacheClusterId` dihilangkan, detail untuk maksimal `MaxRecords` klaster (default-nya 100) akan ditampilkan. Nilai untuk `MaxRecords` tidak boleh kurang dari 20 atau lebih dari 100.

Kode berikut menampilkan daftar detail untuk `my-cluster`.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DescribeCacheClusters
   &CacheClusterId=my-cluster
   &Version=2015-02-02
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &X-Amz-Credential=<credential>
```

Kode berikut menampilkan daftar detail untuk maksimal 25 klaster.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DescribeCacheClusters
   &MaxRecords=25
   &Version=2015-02-02
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya, lihat topik referensi ElastiCache API `DescribeCacheClusters`.

Memodifikasi cluster ElastiCache

Selain menambahkan atau menghapus node dari ElastiCache cluster, mungkin ada saat-saat di mana Anda perlu membuat perubahan lain seperti menambahkan grup keamanan, mengubah jendela pemeliharaan atau grup parameter.

Sebaiknya atur periode pemeliharaan Anda pada waktu penggunaan terendah. Jadi, Anda mungkin perlu mengubahnya dari waktu ke waktu.

Saat Anda mengubah parameter klaster, perubahan diterapkan pada klaster secara langsung atau setelah klaster dimulai ulang. Ini berlaku terlepas dari apakah Anda mengubah grup parameter klaster itu sendiri atau nilai parameter di dalam grup parameter klaster. Untuk menentukan kapan perubahan parameter tertentu diterapkan, lihat bagian Perubahan Berpengaruh pada kolom Detail dalam tabel untuk Parameter spesifik Memcacheddan Parameter Valkey dan Redis OSS. Untuk informasi tentang cara melakukan boot ulang simpul klaster, lihat Mem-boot ulang node.

Menggunakan ElastiCache AWS Management Console

Untuk mengubah klaster

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari daftar di sudut kanan atas, pilih AWS Wilayah tempat cluster yang ingin Anda modifikasi berada.
3. Di panel navigasi, pilih mesin yang berjalan pada klaster yang ingin diubah.

   Daftar klaster mesin yang dipilih akan muncul.
4. Dalam daftar klaster tersebut, pilih nama klaster yang ingin Anda ubah.
5. Pilih Tindakan, lalu pilih Ubah.

   Jendela Ubah Klaster akan muncul.
6. Di jendela Ubah Klaster, lakukan perubahan yang Anda inginkan. Opsinya meliputi:

   - Deskripsi
   - Mode klaster - Untuk mengubah mode klaster dari Dinonaktifkan ke Diaktifkan, Anda harus terlebih dahulu mengatur mode klaster ke Kompatibel.

     Mode yang kompatibel memungkinkan klien Valkey atau Redis OSS Anda terhubung menggunakan mode cluster diaktifkan dan mode cluster dinonaktifkan. Setelah Anda

memigrasikan semua klien Valkey atau Redis OSS untuk menggunakan mode cluster yang diaktifkan, Anda kemudian dapat menyelesaikan konfigurasi mode cluster dan mengatur mode cluster ke Diaktifkan.

- Kompatibilitas Versi Mesin

> **Important**
>
> Anda dapat meningkatkan ke versi mesin yang lebih baru. Jika Anda meningkatkan versi utama mesin, misalnya dari 5.0.6 ke 6.0, Anda perlu memilih keluarga grup parameter yang kompatibel dengan versi mesin yang baru. Untuk informasi selengkapnya tentang cara melakukannya, lihat Manajemen Versi untuk ElastiCache. Namun, Anda tidak dapat menurunkan ke versi mesin yang lebih lama kecuali dengan menghapus klaster yang ada dan membuatnya lagi.

- Grup Keamanan VPC
- Grup Parameter
- Jenis Simpul

> **Note**
>
> Jika klaster menggunakan jenis simpul dari keluarga r6gd, Anda hanya dapat memilih ukuran simpul yang berbeda dari dalam keluarga tersebut. Jika Anda memilih jenis simpul dari keluarga r6gd, tingkatan data akan diaktifkan secara otomatis. Untuk informasi selengkapnya, lihat Tingkatan data.

- Multi-AZ
- Failover otomatis (hanya mode klaster dinonaktifkan)
- Aktifkan Pencadangan Otomatis
- ID Simpul Cadangan
- Periode Retensi Cadangan
- Periode Pencadangan
- Topik untuk Notifikasi SNS

- Kompatibilitas Versi Mesin Memcached
- Jenis jaringan

> **Note**
>
> Jika Anda beralih dari IPv4 ke IPv6, Anda harus memilih atau membuat grup subnet yang kompatibel dengannya IPv6. Untuk informasi selengkapnya, lihat Memilih jenis jaringan di ElastiCache.

- Grup Keamanan VPC
- Grup Parameter
- Periode Pemeliharaan
- Topik untuk Notifikasi SNS

Kotak Terapkan Segera hanya berlaku untuk versi mesin dan modifikasi tipe node. Untuk menerapkan perubahan secara langsung, pilih kotak centang Terapkan Segera. Jika kotak ini tidak dipilih, perubahan versi mesin diterapkan pada periode pemeliharaan berikutnya. Perubahan lain, seperti mengubah jendela pemeliharaan, akan diterapkan segera.

Untuk enable/disable mencatat pengiriman untuk Redis

1. Dari daftar klaster, pilih klaster yang ingin diubah. Pilih Nama klaster, bukan kotak centang di sampingnya.
2. Pada halaman detail Cluster, pilih tab Log.
3. Untuk mengaktifkan atau menonaktifkan log lambat, pilih Aktifkan atau Nonaktifkan.

   Jika Anda memilih Aktifkan:

   a. Di bagian Format log, pilih Teks atau JSON.

   b. Di bawah Jenis tujuan Log, pilih CloudWatch Log atau Kinesis Firehose.

   c. Di bawah Tujuan log, Anda dapat memilih Buat baru dan masukkan nama grup CloudWatchLogs log atau nama aliran Kinesis Data Firehose Anda. Anda juga dapat memilih Pilih yang ada dan kemudian memilih nama grup CloudWatchLogs log atau nama aliran Kinesis Data Firehose Anda.

   d. Pilih Aktifkan.

Untuk mengubah konfigurasi Anda untuk Redis:

1. Pilih Ubah.
2. Di bagian Format log, pilih Teks atau JSON.
3. Di bawah Jenis Tujuan, pilih CloudWatch Log atau Kinesis Firehose.
4. Di bawah Tujuan log, pilih Buat baru dan masukkan nama grup CloudWatchLogs log Anda atau nama aliran Kinesis Data Firehose Anda. Atau pilih Pilih yang ada lalu pilih nama grup CloudWatchLogs log atau nama aliran Kinesis Data Firehose Anda.

Menggunakan AWS CLI dengan ElastiCache

Anda dapat memodifikasi cluster yang ada menggunakan AWS CLI `modify-cache-cluster` operasi. Untuk mengubah nilai konfigurasi klaster, tentukan ID klaster, parameter yang akan diubah, dan nilai baru parameter. Contoh berikut mengubah periode pemeliharaan untuk klaster bernama `my-cluster` dan menerapkan perubahan tersebut secara langsung.

> **Important**
>
> Anda dapat meningkatkan ke versi mesin Memcached yang lebih baru. Untuk informasi selengkapnya tentang cara melakukannya, lihat Manajemen Versi untuk ElastiCache. Namun, Anda tidak dapat menurunkan ke versi mesin yang lebih lama kecuali dengan menghapus klaster yang ada dan membuatnya lagi.

> **Important**
>
> Anda dapat meningkatkan ke versi mesin Valkey atau Redis OSS yang lebih baru. Jika Anda meningkatkan versi mesin utama, misalnya dari Redis OSS 5.0.6 ke Redis OSS 6.0, Anda perlu memilih keluarga grup parameter yang kompatibel dengan versi mesin baru. Untuk informasi selengkapnya tentang cara melakukannya, lihat Manajemen Versi untuk ElastiCache. Namun, Anda tidak dapat menurunkan ke versi mesin yang lebih lama kecuali dengan menghapus klaster yang ada dan membuatnya lagi.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-cache-cluster \
```

```
    --cache-cluster-id my-cluster \
    --preferred-maintenance-window sun:23:00-mon:02:00
```

Untuk Windows:

```
aws elasticache modify-cache-cluster ^
    --cache-cluster-id my-cluster ^
    --preferred-maintenance-window sun:23:00-mon:02:00
```

Parameter `--apply-immediately` hanya berlaku untuk perubahan pada jenis simpul, versi mesin, dan jumlah simpul di klaster. Jika Anda ingin menerapkan salah satu perubahan ini segera, gunakan parameter `--apply-immediately`. Jika Anda lebih memilih untuk menunda perubahan ini ke periode pemeliharaan berikutnya, gunakan parameter `--no-apply-immediately`. Perubahan lain, seperti perubahan periode pemeliharaan, akan diterapkan segera.

Untuk informasi lebih lanjut, lihat ElastiCache topik AWS CLI untuk `modify-cache-cluster`.

Menggunakan ElastiCache API

Anda dapat memodifikasi cluster yang ada menggunakan `ModifyCacheCluster` operasi ElastiCache API. Untuk mengubah nilai konfigurasi klaster, tentukan ID klaster, parameter yang akan diubah, dan nilai baru parameter. Contoh berikut mengubah periode pemeliharaan untuk klaster bernama `my-cluster` dan menerapkan perubahan tersebut secara langsung.

Important

Anda dapat meningkatkan ke versi mesin Memcached yang lebih baru. Untuk informasi selengkapnya tentang cara melakukannya, lihat Manajemen Versi untuk ElastiCache. Namun, Anda tidak dapat menurunkan ke versi mesin yang lebih lama kecuali dengan menghapus klaster yang ada dan membuatnya lagi.

Important

Anda dapat meningkatkan ke versi mesin Valkey atau Redis OSS yang lebih baru. Jika Anda meningkatkan versi mesin utama, misalnya dari Redis OSS 5.0.6 ke Redis OSS 6.0, Anda perlu memilih keluarga grup parameter yang kompatibel dengan versi mesin baru. Untuk informasi selengkapnya tentang cara melakukannya, lihat Manajemen Versi untuk

ElastiCache. Namun, Anda tidak dapat menurunkan ke versi mesin yang lebih lama kecuali dengan menghapus klaster yang ada dan membuatnya lagi.

Jeda baris ditambahkan agar dapat lebih mudah dibaca.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=ModifyCacheCluster
    &CacheClusterId=my-cluster
    &PreferredMaintenanceWindow=sun:23:00-mon:02:00
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20150901T220302Z
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Date=20150202T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20150901T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Parameter `ApplyImmediately` hanya berlaku untuk perubahan pada jenis simpul, versi mesin, dan jumlah simpul di klaster. Jika Anda ingin menerapkan salah satu perubahan ini secara langsung, tetapkan parameter `ApplyImmediately` ke `true`. Jika Anda lebih memilih untuk menunda perubahan ini ke periode pemeliharaan berikutnya, tetapkan parameter `ApplyImmediately` ke `false`. Perubahan lain, seperti perubahan periode pemeliharaan, akan diterapkan segera.

Untuk informasi selengkapnya, lihat topik referensi ElastiCache API `ModifyCacheCluster`.

Menambahkan node ke ElastiCache cluster

Penambahan simpul ke klaster Memcached akan meningkatkan jumlah partisi klaster Anda. Saat Anda mengubah jumlah partisi dalam klaster, beberapa ruang kunci Anda perlu dipetakan ulang agar dapat dipetakan ke simpul yang benar. Pemetaan ulang ruang kunci akan sementara waktu meningkatkan jumlah cache miss di klaster. Untuk informasi selengkapnya, lihat Mengkonfigurasi ElastiCache klien Anda untuk penyeimbangan beban yang efisien (Memcached).

Untuk mengkonfigurasi ulang klaster Valkey atau Redis OSS (mode cluster enabled) Anda, lihat Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan)

Anda dapat menggunakan ElastiCache Management Console, AWS CLI atau ElastiCache API untuk menambahkan node ke cluster Anda.

Menggunakan ElastiCache AWS Management Console

Jika Anda ingin menambahkan node ke cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) (satu tanpa replikasi diaktifkan), ini adalah proses dua langkah: pertama tambahkan replikasi, dan kemudian tambahkan simpul replika.

Topik

- Untuk menambahkan replikasi ke cluster Valkey atau Redis OSS tanpa pecahan
- Untuk menambahkan node ke ElastiCache cluster (konsol)

Prosedur berikut menambahkan replikasi ke satu node Valkey atau Redis OSS yang tidak mengaktifkan replikasi. Saat Anda menambahkan replikasi, simpul yang ada menjadi simpul primer dalam klaster yang mengaktifkan replikasi. Setelah replikasi ditambahkan, Anda dapat menambahkan hingga 5 simpul replika ke klaster.

Untuk menambahkan replikasi ke cluster Valkey atau Redis OSS tanpa pecahan

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cluster Valkey atau cluster Redis OSS.

   Daftar cluster yang menjalankan mesin itu ditampilkan.
3. Pilih nama klaster, bukan kotak di sebelah kiri nama klaster, yang ingin ditambahi simpul.

Berikut ini berlaku untuk cluster Redis OSS yang tidak mengaktifkan replikasi:

- Ini menjalankan Redis OSS, bukan Clustered Redis OSS.
- Klaster ini tidak memiliki serpihan.

  Jika klaster memiliki serpihan, artinya replikasi sudah diaktifkan dan Anda dapat melanjutkan ke Untuk menambahkan node ke ElastiCache cluster (konsol).

4. Pilih Tambahkan replikasi.
5. Di bagian Tambahkan replikasi, masukkan deskripsi untuk klaster yang mengaktifkan replikasi ini.
6. Pilih Tambahkan.

   Segera setelah status klaster kembali menjadi tersedia, Anda dapat melanjutkan ke prosedur berikutnya dan menambahkan replika ke klaster.

Untuk menambahkan node ke ElastiCache cluster (konsol)

Prosedur berikut dapat digunakan untuk menambahkan simpul ke klaster.

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih mesin yang berjalan pada klaster yang ingin ditambahi simpul.

   Daftar klaster yang menjalankan mesin yang dipilih akan muncul.
3. Dari daftar klaster tersebut, pilih nama klaster yang ingin ditambahi simpul.

   Jika cluster Anda adalah cluster Valkey atau Redis OSS (mode cluster enabled), lihat. Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan)

   Jika cluster Anda adalah cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan pecahan nol, pertama-tama selesaikan langkah-langkahnya di. Untuk menambahkan replikasi ke cluster Valkey atau Redis OSS tanpa pecahan
4. Pilih Tambahkan simpul.
5. Lengkapi informasi yang diminta dalam kotak dialog Tambahkan Simpul.
6. Pilih tombol Terapkan Segera - Ya untuk menambahkan simpul ini secara langsung, atau pilih Tidak untuk menambahkan simpul ini pada periode pemeliharaan berikutnya untuk klaster.

Dampak Permintaan Penambahan dan Penghapusan Baru pada Permintaan Tertunda

Skenario	Operasi Tertunda	Permintaan Baru	Hasil
Skenario 1	Hapus	Hapus	Permintaan penghapusan baru, tertunda atau langsung, akan menggantikan permintaan penghapusan yang tertunda. Misalnya, jika ada penghapusan tertunda untuk simpul 0001, 0003, dan 0007 dan permintaan baru untuk menghapus simpul 0002 dan 0004 dikirimkan, hanya simpul 0002 dan 0004 yang akan dihapus. Simpul 0001, 0003, dan 0007 tidak akan dihapus.
Skenario 2	Hapus	Buat	Permintaan pembuatan baru, tertunda atau langsung, akan menggantikan permintaan penghapusan tertunda. Misalnya, jika penghapusan simpul 0001, 0003, dan 0007 tertunda dan permintaan baru untuk membuat simpul dibuat, simpul baru akan dibuat dan simpul 0001, 0003, dan 0007 tidak akan dihapus.
Skenario 3	Buat	Hapus	Permintaan penghapusan baru, tertunda atau langsung, akan menggantikan permintaan pembuatan tertunda. Misalnya, jika ada permintaan tertunda untuk membuat dua simpul dan permintaan baru dibuat untuk menghapus simpul 0003, tidak ada simpul baru yang akan dibuat dan simpul 0003 akan dihapus.

Skenario	Operasi Tertunda	Permintaan Baru	Hasil
Skenario 4	Buat	Buat	Permintaan pembuatan baru ditambahkan ke permintaan pembuatan yang tertunda. Misalnya, jika ada permintaan tertunda untuk membuat dua simpul dan permintaan baru dibuat untuk membuat tiga simpul, permintaan baru ditambahkan ke permintaan tertunda dan lima simpul akan dibuat. **Important** Jika permintaan pembuatan baru diatur ke Terapkan Segera - Ya, semua permintaan pembuatan akan dilakukan segera. Jika permintaan pembuatan baru diatur ke Terapkan Segera - Tidak, semua permintaan pembuatan akan ditunda.

Untuk menentukan operasi apa yang tertunda, pilih tab Deskripsi dan periksa untuk melihat berapa banyak pembuatan atau penghapusan tertunda yang ditampilkan. Anda tidak dapat memiliki pembuatan tertunda dan penghapusan tertunda sekaligus.

7. Pilih tombol Tambahkan.

   Setelah beberapa saat, simpul baru akan muncul dalam daftar simpul dengan status membuat. Jika simpul tersebut tidak muncul, segarkan halaman browser Anda. Saat status simpul berubah menjadi tersedia, simpul baru sudah dapat digunakan.

Menggunakan AWS CLI dengan ElastiCache

Untuk menambahkan node ke cluster menggunakan AWS CLI, gunakan AWS CLI operasi `modify-cache-cluster` dengan parameter berikut:

- `--cache-cluster-id` – ID dari klaster cache yang ingin ditambahi simpul.

- `--num-cache-nodes` – Parameter `--num-cache-nodes` menentukan jumlah simpul yang diinginkan dalam klaster ini setelah perubahan diterapkan. Untuk menambahkan simpul ke klaster ini, `--num-cache-nodes` harus lebih besar dari jumlah simpul saat ini dalam klaster. Jika nilai ini kurang dari jumlah node saat ini, ElastiCache mengharapkan parameter `cache-node-ids-to-remove` dan daftar node untuk dihapus dari cluster. Untuk informasi selengkapnya, lihat Menggunakan AWS CLI dengan ElastiCache.
- `--apply-immediately` atau `--no-apply-immediately` yang menentukan apakah akan menambahkan simpul ini secara langsung atau pada periode pemeliharaan berikutnya.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-cache-cluster \
    --cache-cluster-id my-cluster \
    --num-cache-nodes 5 \
    --apply-immediately
```

Untuk Windows:

```
aws elasticache modify-cache-cluster ^
    --cache-cluster-id my-cluster ^
    --num-cache-nodes 5 ^
    --apply-immediately
```

Operasi ini menghasilkan output seperti yang berikut ini (format JSON):

```
{
    "CacheCluster": {
        "Engine": "memcached",
        "CacheParameterGroup": {
            "CacheNodeIdsToReboot": [],
            "CacheParameterGroupName": "default.memcached1.4",
            "ParameterApplyStatus": "in-sync"
        },
        "CacheClusterId": "my-cluster",
        "PreferredAvailabilityZone": "us-west-2b",
        "ConfigurationEndpoint": {
            "Port": 11211,
            "Address": "rlh-mem000.7alc7bf-example.cfg.usw2.cache.amazonaws.com"
        },
        "CacheSecurityGroups": [],
```

```
        "CacheClusterCreateTime": "2016-09-21T16:28:28.973Z",
        "AutoMinorVersionUpgrade": true,
        "CacheClusterStatus": "modifying",
        "NumCacheNodes": 2,
        "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
        "SecurityGroups": [
            {
                "Status": "active",
                "SecurityGroupId": "sg-dbe93fa2"
            }
        ],
        "CacheSubnetGroupName": "default",
        "EngineVersion": "1.4.24",
        "PendingModifiedValues": {
            "NumCacheNodes": 5
        },
        "PreferredMaintenanceWindow": "sat:09:00-sat:10:00",
        "CacheNodeType": "cache.m3.medium",
         "DataTiering": "disabled",
    }
}
```

Untuk informasi lebih lanjut, lihat AWS CLI topiknya `modify-cache-cluster`.

Menggunakan AWS CLI dengan ElastiCache

Jika Anda ingin menambahkan node ke cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) yang sudah ada yang tidak mengaktifkan replikasi, Anda harus terlebih dahulu membuat grup replikasi yang menentukan cluster yang ada sebagai primer. Untuk informasi selengkapnya, lihat Membuat grup replikasi menggunakan cluster cache Valkey atau Redis OSS yang tersedia ()AWS CLI. Setelah grup replikasi menjadi tersedia, Anda dapat melanjutkan proses berikut.

Untuk menambahkan node ke cluster menggunakan AWS CLI, gunakan AWS CLI operasi `increase-replica-count` dengan parameter berikut:

- `--replication-group-id` – ID grup replikasi yang ingin ditambahi simpul.
- `--new-replica-count` menentukan jumlah simpul yang diinginkan dalam grup replikasi ini setelah perubahan diterapkan. Untuk menambahkan simpul ke klaster ini, `--new-replica-count` harus lebih besar dari jumlah simpul saat ini dalam klaster.

- `--apply-immediately` atau `--no-apply-immediately` yang menentukan apakah akan menambahkan simpul ini secara langsung atau pada periode pemeliharaan berikutnya.

Untuk Linux, macOS, atau Unix:

```
aws elasticache increase-replica-count \
    --replication-group-id my-replication-group \
    --new-replica-count 4 \
    --apply-immediately
```

Untuk Windows:

```
aws elasticache increase-replica-count ^
    --replication-group-id my-replication-group ^
    --new-replica-count 4 ^
    --apply-immediately
```

Operasi ini menghasilkan output seperti yang berikut ini (format JSON):

```
{
    "ReplicationGroup": {
        "ReplicationGroupId": "node-test",
        "Description": "node-test",
        "Status": "modifying",
        "PendingModifiedValues": {},
        "MemberClusters": [
            "node-test-001",
            "node-test-002",
            "node-test-003",
            "node-test-004",
            "node-test-005"
        ],
        "NodeGroups": [
            {
                "NodeGroupId": "0001",
                "Status": "modifying",
                "PrimaryEndpoint": {
                    "Address": "node-test.zzzzzz.ng.0001.usw2.cache.amazonaws.com",
                    "Port": 6379
                },
                "ReaderEndpoint": {
                    "Address": "node-test.zzzzzz.ng.0001.usw2.cache.amazonaws.com",
```

```
                    "Port": 6379
                },
                "NodeGroupMembers": [
                    {
                        "CacheClusterId": "node-test-001",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "node-
test-001.zzzzzz.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2a",
                        "CurrentRole": "primary"
                    },
                    {
                        "CacheClusterId": "node-test-002",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "node-
test-002.zzzzzz.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2c",
                        "CurrentRole": "replica"
                    },
                    {
                        "CacheClusterId": "node-test-003",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "node-
test-003.zzzzzz.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2b",
                        "CurrentRole": "replica"
                    }
                ]
            }
        ],
        "SnapshottingClusterId": "node-test-002",
        "AutomaticFailover": "enabled",
        "MultiAZ": "enabled",
        "SnapshotRetentionLimit": 1,
        "SnapshotWindow": "07:30-08:30",
```

```
        "ClusterEnabled": false,
        "CacheNodeType": "cache.r5.large",
         "DataTiering": "disabled",
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false,
        "ARN": "arn:aws:elasticache:us-west-2:123456789012:replicationgroup:node-test"
    }
}
```

Untuk informasi lebih lanjut, lihat AWS CLI topiknya `increase-replica-count`.

Menggunakan ElastiCache API

Jika Anda ingin menambahkan node ke cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) yang sudah ada yang tidak mengaktifkan replikasi, Anda harus terlebih dahulu membuat grup replikasi yang menentukan cluster yang ada sebagai Primer. Untuk informasi selengkapnya, lihat Menambahkan replika ke cluster Valkey atau Redis OSS (Cluster Mode Disabled) mandiri (API) ElastiCache . Setelah grup replikasi menjadi tersedia, Anda dapat melanjutkan proses berikut.

Untuk menambahkan node ke cluster (ElastiCache API)

- Panggil operasi API `IncreaseReplicaCount` dengan parameter berikut ini:
  - `ReplicationGroupId` – ID dari klaster yang ingin ditambahi simpul.
  - `NewReplicaCount` – Parameter `NewReplicaCount` menentukan jumlah simpul yang diinginkan dalam klaster ini setelah perubahan diterapkan. Untuk menambahkan simpul ke klaster ini, `NewReplicaCount` harus lebih besar dari jumlah simpul saat ini dalam klaster. Jika nilai ini kurang dari jumlah simpul saat ini, gunakan API `DecreaseReplicaCount` dengan jumlah simpul yang akan dihapus dari klaster.
  - `ApplyImmediately` – Menentukan apakah akan menambahkan simpul ini secara langsung atau pada periode pemeliharaan berikutnya.
  - `Region`Menentukan AWS Wilayah cluster yang ingin Anda tambahkan node ke.

  Contoh berikut menunjukkan panggilan untuk menambahkan simpul ke klaster.

  Example

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=IncreaseReplicaCount
    &ApplyImmediately=true
```

```
    &NumCacheNodes=4
    &ReplicationGroupId=my-replication-group
    &Region=us-east-2
    &Version=2014-12-01
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Date=20141201T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20141201T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Untuk informasi selengkapnya, lihat topik ElastiCache API `IncreaseReplicaCount`.

Menggunakan ElastiCache API

Untuk menambahkan node ke cluster (ElastiCache API)

- Panggil operasi API `ModifyCacheCluster` dengan parameter berikut ini:
  - `CacheClusterId` – ID dari klaster yang ingin ditambahi simpul.
  - `NumCacheNodes` – Parameter `NumCachNodes` menentukan jumlah simpul yang diinginkan dalam klaster ini setelah perubahan diterapkan. Untuk menambahkan simpul ke klaster ini, `NumCacheNodes` harus lebih besar dari jumlah simpul saat ini dalam klaster. Jika nilai ini kurang dari jumlah node saat ini, ElastiCache mengharapkan parameter `CacheNodeIdsToRemove` dengan daftar node untuk dihapus dari cluster (lihatMenggunakan ElastiCache API dengan Memcached).
  - `ApplyImmediately` – Menentukan apakah akan menambahkan simpul ini secara langsung atau pada periode pemeliharaan berikutnya.
  - `Region`Menentukan AWS Wilayah cluster yang ingin Anda tambahkan node ke.

  Contoh berikut menunjukkan panggilan untuk menambahkan simpul ke klaster.

  Example

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=ModifyCacheCluster
```

```
   &ApplyImmediately=true
   &NumCacheNodes=5
&CacheClusterId=my-cluster
&Region=us-east-2
   &Version=2014-12-01
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20141201T220302Z
   &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
   &X-Amz-Date=20141201T220302Z
   &X-Amz-SignedHeaders=Host
   &X-Amz-Expires=20141201T220302Z
   &X-Amz-Credential=<credential>
   &X-Amz-Signature=<signature>
```

Untuk informasi selengkapnya, lihat topik ElastiCache API `ModifyCacheCluster`.

Menghapus node dari ElastiCache cluster

Anda dapat menghapus node dari cluster Valkey, Memcached, atau Redis OSS menggunakan AWS Management Console, the AWS CLI, atau API. ElastiCache

> **Note**
>
> Setiap kali Anda mengubah jumlah simpul dalam klaster Memcached, Anda harus memetakan ulang setidaknya beberapa ruang kunci Anda agar dipetakan ke simpul yang benar. Untuk informasi yang lebih mendetail tentang penyeimbangan beban klaster Memcached, lihat Mengkonfigurasi ElastiCache klien Anda untuk penyeimbangan beban yang efisien (Memcached).

Menggunakan ElastiCache AWS Management Console

Untuk menghapus simpul dari klaster (konsol)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari daftar di sudut kanan atas, pilih AWS Wilayah cluster yang ingin Anda hapus node.
3. Di panel navigasi, pilih mesin yang berjalan pada klaster yang ingin dihapus simpulnya itu.

   Daftar klaster yang menjalankan mesin yang dipilih akan muncul.
4. Dari daftar klaster, pilih nama klaster yang ingin dihapus simpulnya.

   Daftar simpul klaster akan muncul.
5. Pilih kotak di sebelah kiri ID simpul untuk simpul yang ingin dihapus. Menggunakan konsol ElastiCache , Anda hanya dapat menghapus satu simpul pada satu saat, sehingga memilih beberapa simpul berarti bahwa Anda tidak dapat menggunakan tombol Hapus Simpul.

   Halaman Hapus Simpul muncul.
6. Untuk menghapus simpul, lengkapi halaman Hapus Simpul dan pilih Hapus Simpul. Untuk mempertahankan simpul, pilih Batalkan.

> **Important**
>
> Dengan Valkey atau Redis OSS, jika Anda menghapus hasil node di cluster yang tidak lagi sesuai dengan Multi-AZ, pastikan untuk terlebih dahulu menghapus kotak centang Multi-AZ dan kemudian hapus node. Jika Anda menghapus kotak centang Multi-AZ, Anda dapat memilih untuk mengaktifkan Failover otomatis.

Dampak Permintaan Penambahan dan Penghapusan Baru pada Permintaan Tertunda

Skenario	Operasi Tertunda	Permintaan Baru	Hasil
Skenario 1	Hapus	Hapus	Permintaan penghapusan baru, tertunda atau langsung, akan menggantikan permintaan penghapusan yang tertunda. Misalnya, jika ada penghapusan tertunda untuk simpul 0001, 0003, dan 0007 dan permintaan baru untuk menghapus simpul 0002 dan 0004 dikirimkan, hanya simpul 0002 dan 0004 yang akan dihapus. Simpul 0001, 0003, dan 0007 tidak akan dihapus.
Skenario 2	Hapus	Buat	Permintaan pembuatan baru, tertunda atau langsung, akan menggantikan permintaan penghapusan tertunda. Misalnya, jika penghapusan simpul 0001, 0003, dan 0007 tertunda dan permintaan baru untuk membuat simpul dibuat, simpul baru akan dibuat dan simpul 0001, 0003, dan 0007 tidak akan dihapus.
Skenario 3	Buat	Hapus	Permintaan penghapusan baru, tertunda atau langsung, akan menggantikan permintaan pembuatan tertunda. Misalnya, jika ada permintaan tertunda untuk membuat dua simpul dan permintaan baru dibuat untuk

Skenario	Operasi Tertunda	Permintaan Baru	Hasil
			menghapus simpul 0003, tidak ada simpul baru yang akan dibuat dan simpul 0003 akan dihapus.
Skenario 4	Buat	Buat	Permintaan pembuatan baru ditambahkan ke permintaan pembuatan yang tertunda. Misalnya, jika ada permintaan tertunda untuk membuat dua simpul dan permintaan baru dibuat untuk membuat tiga simpul, permintaan baru ditambahkan ke permintaan tertunda dan lima simpul akan dibuat. **Important** Jika permintaan pembuatan baru diatur ke Terapkan Segera - Ya, semua permintaan pembuatan akan dilakukan segera. Jika permintaan pembuatan baru diatur ke Terapkan Segera - Tidak, semua permintaan pembuatan akan ditunda.

Untuk menentukan operasi apa yang tertunda, pilih tab Deskripsi dan periksa untuk melihat berapa banyak pembuatan atau penghapusan tertunda yang ditampilkan. Anda tidak dapat memiliki pembuatan tertunda dan penghapusan tertunda sekaligus.

Menggunakan AWS CLI dengan ElastiCache

1. IDs Identifikasi node yang ingin Anda hapus. Untuk informasi selengkapnya, lihat Melihat detail ElastiCache cluster.
2. Gunakan operasi CLI `decrease-replica-count` dengan daftar simpul yang akan dihapus, seperti dalam contoh berikut.

   Untuk menghapus simpul dari klaster menggunakan antarmuka baris perintah, gunakan perintah `decrease-replica-count` dengan parameter berikut:

   - `--replication-group-id` – ID grup replikasi yang ingin dihapus simpulnya.

- `--new-replica-count` – Parameter `--new-replica-count` menentukan jumlah simpul yang diinginkan dalam klaster ini setelah perubahan diterapkan.
- `--replicas-to-remove`Daftar node IDs yang ingin Anda hapus dari cluster ini.
- `--apply-immediately` atau `--no-apply-immediately` – Menentukan apakah akan menghapus simpul ini dengan segera atau pada periode pemeliharaan berikutnya.
- `--region`Menentukan AWS Wilayah cluster yang ingin Anda hapus node dari.

> **Note**
>
> Anda bisa meneruskan hanya satu parameter `--replicas-to-remove` atau `--new-replica-count` saat memanggil operasi ini.

Untuk Linux, macOS, atau Unix:

```
aws elasticache decrease-replica-count \
    --replication-group-id my-replication-group \
    --new-replica-count 2 \
    --region us-east-2 \
    --apply-immediately
```

Untuk Windows:

```
aws elasticache decrease-replica-count ^
    --replication-group-id my-replication-group ^
    --new-replica-count 3 ^
    --region us-east-2 ^
    --apply-immediately
```

Operasi ini menghasilkan output seperti yang berikut ini (format JSON):

```
{
    "ReplicationGroup": {
        "ReplicationGroupId": "node-test",
        "Description": "node-test"
       },
        "Status": "modifying",
        "PendingModifiedValues": {},
```

```
        "MemberClusters": [
            "node-test-001",
            "node-test-002",
            "node-test-003",
            "node-test-004",
            "node-test-005",
            "node-test-006"
        ],
        "NodeGroups": [
            {
                "NodeGroupId": "0001",
                "Status": "modifying",
                "PrimaryEndpoint": {
                    "Address": "node-test.zzzzzz.ng.0001.usw2.cache.amazonaws.com",
                    "Port": 6379
                },
                "ReaderEndpoint": {
                    "Address": "node-test-
ro.zzzzzz.ng.0001.usw2.cache.amazonaws.com",
                    "Port": 6379
                },
                "NodeGroupMembers": [
                    {
                        "CacheClusterId": "node-test-001",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "node-
test-001.zzzzzz.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2a",
                        "CurrentRole": "primary"
                    },
                    {
                        "CacheClusterId": "node-test-002",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "node-
test-002.zzzzzz.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2c",
                        "CurrentRole": "replica"
                    },
```

```
                    {
                        "CacheClusterId": "node-test-003",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "node-
test-003.zzzzzz.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2b",
                        "CurrentRole": "replica"
                    },
                    {
                        "CacheClusterId": "node-test-004",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "node-
test-004.zzzzzz.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2c",
                        "CurrentRole": "replica"
                    },
                    {
                        "CacheClusterId": "node-test-005",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "node-
test-005.zzzzzz.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2b",
                        "CurrentRole": "replica"
                    },
                    {
                        "CacheClusterId": "node-test-006",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Address": "node-
test-006.zzzzzz.0001.usw2.cache.amazonaws.com",
                            "Port": 6379
                        },
                        "PreferredAvailabilityZone": "us-west-2b",
                        "CurrentRole": "replica"
                    }
```

```
                ]
            }
        ],
        "SnapshottingClusterId": "node-test-002",
        "AutomaticFailover": "enabled",
        "MultiAZ": "enabled",
        "SnapshotRetentionLimit": 1,
        "SnapshotWindow": "07:30-08:30",
        "ClusterEnabled": false,
        "CacheNodeType": "cache.r5.large",
         "DataTiering": "disabled",
        "TransitEncryptionEnabled": false,
        "AtRestEncryptionEnabled": false,
        "ARN": "arn:aws:elasticache:us-west-2:123456789012:replicationgroup:node-
test"
    }
}
```

Cara lainnya, Anda dapat memanggil `decrease-replica-count` dan daripada meneruskan parameter `--new-replica-count`, Anda bisa meneruskan parameter `--replicas-to-remove`, seperti yang ditunjukkan berikut ini:

Untuk Linux, macOS, atau Unix:

```
aws elasticache decrease-replica-count \
    --replication-group-id my-replication-group \
    --replicas-to-remove node-test-003 \
    --region us-east-2 \
    --apply-immediately
```

Untuk Windows:

```
aws elasticache decrease-replica-count ^
    --replication-group-id my-replication-group ^
    --replicas-to-remove node-test-003 ^
    --region us-east-2 ^
    --apply-immediately
```

Untuk informasi lebih lanjut, lihat AWS CLI topiknya `decrease-replica-count`.

Menggunakan ElastiCache API dengan Valkey atau Redis OSS

Untuk menghapus node menggunakan ElastiCache API, panggil operasi `DecreaseReplicaCount` API dengan Id grup replikasi dan daftar node yang akan dihapus, seperti yang ditunjukkan:

- `ReplicationGroupId` – ID grup replikasi yang ingin dihapus simpulnya.
- `ReplicasToRemove` – Parameter `ReplicasToRemove` menentukan jumlah simpul yang diinginkan dalam klaster ini setelah perubahan diterapkan.
- `ApplyImmediately` – Menentukan apakah akan menghapus simpul ini dengan segera atau pada periode pemeliharaan berikutnya.
- `Region`Menentukan AWS Wilayah cluster yang Anda ingin menghapus node dari.

Contoh berikut segera menghapus simpul 0004 dan 0005 dari klaster my-cluster.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=DecreaseReplicaCount
    &ReplicationGroupId=my-replication-group
    &ApplyImmediately=true
    &ReplicasToRemove=node-test-003
    &Region us-east-2
    &Version=2014-12-01
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Date=20141201T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20141201T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Untuk informasi selengkapnya, lihat topik ElastiCache API `DecreaseReplicaCount`.

Menggunakan ElastiCache API dengan Memcached

Untuk menghapus node menggunakan ElastiCache API, panggil operasi `ModifyCacheCluster` API dengan ID cluster cache dan daftar node yang akan dihapus, seperti yang ditunjukkan:

- `CacheClusterId` – ID dari klaster cache yang ingin dihapus simpulnya.

- `NumCacheNodes` – Parameter `NumCacheNodes` menentukan jumlah simpul yang diinginkan dalam klaster ini setelah perubahan diterapkan.
- `CacheNodeIdsToRemove.member.n`Daftar node IDs untuk menghapus dari cluster.
  - `CacheNodeIdsToRemove.member.1=0004`
  - `CacheNodeIdsToRemove.member.1=0005`
- `ApplyImmediately` – Menentukan apakah akan menghapus simpul ini dengan segera atau pada periode pemeliharaan berikutnya.
- `Region`Menentukan AWS Wilayah cluster yang Anda ingin menghapus node dari.

Contoh berikut segera menghapus simpul 0004 dan 0005 dari klaster my-cluster.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=ModifyCacheCluster
    &CacheClusterId=my-cluster
    &ApplyImmediately=true
    &CacheNodeIdsToRemove.member.1=0004
    &CacheNodeIdsToRemove.member.2=0005
    &NumCacheNodes=3
    &Region us-east-2
    &Version=2014-12-01
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Date=20141201T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20141201T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Untuk informasi selengkapnya, lihat topik ElastiCache API `ModifyCacheCluster`.

Membatalkan operasi tambah atau hapus node yang tertunda di ElastiCache

Jika Anda memilih untuk tidak segera menerapkan perubahan ElastiCache klaster, operasi memiliki status tertunda hingga dilakukan di jendela pemeliharaan berikutnya. Anda dapat membatalkan operasi tertunda.

Untuk membatalkan operasi tambah atau hapus node yang tertunda dengan AWS CLI, gunakan `modify-cache-cluster` perintah. Setel `num-cache-nodes` sama dengan jumlah node cache saat ini di cluster, dan kemudian tambahkan `--apply-immediately` bendera. Ini akan mengesampingkan perubahan yang tertunda.

Untuk membatalkan penambahan atau penghapusan node yang tertunda:

```
aws elasticache modify-cache-cluster
 --cache-cluster-id <your-cluster-id>
 --num-cache-nodes <current-number-of-nodes>
 --apply-immediately
```

Jika tidak jelas apakah ada penambahan atau penghapusan node yang tertunda, Anda dapat mengonfirmasi statusnya dengan perintah: `describe-cache-clusters`

```
aws elasticache describe-cache-clusters
 --cache-cluster-id <your-cluster-id>
```

Setiap node yang tertunda akan muncul di `PendingModifiedValues` output. Misalnya:

```
"PendingModifiedValues": {
 "NumCacheNodes": 3
 },
```

Menghapus cluster di ElastiCache

Selama ElastiCache cluster dalam keadaan tersedia, Anda dikenakan biaya untuk itu, apakah Anda aktif menggunakannya atau tidak. Untuk menghentikan biaya, hapus klaster tersebut.

> **Warning**
>
> Saat Anda menghapus ElastiCache klaster, snapshot manual Anda dipertahankan. Anda juga dapat membuat snapshot akhir sebelum klaster dihapus. Snapshot cache otomatis tidak dipertahankan.

Menggunakan AWS Management Console

Prosedur berikut menghapus satu klaster dari deployment Anda. Untuk menghapus beberapa klaster, ulangi prosedur untuk setiap klaster yang ingin dihapus. Anda tidak perlu menunggu satu klaster selesai dihapus sebelum memulai prosedur untuk menghapus klaster lain.

Untuk menghapus klaster

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di dasbor ElastiCache mesin, pilih mesin yang berjalan di cluster yang ingin Anda hapus.

   Daftar semua klaster yang menjalankan mesin tersebut akan muncul.
3. Untuk memilih klaster yang akan dihapus, pilih nama klaster tersebut dari daftar klaster.

   > **Important**
   >
   > Anda hanya dapat menghapus satu cluster pada satu waktu dari ElastiCache konsol. Memilih beberapa klaster akan menonaktifkan operasi penghapusan.
4. Untuk Tindakan, pilih Hapus.
5. Pada layar konfirmasi Hapus Klaster, pilih Hapus untuk menghapus klaster, atau Batal untuk mempertahankan klaster.

   Jika Anda memilih Hapus, status klaster berubah menjadi menghapus.

Segera setelah klaster Anda tidak lagi tercantum dalam daftar klaster, Anda berhenti dikenai biaya untuk klaster tersebut.

Menggunakan AWS CLI untuk menghapus ElastiCache cluster

Kode berikut menghapus cluster ElastiCache `my-cluster` cache.

```
aws elasticache delete-cache-cluster --cache-cluster-id my-cluster
```

Tindakan CLI `delete-cache-cluster` hanya menghapus satu klaster cache. Untuk menghapus beberapa klaster cache, panggil `delete-cache-cluster` untuk setiap klaster cache yang ingin dihapus. Anda tidak perlu menunggu satu klaster cache selesai dihapus untuk dapat menghapus yang lain.

Untuk Linux, macOS, atau Unix:

```
aws elasticache delete-cache-cluster \
    --cache-cluster-id my-cluster \
    --region us-east-2
```

Untuk Windows:

```
aws elasticache delete-cache-cluster ^
    --cache-cluster-id my-cluster ^
    --region us-east-2
```

Untuk informasi selengkapnya, lihat ElastiCache topik AWS CLI untuk `delete-cache-cluster`.

Menggunakan ElastiCache API

Kode berikut menghapus klaster `my-cluster`.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=DeleteCacheCluster
    &CacheClusterId=my-cluster
    &Region us-east-2
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20150202T220302Z
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Date=20150202T220302Z
```

```
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20150202T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Operasi API `DeleteCacheCluster` hanya menghapus satu klaster cache. Untuk menghapus beberapa klaster cache, panggil `DeleteCacheCluster` untuk setiap klaster cache yang ingin dihapus. Anda tidak perlu menunggu satu klaster cache selesai dihapus untuk dapat menghapus yang lain.

Untuk informasi selengkapnya, lihat topik referensi ElastiCache API `DeleteCacheCluster`.

Mengakses ElastiCache klaster atau grup replikasi

ElastiCache Instans Amazon Anda dirancang untuk diakses melalui EC2 instans Amazon.

Jika meluncurkan ElastiCache instans di Amazon Virtual Private Cloud (Amazon VPC), Anda dapat mengakses ElastiCache instans dari instans Amazon EC2 di VPC Amazon yang sama. Atau, dengan menggunakan VPC peering, Anda dapat mengakses ElastiCache instans Anda dari Amazon EC2 di VPC Amazon yang berbeda.

Jika meluncurkan ElastiCache instans di EC2 Classic, Anda mengizinkan EC2 instans mengakses klaster Anda dengan memberikan grup EC2 keamanan Amazon yang terkait dengan akses instans ke grup keamanan cache Anda. Secara default, akses ke klaster dibatasi pada akun yang meluncurkan klaster tersebut.

Topik

Memberikan akses ke klaster atau grup replikasi Anda

Anda meluncurkan cluster Anda ke EC2 -VPC

Jika meluncurkan klaster ke Amazon Virtual Private Cloud (Amazon VPC), Anda dapat terhubung ke ElastiCache klaster hanya dari EC2 instans Amazon yang berjalan di VPC Amazon yang sama. Dalam hal ini, Anda akan perlu memberikan izin masuk jaringan ke klaster.

> **Note**
>
> Pastikan Anda telah mengaktifkan Zona Lokal jika Anda menggunakannya. Untuk informasi selengkapnya, lihat Mengaktifkan Zona Lokal. Dengan mengaktifkannya, VPC Anda diperluas ke Zona Lokal dan VPC Anda akan memperlakukan subnet seperti subnet apa pun di Zona Ketersediaan yang lain. Gateway, tabel rute dan pertimbangan grup keamanan lainnya yang berkaitan akan menyesuaikan secara otomatis.

Untuk memberikan izin masuk jaringan dari grup keamanan Amazon VPC ke klaster

1. Masuk ke AWS Management Console dan buka EC2 konsol Amazon di https://console.aws.amazon.com/ec2/.

2. Pada panel navigasi, di bagian Jaringan & Keamanan, pilih Grup Keamanan.
3. Dari daftar grup keamanan, pilih grup keamanan untuk Amazon VPC Anda. Kecuali Anda membuat grup keamanan untuk ElastiCache digunakan, grup keamanan ini akan diberi nama default.
4. Pilih tab Masuk, lalu lakukan hal berikut:

    a. Pilih Edit.

    b. Pilih Tambahkan aturan.

    c. Di kolom Jenis, pilih Aturan TCP kustom.

    d. Di kotak Rentang port, ketik nomor port untuk simpul klaster Anda. Nomor ini harus sama dengan yang Anda tentukan saat meluncurkan klaster. Port default untuk Memcached adalah **11211** Port default untuk Valkey dan Redis OSS adalah. **6379**

    e. Di kotak Sumber, pilih Anywhere yang memiliki rentang port (0.0.0.0/0) sehingga EC2 instans Amazon apa pun yang Anda luncurkan dalam VPC Amazon Anda dapat terhubung ke node Anda. ElastiCache

    > **Important**
    >
    > Membuka ElastiCache cluster ke 0.0.0.0/0 tidak mengekspos cluster ke Internet karena tidak memiliki alamat IP publik dan oleh karena itu tidak dapat diakses dari luar VPC. Namun, grup keamanan default dapat diterapkan ke EC2 instans Amazon lainnya di akun pelanggan, dan instans tersebut mungkin memiliki alamat IP publik. Jika instans tersebut menjalankan sesuatu di port default, layanan tersebut dapat terekspos secara tak disengaja. Oleh karena itu, sebaiknya membuat Grup Keamanan VPC yang akan digunakan secara eksklusif oleh ElastiCache. Untuk informasi selengkapnya, lihat Grup Keamanan Kustom.

    f. Pilih Simpan.

Saat Anda meluncurkan EC2 instans Amazon ke VPC Amazon Anda, instance itu akan dapat terhubung ke cluster Anda ElastiCache .

Mengakses ElastiCache sumber daya dari luar AWS

Amazon ElastiCache adalah AWS layanan yang menyediakan penyimpanan nilai kunci dalam memori berbasis cloud. Layanan ini dirancang untuk diakses secara eksklusif dari dalam AWS. Namun, jika ElastiCache cluster di-host di dalam VPC, Anda dapat menggunakan instance Network Address Translation (NAT) untuk menyediakan akses luar.

Persyaratan

Persyaratan berikut harus dipenuhi agar Anda dapat mengakses ElastiCache sumber daya Anda dari luar AWS:

- Klaster harus berada dalam VPC dan diakses melalui instans Network Address Translation (NAT). Tidak ada pengecualian untuk persyaratan ini.
- Instans NAT harus diluncurkan di VPC yang sama dengan klaster.
- Instans NAT harus diluncurkan di subnet publik yang terpisah dari klaster.
- Alamat IP Elastis (EIP) harus dikaitkan dengan instans NAT. Fitur penerusan port iptables digunakan untuk meneruskan port di instans NAT ke port simpul cache dalam VPC.

Pertimbangan

Pertimbangan berikut harus diingat saat mengakses ElastiCache sumber daya Anda dari luar. ElastiCache

- Klien terhubung ke port EIP dan cache dari instans NAT. Penerusan port di instans NAT meneruskan lalu lintas ke simpul klaster cache yang sesuai.
- Jika simpul klaster ditambahkan atau diganti, aturan iptables perlu diperbarui untuk mencerminkan perubahan ini.

Batasan

Pendekatan ini harus digunakan untuk tujuan pengujian dan pengembangan saja. Sebaiknya jangan digunakan untuk produksi karena batasan berikut:

- Instans NAT bertindak sebagai perantara antara klien dan beberapa klaster. Penambahan proksi berdampak pada performa klaster cache. Dampaknya meningkat dengan jumlah klaster cache yang Anda akses melalui instans NAT.

- Lalu lintas dari klien ke instans NAT tidak terenkripsi. Oleh karena itu, Anda harus menghindari pengiriman data sensitif melalui instans NAT.
- Instans NAT menambahkan overhead untuk memelihara instans lain.
- Instans NAT berfungsi sebagai satu titik kegagalan. Untuk informasi tentang cara mengatur NAT ketersediaan tinggi di VPC, lihat Ketersediaan Tinggi untuk Instans NAT Amazon VPC: Contoh.

Cara mengakses ElastiCache sumber daya dari luar AWS

Prosedur berikut menunjukkan cara menghubungkan ke ElastiCache sumber daya Anda menggunakan instance NAT.

Langkah-langkah ini mengasumsikan hal berikut:

- `iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 6380 -j DNAT --to 10.0.1.231:6379`
- `iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 6381 -j DNAT --to 10.0.1.232:6379`

Selanjutnya Anda membutuhkan NAT ke arah yang berlawanan:

```
iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to-source 10.0.0.55
```

Anda juga perlu mengaktifkan penerusan IP, yang dinonaktifkan secara default:

```
sudo sed -i 's/net.ipv4.ip_forward=0/net.ipv4.ip_forward=1/g' /etc/
sysctl.conf sudo sysctl --system
```

- Anda mengakses klaster Memcached dengan:
  - Alamat IP – 10.0.1.230
  - Port Memcached default – 11211
  - Grup keamanan – *10\.0\.0\.55*
- Anda mengakses cluster Valkey atau Redis OSS dengan:
  - Alamat IP – 10.0.1.230
  - Port standar - 6379
  - Grup keamanan – sg-bd56b7da
  - AWS alamat IP instance — 198.99.100.27

- Klien tepercaya Anda memiliki alamat IP 198.51.100.27.
- Instans NAT Anda memiliki Alamat IP Elastis 203.0.113.73.
- Instans NAT Anda memiliki grup keamanan sg-ce56b7a9.

Untuk terhubung ke ElastiCache sumber daya Anda menggunakan instans NAT

1. Buat instans NAT di VPC yang sama dengan klaster cache Anda tetapi di subnet publik.

   Secara default, wizard VPC akan meluncurkan jenis simpul cache.m1.small. Anda harus memilih ukuran simpul berdasarkan kebutuhan Anda. Anda harus menggunakan EC2 NAT AMI untuk dapat mengakses ElastiCache dari luar AWS.

   Untuk informasi tentang membuat instance NAT, lihat Instans NAT di Panduan Pengguna VPC AWS .
2. Buat aturan grup keamanan untuk klaster cache dan instans NAT.

   Grup keamanan instans NAT dan instans klaster harus memiliki aturan berikut:

   - Dua aturan masuk
     - Dengan Memcached, aturan pertama adalah mengizinkan koneksi TCP dari klien tepercaya ke setiap port cache yang diteruskan dari instance NAT (11211 - 11213).
     - Dengan Valkey dan Redis OSS, aturan pertama adalah mengizinkan koneksi TCP dari klien tepercaya ke setiap port cache yang diteruskan dari instance NAT (6379 - 6381).
     - Aturan kedua untuk memungkinkan akses SSH ke klien tepercaya.

   Grup keamanan instance NAT - aturan masuk dengan Memcached

Jenis	Protokol	Rentang port	Sumber
Aturan TCP Kustom	TCP	11211-11213	198.51.100.27/32
SSH	TCP	22	198.51.100.27/32

Grup keamanan instans NAT - aturan masuk dengan Valkey atau Redis OSS

Jenis	Protokol	Rentang port	Sumber
Aturan TCP Kustom	TCP	6379-6380	198.51.100.27/32
SSH	TCP	22	203.0.113.73/32

- Dengan Memcached, aturan keluar untuk memungkinkan koneksi TCP ke port cache (11211).

  Grup keamanan instans NAT - aturan keluar

Jenis	Protokol	Rentang Port	Tujuan
Aturan TCP Kustom	TCP	11211	sg-ce56b7a9 (Grup Keamanan NAT)

- Dengan Valkey atau Redis OSS, aturan keluar untuk memungkinkan koneksi TCP ke port cache (6379).

  Grup keamanan instans NAT - aturan keluar

Jenis	Protokol	Rentang Port	Tujuan
Aturan TCP Kustom	TCP	6379	sg-ce56b7a9 (Grup Keamanan NAT)

- Dengan Memcached, aturan masuk untuk grup keamanan cluster yang memungkinkan koneksi TCP dari instance NAT ke port cache (11211).

  Grup keamanan instans klaster - aturan masuk

Jenis	Protokol	Rentang port	Sumber
Aturan TCP Kustom	TCP	11211	sg-ce56b7a9 (Grup Keamanan NAT)

- Dengan Valkey atau Redis OSS, aturan masuk untuk grup keamanan cluster yang memungkinkan koneksi TCP dari instance NAT ke port cache (6379).

Grup keamanan instans klaster - aturan masuk

Jenis	Protokol	Rentang port	Sumber
Aturan TCP Kustom	TCP	6379	sg-ce56b7a9 (Grup Keamanan NAT)

3. Validasi aturan.

   - Konfirmasikan bahwa klien tepercaya dapat melakukan SSH ke instans NAT.
   - Konfirmasikan bahwa klien tepercaya dapat terhubung ke klaster dari instans NAT.

4. Memcache

   Tambahkan aturan iptables ke instans NAT.

   Aturan iptables harus ditambahkan ke tabel NAT untuk setiap simpul di klaster untuk meneruskan port cache dari instans NAT ke simpul klaster. Contohnya mungkin terlihat seperti berikut:

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 11211 -j DNAT --to
 10.0.1.230:11211
```

   Nomor port harus unik untuk setiap simpul di klaster. Misalnya, jika ada klaster Memcached tiga simpul menggunakan port 11211 - 11213, aturan akan terlihat seperti berikut:

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 11211 -j DNAT --to
 10.0.1.230:11211
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 11212 -j DNAT --to
 10.0.1.231:11211
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 11213 -j DNAT --to
 10.0.1.232:11211
```

   Konfirmasikan bahwa klien tepercaya dapat terhubung ke klaster.

   Klien tepercaya harus terhubung ke EIP yang terkait dengan instans NAT dan port klaster yang sesuai dengan simpul klaster yang sesuai. Misalnya, string koneksi untuk PHP mungkin terlihat seperti berikut:

```
$memcached->connect( '203.0.113.73', 11211 );
```

```
$memcached->connect( '203.0.113.73', 11212 );
$memcached->connect( '203.0.113.73', 11213 );
```

Klien telnet juga dapat digunakan untuk memverifikasi koneksi. Misalnya:

```
telnet 203.0.113.73 11211
telnet 203.0.113.73 11212
telnet 203.0.113.73 11213
```

Valkey atau Redis OSS

Tambahkan aturan iptables ke instans NAT.

Aturan iptables harus ditambahkan ke tabel NAT untuk setiap simpul di klaster untuk meneruskan port cache dari instans NAT ke simpul klaster. Contohnya mungkin terlihat seperti berikut:

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 6379 -j DNAT --to
 10.0.1.230:6379
```

Nomor port harus unik untuk setiap simpul di klaster. Misalnya, jika bekerja dengan cluster Redis OSS tiga node menggunakan port 6379 - 6381, aturannya akan terlihat seperti berikut:

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 6379 -j DNAT --to
 10.0.1.230:6379
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 6380 -j DNAT --to
 10.0.1.231:6379
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 6381 -j DNAT --to
 10.0.1.232:6379
```

Konfirmasikan bahwa klien tepercaya dapat terhubung ke klaster.

Klien tepercaya harus terhubung ke EIP yang terkait dengan instans NAT dan port klaster yang sesuai dengan simpul klaster yang sesuai. Misalnya, string koneksi untuk PHP mungkin terlihat seperti berikut:

```
redis->connect( '203.0.113.73', 6379 );
redis->connect( '203.0.113.73', 6380 );
redis->connect( '203.0.113.73', 6381 );
```

Klien telnet juga dapat digunakan untuk memverifikasi koneksi. Misalnya:

```
telnet 203.0.113.73 6379
telnet 203.0.113.73 6380
telnet 203.0.113.73 6381
```

5. Simpan konfigurasi iptables.

   Simpan aturan setelah Anda menguji dan memverifikasinya. Jika Anda menggunakan distribusi Linux berbasis Redhat (seperti Amazon Linux), jalankan perintah berikut:

   ```
   service iptables save
   ```

Topik terkait

Topik-topik berikut mungkin menarik bagi Anda.

- Pola Akses untuk Mengakses ElastiCache Cache di VPC Amazon
- Mengakses ElastiCache Cache dari Aplikasi yang Berjalan di Pusat Data Pelanggan
- Instans NAT
- Mengkonfigurasi Klien ElastiCache
- Ketersediaan Tinggi untuk Instans NAT Amazon VPC: Contoh

Menemukan titik akhir koneksi di ElastiCache

Aplikasi Anda terhubung ke ElastiCache klaster Anda menggunakan endpoint. Titik akhir adalah alamat unik dari simpul atau klaster.

Anda juga dapat membuat koneksi pribadi antara VPC dan titik akhir ElastiCache API Anda dengan membuat titik akhir VPC antarmuka melalui. AWS PrivateLink Untuk informasi selengkapnya, lihat ElastiCache API dan antarmuka VPC endpoint ()AWS PrivateLink.

Titik akhir mana yang digunakan dengan Valkey atau Redis OSS.

- Untuk node mandiri, gunakan titik akhir node untuk operasi baca dan tulis.

- Untuk cluster Valkey atau Valkey atau Redis OSS (mode cluster dinonaktifkan), gunakan Endpoint Utama untuk semua operasi penulisan. Gunakan Titik Akhir Pembaca untuk membagi koneksi masuk ke titik akhir secara merata di antara semua replika baca. Gunakan Titik Akhir Simpul individual untuk operasi baca (Dalam API/CLI, ini disebut sebagai Titik Akhir Baca).

- Untuk cluster Valkey atau Redis OSS (mode cluster enabled), gunakan Configuration Endpoint cluster untuk semua operasi yang mendukung perintah yang diaktifkan mode cluster. Anda harus menggunakan klien yang mendukung Valkey Cluster, atau Redis OSS Cluster pada Redis OSS 3.2 dan di atasnya. Anda masih dapat membaca dari titik akhir node individual (Dalam hal API/CLI ini disebut sebagai Read Endpoints).

Bagian berikut memandu Anda menemukan titik akhir yang Anda perlukan untuk mesin yang sedang Anda jalankan.

Titik akhir mana yang akan digunakan dengan Memcached.

Untuk cache ElastiCache tanpa server untuk Memcached, cukup dapatkan DNS dan port titik akhir cluster dari konsol.

Dari AWS CLI, gunakan `describe-serverless-caches` perintah untuk memperoleh informasi Endpoint.

Linux

```
aws elasticache describe-serverless-caches --serverless-cache-name CacheName
```

Windows

```
aws elasticache describe-serverless-caches --serverless-cache-name CacheName
```

Output dari operasi di atas terlihat seperti berikut ini (format JSON):

```
{
    "ServerlessCaches": [
        {
            "ServerlessCacheName": "serverless-memcached",
```

```
            "Description": "test",
            "CreateTime": 1697659642.136,
            "Status": "available",
            "Engine": "memcached",
            "MajorEngineVersion": "1.6",
            "FullEngineVersion": "21",
            "SecurityGroupIds": [
                "sg-083eda453e1e51310"
            ],
            "Endpoint": {
                "Address": "serverless-memcached-01.amazonaws.com",
                "Port":11211
            },
            "ARN": "<the ARN>",
            "SubnetIds": [
                "subnet-0cf759df15bd4dc65",
                "subnet-09e1307e8f1560d17"
            ],
            "SnapshotRetentionLimit": 0,
            "DailySnapshotTime": "03:00"
        }
    ]
}
```

Untuk cluster Memcached berbasis instance, jika Anda menggunakan Automatic Discovery maka Anda dapat menggunakan titik akhir konfigurasi cluster untuk mengonfigurasi klien Memcached Anda. Hal ini berarti Anda harus menggunakan klien yang mendukung Penemuan Otomatis.

Jika tidak menggunakan Penemuan Otomatis, Anda harus mengonfigurasi klien Anda untuk menggunakan titik akhir simpul individual untuk operasi baca dan tulis. Anda juga harus melacak titik akhir saat menambahkan dan menghapus simpul.

Menemukan Titik Akhir Cluster Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol)

Jika cluster Valkey atau Valkey atau Redis OSS (mode cluster dinonaktifkan) hanya memiliki satu node, titik akhir node digunakan untuk membaca dan menulis. Jika klaster Valkey atau Valkey atau Redis OSS (mode cluster dinonaktifkan) memiliki beberapa node, ada tiga jenis titik akhir; titik akhir primer, titik akhir pembaca dan titik akhir node.

Titik akhir primer adalah nama DNS yang selalu diresolusi ke simpul primer di klaster. Titik akhir primer tidak terpengaruh oleh perubahan klaster Anda, seperti promosi replika baca ke peran primer. Untuk aktivitas tulis, sebaiknya aplikasi Anda terhubung ke titik akhir primer.

Titik akhir pembaca akan membagi koneksi masuk secara merata ke titik akhir antara semua replika baca di cluster ElastiCache for Redis OSS. Faktor lain seperti saat aplikasi membuat koneksi atau cara aplikasi menggunakan atau menggunakan ulang koneksi akan menentukan distribusi lalu lintas. Titik akhir pembaca tetap mengikuti perubahan klaster dalam waktu nyata saat replika ditambahkan atau dihapus. Anda dapat menempatkan beberapa replika baca klaster Redis OSS Anda ElastiCache di AWS Availability Zones (AZ) yang berbeda untuk memastikan ketersediaan titik akhir pembaca yang tinggi.

> **Note**
>
> Titik akhir pembaca bukan penyeimbang beban. Ini adalah catatan DNS yang akan diresolusi sebagai alamat IP dari salah satu simpul replika dengan metode round robin.

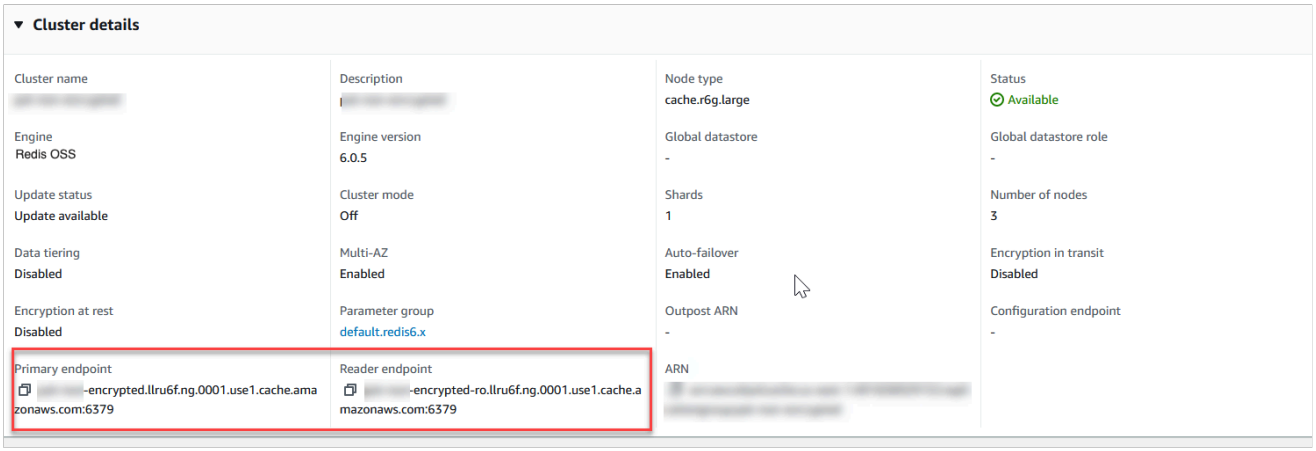
Untuk aktivitas baca, aplikasi juga dapat menghubungkan ke simpul mana pun di klaster. Tidak seperti titik akhir primer, titik akhir simpul diresolusi ke titik akhir tertentu. Jika Anda membuat perubahan dalam klaster Anda, seperti menambahkan atau menghapus replika, Anda harus memperbarui titik akhir simpul di aplikasi Anda.

Untuk menemukan titik akhir cluster Valkey atau Valkey atau Redis OSS (mode cluster dinonaktifkan)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cluster Valkey atau cluster Redis OSS.

   Layar cluster akan muncul dengan daftar Valkey atau Valkey atau Redis OSS (mode cluster dinonaktifkan) dan Valkey atau Redis OSS (mode cluster diaktifkan) cluster.

3. Untuk menemukan titik akhir and/or Pembaca Utama klaster, pilih nama cluster (bukan tombol di sebelah kirinya).

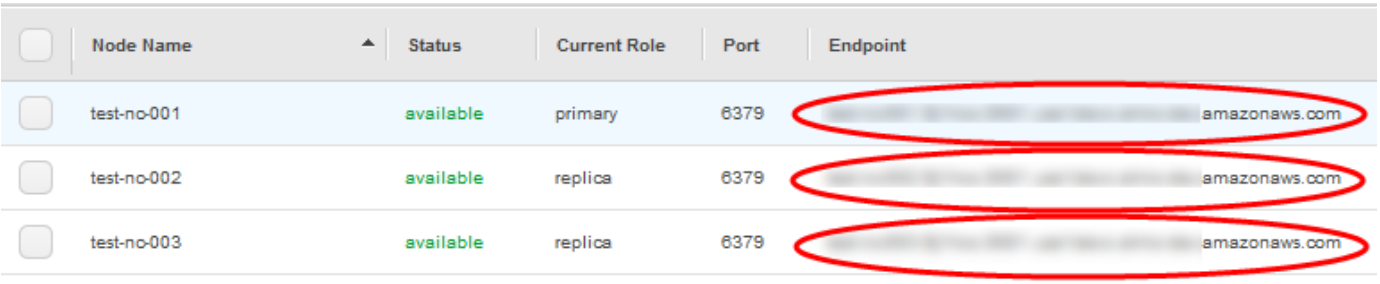


Titik akhir Primer dan Pembaca untuk cluster Valkey atau Valkey atau Redis OSS (mode cluster dinonaktifkan)

Jika hanya ada satu simpul dalam klaster, berarti tidak ada titik akhir primer dan Anda dapat melanjutkan ke langkah berikutnya.

4. Jika cluster Valkey atau Valkey atau Redis OSS (mode cluster dinonaktifkan) memiliki node replika, Anda dapat menemukan titik akhir node replika cluster dengan memilih nama cluster dan kemudian memilih tab Nodes.

Layar simpul muncul dengan setiap simpul yang ada di klaster, primer dan replika, yang tercantum dengan titik akhirnya.



Titik akhir node untuk cluster Valkey atau Valkey atau Redis OSS (mode cluster dinonaktifkan)

5. Untuk menyalin titik akhir ke clipboard Anda:

a. Temukan satu per satu titik akhir yang ingin Anda salin.

b. Pilih ikon salin langsung di depan titik akhir.

Titik akhir sekarang disalin ke clipboard Anda. Untuk informasi tentang menggunakan titik akhir agar terhubung ke simpul, lihat Menghubungkan ke node Memcached.

Titik akhir utama Valkey atau Valkey atau Redis OSS (mode cluster dinonaktifkan) terlihat seperti berikut ini. Ada perbedaan yang tergantung pada apakah enkripsi Bergerak aktif atau tidak.

Enkripsi bergerak tidak diaktifkan

```
clusterName.xxxxxx.nodeId.regionAndAz.cache.amazonaws.com:port

redis-01.7abc2d.0001.usw2.cache.amazonaws.com:6379
```

Enkripsi bergerak diaktifkan

```
master.clusterName.xxxxxx.regionAndAz.cache.amazonaws.com:port

master.ncit.ameaqx.use1.cache.amazonaws.com:6379
```

Menemukan Titik Akhir untuk Cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan) (Konsol)

Cluster Valkey atau Redis OSS (mode cluster enabled) memiliki titik akhir konfigurasi tunggal. Dengan terhubung ke titik akhir konfigurasi, aplikasi Anda mampu menemukan titik akhir primer dan baca untuk setiap serpihan di klaster.

Untuk menemukan titik akhir cluster Valkey atau Redis OSS (mode cluster enabled)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cluster Valkey atau cluster Redis OSS.

   Layar cluster akan muncul dengan daftar cluster. Pilih cluster yang ingin Anda sambungkan.
3. Untuk menemukan titik akhir Konfigurasi klaster, pilih nama klaster (bukan tombol radio).
4. Titik akhir konfigurasi ditampilkan di bagian Detail klaster. Untuk menyalinnya, pilih ikon salin di sebelah kiri titik akhir.

Menemukan Titik Akhir Cluster (Konsol) (Memcached)

Semua titik akhir Memcached adalah titik akhir. read/write Untuk terhubung ke simpul dalam klaster Memcached, aplikasi Anda dapat menggunakan titik akhir untuk setiap simpul, atau titik akhir konfigurasi dari klaster bersama dengan Penemuan Otomatis. Untuk menggunakan Penemuan Otomatis, Anda harus menggunakan klien yang mendukung Penemuan Otomatis.

Saat menggunakan Penemuan Otomatis, aplikasi klien Anda terhubung ke klaster Memcached Anda menggunakan titik akhir konfigurasi. Begitu Anda menskalakan klaster Anda dengan menambahkan atau menghapus simpul, aplikasi Anda akan secara otomatis "mengetahui" semua simpul dalam klaster dan dapat terhubung ke semua simpul tersebut. Tanpa Penemuan Otomatis, aplikasi Anda harus melakukannya, atau Anda harus secara manual memperbarui titik akhir dalam aplikasi Anda setiap kali Anda menambahkan atau menghapus simpul.

Untuk menyalin titik akhir, pilih ikon salin secara langsung di depan alamat titik akhir. Untuk informasi tentang menggunakan titik akhir agar terhubung ke simpul, lihat Menghubungkan ke node Memcached.

Titik akhir konfigurasi dan simpul terlihat sangat mirip. Perbedaannya disorot dengan cetak tebal seperti berikut.

```
myclustername.xxxxxx.cfg.usw2.cache.amazonaws.com:port   # configuration endpoint
 contains "cfg"
myclustername.xxxxxx.0001.usw2.cache.amazonaws.com:port  # node endpoint for node 0001
```

> **Important**
>
> Jika Anda memilih untuk membuat CNAME untuk titik akhir konfigurasi Memcached, agar klien penemuan otomatis Anda mengenali CNAME sebagai titik akhir konfigurasi, Anda harus menyertakan `.cfg.` pada CNAME.

Menemukan Titik Akhir (AWS CLI)

Untuk Memcached, Anda dapat menggunakan for AWS CLI Amazon ElastiCache untuk menemukan titik akhir untuk node dan cluster.

Untuk Redis OSS, Anda dapat menggunakan for AWS CLI Amazon ElastiCache untuk menemukan titik akhir untuk node, cluster, dan juga grup replikasi.

Topik

- Menemukan Titik Akhir untuk Simpul dan Klaster (AWS CLI)
- Menemukan Titik Akhir untuk Grup Replikasi Valkey atau Redis OSS ()AWS CLI

Menemukan Titik Akhir untuk Simpul dan Klaster (AWS CLI)

Anda dapat menggunakan AWS CLI untuk menemukan titik akhir untuk cluster dan node dengan `describe-cache-clusters` perintah. Untuk cluster Valkey atau Redis OSS, perintah mengembalikan titik akhir cluster. Untuk klaster Memcached, perintah tersebut akan menampilkan titik akhir konfigurasi. Jika Anda menyertakan parameter opsional `--show-cache-node-info`, perintah tersebut juga akan menampilkan titik akhir simpul individual di klaster.

Example

Perintah berikut mengambil titik akhir konfigurasi (`ConfigurationEndpoint`) dan titik akhir simpul individual (`Endpoint`) untuk klaster Memcached mycluster.

Untuk Linux, macOS, atau Unix:

```
aws elasticache describe-cache-clusters \
    --cache-cluster-id mycluster \
    --show-cache-node-info
```

Untuk Windows:

```
aws elasticache describe-cache-clusters ^
    --cache-cluster-id mycluster ^
    --show-cache-node-info
```

Output dari operasi di atas akan terlihat seperti berikut (format JSON).

```
{
```

```
    "CacheClusters": [
    {
        "Engine": "memcached",
        "CacheNodes": [
           {
              "CacheNodeId": "0001",
              "Endpoint": {
                 "Port": 11211,
                 "Address": "mycluster.amazonaws.com"
              },
                 "CacheNodeStatus": "available",
                 "ParameterGroupStatus": "in-sync",
                 "CacheNodeCreateTime": "2016-09-22T21:30:29.967Z",
                 "CustomerAvailabilityZone": "us-west-2b"
           },
           {
              "CacheNodeId": "0002",
              "Endpoint": {
                 "Port": 11211,
                 "Address": "mycluster.amazonaws.com"
              },
                 "CacheNodeStatus": "available",
                 "ParameterGroupStatus": "in-sync",
                 "CacheNodeCreateTime": "2016-09-22T21:30:29.967Z",
                 "CustomerAvailabilityZone": "us-west-2b"
           },
           {
                 "CacheNodeId": "0003",
                 "Endpoint": {
                    "Port": 11211,
                    "Address": "mycluster.amazonaws.com"
                 },
                    "CacheNodeStatus": "available",
                    "ParameterGroupStatus": "in-sync",
                    "CacheNodeCreateTime": "2016-09-22T21:30:29.967Z",
                    "CustomerAvailabilityZone": "us-west-2b"
           }
        ],
        "CacheParameterGroup": {
        "CacheNodeIdsToReboot": [],
        "CacheParameterGroupName": "default.memcached1.4",
        "ParameterApplyStatus": "in-sync"
             },
             "CacheClusterId": "mycluster",
```

```
            "PreferredAvailabilityZone": "us-west-2b",
            "ConfigurationEndpoint": {
                "Port": 11211,
                "Address": "mycluster.amazonaws.com"
            },
            "CacheSecurityGroups": [],
            "CacheClusterCreateTime": "2016-09-22T21:30:29.967Z",
            "AutoMinorVersionUpgrade": true,
            "CacheClusterStatus": "available",
            "NumCacheNodes": 3,
            "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
            "CacheSubnetGroupName": "default",
            "EngineVersion": "1.4.24",
            "PendingModifiedValues": {},
            "PreferredMaintenanceWindow": "mon:09:00-mon:10:00",
            "CacheNodeType": "cache.m4.large",
             "DataTiering": "disabled"
        }
    ]
}
```

Important

Jika Anda memilih untuk membuat CNAME untuk titik akhir konfigurasi Memcached agar klien penemuan otomatis Anda mengenali CNAME sebagai titik akhir konfigurasi, Anda harus menyertakan `.cfg.` pada CNAME. Misalnya, `mycluster.cfg.local` dalam file php.ini untuk parameter `session.save_path`.

Example

Untuk Valkey dan Redis OSS, perintah berikut mengambil informasi cluster untuk mycluster single-node cluster.

Important

Parameter `--cache-cluster-id` dapat digunakan dengan id cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) node tunggal atau id node tertentu dalam grup replikasi. Kelompok replikasi adalah nilai 4 digit seperti. `--cache-cluster-id 0001` Jika `--cache-`

`cluster-id` adalah id dari cluster (node) dalam grup replikasi, termasuk dalam output. `replication-group-id`

Untuk Linux, macOS, atau Unix:

```
aws elasticache describe-cache-clusters \
    --cache-cluster-id redis-cluster \
    --show-cache-node-info
```

Untuk Windows:

```
aws elasticache describe-cache-clusters ^
    --cache-cluster-id redis-cluster ^
    --show-cache-node-info
```

Output dari operasi di atas akan terlihat seperti berikut (format JSON).

```
{
    "CacheClusters": [
        {
            "CacheClusterStatus": "available",
            "SecurityGroups": [
                {
                    "SecurityGroupId": "sg-77186e0d",
                    "Status": "active"
                }
            ],
            "CacheNodes": [
                {
                    "CustomerAvailabilityZone": "us-east-1b",
                    "CacheNodeCreateTime": "2018-04-25T18:19:28.241Z",
                    "CacheNodeStatus": "available",
                    "CacheNodeId": "0001",
                    "Endpoint": {
                        "Address": "redis-cluster.amazonaws.com",
                        "Port": 6379
                    },
                    "ParameterGroupStatus": "in-sync"
                }
            ],
            "AtRestEncryptionEnabled": false,
```

```
            "CacheClusterId": "redis-cluster",
            "TransitEncryptionEnabled": false,
            "CacheParameterGroup": {
                "ParameterApplyStatus": "in-sync",
                "CacheNodeIdsToReboot": [],
                "CacheParameterGroupName": "default.redis3.2"
            },
            "NumCacheNodes": 1,
            "PreferredAvailabilityZone": "us-east-1b",
            "AutoMinorVersionUpgrade": true,
            "Engine": "redis",
            "AuthTokenEnabled": false,
            "PendingModifiedValues": {},
            "PreferredMaintenanceWindow": "tue:08:30-tue:09:30",
            "CacheSecurityGroups": [],
            "CacheSubnetGroupName": "default",
            "CacheNodeType": "cache.t2.small",
             "DataTiering": "disabled"
            "EngineVersion": "3.2.10",
            "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
            "CacheClusterCreateTime": "2018-04-25T18:19:28.241Z"
        }
    ]
}
```

Untuk informasi lebih lanjut, lihat topiknya describe-cache-clusters.

Menemukan Titik Akhir untuk Grup Replikasi Valkey atau Redis OSS ()AWS CLI

Anda dapat menggunakan AWS CLI untuk menemukan titik akhir untuk grup replikasi dan klaster dengan perintah. `describe-replication-groups` Perintah ini menampilkan titik akhir primer grup replikasi dan daftar semua klaster (simpul) dalam grup replikasi dengan titik akhirnya, bersama dengan titik akhir pembaca.

Operasi berikut mengambil titik akhir primer dan titik akhir pembaca untuk grup replikasi `myreplgroup`. Gunakan titik akhir primer untuk semua operasi tulis.

```
aws elasticache describe-replication-groups \
    --replication-group-id myreplgroup
```

Untuk Windows:

```
aws elasticache describe-replication-groups ^
    --replication-group-id myreplgroup
```

Output dari operasi ini terlihat seperti berikut (format JSON).

```
{
   "ReplicationGroups": [
     {
       "Status": "available",
       "Description": "test",
       "NodeGroups": [
         {
            "Status": "available",
               "NodeGroupMembers": [
                  {
                     "CurrentRole": "primary",
                     "PreferredAvailabilityZone": "us-west-2a",
                     "CacheNodeId": "0001",
                     "ReadEndpoint": {
                        "Port": 6379,
                        "Address": "myreplgroup-001.amazonaws.com"
                     },
                     "CacheClusterId": "myreplgroup-001"
                  },
                  {
                     "CurrentRole": "replica",
                     "PreferredAvailabilityZone": "us-west-2b",
                     "CacheNodeId": "0001",
                     "ReadEndpoint": {
                        "Port": 6379,
                        "Address": "myreplgroup-002.amazonaws.com"
                     },
                     "CacheClusterId": "myreplgroup-002"
                  },
                  {
                     "CurrentRole": "replica",
                     "PreferredAvailabilityZone": "us-west-2c",
                     "CacheNodeId": "0001",
                     "ReadEndpoint": {
                        "Port": 6379,
                        "Address": "myreplgroup-003.amazonaws.com"
                     },
                     "CacheClusterId": "myreplgroup-003"
```

```
                }
            ],
            "NodeGroupId": "0001",
            "PrimaryEndpoint": {
               "Port": 6379,
               "Address": "myreplgroup.amazonaws.com"
            },
            "ReaderEndpoint": {
               "Port": 6379,
               "Address": "myreplgroup-ro.amazonaws.com"
            }
         }
      ],
      "ReplicationGroupId": "myreplgroup",
      "AutomaticFailover": "enabled",
      "SnapshottingClusterId": "myreplgroup-002",
      "MemberClusters": [
         "myreplgroup-001",
         "myreplgroup-002",
         "myreplgroup-003"
      ],
      "PendingModifiedValues": {}
   }
  ]
}
```

Untuk informasi selengkapnya, lihat describe-replication-groups dalam AWS CLI Referensi Perintah.

Menemukan Titik Akhir (ElastiCache API)

Untuk Memcached, Anda dapat menggunakan Amazon ElastiCache API untuk menemukan titik akhir untuk node dan cluster.

Untuk Redis OSS, Anda dapat menggunakan Amazon ElastiCache API untuk menemukan titik akhir untuk node, cluster, dan juga grup replikasi.

Topik

- Menemukan Endpoint untuk Node dan Cluster (API) ElastiCache
- Menemukan Titik Akhir untuk Grup Replikasi Valkey atau Redis OSS (API) ElastiCache

Menemukan Endpoint untuk Node dan Cluster (API) ElastiCache

Anda dapat menggunakan ElastiCache API untuk menemukan titik akhir untuk klaster dan node-nya dengan `DescribeCacheClusters` tindakan tersebut. Untuk cluster Valkey atau Redis OSS, perintah mengembalikan titik akhir cluster. Untuk klaster Memcached, perintah tersebut akan menampilkan titik akhir konfigurasi. Jika Anda menyertakan parameter opsional `ShowCacheNodeInfo`, tindakan tersebut juga akan menampilkan titik akhir simpul individual di klaster.

Example

Untuk Memcached, perintah berikut mengambil konfigurasi endpoint (*`ConfigurationEndpoint`*) dan titik akhir node individual (*`Endpoint`*) untuk mycluster Memcached cluster.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=DescribeCacheClusters
    &CacheClusterId=mycluster
    &ShowCacheNodeInfo=true
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20150202T192317Z
    &Version=2015-02-02
    &X-Amz-Credential=<credential>
```

> **Important**
>
> Jika Anda memilih untuk membuat CNAME untuk titik akhir konfigurasi Memcached agar klien penemuan otomatis Anda mengenali CNAME sebagai titik akhir konfigurasi, Anda harus

menyertakan `.cfg.` pada CNAME. Misalnya, `mycluster.cfg.local` dalam file php.ini untuk parameter `session.save_path`.

Menemukan Titik Akhir untuk Grup Replikasi Valkey atau Redis OSS (API) ElastiCache

Anda dapat menggunakan ElastiCache API untuk menemukan titik akhir untuk grup replikasi dan klasternya dengan tindakan tersebut. `DescribeReplicationGroups` Perintah ini menampilkan titik akhir primer grup replikasi dan daftar semua klaster dalam grup replikasi dengan titik akhirnya, bersama dengan titik akhir pembaca.

Operasi berikut mengambil endpoint utama (PrimaryEndpoint), endpoint pembaca () dan titik akhir node individu (ReaderEndpointReadEndpoint) untuk grup replikasi. `myreplgroup` Gunakan titik akhir primer untuk semua operasi tulis.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=DescribeReplicationGroups
    &ReplicationGroupId=myreplgroup
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20150202T192317Z
    &Version=2015-02-02
    &X-Amz-Credential=<credential>
```

Lihat informasi yang lebih lengkap di DescribeReplicationGroups.

Bekerja dengan pecahan di ElastiCache

Sebuah shard (API/CLI: node group) adalah kumpulan dari satu sampai enam ElastiCache untuk Valkey atau Redis OSS node. Cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) tidak akan pernah memiliki lebih dari satu pecahan. Dengan pecahan, Anda dapat memisahkan database besar menjadi bagian yang lebih kecil, lebih cepat, dan lebih mudah dikelola yang disebut pecahan data. Hal ini dapat meningkatkan efisiensi database dengan mendistribusikan operasi di beberapa bagian terpisah. Menggunakan pecahan dapat menawarkan banyak manfaat termasuk peningkatan kinerja, skalabilitas, dan efisiensi biaya.

Anda dapat membuat klaster dengan jumlah serpihan lebih banyak dan jumlah replika lebih sedikit dengan jumlah total hingga 90 simpul per klaster. Konfigurasi klaster ini dapat berkisar dari 90 serpihan dan 0 replika hingga 15 serpihan dan 5 replika, yang merupakan jumlah replika maksimum

yang diizinkan. Data klaster dipartisi di seluruh serpihan klaster. Jika ada lebih dari satu node dalam pecahan, shard mengimplementasikan replikasi dengan satu node menjadi node read/write utama dan node lainnya read-only replika node.

Batas node atau shard dapat ditingkatkan hingga maksimum 500 per cluster jika versi mesin Valkey 7.2 dan lebih tinggi, atau Redis OSS 5.0.6 hingga 7.1. Sebagai contoh, Anda dapat memilih untuk mengonfigurasi sebuah klaster dengan 500 simpul yang berkisar antara 83 serpihan (satu primer dan 5 replika per serpihan) dan 500 serpihan (satu primer dan tanpa replika). Pastikan alamat IP yang tersedia mencukupi untuk mengakomodasi peningkatan tersebut. Kesalahan umumnya termasuk subnet dalam grup subnet memiliki rentang CIDR yang terlalu kecil atau subnet dibagikan dan banyak digunakan oleh klaster lainnya. Untuk informasi selengkapnya, lihat Membuat grup subnet.

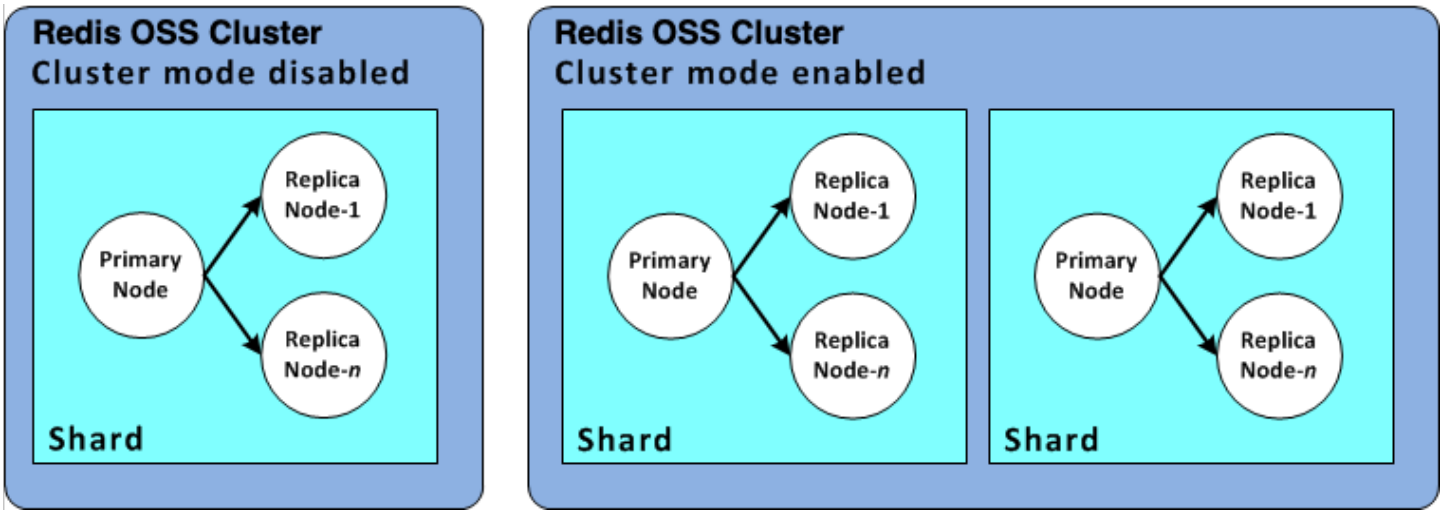
Untuk versi di bawah 5.0.6, batasnya adalah 250 per klaster.

Untuk meminta penambahan batas, lihat Batas layanan AWS dan pilih jenis batas Simpul per klaster per jenis instans.

Saat Anda membuat cluster Valkey atau Redis OSS (mode cluster enabled) menggunakan ElastiCache konsol, Anda menentukan jumlah pecahan di cluster dan jumlah node dalam pecahan. Untuk informasi selengkapnya, lihat Membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) (Konsol). Jika Anda menggunakan ElastiCache API atau AWS CLI membuat klaster (disebut grup replikasi di grup simpulAPI/CLI), you can configure the number of nodes in a shard (API/CLI:) secara independen. Untuk informasi selengkapnya, lihat berikut ini:

- API: CreateReplicationGroup
- CLI: create-replication-group

Setiap simpul dalam serpihan memiliki spesifikasi komputasi, penyimpanan, dan memori yang sama. ElastiCache API memungkinkan Anda mengontrol atribut shard-wide, seperti jumlah node, pengaturan keamanan, dan jendela pemeliharaan sistem.



Konfigurasi pecahan Valkey atau Redis OSS

Untuk informasi selengkapnya, lihat Resharding offline untuk Valkey atau Redis OSS (mode cluster diaktifkan) dan Resharding online untuk Valkey atau Redis OSS (mode cluster diaktifkan).

Menemukan ID serpihan

Anda dapat menemukan ID pecahan menggunakan AWS Management Console, the AWS CLI atau ElastiCache API.

Menggunakan AWS Management Console

Topik

- Untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)
- Untuk Valkey atau Redis OSS (Mode Cluster Diaktifkan)

Untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)

Pecahan grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) selalu ada. IDs `0001`

Untuk Valkey atau Redis OSS (Mode Cluster Diaktifkan)

Prosedur berikut menggunakan AWS Management Console untuk menemukan ID pecahan grup replikasi Valkey atau Redis OSS (mode cluster enabled).

Untuk menemukan ID pecahan dalam grup replikasi Valkey atau Redis OSS (mode cluster diaktifkan)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pada panel navigasi, pilih Valkey atau Redis OSS, lalu pilih nama grup replikasi Valkey atau Redis OSS (mode cluster enabled) yang ingin Anda temukan. IDs
3. Pada kolom Nama Serpihan, ID serpihan adalah empat digit terakhir dari nama serpihan.

Menggunakan AWS CLI

Untuk menemukan id pecahan (grup simpul) untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) atau Valkey atau Redis OSS (mode cluster diaktifkan) gunakan operasi dengan parameter opsional berikut. AWS CLI `describe-replication-groups`

- **`--replication-group-id`**—Parameter opsional yang jika digunakan akan membatasi output pada detail grup replikasi yang ditentukan. Jika parameter ini dihilangkan, detail hingga 100 grup replikasi akan ditampilkan.

Example

Perintah ini akan menampilkan detail untuk `sample-repl-group`.

Untuk Linux, macOS, atau Unix:

```
aws elasticache describe-replication-groups \
    --replication-group-id sample-repl-group
```

Untuk Windows:

```
aws elasticache describe-replication-groups ^
    --replication-group-id sample-repl-group
```

Output dari perintah ini akan terlihat seperti ini. Id shard (grup simpul) ada *highlighted* di sini untuk mempermudah menemukannya.

```
{
    "ReplicationGroups": [
        {
            "Status": "available",
```

```
            "Description": "2 shards, 2 nodes (1 + 1 replica)",
            "NodeGroups": [
                {
                    "Status": "available",
                    "Slots": "0-8191",
                    "NodeGroupId": "0001",
                    "NodeGroupMembers": [
                        {
                            "PreferredAvailabilityZone": "us-west-2c",
                            "CacheNodeId": "0001",
                            "CacheClusterId": "sample-repl-group-0001-001"
                        },
                        {
                            "PreferredAvailabilityZone": "us-west-2a",
                            "CacheNodeId": "0001",
                            "CacheClusterId": "sample-repl-group-0001-002"
                        }
                    ]
                },
                {
                    "Status": "available",
                    "Slots": "8192-16383",
                    "NodeGroupId": "0002",
                    "NodeGroupMembers": [
                        {
                            "PreferredAvailabilityZone": "us-west-2b",
                            "CacheNodeId": "0001",
                            "CacheClusterId": "sample-repl-group-0002-001"
                        },
                        {
                            "PreferredAvailabilityZone": "us-west-2a",
                            "CacheNodeId": "0001",
                            "CacheClusterId": "sample-repl-group-0002-002"
                        }
                    ]
                }
            ],
            "ConfigurationEndpoint": {
                "Port": 6379,
                "Address": "sample-repl-
group.9dcv5r.clustercfg.usw2.cache.amazonaws.com"
            },
            "ClusterEnabled": true,
            "ReplicationGroupId": "sample-repl-group",
```

```
            "SnapshotRetentionLimit": 1,
            "AutomaticFailover": "enabled",
            "SnapshotWindow": "13:00-14:00",
            "MemberClusters": [
                "sample-repl-group-0001-001",
                "sample-repl-group-0001-002",
                "sample-repl-group-0002-001",
                "sample-repl-group-0002-002"
            ],
            "CacheNodeType": "cache.m3.medium",
            "DataTiering": "disabled",
            "PendingModifiedValues": {}
        }
    ]
}
```

Menggunakan ElastiCache API

Untuk menemukan id pecahan (grup simpul) untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) atau Valkey atau Redis OSS (mode cluster diaktifkan) gunakan operasi dengan parameter opsional berikut. AWS CLI `describe-replication-groups`

- **`ReplicationGroupId`**—Parameter opsional yang jika digunakan akan membatasi output pada detail grup replikasi yang ditentukan. Jika parameter ini dihilangkan, detail hingga grup *`xxx`* replikasi dikembalikan.

Example

Perintah ini akan menampilkan detail untuk `sample-repl-group`.

Untuk Linux, macOS, atau Unix:

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DescribeReplicationGroup
   &ReplicationGroupId=sample-repl-group
   &Version=2015-02-02
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &X-Amz-Credential=<credential>
```

Membandingkan cache yang dirancang sendiri Valkey, Memcached, dan Redis OSS

Amazon ElastiCache mendukung mesin cache Valkey, Memcached, dan Redis OSS. Setiap mesin menyediakan beberapa kelebihan. Gunakan informasi dalam topik ini untuk membantu Anda memilih mesin dan versi yang paling sesuai dengan kebutuhan Anda.

> **Important**
>
> Setelah Anda membuat cache, cluster yang dirancang sendiri, atau grup replikasi, Anda dapat meningkatkan ke versi mesin yang lebih baru, tetapi Anda tidak dapat menurunkan versi ke versi mesin yang lebih lama. Jika Anda ingin menggunakan versi mesin yang lebih lama, Anda harus menghapus cache yang ada, cluster yang dirancang sendiri atau grup replikasi dan membuatnya lagi dengan versi mesin sebelumnya.

Secara umum, mesin yang ada terlihat serupa. Masing-masing mesin adalah penyimpanan nilai-kunci dalam memori. Namun, dalam praktiknya terdapat perbedaan yang signifikan.

Pilih Memcached jika hal berikut berlaku untuk Anda:

- Anda membutuhkan model yang paling sederhana.
- Anda perlu menjalankan simpul besar dengan beberapa inti atau thread.
- Anda membutuhkan kemampuan untuk menskalakan ke luar dan ke dalam, yakni menambahkan dan menghapus simpul seiring peningkatan dan penurunan permintaan pada sistem.
- Anda perlu membuat cache untuk obyek.

Pilih Valkey atau Redis OSS dengan ElastiCache jika berikut ini berlaku untuk Anda:

- ElastiCache versi 7.2 untuk Valkey atau versi 7.0 (Ditingkatkan) untuk Redis OSS

  Anda ingin menggunakan Functions, Sharded Pub/Sub, atau perbaikan ACL. Untuk informasi selengkapnya, lihat Redis OSS Versi 7.0 (Ditingkatkan).
- ElastiCache versi 6.2 (Ditingkatkan) untuk Redis OSS

  Anda menginginkan kemampuan untuk mengatur tingkatan data antara memori dan SSD menggunakan jenis simpul r6gd. Untuk informasi selengkapnya, lihat Tingkatan data.

- ElastiCache versi 6.0 (Ditingkatkan) untuk Redis OSS

  Anda ingin mengautentikasi pengguna dengan kontrol akses berbasis peran.

  Untuk informasi selengkapnya, lihat Redis OSS Versi 6.0 (Ditingkatkan).

- ElastiCache versi 5.0.0 (Ditingkatkan) untuk Redis OSS

  Anda ingin menggunakan aliran Redis OSS, struktur data log yang memungkinkan produsen menambahkan item baru secara real time dan juga memungkinkan konsumen untuk mengkonsumsi pesan baik dengan cara pemblokiran atau non-pemblokiran.

  Untuk informasi selengkapnya, lihat Redis OSS Versi 5.0.0 (Ditingkatkan).

- ElastiCache versi 4.0.10 (Ditingkatkan) untuk Redis OSS

  Mendukung enkripsi dan menambahkan atau menghapus pecahan secara dinamis dari cluster Valkey atau Redis OSS (mode cluster enabled) Anda.

  Untuk informasi selengkapnya, lihat Redis OSS Versi 4.0.10 (Ditingkatkan).

Versi berikut tidak digunakan lagi, telah mencapai atau segera mencapai akhir masa pakainya.

- ElastiCache versi 3.2.10 (Ditingkatkan) untuk Redis OSS

  Mendukung kemampuan untuk menambahkan atau menghapus pecahan secara dinamis dari cluster Valkey atau Redis OSS (mode cluster diaktifkan) Anda.

  > **Important**
  >
  > Saat ini ElastiCache 3.2.10 untuk Redis OSS tidak mendukung enkripsi.

  Untuk informasi selengkapnya, lihat berikut ini:

  - Redis OSS Versi 3.2.10 (Ditingkatkan)
  - Praktik terbaik resharding online untuk Redis OSS, Untuk informasi lebih lanjut, lihat berikut ini:
    - Praktik Terbaik: Resharding Online
    - Resharding Online dan Rebalancing Shard untuk Valkey atau Redis OSS (Mode Cluster Diaktifkan)
    - Untuk informasi selengkapnya tentang penskalaan klaster Redis OSS, lihat Penskalaan

- ElastiCache versi 3.2.6 (Ditingkatkan) untuk Redis OSS

  Jika Anda memerlukan fungsionalitas versi Redis OSS sebelumnya ditambah fitur-fitur berikut, pilih 3.2.6:

  - Enkripsi bergerak. Untuk informasi selengkapnya, lihat Amazon ElastiCache for Redis OSS In-Transit Encryption.
  - Enkripsi diam. Untuk informasi selengkapnya, lihat Amazon ElastiCache for Redis OSS At-Rest Encryption.
- ElastiCache (Mode cluster diaktifkan) versi 3.2.4 untuk Redis OSS

  Jika Anda memerlukan fungsionalitas 2.8.x ditambah fitur-fitur berikut, pilih 3.2.4 (mode berkerumun):

  - Anda perlu membuat partisi data Anda di dua hingga 500 grup simpul (mode berklaster saja).
  - Anda membutuhkan pengindeksan geospasial (mode berklaster atau mode tanpa berklaster).
  - Anda tidak perlu mendukung beberapa basis data.
- ElastiCache (mode tidak berkerumun) 2.8.x dan 3.2.4 (Ditingkatkan) untuk Redis OSS

  Jika hal berikut berlaku untuk Anda, 2.8.x atau 3.2.4 (mode non-clustered):

  - Anda memerlukan jenis data yang kompleks, seperti string, hash, list, set, sorted set, dan bitmap.
  - Anda perlu mengurutkan atau membuat peringkat set data dalam memori.
  - Anda perlu persistensi pada penyimpanan kunci Anda.
  - Anda perlu mereplikasi data Anda dari primer ke satu atau beberapa replika baca untuk aplikasi sarat operasi baca.
  - Anda perlu melakukan failover otomatis jika simpul primer Anda gagal.
  - Anda memerlukan kemampuan memublikasikan dan berlangganan (pub/sub)—untuk memberi tahu klien tentang peristiwa di server.
  - Anda memerlukan kemampuan cadangan dan pemulihan untuk cluster yang dirancang sendiri serta cache tanpa server.
  - Anda perlu mendukung beberapa basis data.

Ringkasan perbandingan Memcached, Valkey atau Redis OSS (mode cluster dinonaktifkan), dan Valkey atau Redis OSS (mode cluster diaktifkan)

	Memcached	Valkey atau Redis OSS (mode cluster dinonaktifkan)	Valkey atau Redis OSS (mode cluster diaktifkan)
Versi mesin+	1.4.5 dan kemudian	4.0.10 dan yang lebih baru	4.0.10 dan yang lebih baru
Jenis Data	Sederhana	2.8.x - Kompleks * Kompleks	3.2.x dan setelahnya - Kompleks
Pembuatan partisi data	Ya	Tidak	Ya
Klaster dapat dimodifikasi	Ya	Ya	3.2.10 dan setelahnya - Terbatas
Resharding online	Tidak	Tidak	3.2.10 dan setelahnya
Enkripsi	in-transit 1.6.12 dan yang lebih baru	4.0.10 dan yang lebih baru	4.0.10 dan yang lebih baru
Tingkatan data	Tidak	6.2 dan kemudian	6.2 dan kemudian
Sertifikasi kepatuhan			
Sertifikasi Kepatuhan			
FedRAMP	Ya - 1.6.12 dan yang lebih baru	4.0.10 dan yang lebih baru	4.0.10 dan yang lebih baru
HIPAA	Ya - 1.6.12 dan yang lebih baru	4.0.10 dan yang lebih baru	4.0.10 dan yang lebih baru
PCI DSS	Ya	4.0.10 dan yang lebih baru	4.0.10 dan yang lebih baru
Multi-threaded	Ya	Tidak	Tidak

	Memcached	Valkey atau Redis OSS (mode cluster dinonaktifkan)	Valkey atau Redis OSS (mode cluster diaktifkan)
Peningkatan jenis simpul	Tidak	Ya	Ya
Peningkatan mesin	Ya	Ya	Ya
Ketersediaan tinggi (replikasi)	Tidak	Ya	Ya
Failover otomatis	Tidak	Opsional	Wajib
Kemampuan Pub/Sub	Tidak	Ya	Ya
Set yang diurutkan	Tidak	Ya	Ya
Pencadangan dan pemulihan	Hanya untuk Memcached Tanpa Server, bukan untuk cluster Memcached yang dirancang sendiri	Ya	Ya
Pengindeksan geospasial	Tidak	4.0.10 dan yang lebih baru	Ya

Catatan:

string, obyek (seperti basis data)

* string, set, sorted set, list, hash, bitmap, hyperloglog

string, set, sorted set, list, hash, bitmap, hyperloglog, indeks geospatial

+ Tidak termasuk versi yang tidak digunakan lagi, telah mencapai atau segera mencapai akhir masa pakai.

Setelah Anda memilih mesin untuk klaster Anda, sebaiknya gunakan versi terbaru mesin tersebut. Lihat informasi yang lebih lengkap di Jenis simpul yang didukung.

Migrasi online untuk Valkey atau Redis OSS

Dengan menggunakan Migrasi Online, Anda dapat memigrasikan data dari Valkey atau Redis OSS open-source yang dihosting sendiri di Amazon ke Amazon. EC2 ElastiCache

Ini mengacu pada migrasi dari instance yang dihosting sendiri ke ElastiCache layanan. Untuk informasi tentang peningkatan dari Redis OSS ke Valkey di lihat. ElastiCache Memutakhirkan versi mesin termasuk peningkatan mesin silang

> **Note**
>
> Migrasi online tidak didukung ke cache atau cluster ElastiCache tanpa server yang berjalan pada tipe node r6gd.

Gambaran Umum

Untuk memigrasikan data Anda dari sumber terbuka Valkey atau Redis OSS yang berjalan di Amazon EC2 ke Amazon memerlukan penerapan ElastiCache Amazon yang sudah ada atau yang baru dibuat. ElastiCache Deployment harus memiliki konfigurasi yang siap untuk migrasi. Ini juga harus selaras dengan konfigurasi yang Anda inginkan, termasuk atribut seperti jenis instans, jumlah serpihan, dan jumlah replika.

Migrasi online dirancang untuk migrasi data dari Valkey atau Redis OSS open-source yang dihosting sendiri di Amazon EC2 ke ElastiCache, dan bukan untuk memindahkan data antar cluster. ElastiCache

> **Important**
>
> Sangat direkomendasikan agar Anda membaca bagian berikut secara keseluruhan sebelum memulai proses migrasi online.

Migrasi dimulai saat Anda memanggil operasi API `StartMigration` atau perintah AWS CLI . Saat memigrasikan cluster yang dinonaktifkan mode cluster Valkey atau Redis OSS, proses migrasi

membuat simpul utama klaster Valkey atau Redis OSS menjadi replika dari ElastiCache sumber Valkey atau Redis OSS primer Anda. Saat memigrasikan cluster yang diaktifkan mode cluster Valkey atau Redis OSS, proses migrasi membuat simpul utama dari setiap pecahan menjadi replika ElastiCache pecahan terkait cluster sumber Anda yang memiliki slot yang sama.

Setelah perubahan sisi klien siap, panggil operasi API `CompleteMigration`. Operasi API ini mempromosikan penerapan Anda ke ElastiCache penyebaran Valkey atau Redis OSS utama Anda dengan node primer dan replika (sebagaimana berlaku). Sekarang Anda dapat mengalihkan aplikasi klien Anda untuk mulai menulis data ke ElastiCache. Sepanjang migrasi, Anda dapat memeriksa status replikasi dengan menjalankan perintah VALKEY-CLI INFO pada node Valkey Anda dan pada node utama. ElastiCache

Langkah migrasi

Topik berikut menguraikan proses untuk melakukan migrasi data Anda:

- Mempersiapkan sumber dan target untuk migrasi
- Menguji migrasi data
- Memulai migrasi
- Memverifikasi progres migrasi data
- Menyelesaikan migrasi data

Mempersiapkan sumber dan target untuk migrasi

Dengan langkah-langkah ini, Anda dapat mempersiapkan untuk memigrasikan data Anda dari sumber Valkey atau Redis yang dihosting sendiri EC2 ke ElastiCache, atau dari kluster Redis OSS ke cluster Valkey. ElastiCache

Ini mengacu pada migrasi dari instance yang dihosting sendiri ke ElastiCache layanan. Untuk informasi tentang peningkatan dari Redis OSS ke Valkey di lihat. ElastiCache Memutakhirkan versi mesin termasuk peningkatan mesin silang

Anda harus memastikan bahwa keempat prasyarat yang disebutkan berikut ini terpenuhi sebelum memulai migrasi dari ElastiCache konsol, API, atau CLI. AWS

Untuk mempersiapkan sumber dan target Valkey atau Redis OSS Nodes untuk migrasi

1. Identifikasi ElastiCache penerapan target dan pastikan Anda dapat memigrasikan data ke sana.

ElastiCache Penerapan yang sudah ada atau yang baru dibuat harus memenuhi persyaratan berikut untuk migrasi:

- Ini menggunakan Valkey, atau Redis OSS 5.0.6 atau lebih tinggi.
- Itu tidak mengaktifkan enkripsi dalam perjalanan.
- Mengaktifkan Multi-AZ.
- Ini memiliki memori yang cukup tersedia untuk menyesuaikan data dari cluster Valkey atau Redis OSS Anda. Untuk mengonfigurasi pengaturan memori cadangan yang tepat, lihat Mengelola memori cadangan untuk Valkey dan Redis OSS.
- Untuk mode cluster dinonaktifkan, Anda dapat bermigrasi langsung dari Valkey atau Redis OSS versi 2.8.21 dan seterusnya ke Valkey atau Redis OSS versi 5.0.6 dan seterusnya jika menggunakan CLI atau Valkey atau Redis OSS versi 5.0.6 dan seterusnya menggunakan CLI atau konsol. Untuk mode cluster diaktifkan, Anda dapat bermigrasi langsung dari versi Valkey atau Redis OSS yang diaktifkan mode cluster ke Redis OSS versi 5.0.6 dan seterusnya, jika menggunakan CLI atau Redis OSS versi 5.0.6 dan seterusnya menggunakan CLI atau konsol.
- Jumlah serpihan dalam sumber dan target cocok.
- Bukan bagian dari penyimpanan data global.
- Menonaktifkan tingkat data.

2. Pastikan konfigurasi Valkey atau Redis OSS open-source Anda dan penerapannya kompatibel. ElastiCache

   Minimal, semua hal berikut dalam ElastiCache penerapan target harus kompatibel dengan konfigurasi Valkey atau Redis OSS Anda untuk replikasi:

   - Cluster Anda seharusnya tidak mengaktifkan AUTH.
   - Konfigurasi `protected-mode` harus disetel keno.
   - Jika Anda memiliki `bind` konfigurasi di konfigurasi Valkey atau Redis OSS Anda, maka itu harus diperbarui untuk mengizinkan permintaan dari node. ElastiCache
   - Jumlah database logis harus sama pada ElastiCache node dan cluster Valkey atau Redis OSS Anda. Nilai ini diatur menggunakan konfigurasi `databases` Valkey atau Redis OSS.
   - Perintah Valkey atau Redis OSS yang melakukan modifikasi data tidak boleh diganti namanya untuk memungkinkan replikasi data berhasil. misalnya`async`,,,,, dan. `psync` `info` `config` `command` `cluster`

- Untuk mereplikasi data dari cluster Valkey atau Redis OSS Anda ElastiCache, pastikan ada CPU dan memori yang cukup untuk menangani beban tambahan ini. Beban ini berasal dari file RDB yang dibuat oleh cluster Valkey atau Redis OSS Anda dan ditransfer melalui jaringan ke node. ElastiCache
- Semua instance Valkey atau Redis OSS di cluster sumber harus berjalan pada port yang sama.

3. Pastikan instans Anda dapat terhubung ElastiCache dengan melakukan hal berikut:

   - Pastikan bahwa alamat IP instans Anda adalah privat.
   - Tetapkan atau buat ElastiCache penerapan di cloud pribadi virtual (VPC) yang sama dengan Valkey atau Redis OSS Anda pada instans Anda (disarankan).
   - Jika VPCs berbeda, atur VPC peering untuk memungkinkan akses antar node. Untuk informasi selengkapnya tentang peering VPC, lihat Pola Akses untuk Mengakses ElastiCache Cache di VPC Amazon.
   - Grup keamanan yang dilampirkan pada instans Valkey atau Redis OSS Anda harus mengizinkan lalu lintas masuk dari node. ElastiCache

4. Pastikan aplikasi Anda dapat mengarahkan lalu lintas ke ElastiCache node setelah migrasi data selesai. Lihat informasi yang lebih lengkap di Pola Akses untuk Mengakses ElastiCache Cache di VPC Amazon.

Menguji migrasi data

Setelah semua prasyarat selesai, Anda dapat memvalidasi penyiapan migrasi menggunakan API, atau. AWS Management Console ElastiCache AWS CLI Contoh berikut menunjukkan penggunaan CLI.

Uji migrasi dengan memanggil perintah `test-migration` dengan parameter berikut:

- `--replication-group-id` – ID grup replikasi yang menjadi destinasi migrasi data.
- `--customer-node-endpoint-list` – Daftar titik akhir sumber data yang akan dimigrasikan. Daftar harus memiliki hanya satu elemen.

Contoh berikut menunjukkan penggunaan CLI.

```
aws elasticache test-migration --replication-group-id test-cluster --customer-node-
endpoint-list "Address='10.0.0.241',Port=6379"
```

ElastiCache akan memvalidasi pengaturan migrasi tanpa migrasi data aktual.

Memulai migrasi

Setelah semua prasyarat selesai, Anda dapat memulai migrasi data menggunakan API AWS Management Console, ElastiCache atau. AWS CLI Untuk mode klaster diaktifkan, jika migrasi slot berbeda, resharding akan dilakukan sebelum migrasi langsung. Contoh berikut menunjukkan penggunaan CLI.

> **Note**
>
> Kami merekomendasikan untuk menggunakan API `TestMigration` untuk memvalidasi penyiapan migrasi. Tapi ini benar-benar opsional.

Mulai migrasi dengan memanggil perintah `start-migration` dengan parameter berikut:

- `--replication-group-id`— Pengidentifikasi kelompok ElastiCache replikasi target
- `--customer-node-endpoint-list`— Daftar titik akhir dengan alamat DNS atau IP dan port tempat cluster Valkey atau Redis OSS sumber Anda berjalan. Daftar ini hanya dapat mengambil satu elemen baik untuk mode klaster dinonaktifkan dan mode klaster diaktifkan. Jika Anda telah mengaktifkan replikasi berantai, titik akhir dapat menunjuk ke replika alih-alih simpul utama di cluster Valkey atau Redis OSS Anda.

Contoh berikut menunjukkan penggunaan CLI.

```
aws elasticache start-migration --replication-group-id test-cluster --customer-node-
endpoint-list "Address='10.0.0.241',Port=6379"
```

Saat Anda menjalankan perintah ini, simpul ElastiCache utama (di setiap pecahan) mengonfigurasi dirinya sendiri untuk menjadi replika instance Valkey atau Redis OSS Anda (dalam pecahan terkait yang memiliki slot yang sama di redis yang diaktifkan cluster). Status ElastiCache klaster berubah menjadi migrasi dan data mulai bermigrasi dari instans Valkey atau Redis OSS Anda ke node utama. ElastiCache Bergantung pada ukuran data dan pemuatan pada instans Valkey atau Redis OSS

Anda, migrasi dapat memakan waktu beberapa saat untuk diselesaikan. Anda dapat memeriksa kemajuan migrasi dengan menjalankan perintah VALKEY-CLI INFO pada instance Valkey dan node utama Anda. ElastiCache

Setelah replikasi berhasil, semua penulisan ke instance Valkey atau Redis OSS Anda menyebar ke cluster. ElastiCache Anda dapat menggunakan ElastiCache node untuk membaca. Namun, Anda tidak dapat menulis ke klaster ElastiCache. Jika node ElastiCache primer memiliki node replika lain yang terhubung dengannya, node replika ini terus mereplikasi dari node primer. ElastiCache Dengan cara ini, semua data dari cluster Valkey atau Redis OSS Anda direplikasi ke semua node di cluster. ElastiCache

Jika node ElastiCache primer tidak dapat menjadi replika instance Valkey atau Redis OSS Anda, ia mencoba beberapa kali sebelum akhirnya mempromosikan dirinya kembali ke primer. Status klaster ElastiCache kemudian berubah menjadi tersedia, dan peristiwa grup replikasi tentang kegagalan memulai migrasi akan dikirim. Untuk memecahkan masalah seperti kegagalan tersebut, periksa hal berikut:

- Lihat peristiwa grup replikasi. Gunakan informasi spesifik dari peristiwa untuk memperbaiki kegagalan migrasi.
- Jika peristiwa tidak memberikan informasi spesifik apa pun, pastikan bahwa Anda telah mengikuti pedoman di Mempersiapkan sumber dan target untuk migrasi.
- Pastikan bahwa konfigurasi routing untuk VPC dan subnet Anda memungkinkan lalu lintas ElastiCache antar node dan instance Valkey atau Redis OSS Anda.
- Pastikan grup keamanan yang terpasang pada instans Valkey atau Redis OSS Anda memungkinkan lalu lintas input terikat dari node. ElastiCache
- Periksa log Valkey atau Redis OSS untuk instans Anda untuk informasi lebih lanjut tentang kegagalan khusus untuk replikasi.

Memverifikasi progres migrasi data

Setelah migrasi data dimulai, Anda dapat melakukan hal berikut untuk melacak progresnya:

- Verifikasi bahwa Valkey atau Redis OSS `master_link_status` ada up dalam `INFO` perintah pada simpul ElastiCache utama. Anda juga dapat menemukan informasi ini di ElastiCache konsol. Pilih klaster dan di bawah CloudWatch metrik, amati Status Kesehatan Tautan Utama. Setelah nilainya mencapai 1, itu berarti data sudah sinkron.

- Anda dapat memeriksa apakah ElastiCache replika memiliki status online dengan menjalankan `INFO` perintah pada instance Valkey atau Redis OSS Anda. Melakukan hal ini juga menyediakan informasi tentang lag replikasi.
- Verifikasi buffer output klien rendah dengan menggunakan perintah CLIENT LIST pada instance Valkey atau Redis OSS Anda.

Setelah migrasi data selesai, data disinkronkan dengan penulisan baru yang datang ke node utama klaster Valkey atau Redis OSS Anda.

Menyelesaikan migrasi data

Ketika Anda siap untuk memotong ke ElastiCache cluster, gunakan perintah `complete-migration` CLI dengan parameter berikut:

- `--replication-group-id` – Pengidentifikasi untuk grup replikasi.
- `--force` – Nilai yang memaksa migrasi untuk berhenti tanpa memastikan bahwa data sudah sinkron.

Berikut adalah contohnya.

```
aws elasticache complete-migration --replication-group-id test-cluster
```

Saat Anda menjalankan perintah ini, simpul ElastiCache utama (di setiap pecahan) berhenti mereplikasi dari instance Valkey atau Redis OSS Anda dan mempromosikannya ke primer. Promosi ini biasanya selesai dalam beberapa menit. Untuk mengonfirmasi promosi ke primer, periksa peristiwa `Complete Migration successful for test-cluster`. Pada titik ini, Anda dapat mengarahkan aplikasi Anda untuk ElastiCache menulis dan membaca. ElastiCache Status klaster harus berubah dari migrasi ke tersedia.

Jika promosi ke primer gagal, simpul ElastiCache utama terus mereplikasi dari instans Valkey atau Redis OSS Anda. ElastiCache Cluster terus berada dalam status migrasi, dan pesan peristiwa grup replikasi tentang kegagalan dikirim. Untuk memecahkan masalah kegagalan ini, lihat hal berikut:

- Periksa peristiwa grup replikasi. Gunakan informasi spesifik dari peristiwa itu untuk memperbaiki kegagalan.
- Anda mungkin mendapatkan pesan peristiwa tentang data yang tidak sinkron. Jika demikian, pastikan bahwa ElastiCache primer dapat mereplikasi dari instans Valkey atau Redis OSS Anda

dan keduanya sinkron. Jika Anda masih ingin menghentikan migrasi, Anda dapat menjalankan perintah sebelumnya dengan opsi `–force`.

- Anda mungkin mendapatkan pesan acara jika salah satu ElastiCache node sedang menjalani penggantian. Anda dapat mencoba lagi menyelesaikan langkah migrasi setelah penggantian selesai.

Melakukan migrasi data online menggunakan Konsol

Anda dapat menggunakan file AWS Management Console untuk memigrasikan data dari klaster ke klaster Valkey atau Redis OSS Anda.

Untuk melakukan migrasi data online menggunakan konsol

1. Masuk ke konsol dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Buat cluster Valkey atau Redis OSS baru atau pilih cluster yang ada. Pastikan bahwa klaster memenuhi persyaratan berikut:
   - Versi mesin Anda harus Valkey 7.2 dan lebih tinggi, atau Redis OSS 5.0.6 atau lebih tinggi.
   - Cluster Anda seharusnya tidak mengaktifkan AUTH.
   - Konfigurasi `protected-mode` harus disetel keno.
   - Jika Anda memiliki `bind` konfigurasi di konfigurasi Valkey atau Redis OSS Anda, maka itu harus diperbarui untuk mengizinkan permintaan dari node. ElastiCache
   - Jumlah database harus sama antara ElastiCache node dan cluster Valkey atau Redis OSS Anda. Nilai ini diatur menggunakan `databases` konfigurasi mesin.
   - Perintah Valkey atau Redis OSS yang melakukan modifikasi data tidak boleh diganti namanya untuk memungkinkan replikasi data berhasil.
   - Untuk mereplikasi data dari cluster Valkey atau Redis OSS Anda ElastiCache, pastikan ada CPU dan memori yang cukup untuk menangani beban tambahan ini. Beban ini berasal dari file RDB yang dibuat oleh cluster Valkey atau Redis OSS Anda dan ditransfer melalui jaringan ke node. ElastiCache
   - Klaster berada dalam status tersedia.
3. Setelah memilih klaster, pilih Migrasikan Data dari Titik Akhiruntuk Tindakan.
4. Dalam kotak dialog Migrasi Data dari Endpoint, masukkan alamat IP, dan port tempat cluster Valkey atau Redis OSS Anda tersedia.

> **Important**
>
> Alamat IP harus sama persis. Jika Anda salah memasukkan alamat, migrasi akan gagal.

5. Pilih Mulai Migrasi.

   Saat klaster memulai migrasi, statusnya berubah ke Mengubah lalu ke Bermigrasi.

6. Pantau progres migrasi dengan memilih Peristiwa pada panel navigasi.

Anda dapat menghentikan migrasi kapan pun. Untuk melakukannya, pilih klaster Anda dan pilih Hentikan Migrasi Data untuk Tindakan. Klaster akan berubah status menjadi Tersedia.

Jika migrasi berhasil, status klaster akan menjadi Tersedia dan log peristiwa menunjukkan hal berikut:

`Migration operation succeeded for replication group` *`ElastiCacheClusterName`*`.`

Jika migrasi gagal, status klaster akan menjadi Tersedia dan log peristiwa menunjukkan hal berikut:

`Migration operation failed for replication group` *`ElastiCacheClusterName`*`.`

Memilih wilayah dan zona ketersediaan untuk ElastiCache

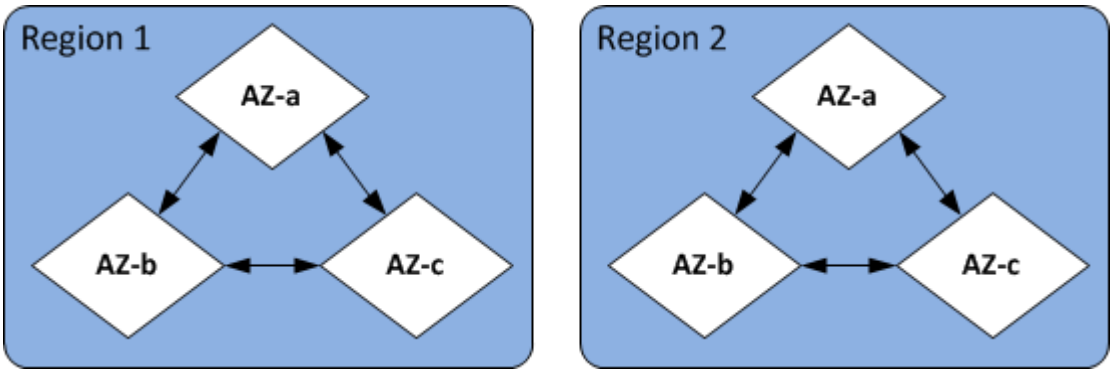
Anda dapat memberikan skalabilitas dan keandalan tambahan ke ElastiCache kluster Anda dengan menetapkan Wilayah dan Zona Ketersediaan menggunakan titik akhir yang sesuai.

AWS Sumber daya komputasi awan ditempatkan di fasilitas pusat data yang sangat tersedia. Untuk memberikan skalabilitas dan keandalan tambahan, fasilitas pusat data ini ditempatkan di beberapa lokasi fisik yang berbeda. Lokasi ini dikategorikan berdasarkan wilayah dan Zona Ketersediaan.

AWS Daerah besar dan tersebar luas ke lokasi geografis yang terpisah. Availability Zone adalah lokasi berbeda dalam AWS Wilayah yang dirancang untuk diisolasi dari kegagalan di Availability Zone lainnya. Mereka menyediakan konektivitas jaringan latensi rendah yang murah ke Availability Zone lainnya di Wilayah yang sama AWS .

> **Important**
>
> Setiap wilayah bersifat independen sepenuhnya. ElastiCache Aktivitas apa pun yang Anda mulai (misalnya, membuat kluster) hanya berjalan di wilayah default Anda saat ini.

Untuk membuat atau menggunakan klaster di wilayah tertentu, gunakan titik akhir layanan regional yang terkait. Untuk titik akhir layanan, lihat Wilayah & titik akhir yang didukung.



Wilayah dan Zona Ketersediaan

Topik

- Pertimbangan Availability Zone dengan Memcached
- Menempatkan simpul Anda
- Wilayah & titik akhir yang didukung
- Menggunakan zona lokal dengan ElastiCache
- Menggunakan Outposts dengan ElastiCache

Pertimbangan Availability Zone dengan Memcached

Mendistribusikan simpul Memcached Anda ke beberapa Zona Ketersediaan dalam suatu wilayah membantu melindungi Anda dari dampak kegagalan besar, seperti pemadaman daya di sebuah Zona Ketersediaan.

Caching Nirserver

ElastiCache caching tanpa server menciptakan cache yang sangat tersedia yang mencakup beberapa Availability Zone. Anda dapat menentukan subnet dari zona ketersediaan yang berbeda

dan VPC yang sama saat Anda membuat klaster tanpa server ElastiCache atau akan memilih subnet secara otomatis dari VPC default Anda.

Merancang sendiri ElastiCache untuk cluster Memcached

Klaster Memcached dapat memiliki hingga 300 simpul. Saat Anda membuat atau menambahkan node ke cluster Memcached Anda, Anda dapat menentukan Availability Zone tunggal untuk semua node Anda, memungkinkan ElastiCache untuk memilih Availability Zone tunggal untuk semua node Anda, menentukan Availability Zone untuk setiap node, atau memungkinkan ElastiCache untuk memilih Availability Zone untuk setiap node. Simpul baru dapat dibuat di Zona Ketersediaan yang berbeda pada saat Anda menambahkan simpul ke klaster Memcached yang sudah ada. Setelah simpul cache dibuat, Zona Ketersediaannya tidak dapat diubah.

Jika Anda ingin klaster dalam satu klaster Availability Zone memiliki node yang didistribusikan di beberapa Availability Zone, ElastiCache dapat membuat node baru di berbagai Availability Zones. Anda kemudian dapat menghapus beberapa atau semua simpul cache yang asli. Kami merekomendasikan pendekatan ini.

Untuk memigrasikan simpul Memcached dari Zona Ketersediaan tunggal ke beberapa Zona Ketersediaan

1. Ubah klaster Anda dengan membuat simpul cache baru di Zona Ketersediaan sesuai tempat yang Anda inginkan. Pada permintaan Anda, lakukan hal berikut:

   - Tetapkan `AZMode` (CLI: `- -az-mode`) ke `cross-az`.
   - Tetapkan `NumCacheNodes` (CLI: `- -num-cache-nodes`) ke jumlah simpul cache yang aktif saat ini ditambah jumlah simpul cache baru yang ingin Anda buat.
   - Tetapkan `NewAvailabilityZones` (CLI: `- -new-availability-zones`) ke daftar zona yang Anda inginkan sebagai tempat untuk simpul cache yang baru dibuat. Untuk membiarkan ElastiCache menentukan Availability Zone untuk setiap simpul baru, jangan menetapkan sebuah daftar.
   - Tetapkan `ApplyImmediately` (CLI: `- -apply-immediately`) menjadi BETUL.

   > **Note**
   >
   > Jika Anda tidak menggunakan penemuan otomatis, pastikan untuk memperbarui aplikasi klien Anda dengan titik akhir simpul cache yang baru.

Sebelum pindah ke langkah berikutnya, pastikan simpul Memcached dibuat sepenuhnya dan tersedia.

2. Ubah klaster Anda dengan menghapus simpul yang tidak diinginkan lagi di Zona Ketersediaan asli. Pada permintaan Anda, lakukan hal berikut:

   - Tetapkan `NumCacheNodes` (CLI: `- -num-cache-nodes`) untuk jumlah simpul cache aktif yang Anda inginkan setelah perubahan ini diterapkan.
   - Tetapkan `CacheNodeIdsToRemove` (CLI: `- -nodes-to-remove`) ke daftar simpul cache yang ingin Anda hapus dari klaster.

     Jumlah node cache yang IDs terdaftar harus sama dengan jumlah node yang saat ini aktif dikurangi nilai dalam`NumCacheNodes`.
   - (Opsional) Tetapkan `ApplyImmediately` (CLI: `- -apply-immediately`) ke benar.

     Jika Anda tidak menetapkan `ApplyImmediately` (CLI: `- -apply-immediately`) menjadi BETUL, maka penghapusan simpul akan berlangsung pada jendela pemeliharaan berikutnya.

Menempatkan simpul Anda

Amazon ElastiCache mendukung lokasi semua node cluster dalam satu atau beberapa Availability Zones (AZs). Selanjutnya, jika Anda memilih untuk menemukan node Anda dalam beberapa AZs (disarankan), ElastiCache memungkinkan Anda untuk memilih AZ untuk setiap node, atau memungkinkan ElastiCache untuk memilihnya untuk Anda.

Dengan menemukan node di tempat yang berbeda AZs, Anda menghilangkan kemungkinan kegagalan, seperti pemadaman listrik, dalam satu AZ akan menyebabkan seluruh sistem Anda gagal. Pengujian telah menunjukkan bahwa tidak ada perbedaan latensi yang signifikan antara menemukan semua node dalam satu AZ atau menyebarkannya ke beberapa. AZs

Anda dapat menentukan AZ untuk setiap node saat membuat klaster, atau dengan menambahkan node saat Anda memodifikasi cluster yang ada. Saat menentukan AZ untuk setiap node saat membuat cluster, AZ harus tersedia di grup subnet tersebut. Untuk informasi selengkapnya, lihat berikut ini:

- Membuat cluster untuk Memcached
- Membuat cluster untuk Valkey atau Redis OSS
- Memodifikasi cluster ElastiCache
- Menambahkan node ke ElastiCache cluster

Wilayah & titik akhir yang didukung

Amazon ElastiCache tersedia di beberapa AWS Wilayah. Ini berarti Anda dapat meluncurkan ElastiCache cluster di lokasi yang memenuhi kebutuhan Anda. Misalnya, Anda dapat meluncurkan di AWS Wilayah terdekat dengan pelanggan Anda, atau meluncurkan di AWS Wilayah tertentu untuk memenuhi persyaratan hukum tertentu.

Setiap Wilayah dirancang untuk terisolasi sepenuhnya dari Wilayah lainnya. Di dalam setiap wilayah terdapat beberapa Zona Ketersediaan (AZ). ElastiCache Cache tanpa server secara otomatis mereplikasi data di beberapa zona ketersediaan (kecuali`us-west-1`, di mana data direplikasi dalam dua zona ketersediaan) untuk ketersediaan tinggi. Saat mendesain ElastiCache cluster Anda sendiri, Anda dapat memilih untuk meluncurkan node Anda secara berbeda AZs untuk mencapai toleransi kesalahan. Untuk informasi selengkapnya tentang Wilayah dan Zona Ketersediaan, lihat Memilih wilayah dan zona ketersediaan untuk ElastiCache di bagian atas topik ini.

Daerah di ElastiCache mana didukung

Nama Wilayah/Wilayah	Titik Akhir	Protokol	
Wilayah AS Timur (Ohio) `us-east-2`	`elasticache.us-east-2.amazonaws.com`	HTTPS	
Wilayah AS Timur (Virginia Utara) `us-east-1`	`elasticache.us-east-1.amazonaws.com`	HTTPS	
Wilayah AS Barat (California Utara) `us-west-1`	`elasticache.us-west-1.amazonaws.com`	HTTPS	
Wilayah AS Barat (Oregon) `us-west-2`	`elasticache.us-west-2.amazonaws.com`	HTTPS	
Wilayah Kanada (Pusat) `ca-central-1`	`elasticache.ca-central-1.amazonaws.com`	HTTPS	
Wilayah Kanada (Barat) `ca-west-1`	`elasticache.ca-west-1.amazonaws.com`	HTTPS	
Asia Pasifik (Jakarta) `ap-southeast-3`	`elasticache.ap-southeast-3.amazonaws.com`	HTTPS	

Nama Wilayah/Wilayah	Titik Akhir	Protokol	
Wilayah Asia Pasifik (Mumbai) `ap-south-1`	`elasticache.ap-south-1.amazonaws.com`	HTTPS	
Wilayah Asia Pasifik (Hyderabad) `ap-south-2`	`elasticache.ap-south-2.amazonaws.com`	HTTPS	
Wilayah Asia Pasifik (Tokyo) `ap-northeast-1`	`elasticache.ap-northeast-1.amazonaws.com`	HTTPS	
Wilayah Asia Pasifik (Seoul) `ap-northeast-2`	`elasticache.ap-northeast-2.amazonaws.com`	HTTPS	
Wilayah Asia Pasifik (Osaka) `ap-northeast-3`	`elasticache.ap-northeast-3.amazonaws.com`	HTTPS	
Wilayah Asia Pasifik (Singapura) `ap-southeast-1`	`elasticache.ap-southeast-1.amazonaws.com`	HTTPS	
Wilayah Asia Pasifik (Sydney) `ap-southeast-2`	`elasticache.ap-southeast-2.amazonaws.com`	HTTPS	

Nama Wilayah/Wilayah	Titik Akhir	Protokol	
Wilayah Eropa (Frankfurt) `eu-central-1`	`elasticache.eu-central-1.amazonaws.com`	HTTPS	
Wilayah Eropa (Zürich) `eu-central-2`	`elasticache.eu-central-2.amazonaws.com`	HTTPS	
Wilayah Eropa (Stockholm) `eu-north-1`	`elasticache.eu-north-1.amazonaws.com`	HTTPS	
Wilayah Timur Tengah (Bahrain) `me-south-1`	`elasticache.me-south-1.amazonaws.com`	HTTPS	
Wilayah Timur Tengah (UEA) `me-central-1`	`elasticache.me-central-1.amazonaws.com`	HTTPS	
Wilayah Eropa (Irlandia) `eu-west-1`	`elasticache.eu-west-1.amazonaws.com`	HTTPS	
Wilayah Eropa (London) `eu-west-2`	`elasticache.eu-west-2.amazonaws.com`	HTTPS	

Nama Wilayah/Wilayah	Titik Akhir	Protokol	
Wilayah Eropa (Paris) `eu-west-3`	`elasticache.eu-west-3.amazonaws.com`	HTTPS	
Wilayah Eropa (Milan) `eu-south-1`	`elasticache.eu-south-1.amazonaws.com`	HTTPS	
Wilayah Eropa (Spanyol) `eu-south-2`	`elasticache.eu-south-2.amazonaws.com`	HTTPS	
Wilayah Amerika Selatan (Sao Paulo) `sa-east-1`	`elasticache.sa-east-1.amazonaws.com`	HTTPS	
Wilayah Tiongkok (Beijing) `cn-north-1`	`elasticache.cn-north-1.amazonaws.com.cn`	HTTPS	
Wilayah Tiongkok (Ningxia) `cn-northwest-1`	`elasticache.cn-northwest-1.amazonaws.com.cn`	HTTPS	
Wilayah Asia Pasifik (Hong Kong) `ap-east-1`	`elasticache.ap-east-1.amazonaws.com`	HTTPS	
Wilayah Afrika (Cape Town) `af-south-1`	`elasticache.af-south-1.amazonaws.com`	HTTPS	

Nama Wilayah/Wilayah	Titik Akhir	Protokol	
Wilayah Israel (Tel Aviv) `il-central-1`	`elasticache.il-central-1.amazonaws.com`	HTTPS	
AWS GovCloud (AS-Barat) `us-gov-west-1`	`elasticache.us-gov-west-1.amazonaws.com`	HTTPS	
AWS GovCloud (AS-Timur) `us-gov-east-1`	`elasticache.us-gov-east-1.amazonaws.com`	HTTPS	
Untuk informasi tentang penggunaan AWS GovCloud (AS) dengan ElastiCache, lihat Layanan di wilayah AWS GovCloud (AS): ElastiCache.			

Beberapa Wilayah mendukung subset tipe node. Untuk tabel tipe node yang didukung menurut AWS Region, lihatJenis simpul yang didukung oleh Wilayah AWS.

Sebagian besar Wilayah mendukung pembuatan koneksi pribadi antara VPC dan titik akhir ElastiCache API Anda, dengan membuat titik akhir VPC antarmuka melalui. AWS PrivateLink Untuk informasi selengkapnya, lihat ElastiCache API dan antarmuka VPC endpoint ()AWS PrivateLink.

Untuk tabel AWS produk dan layanan menurut wilayah, lihat Produk dan Layanan menurut Wilayah.

Menggunakan zona lokal dengan ElastiCache

Zona Lokal adalah perpanjangan dari AWS Wilayah yang secara geografis dekat dengan pengguna Anda. Anda dapat memperluas virtual private cloud (VPC) dari AWS Region induk ke Local Zones dengan membuat subnet baru dan menetapkannya ke Local Zone. Saat membuat subnet di Zona Lokal, VPC Anda diperluas ke Zona Lokal tersebut. Subnet di Zona Lokal beroperasi sama seperti subnet lain di VPC Anda.

Dengan menggunakan Local Zones, Anda dapat menempatkan sumber daya seperti ElastiCache klaster di beberapa lokasi yang dekat dengan pengguna Anda.

Saat membuat ElastiCache cluster, Anda dapat memilih subnet di Local Zone. Local Zone memiliki koneksi sendiri ke internet dan mendukung AWS Direct Connect. Oleh karena itu, sumber daya yang dibuat di Zona Lokal dapat melayani pengguna lokal dengan komunikasi berlatensi sangat rendah. Untuk informasi selengkapnya, lihat Zona Lokal AWS.

Zona Lokal diwakili oleh kode AWS Wilayah diikuti oleh pengidentifikasi yang menunjukkan lokasi, misalnya`us-west-2-lax-1a`.

Saat ini, Zona Lokal yang tersedia adalah `us-west-2-lax-1a` dan `us-west-2-lax-1b`.

Batasan berikut berlaku ElastiCache untuk Local Zones:

- Penyimpanan data global tidak didukung.
- Migrasi online tidak didukung.
- Jenis simpul berikut didukung oleh Zona Lokal saat ini:
  - Generasi saat ini:

    Jenis simpul M5: `cache.m5.large`, `cache.m5.xlarge`, `cache.m5.2xlarge`, `cache.m5.4xlarge`, `cache.m5.12xlarge`, `cache.m5.24xlarge`

    Jenis simpul R5: `cache.r5.large`, `cache.r5.xlarge`, `cache.r5.2xlarge`, `cache.r5.4xlarge`, `cache.r5.12xlarge`, `cache.r5.24xlarge`

    Jenis simpul T3: `cache.t3.micro`, `cache.t3.small`, `cache.t3.medium`

Mengaktifkan zona lokal

1. Aktifkan Zona Lokal di EC2 konsol Amazon.

   Untuk informasi selengkapnya, lihat Mengaktifkan Local Zones di Panduan EC2 Pengguna Amazon.
2. Buat subnet di Zona Lokal.

   Untuk informasi selengkapnya, lihat Membuat subnet dalam VPC Anda dalam Panduan Pengguna Amazon VPC.
3. Buat grup ElastiCache subnet di Zona Lokal.

   Saat Anda membuat grup ElastiCache subnet, pilih grup Availability Zone untuk Local Zone.

Untuk informasi selengkapnya, lihat Membuat grup subnet.

4. Buat cluster ElastiCache untuk Memcached yang menggunakan ElastiCache subnet di Local Zone.

   Untuk informasi selengkapnya, lihat Membuat klaster Memcached (konsol).

5. Buat cluster ElastiCache untuk Redis OSS yang menggunakan ElastiCache subnet di Local Zone. Untuk informasi selengkapnya, lihat salah satu topik berikut:

   - Membuat cluster Valkey (mode cluster dinonaktifkan) (Konsol)
   - Membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) (Konsol)

Menggunakan Outposts dengan ElastiCache

Anda dapat menggunakan AWS Outposts dengan. ElastiCache Outposts adalah layanan yang dikelola sepenuhnya yang memperluas AWS infrastruktur, layanan APIs, dan alat ke tempat pelanggan. Dengan menyediakan akses lokal ke infrastruktur AWS terkelola, AWS Outposts memungkinkan pelanggan untuk membangun dan menjalankan aplikasi di tempat menggunakan antarmuka pemrograman yang sama seperti di AWS Wilayah, sambil menggunakan sumber daya komputasi dan penyimpanan lokal untuk latensi yang lebih rendah dan kebutuhan pemrosesan data lokal. Outpost adalah kumpulan kapasitas AWS komputasi dan penyimpanan yang digunakan di situs pelanggan. AWS mengoperasikan, memantau, dan mengelola kapasitas ini sebagai bagian dari suatu AWS Wilayah. Anda dapat membuat subnet di Outpost Anda dan menentukannya saat Anda membuat AWS sumber daya seperti ElastiCache cluster.

> **Note**
>
> Dalam versi ini, berlaku batasan berikut:
>
> - ElastiCache untuk Outposts hanya mendukung keluarga node M5 dan R5.
> - Multi-AZ (replikasi lintas Outposts tidak didukung).
> - Migrasi langsung tidak didukung.
> - Snapshot lokal tidak didukung.
> - Log mesin dan log lambat tidak dapat diaktifkan.
> - ElastiCache di Outposts tidak mendukung CoIP.

- ElastiCache untuk Outposts tidak didukung di wilayah berikut: cn-north-1, cn-northwest-1 dan ap-northeast-3.

Menggunakan Outposts dengan konsol ElastiCache

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pada panel navigasi, pilih cache Valkey, cache Redis OSS, atau cache Memcached.
3. Jika Anda memilih cache Valkey, pilih Buat cache Valkey. Jika Anda memilih cache Redis OSS, pilih Buat cache Redis OSS. Jika Anda memilih cache Memcached, pilih Buat cache Memcached.
4. Di bawah Pengaturan cluster, pilih Desain cache Anda sendiri dan cache Cluster. Biarkan mode Cluster disetel sebagai Dinonaktifkan. Kemudian buat nama dan deskripsi opsional untuk cache.
5. Untuk lokasi, pilih Di tempat.
6. Di bagian lokal Anda akan melihat bidang Outpost ID. Masukkan ID tempat cluster akan berjalan.

   Semua pengaturan lebih lanjut di bawah pengaturan Cluster dapat tetap sebagai default.
7. Di Konektivitas, pilih Buat grup subnet baru dan masukkan ID VPC. Biarkan sisanya sebagai default, dan pilih Berikutnya.

Mengonfigurasi pilihan on-premise

Anda dapat memilih Outposts yang tersedia untuk menambahkan klaster cache atau, jika tidak ada Outposts yang tersedia, buat Outposts baru menggunakan langkah berikut:

Pada Opsi on-premise:

1. Di bawah pengaturan Valkey, pengaturan Redis OSS, atau pengaturan Memcached, tergantung pada mesin pilihan Anda:

   a. Nama: Masukkan nama untuk cluster

   b. Deskripsi: Masukkan deskripsi untuk cluster.

   c. Kompatibilitas versi mesin: Versi mesin didasarkan pada wilayah Outpost AWS

d. Port: Untuk Valkey atau Redis OSS, terima port default 6379. Untuk Memcached, terima port default 11211. Jika Anda lebih suka menggunakan port yang berbeda, ketikkan nomor port.

e. Grup parameter: Gunakan drop-down untuk memilih grup parameter default atau kustom.

f. Jenis Simpul: Instans yang tersedia didasarkan pada ketersediaan Outposts. Jika menggunakan Valkey atau Redis OSS, Porting Assistant untuk.NET for Outposts hanya mendukung keluarga node M5 dan R5. Dari daftar drop-down, pilih Outposts lalu pilih jenis simpul tersedia yang ingin Anda gunakan untuk klaster ini. Kemudian pilih Simpan.

g. Jumlah Replika: Masukkan jumlah replika baca yang ingin dibuat untuk grup replikasi ini. Anda harus memiliki minimal satu dan maksimal lima replika baca. Nilai default-nya adalah 2.

   Nama-nama replika baca yang dihasilkan otomatis mengikuti pola yang sama seperti nama klaster primer, dengan tanda hubung dan tiga digit nomor berurutan ditambahkan ke akhir, dimulai dengan `-002`. Misalnya, jika grup replikasi Anda bernama `MyGroup`, maka nama-nama berikutnya adalah `MyGroup-002`, `MyGroup-003`, `MyGroup-004`, `MyGroup-005`, `MyGroup-006`.

2. Di bawah Konektivitas:

   a. Grup Subnet: Dari daftar, pilih Buat baru.

      - Nama: Masukkan nama untuk grup subnet
      - Deskripsi: Masukkan deskripsi untuk grup subnet
      - ID VPC: ID VPC harus cocok dengan VPC Outposts. Jika Anda memilih VPC yang tidak memiliki subnet IDs di Outposts, daftar akan kembali kosong.
      - Zona Ketersediaan atau Outposts: Pilih Outposts yang Anda gunakan.
      - ID Subnet: Pilih ID subnet yang tersedia untuk Outposts. Jika tidak ada subnet IDs yang tersedia, Anda harus membuatnya. Untuk informasi selengkapnya, lihat Membuat Subnet.

   b. Pilih Buat.

Melihat detail klaster Outposts

Pada halaman daftar, pilih cluster milik AWS Outpost dan perhatikan hal berikut saat melihat detail Cluster:

- Availability Zone: Ini akan mewakili Outpost, menggunakan ARN (Amazon Resource Name) dan AWS Resource Number.
- Nama pos terdepan: Nama Pos AWS Terdepan.

Menggunakan Outposts dengan CLI AWS

Anda dapat menggunakan AWS Command Line Interface (AWS CLI) untuk mengontrol beberapa AWS layanan dari baris perintah dan mengotomatiskannya melalui skrip. Anda dapat menggunakan AWS CLI untuk operasi ad hoc (satu kali).

Mengunduh dan mengonfigurasi AWS CLI

AWS CLI Berjalan di Windows, macOS, atau Linux. Gunakan prosedur berikut untuk mengunduh dan mengonfigurasinya.

Untuk mengunduh, menginstal, dan mengonfigurasi CLI

1. Unduh AWS CLI di halaman web Antarmuka Baris AWS Perintah.
2. Ikuti petunjuk untuk Menginstal AWS CLI dan Mengkonfigurasi AWS CLI di Panduan Pengguna.AWS Command Line Interface

Menggunakan AWS CLI dengan Outposts

Gunakan operasi CLI berikut untuk membuat klaster cache yang menggunakan Outposts:

- create-cache-cluster— Menggunakan operasi ini, `outpost-mode` parameter menerima nilai yang menentukan apakah node dalam cluster cache dibuat dalam satu Outpost atau di beberapa Outposts.

> **Note**
>
> Pada saat ini, hanya mode `single-outpost` yang didukung.

```
aws elasticache create-cache-cluster \
   --cache-cluster-id cache cluster id \
   --outpost-mode single-outpost \
```

Bekerja dengan ElastiCache

Di bagian ini Anda dapat menemukan detail tentang cara mengelola berbagai komponen ElastiCache implementasi Anda.

Topik

Melakukan snapshot dan pemulihan

Amazon ElastiCache cache yang menjalankan Valkey, Redis OSS, atau Serverless Memcached dapat mencadangkan data mereka dengan membuat snapshot. Anda dapat menggunakan cadangan untuk memulihkan cache atau melakukan seeding data ke cache baru. Cadangan terdiri dari metadata cache, beserta semua data dalam cache. Semua cadangan ditulis ke Amazon Simple Storage Service (Amazon S3), yang menyediakan penyimpanan durabel. Kapan saja, Anda dapat memulihkan data Anda dengan membuat cache Valkey, Redis OSS, atau Memcached Tanpa Server baru dan mengisinya dengan data dari cadangan. Dengan ElastiCache, Anda dapat mengelola backup menggunakan AWS Management Console, the AWS Command Line Interface (AWS CLI), dan API. ElastiCache

Jika Anda ingin menghapus cache dan perlu mempertahankan datanya, Anda dapat mengambil tindakan pencegahan tambahan. Untuk melakukannya, buat cadangan manual terlebih dahulu, pastikan bahwa statusnya tersedia, lalu hapus cache. Dengan melakukannya, Anda dapat

memastikan bahwa jika cadangan gagal, Anda masih memiliki data cache yang tersedia. Anda dapat mencoba lagi membuat cadangan, dengan mengikuti praktik terbaik yang diuraikan sebelumnya.

Topik

Batasan pencadangan

Pertimbangkan batasan berikut saat merencanakan atau membuat cadangan:

- Backup dan restore didukung hanya untuk cache yang berjalan di Valkey, Redis OSS atau Serverless Memcached.
- Untuk cluster Valkey atau Redis OSS (mode cluster dinonaktifkan), pencadangan dan pemulihan tidak didukung pada node. `cache.t1.micro` Semua jenis simpul cache lain didukung.
- Untuk cluster Valkey atau Redis OSS (mode cluster enabled), backup dan restore didukung untuk semua jenis node.
- Selama periode 24 jam yang berdekatan, Anda dapat membuat tidak lebih dari 24 cadangan manual per cache tanpa server. Untuk cluster yang dirancang sendiri oleh Valkey dan Redis OSS, Anda dapat membuat tidak lebih dari 20 backup manual per node di cluster.
- Valkey atau Redis OSS (mode cluster diaktifkan) hanya mendukung pengambilan cadangan pada tingkat cluster (untuk API atau CLI, tingkat grup replikasi). Valkey atau Redis OSS (mode cluster

diaktifkan) tidak mendukung pengambilan cadangan pada tingkat pecahan (untuk API atau CLI, tingkat grup node).

- Selama proses pencadangan, Anda tidak dapat menjalankan operasi API atau CLI lainnya di cache tanpa server. Anda dapat menjalankan operasi API atau CLI pada cluster yang dirancang sendiri selama pencadangan.
- Jika Anda menggunakan cache Valkey atau Redis OSS dengan tiering data, Anda tidak dapat mengekspor cadangan ke Amazon S3.
- Anda dapat memulihkan cadangan klaster yang menggunakan jenis simpul r6gd hanya untuk klaster yang menggunakan jenis simpul r6gd.

Dampak performa pencadangan klaster yang dirancang sendiri

Pencadangan di cache nirserver bersifat transparan untuk aplikasi tanpa adanya dampak performa. Namun, saat membuat cadangan untuk klaster yang dirancang sendiri, mungkin ada beberapa dampak performa bergantung pada memori cadangan yang tersedia. Cadangan untuk cluster yang dirancang sendiri tidak tersedia ElastiCache untuk Memcached tetapi tersedia untuk Redis OSS. ElastiCache

Berikut ini adalah panduan untuk meningkatkan performa pencadangan untuk klaster yang dirancang sendiri.

- Mengatur `reserved-memory-percent` parameter — Untuk mengurangi paging yang berlebihan, kami sarankan Anda mengatur parameter. reserved-memory-percent Parameter ini mencegah Valkey dan Redis OSS mengkonsumsi semua memori node yang tersedia, dan dapat membantu mengurangi jumlah paging. Anda mungkin juga melihat peningkatan performa hanya menggunakan simpul yang lebih besar. Untuk informasi selengkapnya tentang memori cadangan dan reserved-memory-percentparameter, lihat. Mengelola memori cadangan untuk Valkey dan Redis OSS

- Buat cadangan dari replika baca - Jika Anda menjalankan Valkey atau Redis OSS dalam grup node dengan lebih dari satu node, Anda dapat mengambil cadangan dari node utama atau salah satu replika baca. Karena sumber daya sistem yang diperlukan selama BGSAVE, sebaiknya buat cadangan dari salah satu replika baca. Saat cadangan sedang dibuat dari replika, simpul primer tetap tidak terpengaruh oleh kebutuhan sumber daya BGSAVE. Simpul primer dapat terus melayani permintaan tanpa menjadi lambat.

Untuk melakukannya, lihat Membuat cadangan manual (Konsol) dan di bidang Nama Klaster di jendela Buat Cadangan, pilih replika, bukan simpul primer default.

Jika Anda menghapus grup replikasi dan meminta cadangan akhir, ElastiCache selalu ambil cadangan dari simpul utama. Ini memastikan bahwa Anda menangkap data Valkey atau Redis OSS terbaru, sebelum grup replikasi dihapus.

Menjadwalkan pencadangan otomatis

Anda dapat mengaktifkan pencadangan otomatis untuk cache tanpa server Valkey atau Redis OSS atau cluster yang dirancang sendiri. Saat pencadangan otomatis diaktifkan, ElastiCache buat cadangan cache setiap hari. Tidak ada dampak pada cache dan perubahan terjadi seketika. Pencadangan otomatis dapat membantu mencegah kehilangan data. Jika terjadi kegagalan, Anda dapat membuat cache baru, memulihkan data Anda dari cadangan terakhir. Hasilnya adalah cache dengan proses warm start, yang dimuat di awal dengan data Anda dan siap digunakan. Untuk informasi selengkapnya, lihat Melakukan pemulihan dari cadangan ke dalam cache baru.

Anda dapat mengaktifkan pencadangan otomatis untuk cache Tanpa Server Memcached apa pun. Saat pencadangan otomatis diaktifkan, ElastiCache buat cadangan cache setiap hari. Tidak ada dampak pada cache dan perubahan terjadi seketika. Pencadangan otomatis dapat membantu mencegah kehilangan data. Jika terjadi kegagalan, Anda dapat membuat cache baru, memulihkan data Anda dari cadangan terakhir. Hasilnya adalah cache dengan proses warm start, yang dimuat di awal dengan data Anda dan siap digunakan. Untuk informasi selengkapnya, lihat Melakukan pemulihan dari cadangan ke dalam cache baru.

Saat Anda menjadwalkan cadangan otomatis, Anda harus merencanakan pengaturan berikut:

- Waktu mulai Backup — Waktu hari ketika ElastiCache mulai membuat cadangan. Anda dapat mengatur periode pencadangan setiap saat pada waktu yang paling sesuai. Jika Anda tidak menentukan jendela cadangan, ElastiCache tetapkan satu secara otomatis.

- Batas retensi cadangan – Jumlah hari cadangan dipertahankan di Amazon S3. Contohnya, jika Anda menetapkan batas retensi ke 5, cadangan yang diambil hari ini akan dipertahankan selama 5 hari. Jika batas retensi berakhir, cadangan akan dihapus secara otomatis.

  Batas retensi cadangan maksimum adalah 35 hari. Jika batas retensi cadangan ditetapkan ke 0, cadangan otomatis akan dinonaktifkan untuk cache.

Ketika Anda menjadwalkan backup otomatis, ElastiCache akan mulai membuat backup. Anda dapat mengatur periode pencadangan setiap saat pada waktu yang paling sesuai. Jika Anda tidak menentukan jendela cadangan, ElastiCache tetapkan satu secara otomatis.

Anda dapat mengaktifkan atau menonaktifkan pencadangan otomatis saat membuat cache baru atau memperbarui cache yang ada, dengan menggunakan ElastiCache konsol, AWS CLI, atau

API. ElastiCache Untuk Valkey dan Redis OSS, ini dilakukan dengan mencentang kotak Aktifkan Pencadangan Otomatis di bagian Advanced Valkey Settings atau Advanced Redis OSS Settings. Untuk Memcached, ini dilakukan dengan mencentang kotak Aktifkan Pencadangan Otomatis di bagian Pengaturan Memcached Lanjutan.

Membuat cadangan manual

Selain cadangan otomatis, Anda dapat membuat cadangan manual kapan saja. Tidak seperti cadangan otomatis, yang secara otomatis dihapus setelah berakhirnya periode retensi yang ditentukan, cadangan manual tidak memiliki periode retensi yang jika berakhir, akan membuat cadangan manual dihapus secara otomatis. Bahkan jika Anda menghapus cache, setiap cadangan manual dari cache akan dipertahankan. Jika Anda tidak ingin lagi menyimpan cadangan manual, Anda harus menghapusnya sendiri secara eksplisit.

Selain membuat cadangan manual secara langsung, Anda dapat membuat cadangan manual dengan salah satu cara berikut:

- Menyalin cadangan. Tidak menjadi masalah apakah cadangan sumber dibuat secara otomatis atau manual.
- Membuat cadangan akhir. Buat cadangan segera sebelum menghapus klaster atau simpul.

Anda dapat membuat cadangan manual cache menggunakan AWS Management Console, AWS CLI, atau ElastiCache API.

Anda dapat membuat cadangan manual dari replika yang mengaktifkan mode cluster, dan mode cluster dinonaktifkan.

Membuat cadangan manual (Konsol)

Untuk membuat cadangan dari cache (konsol)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cache Valkey, cache Redis OSS, atau cache Memcached, tergantung pada preferensi Anda.
3. Pilih kotak di sebelah kiri nama cache yang ingin Anda cadangkan.
4. Pilih Cadangkan.
5. Di dialog Buat Cadangan, ketik nama untuk cadangan Anda di kotak Nama Cadangan. Sebaiknya berikan nama yang menunjukkan klaster yang dicadangkan serta tanggal dan waktu cadangan dibuat.

   Batasan penamaan klaster adalah sebagai berikut:

- Harus berisi 1–40 karakter alfanumerik atau tanda hubung.
- Harus diawali dengan huruf.
- Tidak boleh berisi dua tanda hubung berurutan.
- Tidak boleh diakhiri dengan tanda hubung.

6. Pilih Buat Cadangan.

   Status klaster berubah menjadi snapshotting.

Membuat cadangan manual (AWS CLI)

Cadangan manual dari cache tanpa server dengan AWS CLI

Untuk membuat cadangan manual cache menggunakan AWS CLI, gunakan `create-serverless-snapshot` AWS CLI operasi dengan parameter berikut:

- `--serverless-cache-name` – Nama cache nirserver yang Anda cadangkan.
- `--serverless-cache-snapshot-name` – Nama snapshot yang akan dibuat.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-serverless-snapshot \
                        --serverless-cache-name CacheName \
                        --serverless-cache-snapshot-name bkup-20231127
```

Untuk Windows:

```
aws elasticache create-serverless-snapshot ^
    --serverless-cache-name CacheName ^
    --serverless-cache-snapshot-name bkup-20231127
```

Cadangan manual dari cluster yang dirancang sendiri dengan AWS CLI

Untuk membuat cadangan manual dari cluster yang dirancang sendiri menggunakan AWS CLI, gunakan `create-snapshot` AWS CLI operasi dengan parameter berikut:

- `--cache-cluster-id`

- Jika cluster yang Anda cadangkan tidak memiliki node replika, `--cache-cluster-id` adalah nama cluster yang Anda cadangkan, misalnya. *`mycluster`*
- Jika klaster yang Anda cadangkan memiliki satu atau beberapa simpul replika, `--cache-cluster-id` adalah nama simpul di klaster yang dapat Anda gunakan untuk cadangan. Misalnya, namanya mungkin*`mycluster-002`*.

Gunakan parameter ini hanya saat mencadangkan cluster Valkey atau Redis OSS (mode cluster dinonaktifkan).

- `--replication-group-id`— Nama cluster Valkey atau Redis OSS (mode cluster enabled) (CLI/API: grup replikasi) untuk digunakan sebagai sumber cadangan. Gunakan parameter ini saat membuat cadangan cluster Valkey atau Redis OSS (mode cluster enabled).

- `--snapshot-name` – Nama snapshot yang akan dibuat.

  Batasan penamaan klaster adalah sebagai berikut:
  - Harus berisi 1–40 karakter alfanumerik atau tanda hubung.
  - Harus diawali dengan huruf.
  - Tidak boleh berisi dua tanda hubung berurutan.
  - Tidak boleh diakhiri dengan tanda hubung.

Contoh 1: Mencadangkan cluster Valkey atau Redis OSS (Cluster Mode Disabled) yang tidak memiliki node replika

AWS CLI Operasi berikut membuat cadangan `bkup-20150515` dari cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) `myNonClusteredRedis` yang tidak memiliki replika baca.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-snapshot \
    --cache-cluster-id myNonClusteredRedis \
    --snapshot-name bkup-20150515
```

Untuk Windows:

```
aws elasticache create-snapshot ^
```

```
    --cache-cluster-id myNonClusteredRedis ^
    --snapshot-name bkup-20150515
```

Contoh 2: Mencadangkan cluster Valkey atau Redis OSS (Cluster Mode Disabled) dengan node replika

AWS CLI Operasi berikut membuat cadangan `bkup-20150515` dari cluster Valkey atau Redis OSS (mode cluster dinonaktifkan). `myNonClusteredRedis` Cadangan ini memiliki satu atau beberapa replika baca.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-snapshot \
    --cache-cluster-id myNonClusteredRedis-001 \
    --snapshot-name bkup-20150515
```

Untuk Windows:

```
aws elasticache create-snapshot ^
    --cache-cluster-id myNonClusteredRedis-001 ^
    --snapshot-name bkup-20150515
```

Contoh Output: Mencadangkan Cluster Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) dengan Node Replika

Output dari operasi ini akan terlihat seperti berikut.

```
{
    "Snapshot": {
        "Engine": "redis",
        "CacheParameterGroupName": "default.redis6.x",
        "VpcId": "vpc-91280df6",
        "CacheClusterId": "myNonClusteredRedis-001",
        "SnapshotRetentionLimit": 0,
        "NumCacheNodes": 1,
        "SnapshotName": "bkup-20150515",
        "CacheClusterCreateTime": "2017-01-12T18:59:48.048Z",
        "AutoMinorVersionUpgrade": true,
        "PreferredAvailabilityZone": "us-east-1c",
        "SnapshotStatus": "creating",
        "SnapshotSource": "manual",
        "SnapshotWindow": "08:30-09:30",
```

```
        "EngineVersion": "6.0",
        "NodeSnapshots": [
            {
                "CacheSize": "",
                "CacheNodeId": "0001",
                "CacheNodeCreateTime": "2017-01-12T18:59:48.048Z"
            }
        ],
        "CacheSubnetGroupName": "default",
        "Port": 6379,
        "PreferredMaintenanceWindow": "wed:07:30-wed:08:30",
        "CacheNodeType": "cache.m3.2xlarge",
        "DataTiering": "disabled"
    }
}
```

Contoh 3: Mencadangkan cluster untuk Valkey atau Redis OSS (Mode Cluster Diaktifkan)

AWS CLI Operasi berikut membuat cadangan `bkup-20150515` dari cluster Valkey atau Redis OSS (mode cluster enabled). `myClusteredRedis` Perhatikan penggunaan `--replication-group-id`, bukan `--cache-cluster-id` untuk mengidentifikasi sumber. Perhatikan juga bahwa ElastiCache mengambil cadangan menggunakan node replika saat ada, dan akan default ke node utama jika node replika tidak tersedia.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-snapshot \
    --replication-group-id myClusteredRedis \
    --snapshot-name bkup-20150515
```

Untuk Windows:

```
aws elasticache create-snapshot ^
    --replication-group-id myClusteredRedis ^
    --snapshot-name bkup-20150515
```

Contoh Output: Mencadangkan Cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan)

Output dari operasi ini akan terlihat seperti berikut.

```
{
    "Snapshot": {
```

```
        "Engine": "redis",
        "CacheParameterGroupName": "default.redis6.x.cluster.on",
        "VpcId": "vpc-91280df6",
        "NodeSnapshots": [
            {
                "CacheSize": "",
                "NodeGroupId": "0001"
            },
            {
                "CacheSize": "",
                "NodeGroupId": "0002"
            }
        ],
        "NumNodeGroups": 2,
        "SnapshotName": "bkup-20150515",
        "ReplicationGroupId": "myClusteredRedis",
        "AutoMinorVersionUpgrade": true,
        "SnapshotRetentionLimit": 1,
        "AutomaticFailover": "enabled",
        "SnapshotStatus": "creating",
        "SnapshotSource": "manual",
        "SnapshotWindow": "10:00-11:00",
        "EngineVersion": "6.0",
        "CacheSubnetGroupName": "default",
        "ReplicationGroupDescription": "2 shards 2 nodes each",
        "Port": 6379,
        "PreferredMaintenanceWindow": "sat:03:30-sat:04:30",
        "CacheNodeType": "cache.r3.large",
        "DataTiering": "disabled"
    }
}
```

Topik terkait

Untuk informasi selengkapnya, lihat create-snapshot dalam Referensi Perintah AWS CLI .

Membuat cadangan menggunakan AWS CloudFormation

Anda dapat menggunakan AWS CloudFormation untuk membuat cadangan ElastiCache Redis OSS atau Valkey cache Anda, menggunakan properti atau. `AWS::ElastiCache::ServerlessCache` `AWS::ElastiCache::ReplicationGroup`

Menggunakan sumber **`AWS::ElastiCache::ServerlessCache`** daya

Gunakan ini untuk membuat cadangan menggunakan AWS::ElastiCache::ServerlessCache sumber daya:

```
Resources:
                    iotCatalog:
                        Type: AWS::ElastiCache::ServerlessCache
                            Properties:
                            ...
                            ServerlessCacheName: "your-cache-name"
                            Engine: "redis"
                            CacheUsageLimits
```

Menggunakan sumber AWS::ElastiCache::ReplicationGroup daya

Gunakan sumber `AWS::ElastiCache::ReplicationGroup` daya:

```
Resources:
                    iotCatalog:
                        Type: AWS::ElastiCache::ReplicationGroup
                            Properties:
                            ...
                            ReplicationGroupDescription: "Description of your
 replication group"
                            Engine: "redis"
                            CacheNodeType
                            NumCacheClusters
                            AutomaticFailoverEnabled
                            AtRestEncryptionEnabled
```

Membuat cadangan akhir

Anda dapat membuat cadangan akhir menggunakan ElastiCache konsol, file AWS CLI, atau ElastiCache API.

Membuat cadangan akhir (konsol)

Anda dapat membuat cadangan akhir saat menghapus cache tanpa server Valkey, Memcached, atau Redis OSS, atau cluster yang dirancang sendiri Valkey atau Redis OSS, dengan menggunakan konsol. ElastiCache

Untuk membuat cadangan akhir saat menghapus cache, pada kotak dialog hapus pilih Ya di bawah Buat cadangan dan beri nama cadangan.

Topik terkait

- Menggunakan AWS Management Console
- Menghapus Grup Replikasi (Konsol)

Membuat cadangan akhir (AWS CLI)

Anda dapat membuat cadangan akhir saat menghapus cache menggunakan file. AWS CLI

Topik

- Saat menghapus cache Valkey, cache tanpa server Memcached, atau cache Redis OSS
- Saat menghapus cluster Valkey atau Redis OSS tanpa replika baca
- Saat menghapus cluster Valkey atau Redis OSS dengan replika baca

Saat menghapus cache Valkey, cache tanpa server Memcached, atau cache Redis OSS

Untuk membuat cadangan akhir, gunakan `delete-serverless-cache` AWS CLI operasi dengan parameter berikut.

- `--serverless-cache-name` – Nama cache yang dihapus.
- `--final-snapshot-name` – Nama cadangan.

Kode berikut membuat cadangan akhir `bkup-20231127-final` saat menghapus cache `myserverlesscache`.

Untuk Linux, macOS, atau Unix:

```
aws elasticache delete-serverless-cache \
        --serverless-cache-name myserverlesscache \
        --final-snapshot-name bkup-20231127-final
```

Untuk Windows:

```
aws elasticache delete-serverless-cache ^
        --serverless-cache-name myserverlesscache ^
        --final-snapshot-name bkup-20231127-final
```

Untuk informasi selengkapnya, lihat delete-serverless-cache dalam AWS CLI Referensi Perintah.

Saat menghapus cluster Valkey atau Redis OSS tanpa replika baca

Untuk membuat cadangan akhir untuk cluster yang dirancang sendiri tanpa replika baca, gunakan `delete-cache-cluster` AWS CLI operasi dengan parameter berikut.

- `--cache-cluster-id` – Nama klaster yang dihapus.
- `--final-snapshot-identifier` – Nama cadangan.

Kode berikut membuat cadangan akhir `bkup-20150515-final` saat menghapus klaster `myRedisCluster`.

Untuk Linux, macOS, atau Unix:

```
aws elasticache delete-cache-cluster \
        --cache-cluster-id myRedisCluster \
        --final-snapshot-identifier bkup-20150515-final
```

Untuk Windows:

```
aws elasticache delete-cache-cluster ^
        --cache-cluster-id myRedisCluster ^
        --final-snapshot-identifier bkup-20150515-final
```

Untuk informasi selengkapnya, lihat delete-cache-cluster dalam AWS CLI Referensi Perintah.

Saat menghapus cluster Valkey atau Redis OSS dengan replika baca

Untuk membuat cadangan akhir saat menghapus grup replikasi, gunakan `delete-replication-group` AWS CLI operasi, dengan parameter berikut:

- `--replication-group-id` – Nama grup replikasi yang dihapus.
- `--final-snapshot-identifier` – Nama cadangan akhir.

Kode berikut membuat cadangan akhir `bkup-20150515-final` saat menghapus grup replikasi `myReplGroup`.

Untuk Linux, macOS, atau Unix:

```
aws elasticache delete-replication-group \
        --replication-group-id myReplGroup \
        --final-snapshot-identifier bkup-20150515-final
```

Untuk Windows:

```
aws elasticache delete-replication-group ^
        --replication-group-id myReplGroup ^
        --final-snapshot-identifier bkup-20150515-final
```

Untuk informasi selengkapnya, lihat delete-replication-group dalam AWS CLI Referensi Perintah.

Menjelaskan cadangan

Prosedur berikut menunjukkan cara menampilkan daftar cadangan Anda. Jika ingin, Anda juga dapat melihat detail cadangan tertentu.

Mendeskripsikan cadangan (Konsol)

Untuk menampilkan cadangan menggunakan AWS Management Console

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih Cadangan.
3. Untuk melihat detail cadangan tertentu, pilih kotak di sebelah kiri nama cadangan.

Menjelaskan cadangan nirserver (AWS CLI)

Untuk menampilkan daftar cadangan nirserver, dan detail tentang cadangan tertentu (opsional), gunakan operasi CLI `describe-serverless-cache-snapshots`.

Contoh

Operasi berikut menggunakan parameter `--max-records` untuk menampilkan hingga 20 cadangan yang terkait dengan akun Anda. Menghilangkan parameter `--max-records` akan menampilkan hingga 50 cadangan.

```
aws elasticache describe-serverless-cache-snapshots --max-records 20
```

Operasi berikut menggunakan parameter `--serverless-cache-name` untuk hanya menampilkan cadangan yang terkait dengan cache `my-cache`.

```
aws elasticache describe-serverless-cache-snapshots --serverless-cache-name my-cache
```

Operasi berikut menggunakan parameter `--serverless-cache-snapshot-name` untuk menampilkan detail cadangan `my-backup`.

```
aws elasticache describe-serverless-cache-snapshots --serverless-cache-snapshot-
name my-backup
```

Untuk informasi selengkapnya, lihat describe-serverless-cache-snapshotsdi Referensi AWS CLI Perintah.

Menjelaskan cadangan klaster yang dirancang sendiri (AWS CLI)

Untuk menampilkan daftar cadangan klaster yang dirancang sendiri dan, detail tentang cadangan tertentu (opsional), gunakan operasi CLI `describe-snapshots`.

Contoh

Operasi berikut menggunakan parameter `--max-records` untuk menampilkan hingga 20 cadangan yang terkait dengan akun Anda. Menghilangkan parameter `--max-records` akan menampilkan hingga 50 cadangan.

```
aws elasticache describe-snapshots --max-records 20
```

Operasi berikut menggunakan parameter `--cache-cluster-id` untuk menampilkan hanya cadangan yang terkait dengan klaster `my-cluster`.

```
aws elasticache describe-snapshots --cache-cluster-id my-cluster
```

Operasi berikut menggunakan parameter `--snapshot-name` untuk menampilkan detail cadangan `my-backup`.

```
aws elasticache describe-snapshots --snapshot-name my-backup
```

Untuk informasi selengkapnya, lihat deskripsi-snapshot di Referensi Perintah. AWS CLI

Menyalin cadangan

Anda dapat membuat salinan cadangan apa pun, baik dibuat secara otomatis maupun manual. Anda juga dapat mengekspor cadangan Anda sehingga Anda dapat mengaksesnya dari luar ElastiCache. Untuk panduan tentang cara mengekspor cadangan, lihat Mengekspor cadangan.

Langkah-langkah berikut menunjukkan cara menyalin cadangan.

Menyalin cadangan (Konsol)

Untuk menyalin cadangan (konsol)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Untuk melihat daftar cadangan Anda, dari panel navigasi sebelah kiri, pilih Cadangan.
3. Dari daftar cadangan, pilih kotak di sebelah kiri nama cadangan yang ingin Anda salin.
4. Pilih Tindakan lalu Salin.
5. Pada kotak Nama cadangan baru, ketikkan nama untuk cadangan baru Anda.
6. Pilih Salin.

Menyalin cadangan nirserver (AWS CLI)

Untuk menyalin cadangan cache nirserver, gunakan operasi `copy-serverless-cache-snapshot`.

Parameter

- `--source-serverless-cache-snapshot-name` – Nama cadangan yang akan disalin.
- `--target-serverless-cache-snapshot-name` – Nama salinan cadangan.

Contoh berikut membuat salinan dari cadangan otomatis.

Untuk Linux, macOS, atau Unix:

```
aws elasticache copy-serverless-cache-snapshot \
    --source-serverless-cache-snapshot-name automatic.my-cache-2023-11-27-03-15 \
    --target-serverless-cache-snapshot-name my-backup-copy
```

Untuk Windows:

```
aws elasticache copy-serverless-cache-snapshot ^
    --source-serverless-cache-snapshot-name automatic.my-cache-2023-11-27-03-15 ^
    --target-serverless-cache-snapshot-name my-backup-copy
```

Untuk informasi selengkapnya, lihat copy-serverless-cache-snapshot di AWS CLI.

Menyalin cadangan klaster yang dirancang sendiri (AWS CLI)

Untuk menyalin cadangan klaster yang dirancang sendiri, gunakan operasi `copy-snapshot`.

Parameter

- `--source-snapshot-name` – Nama cadangan yang akan disalin.
- `--target-snapshot-name` – Nama salinan cadangan.
- `--target-bucket` – Dikhususkan untuk mengekspor cadangan. Jangan menggunakan parameter ini saat membuat salinan cadangan. Untuk informasi selengkapnya, lihat Mengekspor cadangan.

Contoh berikut membuat salinan dari cadangan otomatis.

Untuk Linux, macOS, atau Unix:

```
aws elasticache copy-snapshot  \
    --source-snapshot-name automatic.my-redis-primary-2014-03-27-03-15 \
    --target-snapshot-name amzn-s3-demo-bucket
```

Untuk Windows:

```
aws elasticache copy-snapshot ^
    --source-snapshot-name automatic.my-redis-primary-2014-03-27-03-15 ^
    --target-snapshot-name amzn-s3-demo-bucket
```

Untuk informasi selengkapnya, lihat copy-snapshot di AWS CLI.

Mengekspor cadangan

Amazon ElastiCache mendukung ekspor cadangan Redis OSS Anda ElastiCache ke bucket Amazon Simple Storage Service (Amazon S3), yang memberi Anda akses ke sana dari luar. ElastiCache Anda dapat mengekspor cadangan menggunakan ElastiCache konsol, file AWS CLI, atau ElastiCache API.

Mengekspor cadangan dapat membantu jika Anda perlu meluncurkan cluster di AWS Wilayah lain. Anda dapat mengekspor data Anda dalam satu AWS Wilayah, menyalin file.rdb ke AWS Wilayah baru, dan kemudian menggunakan file.rdb itu untuk menyemai cache baru alih-alih menunggu cluster baru diisi melalui penggunaan. Untuk informasi tentang melakukan seeding klaster baru, lihat Tutorial: Menyemai cluster baru yang dirancang sendiri dengan cadangan yang dibuat secara eksternal. Alasan lain Anda mungkin ingin mengekspor data cache Anda adalah dengan menggunakan file.rdb untuk pemrosesan offline.

Important

- ElastiCache Cadangan dan ember Amazon S3 yang ingin Anda salin harus berada di Wilayah yang sama AWS .

  Meskipun cadangan disalin ke bucket Amazon S3 dalam keadaan terenkripsi, sebaiknya jangan memberi orang lain akses ke bucket Amazon S3 tempat Anda ingin menyimpan cadangan Anda.
- Mengekspor cadangan ke Amazon S3 tidak didukung untuk klaster yang menggunakan tingkatan data. Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.
- Mengekspor cadangan tersedia untuk cluster yang dirancang sendiri Valkey dan Redis OSS, Valkey Tanpa Server dan Redis OSS, dan Memcached Tanpa Server. Mengekspor cadangan tidak tersedia untuk cluster Memcached yang dirancang sendiri.

Sebelum Anda dapat mengekspor cadangan ke bucket Amazon S3, Anda harus memiliki bucket Amazon S3 di Wilayah AWS yang sama dengan cadangan. Berikan ElastiCache akses ke ember. Dua langkah pertama menunjukkan cara melakukannya.

Buat bucket Amazon S3.

Langkah-langkah berikut menggunakan konsol Amazon S3 untuk membuat bucket Amazon S3 tempat Anda mengekspor dan menyimpan cadangan. ElastiCache

Untuk membuat bucket Amazon S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di. https://console.aws.amazon.com/s3/
2. Pilih Buat Bucket.
3. Di Buat Bucket - Pilih Nama Bucket dan Wilayah, lakukan hal berikut:

    a. Di Nama Bucket, ketikkan nama untuk bucket Amazon S3 Anda.

    Nama bucket Amazon S3 Anda harus sesuai dengan persyaratan DNS. Jika tidak, tidak ElastiCache dapat mengakses file cadangan Anda. Aturan untuk kesesuaian DNS adalah:

    - Nama harus minimal 3 dan tidak lebih dari 63 karakter.
    - Nama harus serangkaian satu atau beberapa label yang dipisahkan oleh titik (.) dengan setiap label:
        - Dimulai dengan huruf kecil atau angka.
        - Diakhiri dengan huruf kecil atau angka.
        - Hanya berisi huruf kecil, angka, dan tanda hubung.
    - Nama tidak dapat diformat sebagai alamat IP (misalnya, 192.0.2.0).

    b. Dari daftar Wilayah, pilih AWS Wilayah untuk bucket Amazon S3 Anda. AWS Wilayah ini harus AWS Wilayah yang sama dengan ElastiCache cadangan yang ingin Anda ekspor.

    c. Pilih Buat.

Untuk informasi selengkapnya tentang cara membuat bucket Amazon S3, lihat Membuat bucket dalam Panduan Pengguna Amazon Simple Storage Service.

Berikan ElastiCache akses ke bucket Amazon S3 Anda

ElastiCache Agar dapat menyalin snapshot ke bucket Amazon S3, Anda harus memperbarui kebijakan bucket IAM untuk ElastiCache memberikan akses ke bucket.

> **Warning**
>
> Meskipun cadangan yang disalin ke bucket Amazon S3 sudah terenkripsi, data Anda dapat diakses oleh siapa saja dengan akses ke bucket Amazon S3 Anda. Oleh karena itu,

sebaiknya siapkan kebijakan IAM untuk mencegah akses tidak sah ke bucket Amazon S3 ini. Untuk informasi selengkapnya, lihat Mengelola akses dalam Panduan Pengguna Amazon S3.

Untuk membuat izin yang tepat di bucket Amazon S3, lakukan langkah-langkah yang dijelaskan sebagai berikut.

Untuk memberikan ElastiCache akses ke bucket S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di. https://console.aws.amazon.com/s3/
2. Pilih nama bucket Amazon S3 tempat cadangan akan disalin. Bucket tersebut harus berupa bucket S3 yang Anda buat di Buat bucket Amazon S3..
3. Pilih tab Izin dan di bagian Izin, pilih Daftar kontrol akses (ACL), lalu pilih Edit.
4. Tambahkan ID Kanonis penerima izin `540804c33a284a299d2547575ce1010f2312ef3da9b3a053c8bc45bf233e4353` dengan opsi berikut:
   - Objek: Daftar, Tulis
   - Bucket ACL: Baca, Tulis

   > **Note**
   > - Untuk GovCloud Wilayah PDT, Id Kanonik adalah. `40fa568277ad703bd160f66ae4f83fc9dfdfd06c2f1b5060ca22442ac3ef8be6`
   > - Untuk GovCloud Wilayah OSU, Id Kanonik adalah. `c54286759d2a83da9c480405349819c993557275cf37d820d514b42da6893f5c`
5. Pilih Simpan.

Ekspor ElastiCache cadangan

Sekarang Anda telah membuat bucket S3 dan memberikan ElastiCache izin untuk mengaksesnya. Selanjutnya, Anda dapat menggunakan ElastiCache konsol, AWS CLI, atau ElastiCache API untuk mengekspor snapshot Anda ke sana.

Berikut adalah contoh tampilan kebijakan yang sudah diperbarui.

JSON

```
{
    "Version": "2012-10-17",
    "Id": "Policy15397346",
    "Statement": [
        {
            "Sid": "Stmt15399484",
            "Effect": "Allow",

            "Principal": {
                "Service": "ap-east-1.elasticache-snapshot.amazonaws.com"
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::hkg-elasticache-backup",
                "arn:aws:s3:::hkg-elasticache-backup/*"
            ]
        }
    ]
}
```

Untuk Wilayah keikutsertaan, berikut ini adalah contoh seperti apa kebijakan IAM yang diperbarui untuk bucket S3. (Contoh ini menggunakan Wilayah Asia Pasifik (Hong Kong).)

JSON

```
{
    "Version": "2012-10-17",
    "Id": "Policy15397346",
    "Statement": [
        {
            "Sid": "Stmt15399483",
            "Effect": "Allow",
            "Principal": {
                "Service": "elasticache.amazonaws.com"
            },
            "Action": "s3:*",
```

```
            "Resource": [
                "arn:aws:s3:::hkg-elasticache-backup",
                "arn:aws:s3:::hkg-elasticache-backup/*"
            ]
        },
        {
            "Sid": "Stmt15399484",
            "Effect": "Allow",

            "Principal": {
                "Service": "ap-east-1.elasticache-snapshot.amazonaws.com"
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::hkg-elasticache-backup",
                "arn:aws:s3:::hkg-elasticache-backup/*"
            ]
        }
    ]
}
```

Mengekspor ElastiCache cadangan (Konsol)

Langkah-langkah berikut menggunakan ElastiCache konsol untuk mengekspor cadangan ke bucket Amazon S3 sehingga Anda dapat mengaksesnya dari luar. ElastiCache Bucket Amazon S3 harus berada di AWS Wilayah yang sama dengan cadangan. ElastiCache

Untuk mengekspor ElastiCache cadangan ke bucket Amazon S3

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Untuk melihat daftar cadangan Anda, dari panel navigasi sebelah kiri, pilih Cadangan.
3. Dari daftar cadangan, pilih kotak di sebelah kiri nama cadangan yang ingin Anda ekspor.
4. Pilih Salin.
5. Di Buat Salinan Cadangan?, lakukan hal berikut:

    a. Di kotak Nama cadangan baru, ketikkan nama untuk cadangan baru Anda.

    Nama itu harus di antara 1 dan 1.000 karakter serta dapat dienkode ke UTF-8.

ElastiCache menambahkan pengidentifikasi instance dan nilai `.rdb` yang Anda masukkan di sini. Misalnya, jika Anda memasukkan `my-exported-backup`, ElastiCache membuat `my-exported-backup-0001.rdb`.

b. Dari daftar Lokasi S3 Target, pilih nama bucket Amazon S3 yang menjadi tujuan Anda menyalin cadangan (bucket yang telah Anda buat di Buat bucket Amazon S3.).

   Lokasi Target S3 harus berupa bucket Amazon S3 di Wilayah AWS cadangan dengan izin berikut agar proses ekspor berhasil.

   - Akses objek – Baca dan Tulis.
   - Akses izin – Baca.

   Untuk informasi selengkapnya, lihat Berikan ElastiCache akses ke bucket Amazon S3 Anda.

c. Pilih Salin.

> **Note**
>
> Jika bucket S3 Anda tidak memiliki izin yang diperlukan untuk mengekspor cadangan ElastiCache ke sana, Anda menerima salah satu pesan galat berikut. Kembali ke Berikan ElastiCache akses ke bucket Amazon S3 Anda untuk menambahkan izin yang ditentukan dan mencoba mengekspor cadangan Anda kembali.
>
> - ElastiCache belum diberikan izin BACA %s pada Bucket S3.
>
>   Solusi: Tambahkan izin Baca pada bucket.
> - ElastiCache belum diberikan izin WRITE %s pada Bucket S3.
>
>   Solusi: Tambahkan izin Tulis pada bucket.
> - ElastiCache belum diberikan izin READ_ACP %s pada Bucket S3.
>
>   Solusi: Tambahkan Baca untuk akses Izin pada bucket.

Jika Anda ingin menyalin cadangan Anda ke AWS Wilayah lain, gunakan Amazon S3 untuk menyalinnya. Untuk informasi selengkapnya, lihat Menyalin objek dalam Panduan Pengguna Amazon Simple Storage.

Mengekspor cadangan ElastiCache tanpa server ()AWS CLI

Mengekspor cadangan cache nirserver

Ekspor cadangan ke bucket Amazon S3 menggunakan operasi CLI `export-serverless-cache-snapshot` dengan parameter berikut:

Parameter

- `--serverless-cache-snapshot-name` – Nama cadangan yang akan disalin.
- `--s3-bucket-name` – Nama bucket Amazon S3 tempat tujuan Anda mengekspor cadangan. Salinan cadangan dibuat dalam bucket yang ditentukan.

  `--s3-bucket-name`Harus berupa bucket Amazon S3 di AWS Wilayah cadangan dengan izin berikut agar proses ekspor berhasil.

  - Akses objek – Baca dan Tulis.
  - Akses izin – Baca.

Operasi berikut menyalin cadangan ke my-s3-bucket.

Untuk Linux, macOS, atau Unix:

```
aws elasticache export-serverless-cache-snapshot \
    --serverless-cache-snapshot-name automatic.my-redis-2023-11-27 \
    --s3-bucket-name my-s3-bucket
```

Untuk Windows:

```
aws elasticache export-serverless-cache-snapshot ^
    --serverless-cache-snapshot-name automatic.my-redis-2023-11-27 ^
    --s3-bucket-name my-s3-bucket
```

Mengekspor cadangan ElastiCache cluster yang dirancang sendiri ()AWS CLI

Mengekspor cadangan klaster yang dirancang sendiri

Ekspor cadangan ke bucket Amazon S3 menggunakan operasi CLI `copy-snapshot` dengan parameter berikut:

Parameter

- `--source-snapshot-name` – Nama cadangan yang akan disalin.
- `--target-snapshot-name` – Nama salinan cadangan.

  Nama tersebut harus berisi antara 1 dan 1.000 karakter serta dienkode dengan UTF-8.

  ElastiCache menambahkan pengidentifikasi instance dan nilai `.rdb` yang Anda masukkan di sini. Misalnya, jika Anda memasukkan `my-exported-backup`, ElastiCache membuat `my-exported-backup-0001.rdb`.
- `--target-bucket` – Nama bucket Amazon S3 tempat tujuan Anda mengekspor cadangan. Salinan cadangan dibuat dalam bucket yang ditentukan.

  `--target-bucket`Harus berupa bucket Amazon S3 di AWS Wilayah cadangan dengan izin berikut agar proses ekspor berhasil.
  - Akses objek – Baca dan Tulis.
  - Akses izin – Baca.

  Untuk informasi selengkapnya, lihat Berikan ElastiCache akses ke bucket Amazon S3 Anda.

Operasi berikut menyalin cadangan ke my-s3-bucket.

Untuk Linux, macOS, atau Unix:

```
aws elasticache copy-snapshot \
    --source-snapshot-name automatic.my-redis-primary-2016-06-27-03-15 \
    --target-snapshot-name my-exported-backup \
    --target-bucket my-s3-bucket
```

Untuk Windows:

```
aws elasticache copy-snapshot ^
    --source-snapshot-name automatic.my-redis-primary-2016-06-27-03-15 ^
    --target-snapshot-name my-exported-backup ^
    --target-bucket my-s3-bucket
```

Melakukan pemulihan dari cadangan ke dalam cache baru

Anda dapat memulihkan cadangan yang ada dari Valkey ke cache Valkey baru atau cluster yang dirancang sendiri, dan memulihkan cadangan Redis OSS yang ada ke cache Redis OSS baru atau cluster yang dirancang sendiri. Anda juga dapat memulihkan cadangan cache tanpa server Memcached yang ada ke cache tanpa server Memcached yang baru.

Memulihkan cadangan ke dalam cache nirserver (Konsol)

> **Note**
>
> ElastiCache Tanpa server mendukung file RDB yang kompatibel dengan Valkey 7.2 ke atas, dan versi Redis OSS antara 5.0 dan versi terbaru yang tersedia.

Untuk memulihkan cadangan ke cache nirserver (konsol)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih Cadangan.
3. Dalam daftar cadangan, pilih kotak di sebelah kiri nama cadangan yang ingin Anda pulihkan.
4. Pilih Tindakan, lalu pilih Pulihkan.
5. Masukkan nama untuk cache nirserver baru dan deskripsi opsional.
6. Klik Buat untuk membuat cache baru dan mengimpor data dari cadangan Anda.

Memulihkan cadangan ke dalam klaster yang dirancang sendiri (Konsol)

Untuk memulihkan cadangan ke klaster yang dirancang sendiri (konsol)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih Cadangan.
3. Di daftar cadangan, pilih kotak di sebelah kiri nama cadangan yang ingin Anda pulihkan.
4. Pilih Tindakan, lalu pilih Pulihkan.
5. Pilih Rancang cache sendiri dan sesuaikan pengaturan klaster, seperti jenis simpul, ukuran, jumlah serpihan, replika, penempatan AZ, dan pengaturan keamanan.

6. Pilih Buat untuk membuat cache baru dan mengimpor data dari cadangan Anda.

Memulihkan cadangan ke dalam cache nirserver (AWS CLI)

> **Note**
>
> ElastiCache Tanpa server mendukung file RDB yang kompatibel dengan Valkey 7.2 ke atas, dan versi Redis OSS antara 5.0 dan versi terbaru yang tersedia.

Untuk memulihkan cadangan ke cache nirserver baru (AWS CLI)

AWS CLI Contoh berikut membuat cache baru menggunakan `create-serverless-cache` dan mengimpor data dari cadangan.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-serverless-cache \

    --serverless-cache-name CacheName \
    --engine redis
    --snapshot-arns-to-restore Snapshot-ARN
```

Untuk Windows:

```
aws elasticache create-serverless-cache ^

    --serverless-cache-name CacheName ^
    --engine redis ^
    --snapshot-arns-to-restore Snapshot-ARN
```

Memulihkan cadangan ke dalam klaster yang dirancang sendiri (AWS CLI)

Untuk memulihkan cadangan ke klaster yang dirancang sendiri (AWS CLI)

Anda dapat mengembalikan cadangan Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan dua cara.

-
```
aws elasticache create-serverless-cache \
```

```
    --serverless-cache-name CacheName \
    --engine redis
    --snapshot-arns-to-restore Snapshot-ARN
```

- Untuk Windows:

```
aws elasticache create-serverless-cache ^
    --serverless-cache-name CacheName ^
    --engine redis ^
    --snapshot-arns-to-restore Snapshot-ARN
```

Memulihkan cadangan ke dalam klaster yang dirancang sendiri (AWS CLI)

Untuk memulihkan cadangan ke klaster yang dirancang sendiri (AWS CLI)

Anda dapat memulihkan cadangan cache tanpa server Valkey atau Redis OSS, dan Anda juga dapat memulihkan cluster Valkey atau Redis OSS yang dirancang sendiri.

Anda dapat memulihkan cadangan cache tanpa server Valkey atau Redis OSS dengan dua cara.

- Anda dapat mengembalikan ke cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) node tunggal menggunakan operasi. AWS CLI create-cache-cluster
- Anda dapat mengembalikan ke cluster Valkey atau Redis OSS dengan replika baca (grup replikasi). Untuk melakukan ini, Anda dapat menggunakan Valkey atau Redis OSS (mode cluster dinonaktifkan) atau Valkey atau Redis OSS (mode cluster diaktifkan) dengan operasi. AWS CLI create-replication-group Dalam hal ini, Anda menyemai pemulihan dengan file Valkey atau Redis OSS.rdb. Untuk informasi selengkapnya tentang seeding klaster baru yang dirancang sendiri, lihat Tutorial: Menyemai cluster baru yang dirancang sendiri dengan cadangan yang dibuat secara eksternal.

Anda dapat mengembalikan cadangan Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan dua cara.

- Anda dapat mengembalikan ke cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) node tunggal menggunakan operasi. AWS CLI create-cache-cluster
- Anda dapat mengembalikan ke cluster Valkey atau Redis OSS dengan replika baca (grup replikasi). Untuk melakukan ini, Anda dapat menggunakan Valkey atau Redis OSS (mode cluster dinonaktifkan) atau Valkey atau Redis OSS (mode cluster diaktifkan) dengan operasi. AWS CLI

create-replication-group Dalam hal ini, Anda menyemai pemulihan dengan file Valkey atau Redis OSS.rdb. Untuk informasi selengkapnya tentang seeding klaster baru yang dirancang sendiri, lihat Tutorial: Menyemai cluster baru yang dirancang sendiri dengan cadangan yang dibuat secara eksternal.

Saat menggunakan operasi `create-cache-cluster` atau `create-replication-group`, pastikan untuk memasukkan parameter `--snapshot-name` atau `--snapshot-arn` untuk melakukan seeding klaster atau grup replikasi baru dengan data dari cadangan.

Menghapus cadangan

Cadangan otomatis akan dihapus secara otomatis jika batas retensinya berakhir. Jika Anda menghapus klaster, semua cadangan otomatis juga dihapus. Jika Anda menghapus grup replikasi, semua cadangan otomatis dari klaster di grup tersebut juga dihapus.

ElastiCache menyediakan operasi penghapusan API yang memungkinkan Anda menghapus cadangan kapan saja, terlepas dari apakah cadangan dibuat secara otomatis atau manual. Karena cadangan manual tidak memiliki batas retensi, penghapusan manual adalah satu-satunya cara untuk menghapusnya.

Anda dapat menghapus cadangan menggunakan ElastiCache konsol, file AWS CLI, atau ElastiCache API.

Menghapus cadangan (Konsol)

Prosedur berikut menghapus cadangan menggunakan ElastiCache konsol.

Untuk menghapus cadangan

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Cadangan.

   Layar Cadangan muncul dengan daftar cadangan Anda.
3. Pilih kotak di sebelah kiri nama cadangan yang ingin Anda hapus.
4. Pilih Hapus.
5. Jika Anda ingin menghapus cadangan ini, pilih Hapus di layar konfirmasi Hapus Cadangan. Status berubah menjadi menghapus.

Menghapus cadangan nirserver (AWS CLI)

Gunakan AWS CLI operasi delete-snapshot dengan parameter berikut untuk menghapus cadangan tanpa server.

- `--serverless-cache-snapshot-name` – Nama cadangan yang akan dihapus.

Kode berikut menghapus cadangan `myBackup`.

```
aws elasticache delete-serverless-cache-snapshot --serverless-cache-snapshot-
name myBackup
```

Untuk informasi selengkapnya, lihat delete-serverless-cache-snapshot dalam AWS CLI Referensi Perintah.

Menghapus cadangan klaster yang dirancang sendiri (AWS CLI)

Gunakan AWS CLI operasi delete-snapshot dengan parameter berikut untuk menghapus cadangan cluster yang dirancang sendiri.

- `--snapshot-name` – Nama cadangan yang akan dihapus.

Kode berikut menghapus cadangan `myBackup`.

```
aws elasticache delete-snapshot --snapshot-name myBackup
```

Untuk informasi selengkapnya, lihat delete-snapshot dalam Referensi Perintah AWS CLI .

Memberikan tag pada cadangan

Anda dapat memberikan metadata Anda sendiri ke setiap cadangan dalam bentuk tag. Tag memungkinkan Anda mengategorikan cadangan dengan berbagai cara, misalnya, berdasarkan tujuan, pemilik, atau lingkungan. Hal ini berguna ketika Anda memiliki banyak sumber daya dengan jenis yang sama—Anda dapat dengan cepat mengidentifikasi sumber daya tertentu berdasarkan tag yang telah Anda tetapkan. Untuk informasi selengkapnya, lihat Sumber daya yang dapat Anda beri tag.

Tag alokasi biaya adalah sarana untuk melacak biaya Anda di beberapa AWS layanan dengan mengelompokkan pengeluaran Anda pada faktur berdasarkan nilai tag. Untuk mempelajari selengkapnya tentang tag alokasi biaya, lihat Menggunakan tag alokasi biaya.

Dengan menggunakan ElastiCache konsol, ElastiCache API AWS CLI, atau Anda dapat menambahkan, membuat daftar, memodifikasi, menghapus, atau menyalin tag alokasi biaya pada cadangan Anda. Lihat informasi yang lebih lengkap di Memantau biaya dengan tag alokasi biaya.

Tutorial: Menyemai cluster baru yang dirancang sendiri dengan cadangan yang dibuat secara eksternal

Saat Anda membuat cluster Valkey atau Redis OSS yang dirancang sendiri, Anda dapat menyemai dengan data dari file cadangan Valkey atau Redis OSS.rdb. Penyemaian cluster berguna jika saat ini Anda mengelola instans Valkey atau Redis OSS di luar ElastiCache dan ingin mengisi cluster baru Anda untuk Redis OSS yang dirancang sendiri dengan ElastiCache data Valkey atau Redis OSS yang ada.

Untuk menyemai cluster Valkey atau Redis OSS baru yang dirancang sendiri dari cadangan Valkey atau Redis OSS yang dibuat di Amazon, lihat. ElastiCache Melakukan pemulihan dari cadangan ke dalam cache baru

Saat Anda menggunakan file Valkey atau Redis OSS.rdb untuk menyemai cluster baru yang dirancang sendiri, Anda dapat melakukan hal berikut:

- Tingkatkan dari cluster yang tidak dipartisi ke cluster yang dirancang sendiri Valkey atau Redis OSS (mode cluster diaktifkan) yang menjalankan Redis OSS versi 3.2.4.
- Tentukan jumlah serpihan (disebut grup simpul di API dan CLI) di klaster yang dirancang sendiri. Jumlah ini dapat berbeda dari jumlah serpihan di klaster yang dirancang sendiri yang digunakan untuk membuat file cadangan.
- Tentukan jenis simpul yang berbeda untuk klaster yang dirancang sendiri – lebih besar atau lebih kecil dari yang digunakan di klaster yang membuat cadangan. Jika Anda menskalakan ke tipe node yang lebih kecil, pastikan bahwa tipe node baru memiliki memori yang cukup untuk data Anda dan overhead Valkey atau Redis OSS. Untuk informasi selengkapnya, lihat Memastikan Anda memiliki cukup memori untuk membuat snapshot Valkey atau Redis OSS.
- Bagikan kunci Anda di slot cluster Valkey atau Redis OSS (mode cluster diaktifkan) baru secara berbeda dari pada cluster yang digunakan untuk membuat file cadangan.

> **Note**
>
> Anda tidak dapat menyemai klaster Valkey atau Redis OSS (mode cluster dinonaktifkan) dari file.rdb yang dibuat dari cluster Valkey atau Redis OSS (mode cluster enabled).

Important

- Anda harus memastikan bahwa data cadangan Valkey atau Redis OSS Anda tidak melebihi sumber daya node. Misalnya, Anda tidak dapat mengunggah file.rdb dengan 5 GB data Valkey atau Redis OSS ke node cache.m3.medium yang memiliki memori 2,9 GB.

  Jika cadangan terlalu besar, klaster yang dihasilkan akan memiliki status `restore-failed`. Jika hal ini terjadi, Anda harus menghapus klaster tersebut dan memulai dari awal.

  Untuk daftar lengkap jenis dan spesifikasi node, lihat Parameter spesifik tipe node Redis OSS serta fitur dan detail ElastiCache produk Amazon.
- Anda dapat mengenkripsi file Valkey atau Redis OSS.rdb dengan enkripsi sisi server Amazon S3 (SSE-S3) saja. Untuk informasi selengkapnya, lihat Melindungi data menggunakan enkripsi sisi server.

Berikut ini, Anda dapat menemukan topik yang memandu Anda melalui migrasi cluster Anda dari luar ElastiCache untuk Valkey atau Redis OSS ke ElastiCache Redis OSS.

Bermigrasi ke ElastiCache untuk Redis OSS

- Langkah 1: Buat cadangan Valkey atau Redis OSS
- Langkah 2: Buat folder dan bucket Amazon S3
- Langkah 3: Unggah cadangan Anda ke Amazon S3
- Langkah 4: Berikan akses ElastiCache baca ke file.rdb

Bermigrasi dari layanan eksternal ke ElastiCache Redis OSS.

- Langkah 1: Buat cadangan Valkey atau Redis OSS
- Langkah 2: Buat folder dan bucket Amazon S3
- Langkah 3: Unggah cadangan Anda ke Amazon S3
- Langkah 4: Berikan akses ElastiCache baca ke file.rdb

Langkah 1: Buat cadangan Valkey atau Redis OSS

Untuk membuat cadangan Valkey atau Redis OSS untuk menyemai instans Redis OSS Anda ElastiCache

1. Connect ke instans Valkey atau Redis OSS yang ada.
2. Jalankan salah satu `BGSAVE` atau `SAVE` operasi untuk membuat cadangan. Catat tempat file .rdb Anda berada.

   `BGSAVE` bersifat asinkron dan tidak memblokir klien lain saat melakukan pemrosesan. Untuk informasi lebih lanjut, lihat BGSAVE di situs web Valkey.

   `SAVE` bersifat sinkron dan memblokir proses lainnya hingga selesai. Untuk informasi lebih lanjut, lihat SIMPAN di situs web Valkey.

Untuk informasi tambahan tentang membuat cadangan, lihat Persistence di situs web Valkey.

Langkah 2: Buat folder dan bucket Amazon S3

Saat Anda telah membuat file cadangan, Anda perlu mengunggahnya ke folder dalam bucket Amazon S3. Untuk melakukannya, Anda harus memiliki bucket Amazon S3 dan folder dalam bucket tersebut terlebih dahulu. Jika Anda sudah memiliki bucket Amazon S3 dan folder dengan izin yang sesuai, Anda dapat melanjutkan ke Langkah 3: Unggah cadangan Anda ke Amazon S3.

Untuk membuat bucket Amazon S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di. https://console.aws.amazon.com/s3/
2. Ikuti petunjuk untuk membuat bucket Amazon S3 di Membuat bucket dalam Panduan Pengguna Amazon Simple Storage Service.

   Nama bucket Amazon S3 Anda harus sesuai dengan persyaratan DNS. Jika tidak, tidak ElastiCache dapat mengakses file cadangan Anda. Aturan untuk kesesuaian DNS adalah:

   - Nama harus minimal 3 dan tidak lebih dari 63 karakter.
   - Nama harus serangkaian satu atau beberapa label yang dipisahkan oleh titik (.) dengan setiap label:
     - Dimulai dengan huruf kecil atau angka.

  - Diakhiri dengan huruf kecil atau angka.
  - Hanya berisi huruf kecil, angka, dan tanda hubung.
- Nama tidak dapat diformat sebagai alamat IP (misalnya, 192.0.2.0).

Anda harus membuat bucket Amazon S3 di AWS Wilayah yang sama dengan cluster Redis OSS baru ElastiCache Anda. Pendekatan ini memastikan bahwa kecepatan transfer data tertinggi saat ElastiCache membaca file.rdb Anda dari Amazon S3.

> **Note**
>
> Untuk menjaga data Anda seaman mungkin, buat izin di bucket Amazon S3 Anda seketat mungkin. Pada saat yang sama, izin masih perlu mengizinkan bucket dan isinya digunakan untuk menyemai cluster Valkey atau Redis OSS baru Anda.

Untuk menambahkan folder ke bucket Amazon S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di. https://console.aws.amazon.com/s3/
2. Pilih nama bucket untuk mengunggah file .rdb Anda.
3. Pilih Buat folder.
4. Masukkan nama untuk folder baru Anda.
5. Pilih Simpan.

   Catat nama dari bucket dan folder.

Langkah 3: Unggah cadangan Anda ke Amazon S3

Sekarang, unggah file .rdb yang Anda buat di Langkah 1: Buat cadangan Valkey atau Redis OSS. Anda mengunggahnya ke bucket Amazon S3 dan folder yang Anda buat di Langkah 2: Buat folder dan bucket Amazon S3. Untuk informasi selengkapnya tentang tugas ini, lihat Menambahkan Objek ke Bucket. Di antara langkah 2 dan 3, pilih nama folder yang Anda buat.

Untuk mengunggah file .rdb Anda ke folder Amazon S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di. https://console.aws.amazon.com/s3/
2. Pilih nama bucket Amazon S3 yang Anda buat di Langkah 2.
3. Pilih nama folder yang Anda buat di Langkah 2.
4. Pilih Unggah.
5. Pilih Tambahkan file.
6. Telusuri untuk mencari file atau beberapa file yang ingin diunggah, lalu pilih file atau beberapa file tersebut. Untuk memilih beberapa file, tahan tombol Ctrl saat memilih setiap nama file.
7. Pilih Buka.
8. Konfirmasikan file atau beberapa file yang tercantum sudah benar dalam kotak dialog Unggah, lalu pilih Unggah.

Catat jalur ke file .rdb Anda. Misalnya, jika nama bucket Anda `myBucket` dan jalurnya adalah `myFolder/redis.rdb`, masukkan `myBucket/myFolder/redis.rdb`. Anda memerlukan jalur ini untuk melakukan seeding klaster baru dengan data dalam cadangan ini.

Untuk informasi tambahan, lihat Pembatasan dan batasan bucket dalam Panduan Pengguna Amazon Simple Storage Service.

Langkah 4: Berikan akses ElastiCache baca ke file.rdb

Sekarang, berikan akses ElastiCache baca ke file cadangan.rdb Anda. Anda memberikan ElastiCache akses ke file cadangan dengan cara yang berbeda tergantung apakah bucket Anda berada di AWS Wilayah default atau AWS Wilayah keikutsertaan.

AWS Wilayah yang diperkenalkan sebelum 20 Maret 2019, diaktifkan secara default. Anda dapat mulai bekerja di AWS Wilayah ini segera. Wilayah yang diperkenalkan setelah 20 Maret 2019, seperti Asia Pasifik (Hong Kong) dan Timur Tengah (Bahrain) dinonaktifkan secara default. Anda harus mengaktifkan, atau memilih, Wilayah ini sebelum dapat menggunakannya, seperti dijelaskan pada dalam Mengelola Wilayah AWS dalam Referensi Umum AWS.

Pilih pendekatan Anda tergantung pada AWS Wilayah Anda:

- Untuk Wilayah default, gunakan prosedur di Berikan akses ElastiCache baca ke file.rdb di Wilayah default.

- Untuk Wilayah pilihan, gunakan prosedur di Berikan akses ElastiCache baca ke file.rdb di Wilayah keikutsertaan.

Berikan akses ElastiCache baca ke file.rdb di Wilayah default

AWS Wilayah yang diperkenalkan sebelum 20 Maret 2019, diaktifkan secara default. Anda dapat mulai bekerja di AWS Wilayah ini segera. Wilayah yang diperkenalkan setelah 20 Maret 2019, seperti Asia Pasifik (Hong Kong) dan Timur Tengah (Bahrain) dinonaktifkan secara default. Anda harus mengaktifkan, atau memilih, Wilayah ini sebelum dapat menggunakannya, seperti dijelaskan pada dalam Mengelola Wilayah AWS dalam Referensi Umum AWS.

Untuk memberikan akses ElastiCache baca ke file cadangan di AWS Wilayah diaktifkan secara default

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di. https://console.aws.amazon.com/s3/
2. Pilih nama bucket S3 yang berisi file .rdb Anda.
3. Pilih nama folder yang berisi file .rdb Anda.
4. Pilih nama file cadangan .rdb Anda. Nama file yang dipilih muncul di atas tab di bagian atas halaman.
5. Pilih Izin.
6. Jika aws-scs-s3-readonly atau salah satu kanonik IDs dalam daftar berikut tidak terdaftar sebagai pengguna, lakukan hal berikut:

   a. Di bawah Akses untuk AWS akun lain, pilih Tambahkan penerima hibah.

   b. Di dalam kotak, tambahkan ID kanonik AWS Wilayah seperti yang ditunjukkan berikut:

   - AWS GovCloud (AS-Barat) Wilayah:

     ```
     40fa568277ad703bd160f66ae4f83fc9dfdfd06c2f1b5060ca22442ac3ef8be6
     ```

     > **Important**
     >
     > Cadangan harus ditempatkan di bucket S3 AWS GovCloud (US) agar Anda dapat mengunduhnya ke cluster Valkey atau Redis OSS di. AWS GovCloud (US)

   - AWS Wilayah diaktifkan secara default:

```
540804c33a284a299d2547575ce1010f2312ef3da9b3a053c8bc45bf233e4353
```

c. Tetapkan izin pada bucket dengan memilih Ya untuk berikut ini:

   - Daftar/tulis objek
   - Baca/tulis izin ACL objek

d. Pilih Simpan.

7. Pilih Gambaran Umum, lalu pilih Unduh.

Berikan akses ElastiCache baca ke file.rdb di Wilayah keikutsertaan

AWS Wilayah yang diperkenalkan sebelum 20 Maret 2019, diaktifkan secara default. Anda dapat mulai bekerja di AWS Wilayah ini segera. Wilayah yang diperkenalkan setelah 20 Maret 2019, seperti Asia Pasifik (Hong Kong) dan Timur Tengah (Bahrain) dinonaktifkan secara default. Anda harus mengaktifkan, atau memilih, Wilayah ini sebelum dapat menggunakannya, seperti dijelaskan pada dalam Mengelola Wilayah AWS dalam Referensi Umum AWS.

Sekarang, berikan akses ElastiCache baca ke file cadangan.rdb Anda.

Untuk memberikan akses ElastiCache baca ke file cadangan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di. https://console.aws.amazon.com/s3/
2. Pilih nama bucket S3 yang berisi file .rdb Anda.
3. Pilih nama folder yang berisi file .rdb Anda.
4. Pilih nama file cadangan .rdb Anda. Nama file yang dipilih muncul di atas tab di bagian atas halaman.
5. Pilih tab Izin.
6. Di bagian Izin, pilih Kebijakan bucket, lalu pilih Edit.
7. Memperbarui kebijakan untuk memberikan izin ElastiCache yang diperlukan untuk melakukan operasi:

   - Tambahkan `[ "Service" : "region-full-name.elasticache-snapshot.amazonaws.com" ]` ke `Principal`.
   - Menambahkan izin yang diperlukan berikut untuk mengekspor snapshot ke bucket Amazon S3:

- "s3:GetObject"
- "s3:ListBucket"
- "s3:GetBucketAcl"

Berikut adalah contoh tampilan kebijakan yang sudah diperbarui.

JSON

```
{
    "Version": "2012-10-17",
    "Id": "Policy15397346",
    "Statement": [
        {
            "Sid": "Stmt15399483",
            "Effect": "Allow",
            "Principal": {
                "Service": "ap-east-1.elasticache-snapshot.amazonaws.com"
            },
            "Action": [
                "s3:GetObject",
                "s3:ListBucket",
                "s3:GetBucketAcl"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/backup1.rdb",
                "arn:aws:s3:::amzn-s3-demo-bucket/backup2.rdb"
            ]
        }
    ]
}
```

8. Pilih Simpan perubahan.

Benih ElastiCache cluster dengan data file.rdb

Sekarang Anda siap untuk membuat ElastiCache cluster dan menyemai dengan data dari file.rdb. Untuk membuat klaster, ikuti petunjuk di Membuat cluster untuk Valkey atau Redis OSS atau

Membuat grup replikasi Valkey atau Redis OSS dari awal. Pastikan untuk memilih Valkey atau Redis OSS sebagai mesin cluster Anda.

Metode yang Anda gunakan untuk mengetahui ElastiCache di mana menemukan cadangan yang Anda unggah ke Amazon S3 bergantung pada metode yang Anda gunakan untuk membuat cluster:

Benih ElastiCache untuk cluster Redis OSS atau grup replikasi dengan data file.rdb

- Menggunakan ElastiCache konsol

  Saat memilih Pengaturan klaster, pilih Pulihkan dari cadangan sebagai metode pembuatan klaster Anda, lalu pilih Cadangan lain sebagai Sumber Anda di bagian Sumber cadangan. Di kotak Lokasi S3 file RDB seed, ketikkan jalur Amazon S3 untuk file tersebut. Jika Anda memiliki beberapa file .rdb, ketikkan jalur untuk setiap file dalam daftar yang dipisahkan koma. Jalur Amazon S3 terlihat seperti *myBucket*/*myFolder*/*myBackupFilename*.rdb.

- Menggunakan AWS CLI

  Jika Anda menggunakan operasi `create-cache-cluster` atau `create-replication-group`, gunakan parameter `--snapshot-arns` untuk menentukan ARN yang memenuhi syarat sepenuhnya untuk setiap file .rdb. Misalnya, `arn:aws:s3:::`*myBucket*/*myFolder*/*myBackupFilename*`.rdb`. ARN harus dapat diresolusi ke file cadangan yang Anda simpan di Amazon S3.

- Menggunakan ElastiCache API

  Jika Anda menggunakan operasi `CreateCacheCluster` atau `CreateReplicationGroup` ElastiCache API, gunakan parameter `SnapshotArns` untuk menentukan ARN yang sepenuhnya memenuhi syarat untuk setiap file.rdb. Misalnya, `arn:aws:s3:::`*myBucket*/*myFolder*/*myBackupFilename*`.rdb`. ARN harus dapat diresolusi ke file cadangan yang Anda simpan di Amazon S3.

Important

Saat menyemai cluster Valkey atau Redis OSS (mode cluster enabled), Anda harus mengonfigurasi setiap grup node (shard) di cluster atau grup replikasi baru. Gunakan parameter `--node-group-configuration` (API: `NodeGroupConfiguration`) untuk melakukannya. Untuk informasi selengkapnya, lihat berikut ini:

- CLI: create-replication-groupdalam Referensi AWS CLI

- API: CreateReplicationGroupdi Referensi ElastiCache API

Selama proses pembuatan cluster Anda, data dalam cadangan Valkey atau Redis OSS Anda ditulis ke cluster. Anda dapat memantau kemajuan dengan melihat pesan ElastiCache acara. Untuk melakukan ini, lihat ElastiCache konsol dan pilih Acara Cache. Anda juga dapat menggunakan antarmuka baris AWS ElastiCache perintah atau ElastiCache API untuk mendapatkan pesan acara. Lihat informasi yang lebih lengkap di Melihat ElastiCache acara.

Versi mesin dan peningkatan di ElastiCache

Bagian ini mencakup mesin Valkey, Memcached, dan Redis OSS yang didukung dan cara meng-upgrade. Perhatikan bahwa semua fitur yang tersedia dengan Redis OSS 7.2 tersedia di Valkey 7.2 dan di atas secara default. Anda juga dapat meningkatkan dari beberapa yang ada ElastiCache untuk mesin Redis OSS ke mesin Valkey.

Memutakhirkan versi mesin termasuk peningkatan mesin silang

Valkey dan Redis OSS

Dengan Valkey dan Redis OSS, Anda memulai upgrade versi ke cluster atau grup replikasi Anda dengan memodifikasinya menggunakan ElastiCache konsol, API AWS CLI, atau ElastiCache API dan menentukan versi engine yang lebih baru.

Anda juga dapat melakukan upgrade silang dari Redis OSS ke Valkey. Untuk informasi lebih lanjut tentang peningkatan silang, lihatCara meningkatkan dari Redis OSS ke Valkey.

Topik

- Cara meningkatkan dari Redis OSS ke Valkey
- Menyelesaikan peningkatan mesin Valkey atau Redis OSS yang diblokir

Cara mengubah klaster dan grup replikasi	
Cache	Grup replikasi
Menggunakan ElastiCache AWS Management Console	Menggunakan AWS Management Console
Menggunakan AWS CLI dengan ElastiCache	Menggunakan AWS CLI
Menggunakan ElastiCache API	Menggunakan ElastiCache API

Memcache

Dengan Memcached, untuk memulai upgrade versi ke cluster Anda, Anda memodifikasinya dan menentukan versi mesin yang lebih baru. Anda dapat melakukan ini dengan menggunakan ElastiCache konsol, the AWS CLI, atau ElastiCache API:

- Untuk menggunakan AWS Management Console, lihat —Menggunakan ElastiCache AWS Management Console.
- Untuk menggunakan AWS CLI, lihatMenggunakan AWS CLI dengan ElastiCache.
- Untuk menggunakan ElastiCache API, lihatMenggunakan ElastiCache API.

Cara meningkatkan dari Redis OSS ke Valkey

Valkey dirancang sebagai pengganti drop-in untuk Redis OSS 7. Anda dapat meningkatkan dari Redis OSS ke Valkey menggunakan Konsol, API, atau CLI, dengan menentukan mesin baru dan versi mesin utama. Alamat IP endpoint dan semua aspek lain dari aplikasi tidak akan diubah oleh upgrade. Saat memutakhirkan dari Redis OSS 5.0.6 dan yang lebih tinggi, Anda tidak akan mengalami downtime.

> **Note**
>
> AWS Persyaratan versi CLI untuk peningkatan Redis OSS ke Valkey:
>
> - Untuk AWS CLI v1: Versi minimum yang diperlukan 1.35.2 (Versi saat ini: 1.40.22)
> - Untuk AWS CLI v2: Minimum yang diperlukan versi 2.18.2 (Versi saat ini: 2.27.22)

> **Note**
>
> - Saat memutakhirkan dari versi Redis OSS sebelumnya dari 5.0.6, Anda mungkin mengalami waktu failover 30 hingga 60 detik selama propagasi DNS.
> - Untuk memutakhirkan kluster simpul tunggal Redis OSS (mode cluster dinonaktifkan) yang ada ke mesin Valkey, pertama-tama ikuti langkah-langkah berikut:. Membuat grup replikasi menggunakan klaster yang sudah ada Setelah cluster node tunggal Redis OSS (mode cluster dinonaktifkan) ditambahkan ke grup replikasi, Anda dapat melakukan upgrade lintas mesin ke Valkey.

Memutakhirkan grup replikasi dari Redis OSS ke Valkey

Jika Anda memiliki grup replikasi Redis OSS yang menggunakan grup parameter cache default, Anda dapat meningkatkan ke Valkey dengan menentukan versi mesin dan mesin baru dengan API. modify-replication-group

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
   --replication-group-id myReplGroup \
   --engine valkey \
   --engine-version 8.0
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
   --replication-group-id myReplGroup ^
   --engine valkey ^
   --engine-version 8.0
```

Jika Anda memiliki grup parameter cache kustom yang diterapkan ke grup replikasi Redis OSS yang ada yang ingin Anda tingkatkan, Anda juga harus melewati grup parameter cache Valkey kustom dalam permintaan. Grup parameter kustom Valkey input harus memiliki nilai parameter statis Redis OSS yang sama dengan grup parameter kustom Redis OSS yang ada.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
   --replication-group-id myReplGroup \
   --engine valkey \
   --engine-version 8.0 \
   --cache-parameter-group-name myParamGroup
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
   --replication-group-id myReplGroup ^
   --engine valkey ^
   --engine-version 8.0 ^
   --cache-parameter-group-name myParamGroup
```

Memutakhirkan cache tanpa server Redis OSS ke Valkey dengan CLI

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-serverless-cache \
   --serverless-cache-name myCluster \
```

```
   --engine valkey \
   --major-engine-version 8
```

Untuk Windows:

```
aws elasticache modify-serverless-cache ^
   --serverless-cache-name myCluster ^
   --engine valkey ^
   --major-engine-version 8
```

Memutakhirkan Redis OSS ke Valkey dengan Konsol

Upgrade dari Redis OSS 5 ke Valkey

1. Pilih cache Redis OSS untuk meng-upgrade.
2. Jendela Upgrade ke Valkey akan muncul. Pilih tombol Upgrade ke Valkey.
3. Buka Pengaturan cache, lalu pilih Versi mesin. Versi terbaru dari Valkey direkomendasikan.
4. Jika cache ini tanpa server, maka Anda perlu memperbarui grup parameter. Pergi ke area grup Parameter pengaturan Cache, pilih grup parameter yang sesuai seperti default.valkey8.
5. Pilih Upgrade.

Cache ini sekarang akan terdaftar di area Valkey konsol.

> **Note**
>
> Memutakhirkan langsung dari Redis OSS 4 atau lebih rendah ke Valkey dapat mencakup waktu failover yang lebih lama dari 30 hingga 60 detik selama propagasi DNS.

Menyelesaikan peningkatan mesin Valkey atau Redis OSS yang diblokir

Seperti yang ditunjukkan pada tabel berikut, operasi peningkatan mesin Valkey atau Redis OSS Anda diblokir jika Anda memiliki operasi peningkatan skala yang tertunda.

Operasi yang tertunda	Operasi yang diblokir
Penaikan skala	Peningkatan mesin segera

Operasi yang tertunda	Operasi yang diblokir
Peningkatan mesin	Penaikan skala segera
Penaikan skala dan peningkatan mesin	Penaikan skala segera
	Peningkatan mesin segera

Untuk menyelesaikan upgrade mesin Valkey atau Redis OSS yang diblokir

- Lakukan salah satu tindakan berikut:
  - Jadwalkan operasi upgrade mesin Redis OSS atau Valkey Anda untuk jendela perawatan berikutnya dengan membersihkan kotak centang Terapkan segera.

    Dengan CLI, gunakan `--no-apply-immediately`. Dengan API, gunakan `ApplyImmediately=false`.
  - Tunggu hingga jendela perawatan berikutnya (atau setelah) untuk melakukan operasi upgrade engine Redis OSS Anda.
  - Tambahkan operasi skala Redis OSS ke modifikasi cluster ini dengan kotak centang Terapkan Segera yang dipilih.

    Dengan CLI, gunakan `--apply-immediately`. Dengan API, gunakan `ApplyImmediately=true`.

    Pendekatan ini secara efektif membatalkan peningkatan mesin selama periode pemeliharaan berikutnya dengan melakukannya segera.

ElastiCache Extended Support

Dengan ElastiCache Extended Support, Anda dapat terus menjalankan cache Anda pada versi mesin utama melewati akhir tanggal dukungan standar dengan biaya tambahan. Jika Anda tidak meningkatkan setelah akhir tanggal dukungan standar, Anda akan dikenakan biaya.

Extended Support menyediakan pembaruan dan dukungan teknis berikut:

- Pembaruan keamanan untuk kritis dan tinggi CVEs untuk mesin cache dan cache Anda
- Perbaikan bug dan tambalan untuk masalah kritis

- Kemampuan untuk membuka kasus dukungan dan menerima bantuan pemecahan masalah dalam perjanjian tingkat ElastiCache layanan standar

Penawaran berbayar ini memberi Anda lebih banyak waktu untuk meningkatkan ke versi mesin utama yang didukung.

Misalnya, ElastiCache akhir tanggal dukungan standar untuk Redis OSS 4.0.10 adalah 31 Januari 2026. Jika Anda belum siap untuk secara manual meng-upgrade ke Valkey atau Redis OSS 6 atau lebih baru pada tanggal tersebut, secara otomatis ElastiCache akan mendaftarkan cache Anda di Extended Support dan Anda dapat terus menjalankan Redis OSS 4.0.10. Mulai hari pertama setiap bulan setelah dukungan standar berakhir, 1 Februari 2026, ElastiCache secara otomatis menagih Anda untuk Extended Support.

Extended Support tersedia hingga 3 tahun setelah akhir tanggal dukungan standar untuk versi mesin utama. Untuk Elasticache untuk Redis OSS versi 4 dan 5, itu akan menjadi 31 Januari 2029. Setelah tanggal ini, cache apa pun yang masih menjalankan Redis OSS versi 4 dan 5 akan secara otomatis ditingkatkan ke versi terbaru Valkey.

Setelah periode dukungan mesin berakhir, cache yang terus menjalankan versi lama itu akan secara otomatis beralih ke Extended Support. Anda akan diberi tahu sebelum tanggal mulai penetapan harga Extended Support sehingga Anda dapat meningkatkan instans Anda. Anda juga dapat secara eksplisit memilih keluar kapan saja dengan meningkatkan ke versi yang didukung.

Untuk informasi selengkapnya tentang akhir tanggal dukungan standar dan akhir tanggal Extended Support, lihat ElastiCache versi untuk jadwal akhir hidup Redis OSS Valkey, Memcached, atau Redis OSS.

Topik

- ElastiCache Biaya Extended Support
- Versi dengan ElastiCache Extended Support
- ElastiCache dan tanggung jawab pelanggan dengan ElastiCache Extended Support

ElastiCache Biaya Extended Support

Anda akan dikenakan biaya untuk semua mesin yang terdaftar di ElastiCache Extended Support mulai hari setelah akhir dukungan standar. Untuk ElastiCache akhir tanggal dukungan standar, lihatVersi dengan ElastiCache Extended Support.

Biaya tambahan untuk ElastiCache Extended Support secara otomatis berhenti ketika Anda melakukan salah satu tindakan berikut:

- Tingkatkan ke versi mesin yang tercakup dalam dukungan standar.
- Hapus cache yang menjalankan versi utama melewati ElastiCache akhir tanggal dukungan standar.

Pengisian daya akan dimulai kembali jika versi mesin target Anda memasuki Extended Support di masa mendatang.

Misalnya, katakanlah ElastiCache versi 4 untuk Redis OSS memasuki Extended Support pada 1 Februari 2026, dan Anda memutakhirkan cache Anda di v4 ke v6 pada 1 Januari 2027. Anda hanya akan dikenakan biaya selama 11 bulan Extended Support, pada ElastiCache versi 4 untuk Redis OSS. Jika Anda terus menjalankan ElastiCache versi 6 untuk Redis OSS melewati akhir tanggal dukungan standar 31 Januari 2027, maka cache tersebut akan dikenakan biaya Extended Support lagi mulai 1 Februari 2027.

Anda dapat menghindari biaya untuk ElastiCacheExtended Support dengan ElastiCache mencegah membuat atau memulihkan cache melewati ElastiCache akhir tanggal dukungan standar.

Untuk informasi selengkapnya, lihat ElastiCache harga Amazon.

Versi dengan ElastiCache Extended Support

Redis Open Source Software (OSS) versi 4 dan 5 mencapai komunitas mereka End of Life masing-masing pada tahun 2020 dan 2022. Ini berarti tidak ada pembaruan lebih lanjut, perbaikan bug, atau patch keamanan yang dirilis oleh komunitas. Dukungan standar untuk ElastiCache Redis OSS versi 4 dan 5 ElastiCache akan berakhir pada 31 Januari 2026. Terus menggunakan versi Redis OSS yang tidak didukung dapat membuat data Anda rentan terhadap Kerentanan Umum dan Eksposur () yang diketahui. CVEs

Mulai 1 Februari 2026, ElastiCache cache yang masih berjalan pada Redis OSS versi 4 dan 5 akan secara otomatis terdaftar di Extended Support, untuk memberikan ketersediaan dan keamanan berkelanjutan. Meskipun Extended Support menawarkan fleksibilitas, kami merekomendasikan untuk memperlakukan akhir dari dukungan standar sebagai tonggak perencanaan untuk beban kerja produksi Anda. Kami sangat menyarankan Anda untuk meningkatkan cache Redis OSS v4 dan v5 Anda ke ElastiCache Valkey atau Redis OSS v6 atau yang lebih baru, sebelum akhir dukungan standar.

Tabel berikut merangkum ElastiCache akhir Amazon dari tanggal dukungan standar dan tanggal Extended Support.

Dukungan yang diperpanjang dan jadwal Akhir Kehidupan

Versi Mesin Utama	Akhir dari Standard Support	Mulai dari Extended Support Y1 Premium	Mulai dari Extended Support Y2 Premium	Mulai dari Extended Support Y3 Premium	Akhir dari Extended Support dan versi EOL
Redis OSS v4	1/31/2026	2/1/2026	2/1/2027	2/1/2028	1/31/2029
Redis OSS v5	1/31/2026	2/1/2026	2/1/2027	2/1/2028	1/31/2029
Redis OSS v6	1/31/2027	2/1/2027	2/1/2028	2/1/2029	1/31/2030

Extended Support hanya akan ditawarkan untuk versi patch terbaru yang didukung dari setiap versi utama Redis OSS. Ketika Extended Support dimulai pada 1 Februari 2026, jika cluster Redis OSS v4 dan v5 Anda belum ada di versi patch terbaru, mereka akan secara otomatis ditingkatkan ke v4.0.10 untuk Redis OSS v4, dan v5.0.6 untuk Redis OSS v5, sebelum terdaftar di Extended Support. Ini memastikan bahwa Anda menerima pembaruan keamanan dan perbaikan bug melalui Extended Support. Anda tidak perlu mengambil tindakan apa pun untuk meningkatkan ke versi patch terbaru ini sebagai bagian dari transisi Extended Support.

ElastiCache dan tanggung jawab pelanggan dengan ElastiCache Extended Support

Berikut ini adalah tanggung jawab Amazon ElastiCache, dan tanggung jawab Anda dengan ElastiCache Extended Support.

ElastiCache Tanggung jawab Amazon

Setelah ElastiCache akhir tanggal dukungan standar, Amazon ElastiCache akan menyediakan patch, perbaikan bug, dan upgrade untuk engine yang terdaftar di Extended Support. ElastiCache Ini akan terjadi hingga 3 tahun, atau sampai Anda berhenti menggunakan mesin di Extended Support, mana yang terjadi lebih dulu.

Tanggung jawab Anda

Anda bertanggung jawab untuk menerapkan patch, perbaikan bug, dan upgrade yang diberikan untuk cache di Extended Support. ElastiCache Amazon ElastiCache berhak mengubah, mengganti, atau menarik tambalan, perbaikan bug, dan peningkatan tersebut kapan saja. Jika tambalan diperlukan untuk mengatasi masalah keamanan atau stabilitas kritis, Amazon ElastiCache berhak memperbarui cache Anda dengan tambalan, atau mengharuskan Anda menginstal tambalan.

Anda juga bertanggung jawab untuk meningkatkan mesin Anda ke versi engine yang lebih baru sebelum ElastiCache akhir tanggal Extended Support. ElastiCache Akhir tanggal Extended Support biasanya 3 tahun setelah ElastiCache akhir tanggal dukungan standar.

Jika Anda tidak memutakhirkan mesin Anda, maka setelah tanggal Extended Support ElastiCache berakhir, Amazon ElastiCache akan mencoba meningkatkan mesin Anda ke versi mesin yang lebih baru yang didukung di bawah dukungan ElastiCache standar. Jika pemutakhiran gagal, Amazon ElastiCache berhak menghapus cache yang menjalankan mesin melewati ElastiCache akhir tanggal dukungan standar. Namun, sebelum melakukannya, Amazon ElastiCache akan menyimpan data Anda dari mesin itu.

Manajemen Versi untuk ElastiCache

Kelola bagaimana Anda ingin memperbarui ElastiCache cache dan cluster yang dirancang sendiri yang diperbarui untuk mesin Valkey, Memcached, dan Redis OSS.

Manajemen versi untuk ElastiCache Cache Tanpa Server

Kelola jika dan kapan cache ElastiCache Tanpa Server ditingkatkan dan lakukan peningkatan versi berdasarkan persyaratan dan jadwal Anda sendiri.

ElastiCache Tanpa server secara otomatis menerapkan versi perangkat lunak minor dan patch terbaru ke cache Anda, tanpa dampak atau waktu henti apa pun ke aplikasi Anda. Anda tidak perlu melakukan tindakan apa pun.

Ketika versi utama baru tersedia, ElastiCache Serverless akan mengirimi Anda pemberitahuan di konsol dan acara di. EventBridge Anda dapat memilih untuk meningkatkan cache Anda ke versi utama terbaru dengan mengubah cache menggunakan Konsol, CLI, atau API, dan memilih versi mesin terbaru. Mirip dengan upgrade minor dan patch, upgrade versi mayor dilakukan tanpa downtime ke aplikasi Anda.

Manajemen versi untuk cluster yang dirancang sendiri ElastiCache

Saat bekerja dengan ElastiCache cluster yang dirancang sendiri, Anda dapat mengontrol kapan perangkat lunak yang menyalakan cluster cache Anda ditingkatkan ke versi baru yang didukung oleh. ElastiCache Anda dapat mengontrol kapan harus meng-upgrade cache Anda ke versi mayor, minor, dan patch terbaru yang tersedia. Anda dapat memulai peningkatan versi mesin pada grup klaster atau replikasi Anda dengan mengubahnya dan menentukan versi mesin baru.

Anda dapat mengontrol jika dan kapan perangkat lunak yang sesuai dengan protokol yang memberi daya pada cluster cache Anda ditingkatkan ke versi baru yang didukung oleh. ElastiCache Dengan tingkat kontrol ini, Anda dapat memelihara kompatibilitas dengan versi tertentu, menguji versi baru dengan aplikasi Anda sebelum di-deploy ke sistem produksi, dan melakukan peningkatan versi sesuai syarat dan waktu Anda sendiri.

Karena peningkatan versi mungkin menimbulkan beberapa risiko kompatibilitas, peningkatan tidak dilakukan secara otomatis. Anda sendiri yang harus memulai prosesnya.

Cluster Valkey dan Redis OSS

> **Note**
>
> - Jika klaster Valkey atau Redis OSS direplikasi di satu atau lebih Wilayah, versi mesin ditingkatkan untuk Wilayah sekunder dan kemudian untuk Wilayah utama.
> - ElastiCache untuk Redis OSS versi diidentifikasi dengan versi semantik yang terdiri dari komponen mayor dan minor. Misalnya, di Redis OSS 6.2, versi utama adalah 6, dan versi minor 2. Saat mengoperasikan cluster yang dirancang sendiri, ElastiCache untuk Redis OSS juga mengekspos komponen patch, misalnya Redis OSS 6.2.1, dan versi tambalannya adalah 1.
>
>   Versi utama adalah untuk perubahan API yang tidak kompatibel dan versi minor untuk fungsionalitas baru yang ditambahkan dengan cara yang kompatibel ke belakang. Versi patch adalah untuk perbaikan bug yang kompatibel ke belakang dan perubahan non-fungsional.

Dengan Valkey dan Redis OSS, Anda memulai upgrade versi engine ke cluster atau grup replikasi Anda dengan memodifikasinya dan menentukan versi mesin baru. Untuk informasi selengkapnya, lihat Mengubah grup replikasi.

Memcache

Dengan Memcached, untuk meningkatkan ke versi yang lebih baru Anda harus memodifikasi cluster cache Anda dan menentukan versi mesin baru yang ingin Anda gunakan. Peningkatan ke versi Memcached yang lebih baru merupakan proses destruktif – Data Anda akan hilang dan Anda akan memulai dengan cache "cold" atau kosong. Untuk informasi selengkapnya, lihat Memodifikasi cluster ElastiCache .

Anda harus menyadari bahwa persyaratan berikut ketika melakukan peningkatan dari versi lebih lama dari Memcached ke Memcached versi 1.4.33 atau yang lebih baru. `CreateCacheCluster` dan `ModifyCacheCluster` akan gagal dalam kondisi berikut:

- Jika `slab_chunk_max > max_item_size`.
- Jika `max_item_size modulo slab_chunk_max != 0`.
- Jika `max_item_size > ((max_cache_memory - memcached_connections_overhead) / 4)`.

  Nilai `(max_cache_memory - memcached_connections_overhead)` adalah memori simpul yang dapat digunakan untuk data. Untuk informasi selengkapnya, lihat Overhead koneksi Memcached.

Mesin dan versi yang didukung

ElastiCache cache tanpa server mendukung ElastiCache versi 7.2 untuk Valkey dan di atasnya, ElastiCache versi 1.6 untuk Memcached dan di atasnya, dan 7.0 untuk Redis OSS dan di atasnya. ElastiCache

ElastiCache cache yang dirancang sendiri mendukung ElastiCache versi 7.2 untuk Valkey dan di atasnya, ElastiCache versi 1.4.5 untuk Memcached dan di atasnya, dan 4.0.10 untuk Redis OSS dan di atasnya. ElastiCache

ElastiCache Cluster yang dirancang sendiri mendukung versi Valkey berikut:

- Versi Valkey yang didukung
- ElastiCache versi 8.1 untuk Valkey
- ElastiCache versi 8.0 untuk Valkey
- ElastiCache versi 7.2.6 untuk Valkey

Versi Valkey yang didukung

Versi Valkey yang didukung di bawah ini. Perhatikan bahwa Valkey mendukung sebagian besar fitur yang tersedia di ElastiCache versi 7.2 untuk Redis OSS secara default.

- Anda juga dapat meng-upgrade ElastiCache cluster Anda dengan versi lebih awal dari 5.0.6. Tindakan ini melibatkan proses yang sama tetapi mungkin mengalami waktu failover lebih lama selama penyebaran DNS (30 detik - 1 menit).
- Dimulai dengan Redis OSS 7, ElastiCache mendukung peralihan antara Valkey atau Redis OSS (mode cluster dinonaktifkan) dan Valkey atau Redis OSS (mode cluster diaktifkan).
- Proses upgrade mesin Amazon ElastiCache for Redis OSS dirancang untuk melakukan upaya terbaik untuk mempertahankan data Anda yang ada dan memerlukan replikasi Redis OSS yang berhasil.
- Saat memutakhirkan mesin, ElastiCache akan menghentikan koneksi klien yang ada. Untuk meminimalkan waktu henti selama peningkatan engine, kami sarankan Anda menerapkan praktik terbaik untuk klien Redis OSS dengan percobaan ulang kesalahan dan backoff eksponensial dan praktik terbaik untuk meminimalkan waktu henti selama pemeliharaan.
- Anda tidak dapat memutakhirkan langsung dari Valkey atau Redis OSS (mode cluster dinonaktifkan) ke Valkey atau Redis OSS (mode cluster diaktifkan) saat Anda meningkatkan mesin Anda. Prosedur berikut menunjukkan kepada Anda cara meningkatkan dari Valkey atau Redis OSS (mode cluster dinonaktifkan) ke Valkey atau Redis OSS (mode cluster diaktifkan).

  Untuk meningkatkan dari Valkey atau Redis OSS (mode cluster dinonaktifkan) ke Valkey atau Redis OSS (mode cluster diaktifkan) versi mesin

  1. Buat cadangan cluster atau grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) Anda. Untuk informasi selengkapnya, lihat Membuat cadangan manual.
  2. Gunakan cadangan untuk membuat dan menyemai klaster Valkey atau Redis OSS (mode cluster enabled) dengan satu shard (grup node). Tentukan versi mesin baru dan aktifkan mode klaster saat membuat klaster atau grup replikasi. Untuk informasi selengkapnya, lihat Tutorial: Menyemai cluster baru yang dirancang sendiri dengan cadangan yang dibuat secara eksternal.
  3. Hapus cluster atau grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) lama. Untuk informasi selengkapnya, lihat Menghapus cluster di ElastiCache atau Menghapus grup replikasi.

4. Skala klaster Valkey atau Redis OSS (mode cluster diaktifkan) baru atau grup replikasi ke jumlah pecahan (grup simpul) yang Anda butuhkan. Untuk informasi selengkapnya, lihat Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan)

- Saat meningkatkan versi utama mesin, misalnya dari 5.0.6 ke 6.0, Anda juga harus memilih grup parameter baru yang kompatibel dengan versi mesin yang baru tersebut.
- Untuk cluster dan cluster Redis OSS tunggal dengan multi-AZ dinonaktifkan, kami menyarankan agar memori yang cukup tersedia untuk Redis OSS seperti yang dijelaskan dalam. Memastikan Anda memiliki cukup memori untuk membuat snapshot Valkey atau Redis OSS Dalam kasus ini, klaster primer tidak tersedia untuk melayani permintaan selama proses peningkatan.
- Untuk kluster Redis OSS dengan Multi-AZ diaktifkan, kami juga menyarankan Anda menjadwalkan peningkatan mesin selama periode lalu lintas tulis masuk yang rendah. Saat memutakhirkan ke Redis OSS 5.0.6 atau lebih tinggi, klaster utama terus tersedia untuk permintaan layanan selama proses peningkatan.

  Klaster dan grup replikasi dengan beberapa serpihan diproses dan di-patch sebagai berikut:

  - Semua serpihan diproses secara paralel. Hanya satu operasi peningkatan yang akan dilakukan pada satu serpihan kapan saja.
  - Di setiap serpihan, semua replika diproses sebelum primer diproses. Jika terdapat lebih sedikit replika dalam serpihan, primer dalam serpihan itu mungkin diproses sebelum replika di serpihan lainnya selesai diproses.
  - Di semua serpihan, simpul primer diproses secara berurutan. Hanya satu simpul primer yang ditingkatkan dalam satu waktu.
- Jika enkripsi diaktifkan di klaster atau grup replikasi Anda saat ini, Anda tidak dapat melakukan peningkatan ke versi mesin yang tidak mendukung enkripsi, seperti dari 3.2.6 ke 3.2.10.

Pertimbangan memcached

Saat memutakhirkan cluster Memcached yang dirancang sendiri, pertimbangkan hal berikut.

- Manajemen versi mesin dirancang agar Anda dapat memiliki kontrol sebanyak mungkin terkait cara melakukan patching. Namun, ElastiCache berhak untuk menambal klaster Anda atas nama Anda jika terjadi kerentanan keamanan kritis dalam sistem atau perangkat lunak cache.
- Karena mesin Memcached tidak mendukung persistensi, peningkatan versi mesin Memcached merupakan proses disruptif yang menghilangkan semua data cache di klaster.

ElastiCache versi 8.1 untuk Valkey

Berikut adalah beberapa fitur baru yang diperkenalkan di Valkey 8.1 (dibandingkan dengan ElastiCache Valkey 8.0):

- Implementasi tabel hash baru yang mengurangi overhead memori untuk menurunkan penggunaan memori sebanyak 20% untuk pola umum key/value .
- Dukungan asli untuk filter Bloom, tipe data baru yang memungkinkan Anda melakukan pencarian menggunakan memori sebanyak 98% lebih sedikit dibandingkan dengan menggunakan tipe data Set.
- Perintah baru COMMANDLOG yang mencatat eksekusi lambat, permintaan besar, dan balasan besar.
- Dukungan pembaruan bersyarat baru untuk perintah SET menggunakan argumen IFEQ.
- Peningkatan kinerja, termasuk latensi hingga 45% lebih rendah untuk perintah ZRANK, kinerja hingga 12x lebih cepat untuk PFMERGE dan PFCOUNT, dan throughput hingga 514% lebih tinggi untuk BITCOUNT.

Untuk informasi lebih lanjut tentang Valkey, lihat Valkey

Untuk informasi lebih lanjut tentang rilis Valkey 8.1, lihat Catatan Rilis Valkey 8.1

ElastiCache versi 8.0 untuk Valkey

Berikut adalah beberapa fitur baru yang diperkenalkan di Valkey 8.0 (dibandingkan dengan ElastiCache Valkey 7.2.6):

- Peningkatan efisiensi memori, memungkinkan pengguna untuk menyimpan hingga 20% lebih banyak data per node tanpa perubahan aplikasi apa pun.
- Infrastruktur metrik per slot yang baru diperkenalkan untuk cache yang dirancang sendiri, memberikan visibilitas terperinci ke dalam kinerja dan penggunaan sumber daya slot individu.
- ElastiCache Tanpa server untuk Valkey 8.0 dapat menggandakan permintaan per detik (RPS) yang didukung setiap 2-3 menit, mencapai 5M RPS per cache dari nol dalam waktu kurang dari 13 menit, dengan latensi baca p50 sub-milidetik yang konsisten.

Untuk informasi lebih lanjut tentang Valkey, lihat Valkey

Untuk informasi lebih lanjut tentang rilis Valkey 8, lihat Catatan Rilis Valkey 8

ElastiCache versi 7.2.6 untuk Valkey

Pada 10 Oktober 2024, ElastiCache versi 7.2.6 untuk Valkey dirilis. Berikut adalah beberapa fitur baru yang diperkenalkan di 7.2 (dibandingkan dengan ElastiCache versi 7.1 untuk Redis OSS):

- Pengoptimalan kinerja dan memori untuk berbagai tipe data: optimasi memori untuk daftar dan set tombol jenis, optimasi kecepatan untuk perintah set yang diurutkan, pengoptimalan kinerja untuk perintah dengan beberapa tombol dalam mode cluster, peningkatan pub/sub kinerja, optimasi kinerja untuk SCAN, SSCAN, HSCAN, perintah ZSCAN dan banyak pengoptimalan kecil lainnya.
- Opsi WITHSCORE baru untuk perintah ZRANK dan ZREVRANK
- CLIENT NO-TOUCH bagi klien untuk menjalankan perintah tanpa mempengaruhi LRU/LFU kunci.
- Perintah baru CLUSTER MYSHARDID yang mengembalikan ID Shard dari node untuk secara logis mengelompokkan node dalam modus cluster berdasarkan replikasi.

Untuk informasi lebih lanjut tentang Valkey, lihat Valkey

Untuk informasi lebih lanjut tentang ElastiCache versi 7.2 untuk rilis Valkey, lihat Redis OSS 7.2.4 Catatan Rilis (ElastiCache versi 7.2 untuk Valkey mencakup semua perubahan dari versi 7.1 untuk Redis OSS hingga ElastiCache versi 7.2.4 untuk Redis OSS). ElastiCache Catatan rilis Valkey 7.2 di Valkey on. GitHub

ElastiCache versi 8.1 untuk Valkey

Berikut adalah beberapa fitur baru yang diperkenalkan di Valkey 8.1 (dibandingkan dengan ElastiCache Valkey 8.0):

- Implementasi tabel hash baru yang mengurangi overhead memori untuk menurunkan penggunaan memori sebanyak 20% untuk pola umum key/value .
- Dukungan asli untuk filter Bloom, tipe data baru yang memungkinkan Anda melakukan pencarian menggunakan memori sebanyak 98% lebih sedikit dibandingkan dengan menggunakan tipe data Set.
- Perintah baru COMMANDLOG yang mencatat eksekusi lambat, permintaan besar, dan balasan besar.
- Dukungan pembaruan bersyarat baru untuk perintah SET menggunakan argumen IFEQ.
- Peningkatan kinerja, termasuk latensi hingga 45% lebih rendah untuk perintah ZRANK, kinerja hingga 12x lebih cepat untuk PFMERGE dan PFCOUNT, dan throughput hingga 514% lebih tinggi untuk BITCOUNT.

Untuk informasi lebih lanjut tentang Valkey, lihat Valkey

Untuk informasi lebih lanjut tentang rilis Valkey 8.1, lihat Catatan Rilis Valkey 8.1

ElastiCache versi 8.0 untuk Valkey

Berikut adalah beberapa fitur baru yang diperkenalkan di Valkey 8.0 (dibandingkan dengan ElastiCache Valkey 7.2.6):

- Peningkatan efisiensi memori, memungkinkan pengguna untuk menyimpan hingga 20% lebih banyak data per node tanpa perubahan aplikasi apa pun.
- Infrastruktur metrik per slot yang baru diperkenalkan untuk cache yang dirancang sendiri, memberikan visibilitas terperinci ke dalam kinerja dan penggunaan sumber daya slot individu.
- ElastiCache Tanpa server untuk Valkey 8.0 dapat menggandakan permintaan per detik (RPS) yang didukung setiap 2-3 menit, mencapai 5M RPS per cache dari nol dalam waktu kurang dari 13 menit, dengan latensi baca p50 sub-milidetik yang konsisten.

Untuk informasi lebih lanjut tentang Valkey, lihat Valkey

Untuk informasi lebih lanjut tentang rilis Valkey 8, lihat Catatan Rilis Valkey 8

ElastiCache versi 7.2.6 untuk Valkey

Pada 10 Oktober 2024, ElastiCache versi 7.2.6 untuk Valkey dirilis. Berikut adalah beberapa fitur baru yang diperkenalkan di 7.2 (dibandingkan dengan ElastiCache versi 7.1 untuk Redis OSS):

- Pengoptimalan kinerja dan memori untuk berbagai tipe data: optimasi memori untuk daftar dan set tombol jenis, optimasi kecepatan untuk perintah set yang diurutkan, pengoptimalan kinerja untuk perintah dengan beberapa tombol dalam mode cluster, peningkatan pub/sub kinerja, optimasi kinerja untuk SCAN, SSCAN, HSCAN, perintah ZSCAN dan banyak pengoptimalan kecil lainnya.
- Opsi WITHSCORE baru untuk perintah ZRANK dan ZREVRANK
- CLIENT NO-TOUCH bagi klien untuk menjalankan perintah tanpa mempengaruhi LRU/LFU kunci.
- Perintah baru CLUSTER MYSHARDID yang mengembalikan ID Shard dari node untuk secara logis mengelompokkan node dalam modus cluster berdasarkan replikasi.

Untuk informasi lebih lanjut tentang Valkey, lihat Valkey

Untuk informasi lebih lanjut tentang ElastiCache versi 7.2 untuk rilis Valkey, lihat Redis OSS 7.2.4 Catatan Rilis (ElastiCache versi 7.2 untuk Valkey mencakup semua perubahan dari versi 7.1 untuk Redis OSS hingga ElastiCache versi 7.2.4 untuk Redis OSS). ElastiCache Catatan rilis Valkey 7.2 di Valkey on. GitHub

Versi mesin Redis OSS yang didukung

ElastiCache Cache tanpa server dan cache yang dirancang sendiri mendukung semua Redis OSS versi 7.1 dan sebelumnya.

- ElastiCache versi 7.1 untuk Redis OSS (ditingkatkan)

ElastiCache Cluster yang dirancang sendiri mendukung versi Redis OSS berikut:

- ElastiCache versi 7.1 untuk Redis OSS (ditingkatkan)
- ElastiCache versi 7.0 untuk Redis OSS (ditingkatkan)
- ElastiCache versi 6.2 untuk Redis OSS (ditingkatkan)
- ElastiCache versi 6.0 untuk Redis OSS (ditingkatkan)
- ElastiCache versi 5.0.6 untuk Redis OSS (ditingkatkan)
- ElastiCache versi 5.0.5 untuk Redis OSS (usang, gunakan versi 5.0.6)
- ElastiCache versi 5.0.4 untuk Redis OSS (usang, gunakan versi 5.0.6)
- ElastiCache versi 5.0.3 untuk Redis OSS (usang, gunakan versi 5.0.6)
- ElastiCache versi 5.0.0 untuk Redis OSS (usang, gunakan versi 5.0.6)
- ElastiCache versi 4.0.10 untuk Redis OSS (ditingkatkan)
- Versi Lewat Masa Pakai (EOL) (3.x)
- Versi Lewat Masa Pakai (EOL) (2.x)

ElastiCache versi 7.1 untuk Redis OSS (ditingkatkan)

Rilis ini berisi peningkatan kinerja yang memungkinkan beban kerja mendorong throughput yang lebih tinggi dan latensi operasi yang lebih rendah. ElastiCache versi 7.1 untuk Redis OSS memperkenalkan dua perangkat tambahan utama:

Kami memperluas fungsionalitas I/O thread yang disempurnakan untuk juga menangani logika lapisan presentasi. Dengan layer presentasi, yang kami maksud adalah thread Enhanced I/O yang sekarang tidak hanya membaca input klien, tetapi juga mengurai input ke dalam format perintah biner

Redis OSS. Ini kemudian diteruskan ke thread utama untuk eksekusi yang memberikan peningkatan performa. Peningkatan pola akses memori Redis OSS. Langkah-langkah eksekusi dari banyak operasi struktur data disisipkan, untuk memastikan akses memori paralel dan mengurangi latensi akses memori. Saat berjalan ElastiCache di Graviton3 berbasis `R7g.4xlarge` atau lebih besar, pelanggan dapat mencapai lebih dari 1 juta permintaan per detik per node. Dengan peningkatan kinerja ElastiCache untuk Redis OSS v7.1, pelanggan dapat mencapai throughput hingga 100% lebih banyak dan latensi P99 50% lebih rendah dibandingkan dengan Redis OSS v7.0. ElastiCache Peningkatan ini diaktifkan pada ukuran simpul dengan setidaknya 8 inti fisik (`2xlarge` pada Graviton, dan `4xlarge` pada x86), terlepas dari jenis CPU dan tidak memerlukan perubahan klien.

> **Note**
>
> ElastiCache v7.1 kompatibel dengan Redis OSS v7.0.

ElastiCache versi 7.0 untuk Redis OSS (ditingkatkan)

ElastiCache untuk Redis OSS 7.0 menambahkan sejumlah perbaikan dan dukungan untuk fungsionalitas baru:

- Fungsi: ElastiCache untuk Redis OSS 7 menambahkan dukungan untuk Fungsi Redis OSS, dan memberikan pengalaman terkelola yang memungkinkan pengembang untuk mengeksekusi skrip LUA dengan logika aplikasi yang disimpan di ElastiCache cluster, tanpa mengharuskan klien untuk mengirim ulang skrip ke server dengan setiap koneksi.
- Perbaikan ACL: Valkey dan Redis OSS 7 menambahkan dukungan untuk versi berikutnya dari Daftar Kontrol Akses (). ACLs Klien sekarang dapat menentukan beberapa set izin pada kunci atau ruang kunci tertentu di Valkey dan Redis OSS.
- Sharded Pub/Sub: ElastiCache untuk Valkey dan Redis OSS 7 menambahkan dukungan untuk menjalankan Pub/Sub functionality in a sharded way when running ElastiCache in Cluster Mode Enabled (CME). Pub/Sub kemampuan memungkinkan penerbit mengeluarkan pesan ke sejumlah pelanggan di saluran. Saluran terikat pada pecahan di ElastiCache cluster, menghilangkan kebutuhan untuk menyebarkan informasi saluran di seluruh pecahan sehingga meningkatkan skalabilitas.
- I/O Multiplexing yang disempurnakan: ElastiCache untuk Valkey dan Redis OSS 7 memperkenalkan multiplexing I/O yang disempurnakan, yang memberikan peningkatan throughput dan mengurangi latensi untuk beban kerja throughput tinggi yang memiliki banyak koneksi klien bersamaan ke sebuah cluster. ElastiCache Misalnya, saat menggunakan cluster node r6g.xlarge

dan menjalankan 5200 klien bersamaan, Anda dapat mencapai hingga 72% peningkatan throughput (operasi baca dan tulis per detik) dan latensi P99 penurunan hingga 71%, dibandingkan dengan versi 6 untuk Redis OSS. ElastiCache

Untuk informasi lebih lanjut tentang Valkey, lihat Valkey. Untuk informasi lebih lanjut tentang rilis Redis OSS 7.0, lihat Catatan Rilis Redis OSS 7.0 di Redis OSS pada. GitHub

ElastiCache versi 6.2 untuk Redis OSS (ditingkatkan)

ElastiCache untuk Redis OSS 6.2 mencakup peningkatan kinerja untuk cluster berkemampuan TLS menggunakan tipe node x86 dengan 8 v CPUs atau lebih atau tipe node Graviton2 dengan 4 v atau lebih. CPUs Peningkatan ini meningkatkan throughput dan mengurangi waktu pembentukan koneksi klien dengan membongkar enkripsi ke v. CPUs Dengan Redis OSS 6.2, Anda juga dapat mengelola akses ke saluran Pub/Sub dengan aturan Access Control List (ACL).

Dengan versi ini, kami juga memperkenalkan dukungan untuk tiering data pada node cluster yang berisi SSD yang terpasang NVMe secara lokal. Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.

Mesin Redis OSS versi 6.2.6 juga memperkenalkan dukungan untuk format asli JavaScript Object Notation (JSON), cara sederhana dan tanpa skema untuk menyandikan kumpulan data kompleks di dalam cluster Redis OSS. Dengan dukungan JSON, Anda dapat memanfaatkan kinerja dan Redis OSS APIs untuk aplikasi yang beroperasi melalui JSON. Untuk informasi selengkapnya, lihat Memulai dengan JSON. Juga termasuk metrik terkait JSON, `JsonBasedCmds` dan`JsonBasedCmdsLatency`, yang dimasukkan ke dalam CloudWatch untuk memantau penggunaan tipe data ini. Untuk informasi selengkapnya, lihat Metrik untuk Valkey dan Redis OSS.

Anda menentukan versi mesin dengan menggunakan 6.2. ElastiCache akan secara otomatis memanggil versi patch pilihan Redis OSS 6.2 yang tersedia. Sebagai contoh, ketika Anda membuat/mengubah klaster cache, Anda menetapkan parameter `--engine-version` ke 6.2. Cluster akan diluncurkan dengan versi patch pilihan Redis OSS 6.2 yang tersedia saat ini. creation/modification Menentukan engine versi 6.x di API akan menghasilkan versi minor terbaru dari Redis OSS 6.

Untuk kluster 6.0 yang ada, Anda dapat ikut serta dalam upgrade versi minor otomatis berikutnya dengan menyetel `AutoMinorVersionUpgrade` parameter ke `yes` dalam`CreateCacheCluster`,, `ModifyCacheCluster` atau. `CreateReplicationGroup` `ModifyReplicationGroup` APIs ElastiCache akan memutakhirkan versi minor dari cluster 6.0 Anda yang ada ke 6.2 menggunakan pembaruan layanan mandiri. Untuk informasi selengkapnya, lihat Pembaruan layanan mandiri di Amazon ElastiCache.

Saat memanggil DescribeCacheEngineVersions API, nilai `EngineVersion` parameter akan disetel ke 6.2 dan versi mesin aktual dengan versi tambalan akan dikembalikan di `CacheEngineVersionDescription` bidang.

Untuk informasi lebih lanjut tentang rilis Redis OSS 6.2, lihat Catatan Rilis Redis OSS 6.2 di Redis OSS pada. GitHub

ElastiCache versi 6.0 untuk Redis OSS (ditingkatkan)

Amazon ElastiCache memperkenalkan versi berikutnya ElastiCache untuk mesin Redis OSS, yang mencakup Autentikasi Pengguna dengan Kontrol Akses Berbasis Peran, caching sisi klien, dan peningkatan operasional yang signifikan.

Dimulai dengan Redis OSS 6.0, ElastiCache akan menawarkan versi tunggal untuk setiap rilis minor Redis OSS, daripada menawarkan beberapa versi patch. ElastiCache akan secara otomatis mengelola versi patch dari cluster cache Anda yang sedang berjalan, memastikan peningkatan kinerja dan keamanan yang ditingkatkan.

Anda juga dapat ikut serta dalam peningkatan versi auto minor berikutnya dengan menyetel `AutoMinorVersionUpgrade` parameter ke `yes` dan ElastiCache akan mengelola peningkatan versi minor, melalui pembaruan layanan mandiri. Untuk informasi selengkapnya, lihat Pembaruan layanan di ElastiCache.

Anda menentukan versi mesin dengan menggunakan`6.0`. ElastiCache akan secara otomatis memanggil versi patch pilihan Redis OSS 6.0 yang tersedia. Misalnya, ketika Anda create/modify cluster cache, Anda mengatur `--engine-version` parameter ke 6.0. Cluster akan diluncurkan dengan versi patch pilihan Redis OSS 6.0 yang tersedia saat ini pada waktu pembuatan/modifikasi. Permintaan apa pun dengan nilai versi patch tertentu akan ditolak, pengecualian akan dikeluarkan dan proses akan gagal.

Saat memanggil DescribeCacheEngineVersions API, nilai `EngineVersion` parameter akan disetel ke 6.0 dan versi mesin aktual dengan versi tambalan akan dikembalikan di `CacheEngineVersionDescription` bidang.

Untuk informasi lebih lanjut tentang rilis Redis OSS 6.0, lihat Catatan Rilis Redis OSS 6.0 di Redis OSS pada. GitHub

ElastiCache versi 5.0.6 untuk Redis OSS (ditingkatkan)

Amazon ElastiCache memperkenalkan versi berikutnya ElastiCache untuk mesin Redis OSS, yang mencakup perbaikan bug dan pembaruan kumulatif berikut:

- Jaminan kestabilan mesin dalam kondisi khusus.
- Peningkatan penanganan kesalahan Hyperloglog.
- Peningkatan perintah handshake untuk replikasi yang andal.
- Pelacakan pengiriman pesan yang konsisten melalui perintah `XCLAIM`.
- Peningkatan manajemen bidang `LFU` dalam objek.
- Manajemen transaksi yang ditingkatkan saat menggunakan `ZPOP`.
- Kemampuan untuk mengganti nama perintah: Parameter yang disebut `rename-commands` yang memungkinkan Anda mengganti nama perintah Redis OSS yang berpotensi berbahaya atau mahal yang dapat menyebabkan kehilangan data yang tidak disengaja, seperti atau. `FLUSHALL` `FLUSHDB` Ini mirip dengan konfigurasi rename-command di open source Redis OSS. Namun, ElastiCache telah meningkatkan pengalaman dengan menyediakan alur kerja yang dikelola sepenuhnya. Perubahan nama perintah diterapkan segera, dan secara otomatis disebarkan ke semua simpul dalam klaster yang berisi daftar perintah. Tidak ada intervensi yang diperlukan di sisi Anda, seperti boot ulang simpul.

  Contoh berikut menunjukkan cara mengubah grup parameter yang sudah ada. Yang termasuk di sini adalah parameter `rename-commands`, yang merupakan daftar dipisahkan spasi berisi perintah yang ingin diubah namanya:

  ```
  aws elasticache modify-cache-parameter-group --cache-parameter-group-
  name custom_param_group
  --parameter-name-values "ParameterName=rename-commands,  ParameterValue='flushall
   restrictedflushall'" --region region
  ```

  Pada contoh ini, parameter rename-command digunakan untuk mengubah nama perintah `flushall` menjadi `restrictedflushall`.

  Untuk mengubah nama beberapa perintah, gunakan cara berikut:

  ```
  aws elasticache modify-cache-parameter-group --cache-parameter-group-
  name custom_param_group
  --parameter-name-values "ParameterName=rename-commands,  ParameterValue='flushall
   restrictedflushall flushdb restrictedflushdb'" --region region
  ```

  Untuk membalikkan perubahan apa pun, jalankan kembali perintah dan keluarkan semua nilai perubahan nama dari daftar `ParameterValue` yang ingin dipertahankan, seperti ditunjukkan berikut:

```
aws elasticache modify-cache-parameter-group --cache-parameter-group-
name custom_param_group
--parameter-name-values "ParameterName=rename-commands,  ParameterValue='flushall
 restrictedflushall'" --region region
```

Dalam hal ini, perintah `flushall` diubah namanya menjadi `restrictedflushall` dan perintah perubahan nama lainnya kembali ke nama perintah aslinya.

> **Note**
>
> Saat mengubah nama perintah, Anda dibatasi pada batasan berikut:
>
> - Semua perintah ubah nama harus alfanumerik.
> - Panjang maksimum nama perintah baru adalah 20 karakter alfanumerik.
> - Ketika mengubah nama perintah, pastikan bahwa Anda memperbarui grup parameter yang terkait dengan klaster Anda.
> - Untuk mencegah penggunaan sebuah perintah sepenuhnya, gunakan kata kunci `blocked`, seperti yang ditunjukkan berikut ini:
>
> ```
> aws elasticache modify-cache-parameter-group --cache-parameter-group-
> name custom_param_group
> --parameter-name-values "ParameterName=rename-commands,
>  ParameterValue='flushall blocked'" --region region
> ```

Untuk informasi selengkapnya tentang perubahan parameter dan daftar perintah apa yang memenuhi syarat untuk perubahan nama, lihat Redis OSS 5.0.3 perubahan parameter.

- Redis OSS Streams: Ini memodelkan struktur data log yang memungkinkan produsen untuk menambahkan item baru secara real time. Fitur ini juga memungkinkan pelanggan menerima pesan baik dalam mode memblokir ataupun tidak memblokir. Aliran juga memungkinkan grup pelanggan, yang merepresentasikan sekelompok klien untuk secara kooperatif menggunakan bagian yang berbeda dari aliran pesan yang sama, mirip dengan Apache Kafka. Untuk informasi selengkapnya, lihat Streaming.
- Dukungan untuk keluarga perintah aliran, seperti `XADD`, `XRANGE` dan `XREAD`. Untuk informasi selengkapnya, lihat Perintah Streams.

- Sejumlah parameter baru dan perubahan nama. Untuk informasi selengkapnya, lihat Redis OSS 5.0.0 perubahan parameter.
- Metrik Redis OSS baru,. `StreamBasedCmds`
- Waktu snapshot sedikit lebih cepat untuk node Redis OSS.

> **Important**
>
> ElastiCache telah melakukan back-porting dua perbaikan bug penting dari Redis OSS open source versi 5.0.1. Perbaikan tersebut tercantum sebagai berikut:
>
> - PULIHKAN balasan yang tidak sesuai ketika kunci tertentu telah kedaluwarsa.
> - Perintah `XCLAIM` dapat berpotensi menghasilkan entri yang salah atau mengganggu sinkronisasi protokol.
>
> Kedua perbaikan bug ini disertakan dalam ElastiCache untuk dukungan Redis OSS untuk mesin Redis OSS versi 5.0.0 dan dikonsumsi dalam pembaruan versi masa depan.

Untuk informasi lebih lanjut, lihat Catatan Rilis Redis OSS 5.0.6 di Redis OSS di. GitHub

ElastiCache versi 5.0.5 untuk Redis OSS (usang, gunakan versi 5.0.6)

Amazon ElastiCache memperkenalkan versi berikutnya ElastiCache untuk mesin Redis OSS;. Ini termasuk perubahan konfigurasi online untuk ElastiCache cluster auto-failover selama semua operasi yang direncanakan. Anda sekarang dapat menskalakan klaster Anda, meningkatkan versi mesin Redis OSS dan menerapkan patch dan pembaruan pemeliharaan saat cluster tetap online dan terus melayani permintaan yang masuk. Pembaruan ini juga menyertakan perbaikan bug.

Untuk informasi lebih lanjut, lihat Catatan Rilis Redis OSS 5.0.5 di Redis OSS di. GitHub

ElastiCache versi 5.0.4 untuk Redis OSS (usang, gunakan versi 5.0.6)

Amazon ElastiCache memperkenalkan versi berikutnya dari mesin Redis OSS yang didukung oleh. ElastiCache Versi ini mencakup perbaikan berikut:

- Jaminan kestabilan mesin dalam kondisi khusus.
- Peningkatan penanganan kesalahan Hyperloglog.

- Peningkatan perintah handshake untuk replikasi yang andal.
- Pelacakan pengiriman pesan yang konsisten melalui perintah `XCLAIM`.
- Peningkatan manajemen bidang `LFU` dalam objek.
- Manajemen transaksi yang ditingkatkan saat menggunakan `ZPOP`.

Untuk informasi lebih lanjut, lihat Catatan Rilis Redis OSS 5.0.4 di Redis OSS di. GitHub

ElastiCache versi 5.0.3 untuk Redis OSS (usang, gunakan versi 5.0.6)

Amazon ElastiCache memperkenalkan versi berikutnya ElastiCache untuk mesin Redis OSS, yang mencakup perbaikan bug.

ElastiCache versi 5.0.0 untuk Redis OSS (usang, gunakan versi 5.0.6)

Amazon ElastiCache memperkenalkan versi utama berikutnya ElastiCache untuk mesin Redis OSS. ElastiCache versi 5.0.0 untuk Redis OSS membawa dukungan untuk perbaikan berikut:

- Redis OSS Streams: Ini memodelkan struktur data log yang memungkinkan produsen untuk menambahkan item baru secara real time. Fitur ini juga memungkinkan pelanggan menerima pesan baik dalam mode memblokir ataupun tidak memblokir. Aliran juga memungkinkan grup pelanggan, yang merepresentasikan sekelompok klien untuk secara kooperatif menggunakan bagian yang berbeda dari aliran pesan yang sama, mirip dengan Apache Kafka. Untuk informasi selengkapnya, lihat Streaming.
- Dukungan untuk keluarga perintah aliran, seperti `XADD`, `XRANGE` dan `XREAD`. Untuk informasi selengkapnya, lihat Perintah Streams.
- Sejumlah parameter baru dan perubahan nama. Untuk informasi selengkapnya, lihat Redis OSS 5.0.0 perubahan parameter.
- Metrik Redis OSS baru,. `StreamBasedCmds`
- Waktu snapshot sedikit lebih cepat untuk node Redis OSS.

ElastiCache versi 4.0.10 untuk Redis OSS (ditingkatkan)

Amazon ElastiCache memperkenalkan versi utama berikutnya ElastiCache untuk mesin Redis OSS. ElastiCache versi 4.0.10 untuk Redis OSS membawa dukungan untuk perbaikan berikut:

- Baik pengubahan ukuran cluster online dan enkripsi dalam satu ElastiCache versi. Untuk informasi selengkapnya, lihat berikut ini:

  - Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan)
  - Resharding online untuk Valkey atau Redis OSS (mode cluster diaktifkan)
  - Keamanan data di Amazon ElastiCache
- Sejumlah parameter baru. Untuk informasi selengkapnya, lihat Redis OSS 4.0.10 perubahan parameter.
- Dukungan untuk keluarga perintah memori, seperti `MEMORY`. Untuk informasi selengkapnya, lihat Perintah (cari di MEMO).
- Dukungan untuk defragmentasi memori saat online sehingga memungkinkan pemanfaatan memori yang lebih efisien dan lebih banyak memori yang tersedia untuk data Anda.
- Support untuk pembilasan dan penghapusan asinkron. ElastiCache untuk Redis OSS mendukung perintah sepertiUNLINK, `FLUSHDB` dan `FLUSHALL` berjalan di utas yang berbeda dari utas utama. Melakukan hal ini membantu meningkatkan performa dan waktu respons untuk aplikasi Anda dengan membebaskan memori secara asinkron.
- Metrik Redis OSS baru,. `ActiveDefragHits` Untuk informasi selengkapnya, lihat Metrik untuk Redis OSS.

Redis OSS (mode cluster dinonaktifkan) pengguna yang menjalankan ElastiCache versi 3.2.10 untuk Redis OSS dapat menggunakan konsol untuk meng-upgrade cluster mereka melalui upgrade online.

Membandingkan pengubahan ukuran ElastiCache cluster dan dukungan enkripsi

Fitur	3.2.6	3.2.10	4.0.10 dan yang lebih baru
Perubahan ukuran klaster online *	Tidak	Ya	Ya
Enkripsi bergerak **	Ya	Tidak	Ya
Enkripsi diam **	Ya	Tidak	Ya
* Menambahkan, menghapus, dan menyeimbangkan kembali serpihan.			
** Diwajibkan untuk aplikasi yang mematuhi FedRAMP, HIPAA, dan PCI DSS. Untuk informasi selengkapnya, lihat Validasi kepatuhan untuk Amazon ElastiCache.			

Versi Lewat Masa Pakai (EOL) (3.x)

ElastiCache versi 3.2.10 untuk Redis OSS (ditingkatkan)

Amazon ElastiCache memperkenalkan versi utama berikutnya ElastiCache untuk mesin Redis OSS. ElastiCache versi 3.2.10 untuk Redis OSS (enchanced) memperkenalkan pengubahan ukuran cluster online untuk menambah atau menghapus pecahan dari cluster sambil terus melayani permintaan yang masuk. I/O ElastiCache untuk Redis OSS 3.2.10 pengguna memiliki semua fungsi versi Redis OSS sebelumnya kecuali kemampuan untuk mengenkripsi data mereka. Kemampuan ini saat ini hanya tersedia pada versi 3.2.6.

Membandingkan ElastiCache versi 3.2.6 dan 3.2.10 untuk Redis OSS

Fitur	3.2.6	3.2.10
Perubahan ukuran klaster online *	Tidak	Ya
Enkripsi bergerak **	Ya	Tidak
Enkripsi diam **	Ya	Tidak
* Menambahkan, menghapus, dan menyeimbangkan kembali serpihan.		
** Diwajibkan untuk aplikasi yang mematuhi FedRAMP, HIPAA, dan PCI DSS. Untuk informasi selengkapnya, lihat Validasi kepatuhan untuk Amazon ElastiCache.		

Untuk informasi selengkapnya, lihat berikut ini:

- Resharding online untuk Valkey atau Redis OSS (mode cluster diaktifkan)
- Perubahan ukuran klaster online

ElastiCache versi 3.2.6 untuk Redis OSS (ditingkatkan)

Amazon ElastiCache memperkenalkan versi utama berikutnya ElastiCache untuk mesin Redis OSS. ElastiCache versi 3.2.6 untuk pengguna Redis OSS memiliki akses ke semua fungsi versi Redis OSS sebelumnya, ditambah opsi untuk mengenkripsi data mereka. Untuk informasi selengkapnya, lihat berikut ini:

- ElastiCache enkripsi dalam transit (TLS)

- Enkripsi At-Rest di ElastiCache
- Validasi kepatuhan untuk Amazon ElastiCache

ElastiCache versi 3.2.4 untuk Redis OSS (ditingkatkan)

Amazon ElastiCache versi 3.2.4 memperkenalkan versi utama berikutnya ElastiCache untuk mesin Redis OSS. ElastiCache 3.2.4 pengguna memiliki semua fungsi versi Redis OSS sebelumnya yang tersedia bagi mereka, ditambah opsi untuk berjalan dalam mode cluster atau mode non-cluster. Tabel berikut merangkum hal ini.

Membandingkan Redis OSS 3.2.4 mode non-cluster dan mode cluster

Fitur	Mode non-klaster	Mode klaster
Pembuatan Partisi Data	Tidak	Ya
Pengindeksan geospasial	Ya	Ya
Ubah jenis simpul	Ya	Ya *
Penskalaan replika	Ya	Ya *
Menskalakan ke luar	Tidak	Ya *
Dukungan basis data	Beberapa	Tunggal
Grup parameter	`default.redis3.2` **	`default.redis3.2.cluster.on` **
* Lihat Melakukan pemulihan dari cadangan ke dalam cache baru		
** Atau satu yang berasal dari itu.		

Catatan:

- Pembuatan partisi – kemampuan untuk membagi data Anda pada 2 hingga 500 grup simpul (serpihan) dengan dukungan replikasi untuk setiap grup simpul.

- Pengindeksan geospasial - Redis OSS 3.2.4 memperkenalkan dukungan untuk pengindeksan geospasial melalui enam perintah GEO. Untuk informasi lebih lanjut, lihat Redis OSS GEO* dokumentasi perintah Perintah: GEO di halaman Perintah Valkey (difilter untuk GEO).

Untuk informasi tentang fitur tambahan Redis OSS 3, lihat Catatan rilis Redis OSS 3.2 dan catatan rilis Redis OSS 3.0.

Saat ini ElastiCache dikelola Valkey atau Redis OSS (mode cluster diaktifkan) tidak mendukung fitur Redis OSS 3.2 berikut:

- Migrasi replika
- Penyeimbangan kembali klaster
- Lua debugger

ElastiCache menonaktifkan perintah manajemen Redis OSS 3.2 berikut:

- `cluster meet`
- `cluster replicate`
- `cluster flushslots`
- `cluster addslots`
- `cluster delslots`
- `cluster setslot`
- `cluster saveconfig`
- `cluster forget`
- `cluster failover`
- `cluster bumpepoch`
- `cluster set-config-epoch`
- `cluster reset`

Untuk informasi tentang parameter Redis OSS 3.2.4, lihat. Redis OSS 3.2.4 perubahan parameter

Versi Lewat Masa Pakai (EOL) (2.x)

ElastiCache versi 2.8.24 untuk Redis OSS (ditingkatkan)

Perbaikan Redis OSS ditambahkan sejak versi 2.8.23 termasuk perbaikan bug dan logging alamat akses memori yang buruk. Untuk informasi lebih lanjut, lihat catatan rilis Redis OSS 2.8.

ElastiCache versi 2.8.23 untuk Redis OSS (ditingkatkan)

Perbaikan Redis OSS ditambahkan sejak versi 2.8.22 termasuk perbaikan bug. Untuk informasi lebih lanjut, lihat catatan rilis Redis OSS 2.8. Rilis ini mencakup dukungan untuk parameter baru `close-on-slave-write` yang, jika diaktifkan, memutuskan klien yang mencoba menulis ke replika hanya-baca.

Untuk informasi selengkapnya tentang parameter Redis OSS 2.8.23, lihat Redis OSS 2.8.23 (ditingkatkan) menambahkan parameter di Panduan Pengguna. ElastiCache

ElastiCache versi 2.8.22 untuk Redis OSS (ditingkatkan)

Perbaikan Redis OSS ditambahkan sejak versi 2.8.21 meliputi yang berikut:

- Dukungan untuk pencadangan dan sinkronisasi forkless, yang memungkinkan Anda mengalokasikan lebih sedikit memori untuk overhead cadangan dan lebih banyak untuk aplikasi Anda. Untuk informasi selengkapnya, lihat Cara penerapan sinkronisasi dan pencadangan. Proses forkless dapat memengaruhi latensi dan throughput. Saat ada throughput operasi tulis yang tinggi, dan replika melakukan sinkronisasi ulang, operasi tulis dapat menjadi tidak terjangkau selama keseluruhan waktu sinkronisasi itu.
- Jika ada failover, grup replikasi sekarang akan pulih lebih cepat karena replika melakukan sinkronisasi parsial dengan primer bukannya sinkronisasi penuh jika memungkinkan. Selain itu, primer dan replika tidak lagi menggunakan disk selama sinkronisasi, yang menyediakan keuntungan selanjutnya pada kecepatan.
- Support untuk dua CloudWatch metrik baru.
  - `ReplicationBytes` – Jumlah byte yang dikirimkan klaster primer grup replikasi ke replika baca.
  - `SaveInProgress` – Nilai biner yang menunjukkan apakah ada proses simpan di latar belakang yang berjalan.

  Untuk informasi selengkapnya, lihat Pemantauan penggunaan dengan CloudWatch Metrik.

- Sejumlah perbaikan bug penting dalam perilaku replikasi PSYNC. Untuk informasi lebih lanjut, lihat catatan rilis Redis OSS 2.8.
- Untuk mempertahankan kinerja replikasi yang ditingkatkan dalam grup replikasi multi-AZ dan untuk meningkatkan stabilitas klaster, ElastiCache non-replika tidak lagi didukung.
- Untuk memperbaiki konsistensi data antara klaster primer dan replika dalam grup replikasi, replika tidak lagi mengosongkan kunci yang independen dari klaster primer.
- Redis OSS variabel konfigurasi `appendonly` dan tidak `appendfsync` didukung pada Redis OSS versi 2.8.22 dan yang lebih baru.
- Dalam situasi memori rendah, klien dengan buffer output yang besar mungkin terputus dari klaster replika. Jika terputus, klien perlu terhubung kembali. Situasi seperti itu kemungkinan besar terjadi untuk klien PUBSUB.

ElastiCache versi 2.8.21 untuk Redis OSS

Perbaikan Redis OSS ditambahkan sejak versi 2.8.19 mencakup sejumlah perbaikan bug. Untuk informasi lebih lanjut, lihat catatan rilis Redis OSS 2.8.

ElastiCache versi 2.8.19 untuk Redis OSS

Perbaikan Redis OSS ditambahkan sejak versi 2.8.6 meliputi yang berikut:

- Support untuk HyperLogLog. Untuk informasi lebih lanjut, lihat struktur data baru Redis OSS:. HyperLogLog
- Jenis data sorted set sekarang memiliki dukungan untuk kueri kisaran leksikografis dengan perintah baru `ZRANGEBYLEX`, `ZLEXCOUNT`, dan `ZREMRANGEBYLEX`.
- Untuk mencegah simpul primer mengirimkan data usang untuk simpul replika, master SYNC gagal jika proses turunan untuk menyimpan di latar belakang (`bgsave`) dibatalkan.
- Support untuk HyperLogLogBasedCommands CloudWatchmetrik. Untuk informasi selengkapnya, lihat Metrik untuk Valkey dan Redis OSS.

ElastiCache versi 2.8.6 untuk Redis OSS

Perbaikan Redis OSS ditambahkan sejak versi 2.6.13 meliputi yang berikut:

- Peningkatan ketahanan dan toleransi kesalahan untuk replika baca.
- Dukungan untuk sinkronisasi ulang parsial.

- Dukungan untuk jumlah minimum replika baca yang ditentukan pengguna yang harus tersedia setiap saat.
- Dukungan penuh untuk pub/sub—memberi tahu klien tentang peristiwa di server.
- Deteksi otomatis untuk kegagalan simpul primer dan failover dari simpul primer Anda ke simpul sekunder.

ElastiCache versi 2.6.13 untuk Redis OSS

ElastiCache versi 2.6.13 untuk Redis OSS adalah versi awal yang didukung Redis OSS. ElastiCache Multi-AZ tidak didukung pada ElastiCache versi 2.6.13 untuk Redis OSS.

ElastiCache versi untuk jadwal akhir hidup Redis OSS

Bagian ini mendefinisikan tanggal akhir masa pakai (EOL) untuk versi utama yang lebih lama saat diumumkan. Hal ini membantu Anda mengambil keputusan terkait versi dan peningkatan pada masa mendatang.

> **Note**
>
> ElastiCache versi dari 5.0.0 ke 5.0.5 untuk Redis OSS tidak digunakan lagi. Gunakan versi 5.0.6 atau yang lebih baru.

Tabel berikut menunjukkan jadwal Extended Support ElastiCache untuk mesin Redis OSS.

Jadwal Extended Support dan End of Life

Versi Mesin Utama	Akhir dari Standard Support	Mulai dari Extended Support Y1 Premium	Mulai dari Extended Support Y2 Premium	Mulai dari Extended Support Y3 Premium	Akhir dari Extended Support dan versi EOL
Redis OSS v4	1/31/2026	2/1/2026	2/1/2027	2/1/2028	1/31/2029
Redis OSS v5	1/31/2026	2/1/2026	2/1/2027	2/1/2028	1/31/2029

Versi Mesin Utama	Akhir dari Standard Support	Mulai dari Extended Support Y1 Premium	Mulai dari Extended Support Y2 Premium	Mulai dari Extended Support Y3 Premium	Akhir dari Extended Support dan versi EOL
Redis OSS v6	1/31/2027	2/1/2027	2/1/2028	2/1/2029	1/31/2030

Tabel berikut merangkum setiap versi dan tanggal EOL yang diumumkan, serta versi target peningkatan yang direkomendasikan.

EOL terdahulu

V U S	Versi Minor Sumber	Target Peningkatan yang Disarankan	Tanggal EOL
V 3	3.2.4, 3.2.6, dan 3.2.10	Versi 6.2 atau yang lebih baru **Note** Untuk Wilayah US-ISO-EAST -1, US-ISO-WEST -1, dan US-ISOB-EAST -1, kami merekomendasikan 5.0.6 atau lebih tinggi.	31 Juli 2023
V 2	2.8.24, 2.8.23, 2.8.22, 2.8.21, 2.8.19, 2.8.12, 2.8.6, 2.6.13	Versi 6.2 atau yang lebih baru **Note** Untuk Wilayah US-ISO-EAST -1, US-ISO-WEST -1, dan US-ISOB-EAST -1,	13 Januari 2023

V U S	Versi Minor Sumber	Target Peningkatan yang Disarankan	Tanggal EOL
		kami merekomen dasikan 5.0.6 atau lebih tinggi.	

Didukung ElastiCache untuk versi Memcached

ElastiCache mendukung versi Memcached berikut dan meningkatkan ke versi yang lebih baru. Saat meningkatkan ke versi yang lebih baru, perhatikan kondisi yang jika tidak terpenuhi dapat menyebabkan peningkatan Anda gagal.

ElastiCache untuk Versi Memcached

- ElastiCache versi 1.6.22 untuk Memcached
- ElastiCache versi 1.6.17 untuk Memcached
- ElastiCache versi 1.6.12 untuk Memcached
- ElastiCache versi 1.6.6 untuk Memcached
- ElastiCache versi 1.5.16 untuk Memcached
- ElastiCache versi 1.5.10 untuk Memcached
- ElastiCache versi 1.4.34 untuk Memcached
- ElastiCache versi 1.4.33 untuk Memcached
- ElastiCache versi 1.4.24 untuk Memcached
- ElastiCache versi 1.4.14 untuk Memcached
- ElastiCache versi 1.4.5 untuk Memcached

ElastiCache versi 1.6.22 untuk Memcached

ElastiCache untuk Memcached versi 1.6.22 untuk Memcached menambahkan dukungan untuk Memcached versi 1.6.22. Ini tidak menyertakan fitur baru, tetapi mencakup perbaikan bug dan pembaruan kumulatif dari Memcached 1.6.18.

Untuk informasi lebih lanjut, lihat ReleaseNotes1622 di Memcached on. GitHub

ElastiCache versi 1.6.17 untuk Memcached

ElastiCache untuk Memcached versi 1.6.17 untuk Memcached menambahkan dukungan untuk mesin Memcached versi 1.6.17. Ini tidak menyertakan fitur baru, tetapi mencakup perbaikan bug dan pembaruan kumulatif dari Memcached 1.6.17.

Untuk informasi lebih lanjut, lihat ReleaseNotes1617 di Memcached on. GitHub

ElastiCache versi 1.6.12 untuk Memcached

ElastiCache untuk Memcached versi 1.6.12 untuk Memcached menambahkan dukungan untuk mesin Memcached 1.6.12 dan enkripsi dalam perjalanan. Ini juga mencakup perbaikan bug dan pembaruan kumulatif dari Memcached 1.6.6.

Untuk informasi lebih lanjut, lihat ReleaseNotes1612 di Memcached on. GitHub

ElastiCache versi 1.6.6 untuk Memcached

ElastiCache untuk Memcached versi 1.6.6 untuk Memcached menambahkan dukungan untuk Memcached versi 1.6.6. Ini tidak mencakup fitur baru, tetapi termasuk perbaikan bug dan pembaruan kumulatif dari Memcached 1.5.16. ElastiCache untuk Memcached tidak termasuk dukungan untuk Extstore.

Untuk informasi lebih lanjut, lihat ReleaseNotes166 di Memcached on. GitHub

ElastiCache versi 1.5.16 untuk Memcached

ElastiCache versi 1.5.16 untuk Memcached menambahkan dukungan untuk Memcached versi 1.5.16. Dukungan ini tidak menyertakan fitur baru, tetapi mencakup perbaikan bug dan pembaruan kumulatif dari Memcached 1.5.14 dan Memcached 1.5.15.

Untuk informasi lebih lanjut, lihat Memcached 1.5.16 Catatan Rilis di Memcached on. GitHub

ElastiCache versi 1.5.10 untuk Memcached

ElastiCache versi 1.5.10 untuk Memcached mendukung fitur Memcached berikut:

- Penyeimbangan ulang slab otomatis.
- Pencarian tabel hash yang lebih cepat dengan algoritma `murmur3`.
- Algoritma LRU tersegmentasi.

- Perayap LRU untuk memori background-reclaim.
- `--enable-seccomp`: Opsi waktu kompilasi.

Juga diperkenalkan parameter `no_modern` dan `inline_ascii_resp`. Untuk informasi selengkapnya, lihat Perubahan parameter Memcached 1.5.10.

Perbaikan memcached yang ditambahkan sejak ElastiCache versi 1.4.34 untuk Memcached meliputi yang berikut:

- Perbaikan kumulatif, seperti multigets ASCII, CVE-2017-9951, dan batas perayapan untuk `metadumper`.
- Manajemen koneksi yang lebih baik dengan menutup koneksi pada batas koneksi.
- Peningkatan manajemen ukuran item untuk ukuran item di atas 1 MB.
- Performa yang lebih baik dan perbaikan overhead memori dengan mengurangi persyaratan memori per-item sebanyak beberapa byte.

Untuk informasi lebih lanjut, lihat Memcached 1.5.10 Catatan Rilis di Memcached pada. GitHub

ElastiCache versi 1.4.34 untuk Memcached

ElastiCache versi 1.4.34 untuk Memcached tidak menambahkan fitur baru ke versi 1.4.33. Versi 1.4.34 adalah rilis perbaikan bug yang lebih besar dari rilis biasa.

Untuk informasi lebih lanjut, lihat Memcached 1.4.34 Catatan Rilis di Memcached pada. GitHub

ElastiCache versi 1.4.33 untuk Memcached

Perbaikan yang ditambahkan sejak versi 1.4.24 meliputi yang berikut:

- Kemampuan untuk membuang semua metadata untuk kelas slab tertentu, daftar kelas slab, atau semua kelas slab. Untuk informasi selengkapnya, lihat Memcached 1.4.31 Release Notes.
- Peningkatan dukungan untuk item besar melebihi default 1 megabyte. Untuk informasi selengkapnya, lihat Memcached 1.4.29 Release Notes.
- Kemampuan untuk menentukan berapa lama klien dapat idle sebelum diminta untuk menutup.

  Kemampuan untuk secara dinamis meningkatkan jumlah memori yang tersedia untuk Memcached tanpa harus memulai ulang klaster. Untuk informasi selengkapnya, lihat Memcached 1.4.27 Release Notes.

- Saat ini, pencatatan log dari `fetchers`, `mutations`, dan `evictions` didukung. Untuk informasi selengkapnya, lihat Memcached 1.4.26 Release Notes.
- Memori yang sudah dibebaskan dapat diklaim kembali ke dalam pool global dan ditempatkan ulang ke kelas slab baru. Untuk informasi selengkapnya, lihat Memcached 1.4.25 Release Notes.
- Beberapa perbaikan bug.
- Beberapa perintah dan parameter baru. Untuk daftarnya, lihat Parameter yang ditambahkan di Memcached 1.4.33.

ElastiCache versi 1.4.24 untuk Memcached

Perbaikan yang ditambahkan sejak versi 1.4.14 meliputi yang berikut:

- Manajemen least recently used (LRU) menggunakan proses di latar belakang.
- Menambahkan opsi untuk menggunakan jenkins atau murmur3 sebagai algoritma hash Anda.
- Beberapa perintah dan parameter baru. Untuk daftarnya, lihat Parameter yang ditambahkan di Memcached 1.4.24.
- Beberapa perbaikan bug.

ElastiCache versi 1.4.14 untuk Memcached

Perbaikan ditambahkan sejak versi 1.4.5 termasuk yang berikut:

- Kemampuan penyeimbangan kembali slab yang ditingkatkan.
- Peningkatan performa dan skalabilitas.
- Memperkenalkan perintah touch untuk memperbarui waktu kedaluwarsa dari item yang sudah ada tanpa mengambilnya.
- Penemuan otomatis—kemampuan program klien untuk menentukan secara otomatis semua simpul cache pada klaster, dan untuk memulai serta memelihara koneksi ke semua simpul ini.

ElastiCache versi 1.4.5 untuk Memcached

ElastiCache versi 1.4.5 untuk Memcached adalah mesin awal dan versi yang didukung oleh Amazon ElastiCache untuk Memcached.

Perilaku versi mesin utama dan perbedaan kompatibilitas dengan Valkey

Valkey 7.2.6 memiliki perbedaan kompatibilitas yang serupa dengan versi Redis OSS 7.2.4 sebelumnya. Untuk versi terbaru Valkey yang didukung, lihatMesin dan versi yang didukung.

Untuk informasi lebih lanjut tentang rilis Valkey 7.2, lihat Catatan Rilis Redis OSS 7.2.4 (Valkey 7.2 mencakup semua perubahan dari Redis OSS hingga versi 7.2.4) dan catatan rilis Valkey 7.2 di Valkey pada. GitHub

Berikut adalah perubahan perilaku yang berpotensi melanggar antara Valkey 7.2 dan Redis OSS 7.1 (atau 7.0):

- Freeze time sampling terjadi selama eksekusi perintah dan dalam skrip.
- Perintah aliran yang diblokir yang dirilis saat kunci tidak ada lagi membawa kode kesalahan yang berbeda (-NOGROUP atau -WRONGTYPE alih-alih -UNBLOCKED).
- Pelacakan sisi klien untuk skrip sekarang melacak kunci yang dibaca oleh skrip, bukan kunci yang dideklarasikan oleh pemanggil EVAL /FCALL.

Perilaku versi mesin utama dan perbedaan kompatibilitas dengan Redis OSS

> **Important**
>
> Halaman berikut disusun untuk menandakan semua perbedaan inkompatibilitas antarversi dan memberi tahu Anda tentang pertimbangan apa pun yang harus Anda buat saat meningkatkan ke versi yang lebih baru. Daftar ini termasuk masalah inkompatibilitas versi apa pun yang mungkin Anda temui saat melakukan peningkatan.
> Anda dapat meng-upgrade langsung dari versi Redis OSS Anda saat ini ke versi Redis OSS terbaru yang tersedia, tanpa perlu upgrade berurutan. Misalnya, Anda dapat meng-upgrade langsung dari Redis OSS versi 3.0 ke versi 7.0.

Versi Redis OSS diidentifikasi dengan versi semantik yang terdiri dari komponen mayor, minor, dan patch. Misalnya, di Redis OSS 4.0.10, versi utama adalah 4, versi minor 0, dan versi patch adalah 10. Nilai-nilai ini umumnya bertambah berdasarkan konvensi berikut:

- Versi utama adalah untuk perubahan API yang tidak kompatibel

- Versi minor adalah untuk fungsionalitas baru yang ditambahkan dengan cara yang kompatibel ke belakang
- Versi patch adalah untuk perbaikan bug yang kompatibel ke belakang dan perubahan non-fungsional

Kami menyarankan untuk selalu menggunakan versi patch terbaru dalam versi mayor.minor tertentu untuk mendapatkan peningkatan kinerja dan stabilitas terbaru. Dimulai dengan ElastiCache versi 6.0 untuk Redis OSS, ElastiCache akan menawarkan versi tunggal untuk setiap rilis minor Redis OSS daripada menawarkan beberapa versi patch. ElastiCacheakan secara otomatis mengelola versi patch dari cluster cache Anda yang sedang berjalan, memastikan peningkatan kinerja dan keamanan yang ditingkatkan.

Kami juga merekomendasikan melakukan peningkatan secara berkala ke versi utama terbaru, karena sebagian besar perbaikan utama tidak kembali dipindahkan ke versi lama. Karena ElastiCache memperluas ketersediaan ke AWS wilayah baru, ElastiCache untuk Redis OSS mendukung dua versi mayor.minor terbaru pada waktu itu untuk wilayah baru. Misalnya, jika AWS wilayah baru diluncurkan dan versi mayor.minor terbaru untuk Redis OSS adalah 7.0 dan 6.2, ElastiCache akan mendukung Redis OSS ElastiCache versi 7.0 dan 6.2 di wilayah baru. AWS Karena versi mayor.minor yang lebih baru ElastiCache untuk Redis OSS dirilis, ElastiCache akan terus menambahkan dukungan untuk versi yang baru dirilis. Untuk mempelajari selengkapnya tentang memilih wilayah ElastiCache, lihat Memilih wilayah dan zona ketersediaan.

Saat melakukan pemutakhiran yang mencakup versi mayor atau minor, harap pertimbangkan daftar berikut yang mencakup perilaku dan perubahan tidak kompatibel mundur yang dirilis dengan Redis OSS dari waktu ke waktu.

Perilaku Redis OSS 7.0 dan perubahan yang tidak kompatibel ke belakang

Untuk daftar lengkap perubahan, lihat Catatan rilis Redis OSS 7.0.

- `SCRIPT LOAD` dan `SCRIPT FLUSH` tidak lagi disebarkan ke replika. Jika Anda perlu memiliki beberapa daya tahan untuk skrip, kami sarankan Anda mempertimbangkan untuk menggunakan fungsi Redis OSS.
- Saluran Pubsub sekarang diblokir secara default untuk pengguna ACL baru.
- Perintah `STRALGO` diganti dengan perintah `LCS`.

- Format untuk `ACL GETUSER` telah berubah sehingga semua bidang menunjukkan pola string akses standar. Jika Anda menggunakan otomatisasi `ACL GETUSER`, Anda harus memverifikasi bahwa itu akan menangani salah satu format.
- Kategori ACL untuk `SELECT`, `WAIT`, `ROLE`, `LASTSAVE`, `READONLY`, `READWRITE`, dan `ASKING` telah berubah.
- Perintah `INFO` sekarang menunjukkan statistik perintah per sub-perintah, bukan di perintah kontainer tingkat atas.
- Nilai pengembalian perintah `LPOP`, `RPOP`, `ZPOPMIN` dan `ZPOPMAX` telah berubah dalam kasus ekstrem tertentu. Jika Anda menggunakan perintah ini, Anda harus memeriksa catatan rilis dan mengevaluasi apakah Anda terpengaruh.
- Perintah `SORT` dan `SORT_RO` sekarang memerlukan akses ke seluruh ruang kunci untuk menggunakan argumen `GET` dan `BY`.

Redis OSS 6.2 perilaku dan perubahan mundur yang tidak kompatibel

Untuk daftar lengkap perubahan, lihat Catatan rilis Redis OSS 6.2.

- Bendera ACL perintah `TIME`, `ECHO`, `ROLE`, dan `LASTSAVE` telah diubah. Hal ini dapat menyebabkan perintah yang sebelumnya diizinkan untuk ditolak dan sebaliknya.

> **Note**
>
> Tak satu pun dari perintah ini akan mengubah atau memberikan akses ke data.

- Saat memutakhirkan dari Redis OSS 6.0, urutan key/value pasangan yang dikembalikan dari respons peta ke skrip lua diubah. Jika skrip Anda menggunakan `redis.setresp()` atau mengembalikan peta (baru di Redis OSS 6.0), pertimbangkan implikasi bahwa skrip dapat rusak pada peningkatan.

Perilaku Redis OSS 6.0 dan perubahan yang tidak kompatibel ke belakang

Untuk daftar lengkap perubahan, lihat Catatan rilis Redis OSS 6.0.

- Jumlah maksimum basis data yang diizinkan telah dikurangi dari 1,2 juta menjadi 10 ribu. Nilai default-nya adalah 16, dan kami tidak menyarankan penggunaan nilai yang jauh lebih besar dari ini karena kami telah menemukan masalah performa dan memori.

- Setel `AutoMinorVersionUpgrade` parameter ke ya, dan ElastiCache akan mengelola peningkatan versi minor melalui pembaruan layanan mandiri. Hal ini akan ditangani lewat saluran notifikasi pelanggan standar melalui kampanye pembaruan mandiri. Untuk informasi selengkapnya, lihat Pembaruan layanan mandiri di ElastiCache.

Perilaku Redis OSS 5.0 dan perubahan yang tidak kompatibel ke belakang

Untuk daftar lengkap perubahan, lihat Catatan rilis Redis OSS 5.0.

- Skrip dengan direplikasi oleh efek bukannya mengeksekusi ulang skrip pada replika. Hal ini umumnya akan meningkatkan performa tetapi dapat meningkatkan jumlah data yang direplikasi antara primer dan replika. Ada opsi untuk kembali ke perilaku sebelumnya yang hanya tersedia di ElastiCache versi 5.0 untuk Redis OSS.
- Jika Anda memutakhirkan dari Redis OSS 4.0, beberapa perintah dalam skrip LUA akan mengembalikan argumen dalam urutan yang berbeda dari yang mereka lakukan di versi sebelumnya. Dalam Redis OSS 4.0, Redis OSS akan memesan beberapa tanggapan secara leksografis untuk membuat respons deterministik, urutan ini tidak diterapkan ketika skrip direplikasi oleh efek.
- Inn Redis OSS 5.0.3 dan yang lebih baru, ElastiCache untuk Redis OSS akan menurunkan beberapa pekerjaan IO ke inti latar belakang pada tipe instance dengan lebih dari 4. VCPUs Ini dapat mengubah karakteristik kinerja Redis OSS dan mengubah nilai beberapa metrik. Untuk informasi selengkapnya, lihat Metrik Apa yang Harus Saya Pantau? untuk memahami apakah Anda perlu mengubah metrik yang Anda lihat.

Perilaku Redis OSS 4.0 dan perubahan yang tidak kompatibel ke belakang

Untuk daftar lengkap perubahan, lihat Catatan rilis Redis OSS 4.0.

- Log lambat sekarang mencatat log dua argumen tambahan, nama klien dan alamat. Perubahan ini harus kompatibel dengan versi lama kecuali jika Anda secara eksplisit mengandalkan setiap entri log lambat yang berisi 3 nilai.
- Perintah `CLUSTER NODES` sekarang mengembalikan format yang sedikit berbeda, yang tidak kompatibel dengan versi lama. Sebaiknya klien tidak menggunakan perintah ini untuk mempelajari simpul yang ada di klaster, dan sebagai gantinya harus menggunakan `CLUSTER SLOTS`.

EOL terdahulu

Perilaku Redis OSS 3.2 dan perubahan yang tidak kompatibel ke belakang

Untuk daftar lengkap perubahan, lihat Catatan rilis Redis OSS 3.2.

- Tidak ada perubahan kompatibilitas untuk memanggil versi ini.

Untuk informasi selengkapnya, lihat ElastiCache versi untuk jadwal akhir hidup Redis OSS.

Redis OSS 2.8 perilaku dan perubahan mundur yang tidak kompatibel

Untuk daftar lengkap perubahan, lihat Catatan rilis Redis OSS 2.8.

- Mulai Redis OSS 2.8.22, Redis OSS AOF tidak lagi didukung untuk Redis OSS. ElastiCache Sebaiknya gunakan MemoryDB jika data perlu dipertahankan tahan lama.
- Mulai dari Redis OSS 2.8.22, ElastiCache untuk Redis OSS tidak lagi mendukung melampirkan replika ke pendahuluan yang dihosting di dalamnya. ElastiCache Saat melakukan peningkatan, replika eksternal akan terputus dan tidak akan dapat terhubung kembali. Sebaiknya gunakan caching sisi klien, tersedia di Redis OSS 6.0 sebagai alternatif replika eksternal.
- Perintah `TTL` dan `PTTL` sekarang mengembalikan -2 jika kunci tidak ada dan -1 jika ada tetapi tidak memiliki kedaluwarsa terkait. Redis OSS 2.6 dan versi sebelumnya digunakan untuk mengembalikan -1 untuk kedua kondisi.
- `SORT` dengan `ALPHA` sekarang melakukan pengurutan berdasarkan lokal pengumpulan lokal jika tidak ada opsi `STORE` yang digunakan.

Lihat informasi yang lebih lengkap di ElastiCache versi untuk jadwal akhir hidup Redis OSS.

Pertimbangan peningkatan saat menangani klaster yang dirancang sendiri

> **Note**
>
> Pertimbangan berikut hanya berlaku saat meningkatkan klaster yang dirancang sendiri. Mereka tidak berlaku untuk Tanpa ElastiCache Server.

Pertimbangan Valkey dan Redis OSS

Saat memutakhirkan cluster Valkey atau Redis OSS yang dirancang sendiri, pertimbangkan hal berikut.

- Manajemen versi mesin dirancang agar Anda dapat memiliki kontrol sebanyak mungkin terkait cara melakukan patching. Namun, ElastiCache berhak untuk menambal klaster Anda atas nama Anda jika terjadi kerentanan keamanan kritis dalam sistem atau perangkat lunak cache.
- Dimulai dengan ElastiCache versi 7.2 untuk Valkey dan ElastiCache versi 6.0 untuk Redis OSS, ElastiCache akan menawarkan versi tunggal untuk setiap rilis minor, daripada menawarkan beberapa versi patch.
- Dimulai dengan mesin Redis OSS versi 5.0.6, Anda dapat meng-upgrade versi cluster Anda dengan downtime minimal. Klaster tersedia untuk operasi baca selama keseluruhan proses peningkatan dan tersedia untuk operasi tulis untuk sebagian besar durasi peningkatan, kecuali selama operasi failover yang berlangsung beberapa detik.
- Anda juga dapat meng-upgrade ElastiCache cluster Anda dengan versi lebih awal dari 5.0.6. Tindakan ini melibatkan proses yang sama tetapi mungkin mengalami waktu failover lebih lama selama penyebaran DNS (30 detik - 1 menit).
- Dimulai dengan Redis OSS 7, ElastiCache mendukung peralihan antara Valkey atau Redis OSS (mode cluster dinonaktifkan) dan Valkey atau Redis OSS (mode cluster diaktifkan).
- Proses upgrade mesin Amazon ElastiCache for Redis OSS dirancang untuk melakukan upaya terbaik untuk mempertahankan data Anda yang ada dan memerlukan replikasi Redis OSS yang berhasil.
- Saat memutakhirkan mesin, ElastiCache akan menghentikan koneksi klien yang ada. Untuk meminimalkan waktu henti selama peningkatan engine, kami sarankan Anda menerapkan praktik terbaik untuk klien Redis OSS dengan percobaan ulang kesalahan dan backoff eksponensial dan praktik terbaik untuk meminimalkan waktu henti selama pemeliharaan.
- Anda tidak dapat memutakhirkan langsung dari Valkey atau Redis OSS (mode cluster dinonaktifkan) ke Valkey atau Redis OSS (mode cluster diaktifkan) saat Anda meningkatkan mesin Anda. Prosedur berikut menunjukkan kepada Anda cara meningkatkan dari Valkey atau Redis OSS (mode cluster dinonaktifkan) ke Valkey atau Redis OSS (mode cluster diaktifkan).

  Untuk meningkatkan dari Valkey atau Redis OSS (mode cluster dinonaktifkan) ke Valkey atau Redis OSS (mode cluster diaktifkan) versi mesin

  1. Buat cadangan cluster atau grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) Anda. Untuk informasi selengkapnya, lihat Membuat cadangan manual.

2. Gunakan cadangan untuk membuat dan menyemai klaster Valkey atau Redis OSS (mode cluster enabled) dengan satu shard (grup node). Tentukan versi mesin baru dan aktifkan mode klaster saat membuat klaster atau grup replikasi. Untuk informasi selengkapnya, lihat Tutorial: Menyemai cluster baru yang dirancang sendiri dengan cadangan yang dibuat secara eksternal.
3. Hapus cluster atau grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) lama. Untuk informasi selengkapnya, lihat Menghapus cluster di ElastiCache atau Menghapus grup replikasi.
4. Skala klaster Valkey atau Redis OSS (mode cluster diaktifkan) baru atau grup replikasi ke jumlah pecahan (grup simpul) yang Anda butuhkan. Untuk informasi selengkapnya, lihat Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan)

- Saat meningkatkan versi utama mesin, misalnya dari 5.0.6 ke 6.0, Anda juga harus memilih grup parameter baru yang kompatibel dengan versi mesin yang baru tersebut.
- Untuk cluster dan cluster Redis OSS tunggal dengan multi-AZ dinonaktifkan, kami menyarankan agar memori yang cukup tersedia untuk Redis OSS seperti yang dijelaskan dalam. Memastikan Anda memiliki cukup memori untuk membuat snapshot Valkey atau Redis OSS Dalam kasus ini, klaster primer tidak tersedia untuk melayani permintaan selama proses peningkatan.
- Untuk kluster Redis OSS dengan Multi-AZ diaktifkan, kami juga menyarankan Anda menjadwalkan peningkatan mesin selama periode lalu lintas tulis masuk yang rendah. Saat memutakhirkan ke Redis OSS 5.0.6 atau lebih tinggi, klaster utama terus tersedia untuk permintaan layanan selama proses peningkatan.

  Klaster dan grup replikasi dengan beberapa serpihan diproses dan di-patch sebagai berikut:

  - Semua serpihan diproses secara paralel. Hanya satu operasi peningkatan yang akan dilakukan pada satu serpihan kapan saja.
  - Di setiap serpihan, semua replika diproses sebelum primer diproses. Jika terdapat lebih sedikit replika dalam serpihan, primer dalam serpihan itu mungkin diproses sebelum replika di serpihan lainnya selesai diproses.
  - Di semua serpihan, simpul primer diproses secara berurutan. Hanya satu simpul primer yang ditingkatkan dalam satu waktu.
- Jika enkripsi diaktifkan di klaster atau grup replikasi Anda saat ini, Anda tidak dapat melakukan peningkatan ke versi mesin yang tidak mendukung enkripsi, seperti dari 3.2.6 ke 3.2.10.

Pertimbangan memcached

Saat memutakhirkan cluster Memcached yang dirancang sendiri, pertimbangkan hal berikut.

- Manajemen versi mesin dirancang agar Anda dapat memiliki kontrol sebanyak mungkin terkait cara melakukan patching. Namun, ElastiCache berhak untuk menambal klaster Anda atas nama Anda jika terjadi kerentanan keamanan kritis dalam sistem atau perangkat lunak cache.
- Karena mesin Memcached tidak mendukung persistensi, peningkatan versi mesin Memcached merupakan proses disruptif yang menghilangkan semua data cache di klaster.

ElastiCache praktik terbaik dan strategi caching

Di bawah ini Anda dapat menemukan praktik terbaik yang direkomendasikan untuk Amazon ElastiCache. Mengikuti langkah ini akan meningkatkan performa dan keandalan cache Anda.

Topik

Praktik terbaik secara keseluruhan

Di bawah ini Anda dapat menemukan informasi tentang praktik terbaik untuk menggunakan antarmuka Valkey, Memcached, dan Redis OSS di dalamnya. ElastiCache

- Gunakan konfigurasi berkemampuan mode cluster — Mode cluster yang diaktifkan memungkinkan cache untuk menskalakan secara horizontal untuk mencapai penyimpanan dan throughput yang lebih tinggi daripada konfigurasi yang dinonaktifkan mode cluster. ElastiCache tanpa server hanya tersedia dalam konfigurasi yang diaktifkan mode cluster.

- Gunakan koneksi berumur panjang – Pembuatan koneksi baru menghabiskan banyak daya komputasi, serta membutuhkan waktu dan sumber daya CPU dari cache. Gunakan kembali koneksi jika memungkinkan (misalnya dengan pooling koneksi) untuk menyebarkan beban ini melalui banyak perintah.
- Baca dari replika — Jika Anda menggunakan ElastiCache tanpa server atau telah menyediakan replika baca (cluster yang dirancang sendiri), arahkan pembacaan ke replika untuk mencapai skalabilitas yang lebih baik latensi yang lebih rendah. and/or Pembacaan dari replika pada akhirnya akan konsisten dengan yang primer.

  Dalam klaster yang dirancang sendiri, hindari mengarahkan permintaan baca ke satu replika baca karena pembacaan mungkin tidak tersedia sementara jika simpul gagal. Konfigurasikan klien Anda untuk mengarahkan permintaan baca ke setidaknya dua replika baca, atau arahkan permintaan baca ke replika tunggal dan primer.

  Dalam ElastiCache tanpa server, membaca dari port replika (6380) akan mengarahkan pembacaan ke zona ketersediaan lokal klien jika memungkinkan, mengurangi latensi pengambilan. Hal ini akan secara otomatis melakukan fallback ke simpul lainnya selama kegagalan.
- Hindari perintah mahal — Hindari menjalankan operasi komputasi dan I/O intensif, seperti perintah `KEYS` dan `SMEMBERS` perintah. Pendekatan ini disarankan karena operasi ini meningkatkan beban pada klaster dan memiliki dampak pada performa klaster. Sebagai gantinya, gunakan perintah `SCAN` dan `SSCAN`.
- Ikuti praktik terbaik Lua – Hindari menjalankan skrip Lua terlalu lama, dan selalu nyatakan kunci yang digunakan dalam skrip Lua di depan. Pendekatan ini disarankan untuk menentukan bahwa skrip Lua tidak menggunakan perintah cross slot. Pastikan bahwa kunci yang digunakan dalam skrip Lua adalah milik slot yang sama.
- Gunakan pub/sub sharded - Saat menggunakan Valkey atau Redis OSS untuk mendukung pub/sub beban kerja dengan throughput tinggi, kami sarankan Anda menggunakan pub/sub sharded (tersedia dengan Valkey, dan dengan Redis OSS 7 atau lebih baru). Cluster yang diaktifkan mode cluster tradisional pub/sub menyiarkan pesan ke semua node di cluster, yang dapat menghasilkan high. `EngineCPUUtilization` Perhatikan bahwa dalam ElastiCache perintah tradisional pub/sub commands internally use sharded pub/sub tanpa server.

Praktik Terbaik untuk menggunakan Read Replicas

Banyak aplikasi, seperti toko sesi, papan peringkat, dan mesin rekomendasi, memerlukan ketersediaan tinggi dan menangani operasi baca secara signifikan lebih banyak daripada operasi

tulis. Aplikasi ini sering dapat mentolerir data yang sedikit basi (konsistensi akhirnya), yang berarti bahwa itu dapat diterima jika pengguna yang berbeda sejenak melihat versi yang sedikit berbeda dari data yang sama. Misalnya:

- Hasil kueri yang di-cache seringkali dapat mentolerir data yang sedikit basi, terutama untuk pola selain cache di mana sumber kebenarannya adalah eksternal.
- Dalam papan peringkat game, penundaan beberapa detik dalam skor yang diperbarui seringkali tidak akan berdampak signifikan pada pengalaman pengguna.
- Untuk penyimpanan sesi, beberapa penundaan kecil dalam menyebarkan data sesi di seluruh replika jarang memengaruhi fungsionalitas aplikasi.
- Mesin rekomendasi biasanya menggunakan analisis data historis, sehingga konsistensi waktu nyata kurang penting.

Konsistensi akhirnya berarti bahwa semua node replika pada akhirnya akan mengembalikan data yang sama setelah proses replikasi selesai, biasanya dalam milidetik. Untuk kasus penggunaan seperti itu, menerapkan replika baca adalah strategi yang efektif untuk mengurangi latensi saat membaca dari instans Anda ElastiCache.

Menggunakan replika baca di Amazon ElastiCache dapat memberikan manfaat kinerja yang signifikan melalui:

Peningkatan Skalabilitas Baca

- Mendistribusikan operasi baca di beberapa node replika
- Offload membaca lalu lintas dari simpul utama
- Mengurangi latensi baca dengan menyajikan permintaan dari replika yang lebih dekat secara geografis

Kinerja Node Primer yang Dioptimalkan

- Mendedikasikan sumber daya node utama untuk menulis operasi
- Mengurangi overhead koneksi pada node utama
- Meningkatkan kinerja penulisan dan mempertahankan waktu respons yang lebih baik selama periode lalu lintas puncak

Menggunakan Baca dari Replika di Tanpa Server ElastiCache

ElastiCache tanpa server menyediakan dua titik akhir yang berbeda, untuk persyaratan konsistensi yang berbeda. Kedua titik akhir menggunakan nama DNS yang sama tetapi port yang berbeda. Untuk menggunakan read-from-replica port, Anda harus mengotorisasi akses ke kedua port dari aplikasi klien Anda dengan mengonfigurasi grup keamanan dan daftar kontrol akses jaringan VPC Anda.

Titik akhir primer (Port 6379)

- Gunakan untuk operasi yang membutuhkan konsistensi segera
- Jaminan membaca up-to-date data terbanyak
- Terbaik untuk transaksi kritis dan operasi tulis
- Diperlukan untuk operasi menulis
- Contoh: `test-12345.serverless.use1.cache.amazonaws.com:6379`

Titik akhir yang dioptimalkan latensi (Port 6380)

- Dioptimalkan untuk operasi baca yang dapat mentolerir konsistensi akhirnya
- Jika memungkinkan, ElastiCache tanpa server secara otomatis merutekan permintaan baca ke node replika di Availability Zone lokal klien. Optimalisasi ini memberikan latensi yang lebih rendah dengan menghindari latensi jaringan tambahan yang terjadi saat mengambil data dari node di zona ketersediaan yang berbeda.
- ElastiCache tanpa server secara otomatis memilih node yang tersedia di zona lain jika node lokal tidak tersedia
- Contoh: `test-12345.serverless.use1.cache.amazonaws.com:6380`
- Klien seperti Glide dan Lettuce akan secara otomatis mendeteksi dan merutekan pembacaan ke titik akhir yang dioptimalkan latensi jika Anda memberikan konfigurasi replika baca dari. Jika klien Anda tidak mendukung konfigurasi routing (misalnya, valkey-java dan versi jedis yang lebih lama), Anda harus menentukan port dan konfigurasi klien yang tepat untuk membaca dari replika.

Menghubungkan untuk membaca replika di ElastiCache Tanpa Server - Valkey dan Glide

Cuplikan kode berikut menunjukkan bagaimana Anda dapat mengonfigurasi baca dari replika untuk ElastiCache Tanpa Server di pustaka glide Valkey. Anda tidak perlu menentukan port untuk dibaca dari replika, tetapi Anda perlu mengonfigurasi konfigurasi perutean. `ReadFrom.PREFER_REPLICA`

```
package glide.examples;

import glide.api.GlideClusterClient;
import glide.api.logging.Logger;
import glide.api.models.configuration.GlideClusterClientConfiguration;
import glide.api.models.configuration.NodeAddress;
import glide.api.models.exceptions.ClosingException;
import glide.api.models.exceptions.ConnectionException;
import glide.api.models.exceptions.TimeoutException;
import glide.api.models.configuration.ReadFrom;

import java.util.concurrent.CompletableFuture;
import java.util.concurrent.ExecutionException;

public class ClusterExample {

    public static void main(String[] args) {
        // Set logger configuration
        Logger.setLoggerConfig(Logger.Level.INFO);

        GlideClusterClient client = null;

        try {
            System.out.println("Connecting to Valkey Glide...");

            // Configure the Glide Client
            GlideClusterClientConfiguration config =
 GlideClusterClientConfiguration.builder()
                .address(NodeAddress.builder()
                    .host("your-endpoint")
                    .port(6379)
                    .build())
                .useTLS(true)
                .readFrom(ReadFrom.PREFER_REPLICA)
                .build();

            // Create the GlideClusterClient
            client = GlideClusterClient.createClient(config).get();
            System.out.println("Connected successfully.");

            // Perform SET operation
            CompletableFuture<String> setResponse = client.set("key", "value");
            System.out.println("Set key 'key' to 'value': " + setResponse.get());
```

```
            // Perform GET operation
            CompletableFuture<String> getResponse = client.get("key");
            System.out.println("Get response for 'key': " + getResponse.get());

            // Perform PING operation
            CompletableFuture<String> pingResponse = client.ping();
            System.out.println("PING response: " + pingResponse.get());

        } catch (ClosingException | ConnectionException | TimeoutException |
 ExecutionException e) {
            System.err.println("An exception occurred: ");
            e.printStackTrace();
        } catch (InterruptedException e) {
            Thread.currentThread().interrupt();
        } finally {
            // Close the client connection
            if (client != null) {
                try {
                    client.close();
                    System.out.println("Client connection closed.");
                } catch (ClosingException | ExecutionException e) {
                    System.err.println("Error closing client: " + e.getMessage());
                }
            }
        }
    }
}
```

Perintah Valkey, Memcached, dan Redis OSS yang didukung dan dibatasi

Perintah Valkey dan Redis OSS yang didukung

Perintah Valkey dan Redis OSS yang didukung

Perintah Valkey dan Redis OSS berikut didukung oleh cache tanpa server. Selain perintah ini, Perintah Valkey dan Redis OSS yang didukung ini juga didukung.

Untuk informasi tentang perintah Bloom Filter lihat Perintah filter Bloom

Perintah Bitmap

- `BITCOUNT`

Menghitung jumlah bit set (penghitungan populasi) dalam string.

Pelajari selengkapnya

- `BITFIELD`

  Melakukan operasi bilangan bulat bitfield arbitrer pada string.

  Pelajari selengkapnya

- `BITFIELD_RO`

  Melakukan operasi bilangan bulat bitfield hanya-baca arbitrer pada string.

  Pelajari selengkapnya

- `BITOP`

  Melakukan operasi bitwise pada beberapa string, dan menyimpan hasilnya.

  Pelajari selengkapnya

- `BITPOS`

  Menemukan set pertama (1) atau menghapus (0) bit dalam string.

  Pelajari selengkapnya

- `GETBIT`

  Mengembalikan nilai bit dengan offset.

  Pelajari selengkapnya

- `SETBIT`

  Menetapkan atau menghapus bit pada offset dari nilai string. Membuat kunci jika tidak ada.

  Pelajari selengkapnya

Perintah Manajemen Klaster

- `CLUSTER COUNTKEYSINSLOT`

  Mengembalikan jumlah kunci dalam slot hash.

Pelajari selengkapnya

- `CLUSTER GETKEYSINSLOT`

  Mengembalikan nama kunci dalam slot hash.

  Pelajari selengkapnya

- `CLUSTER INFO`

  Mengembalikan informasi tentang keadaan simpul. Dalam cache nirserver, mengembalikan status tentang "serpihan" virtual tunggal yang diekspos ke klien.

  Pelajari selengkapnya

- `CLUSTER KEYSLOT`

  Mengembalikan slot hash untuk kunci.

  Pelajari selengkapnya

- `CLUSTER MYID`

  Mengembalikan ID simpul. Dalam cache nirserver, mengembalikan status tentang "serpihan" virtual tunggal yang diekspos ke klien.

  Pelajari selengkapnya

- `CLUSTER NODES`

  Mengembalikan konfigurasi klaster untuk simpul. Dalam cache nirserver, mengembalikan status tentang "serpihan" virtual tunggal yang diekspos ke klien.

  Pelajari selengkapnya

- `CLUSTER REPLICAS`

  Daftar simpul replika dari simpul utama. Dalam cache nirserver, mengembalikan status tentang "serpihan" virtual tunggal yang diekspos ke klien.

  Pelajari selengkapnya

- `CLUSTER SHARDS`

  Mengembalikan pemetaan slot klaster ke serpihan. Dalam cache nirserver, mengembalikan status tentang "serpihan" virtual tunggal yang diekspos ke klien.

Pelajari selengkapnya

- `CLUSTER SLOTS`

  Mengembalikan pemetaan slot klaster ke simpul. Dalam cache nirserver, mengembalikan status tentang "serpihan" virtual tunggal yang diekspos ke klien.

  Pelajari selengkapnya

- `CLUSTER SLOT-STATS`

  Memungkinkan pelacakan metrik per slot untuk jumlah kunci, pemanfaatan CPU, byte jaringan masuk, dan byte keluar jaringan.

  Pelajari selengkapnya

- `READONLY`

  Mengaktifkan kueri hanya-baca untuk koneksi ke node replika Valkey atau Redis OSS Cluster.

  Pelajari selengkapnya

- `READWRITE`

  Mengaktifkan kueri baca-tulis untuk koneksi ke node replika Valkey atau Redis OSS Cluster.

  Pelajari selengkapnya

- `SCRIPT SHOW`

  Mengembalikan kode sumber asli dari script dalam cache script.

  Pelajari selengkapnya

Perintah Manajemen Koneksi

- `AUTH`

  Mengautentikasi koneksi.

  Pelajari selengkapnya

- `CLIENT GETNAME`

  Mengembalikan nama koneksi.

Pelajari selengkapnya

- `CLIENT REPLY`

  Menginstruksikan server apakah akan membalas perintah.

  Pelajari selengkapnya

- `CLIENT SETNAME`

  Menetapkan nama koneksi.

  Pelajari selengkapnya

- `ECHO`

  Mengembalikan string yang diberikan.

  Pelajari selengkapnya

- `HELLO`

  Jabat tangan dengan server Valkey atau Redis OSS.

  Pelajari selengkapnya

- `PING`

  Mengembalikan respon keaktifan server.

  Pelajari selengkapnya

- `QUIT`

  Menutup koneksi.

  Pelajari selengkapnya

- `RESET`

  Mereset koneksi.

  Pelajari selengkapnya

- `SELECT`

  Mengubah basis data yang dipilih.

Pelajari selengkapnya

Perintah Generik

- `COPY`

  Menyalin nilai kunci ke kunci baru.

  Pelajari selengkapnya

- `DEL`

  Menghapus satu atau beberapa tombol.

  Pelajari selengkapnya

- `DUMP`

  Mengembalikan representasi terserialisasi dari nilai yang disimpan pada kunci.

  Pelajari selengkapnya

- `EXISTS`

  Menentukan apakah ada satu kunci atau lebih.

  Pelajari selengkapnya

- `EXPIRE`

  Menetapkan waktu kedaluwarsa kunci dalam hitungan detik.

  Pelajari selengkapnya

- `EXPIREAT`

  Menetapkan waktu kedaluwarsa kunci ke stempel waktu Unix.

  Pelajari selengkapnya

- `EXPIRETIME`

  Mengembalikan waktu kedaluwarsa kunci sebagai stempel waktu Unix.

  Pelajari selengkapnya

- `PERSIST`

  Menghapus waktu kedaluwarsa kunci.

  Pelajari selengkapnya

- `PEXPIRE`

  Menetapkan waktu kedaluwarsa kunci dalam milidetik.

  Pelajari selengkapnya

- `PEXPIREAT`

  Menetapkan waktu kedaluwarsa kunci ke stempel waktu milidetik Unix.

  Pelajari selengkapnya

- `PEXPIRETIME`

  Mengembalikan waktu kedaluwarsa kunci sebagai stempel waktu milidetik Unix.

  Pelajari selengkapnya

- `PTTL`

  Mengembalikan waktu kedaluwarsa dalam milidetik kunci.

  Pelajari selengkapnya

- `RANDOMKEY`

  Mengembalikan nama kunci acak dari basis data.

  Pelajari selengkapnya

- `RENAME`

  Mengganti nama kunci dan menimpa tujuan.

  Pelajari selengkapnya

- `RENAMENX`

  Mengganti nama kunci hanya jika nama kunci target tidak ada.

  Pelajari selengkapnya

- `RESTORE`

  Membuat kunci dari representasi terserialisasi untuk sebuah nilai.

  Pelajari selengkapnya

- `SCAN`

  Melakukan iterasi pada nama kunci dalam basis data.

  Pelajari selengkapnya

- `SORT`

  Mengurutkan elemen dalam daftar, set, atau sorted set, secara opsional menyimpan hasilnya.

  Pelajari selengkapnya

- `SORT_RO`

  Mengembalikan elemen diurutkan dari daftar, set, atau sorted set.

  Pelajari selengkapnya

- `TOUCH`

  Mengembalikan jumlah kunci yang ada dari yang ditentukan setelah memperbarui waktu kunci tersebut terakhir diakses.

  Pelajari selengkapnya

- `TTL`

  Mengembalikan waktu kedaluwarsa dalam detik kunci.

  Pelajari selengkapnya

- `TYPE`

  Menentukan jenis nilai yang disimpan pada kunci.

  Pelajari selengkapnya

- `UNLINK`

  Menghapus satu kunci atau lebih secara asinkron.

Pelajari selengkapnya

Perintah Geospasial

- `GEOADD`

  Menambahkan satu atau beberapa anggota ke indeks geospasial. Kunci dibuat jika tidak ada.

  Pelajari selengkapnya
- `GEODIST`

  Mengembalikan jarak antara dua anggota indeks geospasial.

  Pelajari selengkapnya
- `GEOHASH`

  Mengembalikan anggota dari indeks geospasial sebagai string geohash.

  Pelajari selengkapnya
- `GEOPOS`

  Mengembalikan bujur dan lintang anggota dari indeks geospasial.

  Pelajari selengkapnya
- `GEORADIUS`

  Meminta indeks geospasial untuk anggota dalam jarak dari koordinat, secara opsional menyimpan hasilnya.

  Pelajari selengkapnya
- `GEORADIUS_RO`

  Mengembalikan anggota dari indeks geospasial yang berada dalam jarak dari koordinat.

  Pelajari selengkapnya
- `GEORADIUSBYMEMBER`

  Meminta indeks geospasial untuk anggota dalam jarak dari anggota, secara opsional menyimpan hasilnya.

Pelajari selengkapnya

- `GEORADIUSBYMEMBER_RO`

  Mengembalikan anggota dari indeks geospasial yang berada dalam jarak dari anggota.

  Pelajari selengkapnya

- `GEOSEARCH`

  Kueri indeks geospasial untuk anggota dalam area kotak atau lingkaran.

  Pelajari selengkapnya

- `GEOSEARCHSTORE`

  Kueri indeks geospasial untuk anggota dalam area kotak atau lingkaran, secara opsional menyimpan hasil.

  Pelajari selengkapnya

Perintah Hash

- `HDEL`

  Menghapus satu atau beberapa bidang dan nilainya dari hash. Menghapus hash jika tidak ada bidang yang tersisa.

  Pelajari selengkapnya

- `HEXISTS`

  Menentukan apakah bidang ada dalam hash.

  Pelajari selengkapnya

- `HGET`

  Mengembalikan nilai bidang dalam hash.

  Pelajari selengkapnya

- `HGETALL`

  Mengembalikan semua bidang dan nilai dalam hash.

Pelajari selengkapnya

- HINCRBY

  Menambah nilai bilangan bulat sebuah bidang dalam sebuah hash sebanyak satu angka. Menggunakan 0 sebagai nilai awal jika bidang tidak ada.

  Pelajari selengkapnya

- HINCRBYFLOAT

  Menambah nilai titik mengambang sebuah bidang sebanyak satu angka. Menggunakan 0 sebagai nilai awal jika bidang tidak ada.

  Pelajari selengkapnya

- HKEYS

  Mengembalikan semua bidang dalam hash.

  Pelajari selengkapnya

- HLEN

  Mengembalikan jumlah bidang dalam hash.

  Pelajari selengkapnya

- HMGET

  Mengembalikan nilai-nilai dari semua bidang dalam hash.

  Pelajari selengkapnya

- HMSET

  Menetapkan nilai dari beberapa bidang.

  Pelajari selengkapnya

- HRANDFIELD

  Mengembalikan satu atau beberapa bidang acak dari hash.

  Pelajari selengkapnya

- HSCAN

Melakukan iterasi pada bidang dan nilai hash.

Pelajari selengkapnya

- HSET

  Membuat atau mengubah nilai bidang dalam hash.

  Pelajari selengkapnya

- HSETNX

  Menetapkan nilai bidang dalam hash hanya jika bidang tidak ada.

  Pelajari selengkapnya

- HSTRLEN

  Mengembalikan panjang nilai bidang.

  Pelajari selengkapnya

- HVALS

  Mengembalikan semua nilai dalam hash.

  Pelajari selengkapnya

HyperLogLog Commands

- PFADD

  Menambahkan elemen ke HyperLogLog kunci. Membuat kunci jika tidak ada.

  Pelajari selengkapnya

- PFCOUNT

  Mengembalikan perkiraan kardinalitas himpunan yang diamati oleh HyperLogLog kunci (s).

  Pelajari selengkapnya

- PFMERGE

  Menggabungkan satu atau lebih HyperLogLog nilai ke dalam satu kunci.

Pelajari selengkapnya

Perintah List

- `BLMOVE`

  Memunculkan elemen dari daftar, mendorongnya ke daftar lain, dan mengembalikannya. Memblokir sampai elemen tersedia. Menghapus daftar jika elemen terakhir dipindahkan.

  Pelajari selengkapnya

- `BLMPOP`

  Memunculkan elemen pertama dari salah satu dari beberapa daftar. Memblokir sampai elemen tersedia. Menghapus daftar jika elemen terakhir di-popping.

  Pelajari selengkapnya

- `BLPOP`

  Menghapus dan mengembalikan elemen pertama dalam daftar. Memblokir sampai elemen tersedia. Menghapus daftar jika elemen terakhir di-popping.

  Pelajari selengkapnya

- `BRPOP`

  Menghapus dan mengembalikan elemen terakhir dalam daftar. Memblokir sampai elemen tersedia. Menghapus daftar jika elemen terakhir di-popping.

  Pelajari selengkapnya

- `BRPOPLPUSH`

  Memunculkan elemen dari daftar, mendorongnya ke daftar lain, dan mengembalikannya. Memblokir sampai elemen tersedia sebaliknya. Menghapus daftar jika elemen terakhir di-popping.

  Pelajari selengkapnya

- `LINDEX`

  Mengembalikan elemen dari daftar dengan indeks.

  Pelajari selengkapnya

- `LINSERT`

  Menyisipkan elemen sebelum atau sesudah elemen lain dalam daftar.

  Pelajari selengkapnya

- `LLEN`

  Mengembalikan panjang daftar.

  Pelajari selengkapnya

- `LMOVE`

  Mengembalikan elemen setelah mem-popping elemen tersebut dari satu daftar dan mendorongnya ke yang lain. Menghapus daftar jika elemen terakhir dipindahkan.

  Pelajari selengkapnya

- `LMPOP`

  Mengembalikan beberapa elemen dari daftar setelah menghapusnya. Menghapus daftar jika elemen terakhir di-popping.

  Pelajari selengkapnya

- `LPOP`

  Mengembalikan elemen pertama dalam daftar setelah menghapusnya. Menghapus daftar jika elemen terakhir di-popping.

  Pelajari selengkapnya

- `LPOS`

  Mengembalikan indeks elemen yang cocok dalam daftar.

  Pelajari selengkapnya

- `LPUSH`

  Menambahkan satu atau beberapa elemen ke awal daftar. Membuat kunci jika tidak ada.

  Pelajari selengkapnya

- `LPUSHX`

Menambahkan satu atau beberapa elemen ke daftar hanya jika daftar ada.

Pelajari selengkapnya

- `LRANGE`

  Mengembalikan berbagai elemen dari daftar.

  Pelajari selengkapnya

- `LREM`

  Menghapus elemen dari daftar. Menghapus daftar jika elemen terakhir telah dihapus.

  Pelajari selengkapnya

- `LSET`

  Menetapkan nilai elemen dalam daftar dengan indeksnya.

  Pelajari selengkapnya

- `LTRIM`

  Menghapus elemen dari kedua ujung daftar. Menghapus daftar jika semua elemen dipangkas.

  Pelajari selengkapnya

- `RPOP`

  Mengembalikan dan menghapus elemen terakhir dari daftar. Menghapus daftar jika elemen terakhir di-popping.

  Pelajari selengkapnya

- `RPOPLPUSH`

  Mengembalikan elemen terakhir dari daftar setelah menghapus dan mendorongnya ke daftar lain. Menghapus daftar jika elemen terakhir di-popping.

  Pelajari selengkapnya

- `RPUSH`

  Menambahkan satu atau beberapa elemen ke daftar. Membuat kunci jika tidak ada.

Pelajari selengkapnya

- RPUSHX

  Menambahkan elemen ke daftar hanya jika daftar ada.

  Pelajari selengkapnya

Perintah Pub/Sub

> **Note**
>
> Perintah PUBSUB secara internal menggunakan PUBSUB serpihan, sehingga nama saluran akan dicampur.

- PUBLISH

  Memposting pesan ke saluran.

  Pelajari selengkapnya

- PUBSUB CHANNELS

  Mengembalikan saluran yang aktif.

  Pelajari selengkapnya

- PUBSUB NUMSUB

  Mengembalikan jumlah pelanggan ke saluran.

  Pelajari selengkapnya

- PUBSUB SHARDCHANNELS

  Mengembalikan saluran serpihan aktif.

  Pelajari selengkapnya

- PUBSUB SHARDNUMSUB

  Mengembalikan jumlah pelanggan saluran serpihan.

Pelajari selengkapnya

- `SPUBLISH`

  Memposting pesan ke saluran serpihan

  Pelajari selengkapnya

- `SSUBSCRIBE`

  Mendengarkan pesan yang dipublikasikan ke saluran serpihan.

  Pelajari selengkapnya

- `SUBSCRIBE`

  Mendengarkan pesan yang dipublikasikan ke saluran.

  Pelajari selengkapnya

- `SUNSUBSCRIBE`

  Berhenti mendengarkan pesan yang diposting ke saluran serpihan.

  Pelajari selengkapnya

- `UNSUBSCRIBE`

  Berhenti mendengarkan pesan yang diposting ke saluran.

  Pelajari selengkapnya

Perintah Scripting

- `EVAL`

  Mengeksekusi skrip Lua sisi server.

  Pelajari selengkapnya

- `EVAL_RO`

  Mengeksekusi skrip Lua sisi server hanya-baca.

  Pelajari selengkapnya

- `EVALSHA`

  Mengeksekusi skrip Lua sisi server dengan digest. SHA1

  Pelajari selengkapnya

- `EVALSHA_RO`

  Mengeksekusi skrip Lua sisi server read-only dengan digest. SHA1

  Pelajari selengkapnya

- `SCRIPT EXISTS`

  Menentukan apakah skrip Lua sisi server ada di cache skrip.

  Pelajari selengkapnya

- `SCRIPT FLUSH`

  Saat ini cache skrip no-op dikelola oleh layanan.

  Pelajari selengkapnya

- `SCRIPT LOAD`

  Memuat skrip Lua sisi server ke cache skrip.

  Pelajari selengkapnya

Perintah Manajemen Server

> **Note**
>
> Saat menggunakan ElastiCache cluster yang dirancang sendiri untuk Valkey dan Redis OSS, perintah flush harus dikirim ke setiap primer oleh klien untuk membersihkan semua kunci. ElastiCache Tanpa server untuk Valkey dan Redis OSS bekerja secara berbeda, karena mengabstraksi topologi cluster yang mendasarinya. Hasilnya adalah bahwa di ElastiCache Serverless, `FLUSHDB` dan `FLUSHALL` perintah akan selalu flush semua kunci di seluruh cluster. Untuk alasan ini, perintah flush tidak dapat dimasukkan dalam transaksi Tanpa Server.

- `ACL CAT`

  Daftar kategori ACL, atau perintah dalam kategori.

  Pelajari selengkapnya

- `ACL GENPASS`

  Menghasilkan pseudorandom, kata sandi aman yang dapat digunakan untuk mengidentifikasi pengguna ACL.

  Pelajari selengkapnya

- `ACL GETUSER`

  Menampilkan daftar aturan ACL pengguna.

  Pelajari selengkapnya

- `ACL LIST`

  Membuang aturan efektif dalam format file ACL.

  Pelajari selengkapnya

- `ACL USERS`

  Menampilkan daftar semua pengguna ACL.

  Pelajari selengkapnya

- `ACL WHOAMI`

  Mengembalikan nama pengguna yang diautentikasi koneksi saat ini.

  Pelajari selengkapnya

- `DBSIZE`

  Mengembalikan jumlah kunci dalam basis data yang dipilih saat ini. Operasi ini tidak dijamin atom di semua slot.

  Pelajari selengkapnya

- `COMMAND`

  Mengembalikan informasi mendetail tentang semua perintah.

Pelajari selengkapnya

- `COMMAND COUNT`

  Mengembalikan hitungan perintah.

  Pelajari selengkapnya

- `COMMAND DOCS`

  Mengembalikan informasi dokumenter tentang satu, beberapa, atau semua perintah.

  Pelajari selengkapnya

- `COMMAND GETKEYS`

  Mengekstrak nama kunci dari perintah arbitrer.

  Pelajari selengkapnya

- `COMMAND GETKEYSANDFLAGS`

  Mengekstrak nama kunci dan bendera akses untuk perintah arbitrer.

  Pelajari selengkapnya

- `COMMAND INFO`

  Mengembalikan informasi tentang satu, beberapa, atau semua perintah.

  Pelajari selengkapnya

- `COMMAND LIST`

  Mengembalikan daftar nama perintah.

  Pelajari selengkapnya

- `COMMANDLOG`

  Sebuah wadah untuk perintah log perintah.

  Pelajari selengkapnya

- `COMMANDLOG GET`

  Mengembalikan entri log perintah yang ditentukan.

Pelajari selengkapnya

- `COMMANDLOG HELP`

  Tampilkan teks bermanfaat tentang subperintah yang berbeda.

  Pelajari selengkapnya

- `COMMANDLOG LEN`

  Mengembalikan jumlah entri dalam jenis tertentu dari perintah log.

  Pelajari selengkapnya

- `COMMANDLOG RESET`

  Menghapus semua entri dari jenis log perintah yang ditentukan.

  Pelajari selengkapnya

- `FLUSHALL`

  Menghapus semua kunci dari semua basis data. Operasi ini tidak dijamin atom di semua slot.

  Pelajari selengkapnya

- `FLUSHDB`

  Menghapus semua kunci dari basis data saat ini. Operasi ini tidak dijamin atom di semua slot.

  Pelajari selengkapnya

- `INFO`

  Mengembalikan informasi dan statistik tentang server.

  Pelajari selengkapnya

- `LOLWUT`

  Menampilkan seni komputer dan versi Valkey atau Redis OSS.

  Pelajari selengkapnya

- `ROLE`

  Mengembalikan peran replikasi.

Pelajari selengkapnya

- `TIME`

  Mengembalikan waktu server.

  Pelajari selengkapnya

Perintah Set

- `SADD`

  Menambahkan satu atau beberapa anggota ke set. Membuat kunci jika tidak ada.

  Pelajari selengkapnya

- `SCARD`

  Mengembalikan jumlah anggota dalam satu set.

  Pelajari selengkapnya

- `SDIFF`

  Mengembalikan perbedaan beberapa set.

  Pelajari selengkapnya

- `SDIFFSTORE`

  Menyimpan perbedaan beberapa set dalam kunci.

  Pelajari selengkapnya

- `SINTER`

  Mengembalikan potongan dari beberapa set.

  Pelajari selengkapnya

- `SINTERCARD`

  Mengembalikan jumlah anggota potongan dari beberapa set.

  Pelajari selengkapnya

- `SINTERSTORE`

  Menyimpan potongan beberapa set dalam kunci.

  Pelajari selengkapnya

- `SISMEMBER`

  Menentukan apakah anggota termasuk dalam set.

  Pelajari selengkapnya

- `SMEMBERS`

  Mengembalikan semua anggota dari satu set.

  Pelajari selengkapnya

- `SMISMEMBER`

  Menentukan apakah beberapa anggota termasuk dalam set.

  Pelajari selengkapnya

- `SMOVE`

  Memindahkan anggota dari satu set ke set lainnya.

  Pelajari selengkapnya

- `SPOP`

  Mengembalikan satu atau beberapa anggota acak dari satu set setelah menghapusnya. Menghapus set jika anggota terakhir di-popping.

  Pelajari selengkapnya

- `SRANDMEMBER`

  Mendapatkan satu atau beberapa anggota acak dari satu set

  Pelajari selengkapnya

- `SREM`

  Menghapus satu atau beberapa anggota dari satu set. Menghapus set jika anggota terakhir telah dihapus.

Pelajari selengkapnya

- SSCAN

  Melakukan iterasi pada anggota set.

  Pelajari selengkapnya

- SUNION

  Mengembalikan gabungan dari beberapa set.

  Pelajari selengkapnya

- SUNIONSTORE

  Menyimpan gabungan beberapa set dalam kunci.

  Pelajari selengkapnya

Perintah Sorted Set

- BZMPOP

  Menghapus dan mengembalikan anggota berdasarkan skor dari satu atau beberapa sorted set. Memblokir sampai anggota tersedia. Menghapus sorted set jika elemen terakhir di-popping.

  Pelajari selengkapnya

- BZPOPMAX

  Menghapus dan mengembalikan anggota dengan skor tertinggi dari satu atau beberapa sorted set. Memblokir sampai anggota tersedia. Menghapus sorted set jika elemen terakhir di-popping.

  Pelajari selengkapnya

- BZPOPMIN

  Menghapus dan mengembalikan anggota dengan skor terendah dari satu atau beberapa sorted set. Memblokir sampai anggota tersedia. Menghapus sorted set jika elemen terakhir di-popping.

  Pelajari selengkapnya

- ZADD

Menambahkan satu atau beberapa anggota ke sorted set, atau memperbarui skornya. Membuat kunci jika tidak ada.

Pelajari selengkapnya

- `ZCARD`

  Mengembalikan jumlah anggota dalam satu sorted set.

  Pelajari selengkapnya

- `ZCOUNT`

  Mengembalikan jumlah anggota dalam satu sorted set yang memiliki skor dalam rentang.

  Pelajari selengkapnya

- `ZDIFF`

  Mengembalikan perbedaan antara beberapa sorted set.

  Pelajari selengkapnya

- `ZDIFFSTORE`

  Menyimpan perbedaan beberapa sorted set dalam kunci.

  Pelajari selengkapnya

- `ZINCRBY`

  Menambah skor anggota dalam sorted set.

  Pelajari selengkapnya

- `ZINTER`

  Mengembalikan potongan dari beberapa sorted set.

  Pelajari selengkapnya

- `ZINTERCARD`

  Mengembalikan jumlah anggota potongan dari beberapa sorted set.

  Pelajari selengkapnya

- `ZINTERSTORE`

  Menyimpan potongan beberapa sorted set dalam kunci.

  Pelajari selengkapnya

- `ZLEXCOUNT`

  Mengembalikan jumlah anggota dalam sorted set dalam rentang leksikografis.

  Pelajari selengkapnya

- `ZMPOP`

  Mengembalikan anggota dengan skor tertinggi atau terendah dari satu atau beberapa sorted set setelah menghapusnya. Menghapus sorted set jika anggota terakhir di-popping.

  Pelajari selengkapnya

- `ZMSCORE`

  Mengembalikan skor dari satu atau beberapa anggota dalam satu sorted set.

  Pelajari selengkapnya

- `ZPOPMAX`

  Mengembalikan anggota dengan skor tertinggi dari satu sorted set setelah menghapusnya. Menghapus sorted set jika anggota terakhir di-popping.

  Pelajari selengkapnya

- `ZPOPMIN`

  Mengembalikan anggota dengan skor terendah dari satu sorted set setelah menghapusnya. Menghapus sorted set jika anggota terakhir di-popping.

  Pelajari selengkapnya

- `ZRANDMEMBER`

  Mengembalikan satu atau beberapa anggota acak dari sorted set.

  Pelajari selengkapnya

- `ZRANGE`

Mengembalikan anggota dalam satu sorted set dalam rentang indeks.

Pelajari selengkapnya

- ZRANGEBYLEX

  Mengembalikan anggota dalam sorted set dalam rentang leksikografis.

  Pelajari selengkapnya

- ZRANGEBYSCORE

  Mengembalikan anggota dalam satu sorted set dalam rentang skor.

  Pelajari selengkapnya

- ZRANGESTORE

  Menyimpan rentang anggota dari sorted set dalam kunci.

  Pelajari selengkapnya

- ZRANK

  Mengembalikan indeks anggota dalam sorted set yang diurutkan berdasarkan skor naik.

  Pelajari selengkapnya

- ZREM

  Menghapus satu atau beberapa anggota dari satu sorted set. Menghapus sorted set jika semua anggota telah dihapus.

  Pelajari selengkapnya

- ZREMRANGEBYLEX

  Menghapus anggota dalam sorted set dalam rentang leksikografis. Menghapus sorted set jika semua anggota telah dihapus.

  Pelajari selengkapnya

- ZREMRANGEBYRANK

  Menghapus anggota dalam satu sorted set dalam rentang indeks. Menghapus sorted set jika semua anggota telah dihapus.

Pelajari selengkapnya

- ZREMRANGEBYSCORE

  Menghapus anggota dalam satu sorted set dalam rentang skor. Menghapus sorted set jika semua anggota telah dihapus.

  Pelajari selengkapnya

- ZREVRANGE

  Mengembalikan anggota dalam satu sorted set dalam rentang indeks dalam urutan mundur.

  Pelajari selengkapnya

- ZREVRANGEBYLEX

  Mengembalikan anggota dalam sorted set rentang leksikografis dalam urutan mundur.

  Pelajari selengkapnya

- ZREVRANGEBYSCORE

  Mengembalikan anggota dalam satu sorted set dalam rentang skor dalam urutan mundur.

  Pelajari selengkapnya

- ZREVRANK

  Mengembalikan indeks anggota dalam sorted set yang diurutkan berdasarkan skor menurun.

  Pelajari selengkapnya

- ZSCAN

  Melakukan iterasi atas anggota dan skor dari sorted set.

  Pelajari selengkapnya

- ZSCORE

  Mengembalikan skor anggota dalam sorted set.

  Pelajari selengkapnya

- ZUNION

Mengembalikan gabungan dari beberapa sorted set.

Pelajari selengkapnya

- `ZUNIONSTORE`

  Menyimpan gabungan beberapa sorted set dalam kunci.

  Pelajari selengkapnya

Perintah Aliran

- `XACK`

  Mengembalikan jumlah pesan yang berhasil diakui oleh anggota grup konsumen aliran.

  Pelajari selengkapnya

- `XADD`

  Menambahkan pesan baru ke aliran. Membuat kunci jika tidak ada.

  Pelajari selengkapnya

- `XAUTOCLAIM`

  Mengubah, atau memperoleh, kepemilikan pesan dalam grup konsumen, seolah-olah pesan yang telah dikirimkan berasal dari anggota grup konsumen.

  Pelajari selengkapnya

- `XCLAIM`

  Mengubah, atau memperoleh, kepemilikan pesan dalam grup konsumen, seolah-olah pesan yang telah terkirim anggota grup konsumen.

  Pelajari selengkapnya

- `XDEL`

  Mengembalikan jumlah pesan setelah menghapusnya dari aliran.

  Pelajari selengkapnya

- `XGROUP CREATE`

Membuat grup konsumen.

Pelajari selengkapnya

- `XGROUP CREATECONSUMER`

  Membuat konsumen dalam grup konsumen.

  Pelajari selengkapnya

- `XGROUP DELCONSUMER`

  Menghapus konsumen dari grup konsumen.

  Pelajari selengkapnya

- `XGROUP DESTROY`

  Menghancurkan grup konsumen.

  Pelajari selengkapnya

- `XGROUP SETID`

  Menetapkan ID terakhir yang dikirimkan dari grup konsumen.

  Pelajari selengkapnya

- `XINFO CONSUMERS`

  Mengembalikan daftar konsumen dalam grup konsumen.

  Pelajari selengkapnya

- `XINFO GROUPS`

  Mengembalikan daftar grup konsumen dari aliran.

  Pelajari selengkapnya

- `XINFO STREAM`

  Mengembalikan informasi tentang aliran.

  Pelajari selengkapnya

- `XLEN`

Mengembalikan jumlah pesan dalam aliran.

Pelajari selengkapnya

- XPENDING

  Mengembalikan informasi dan entri dari daftar entri tertunda grup konsumen aliran.

  Pelajari selengkapnya

- XRANGE

  Mengembalikan pesan dari aliran dalam rentang IDs.

  Pelajari selengkapnya

- XREAD

  Mengembalikan pesan dari beberapa aliran dengan IDs lebih besar dari yang diminta. Memblokir sampai pesan tersedia.

  Pelajari selengkapnya

- XREADGROUP

  Mengembalikan pesan baru atau historis dari aliran untuk konsumen dalam grup. Memblokir sampai pesan tersedia.

  Pelajari selengkapnya

- XREVRANGE

  Mengembalikan pesan dari aliran dalam rentang IDs dalam urutan terbalik.

  Pelajari selengkapnya

- XTRIM

  Menghapus pesan dari awal aliran.

  Pelajari selengkapnya

Perintah String

- APPEND

Menambahkan string ke nilai kunci. Membuat kunci jika tidak ada.

Pelajari selengkapnya

- `DECR`

  Mengurangi nilai bilangan bulat sebuah kunci sebanyak satu. Menggunakan 0 sebagai nilai awal jika kunci tidak ada.

  Pelajari selengkapnya

- `DECRBY`

  Mengurangi angka dari nilai bilangan bulat sebuah kunci. Menggunakan 0 sebagai nilai awal jika kunci tidak ada.

  Pelajari selengkapnya

- `GET`

  Mengembalikan nilai string dari kunci.

  Pelajari selengkapnya

- `GETDEL`

  Mengembalikan nilai string kunci setelah menghapus kunci.

  Pelajari selengkapnya

- `GETEX`

  Mengembalikan nilai string kunci setelah mengatur waktu kedaluwarsa.

  Pelajari selengkapnya

- `GETRANGE`

  Mengembalikan substring dari string yang disimpan pada kunci.

  Pelajari selengkapnya

- `GETSET`

  Mengembalikan nilai string sebelumnya dari kunci setelah mengaturnya ke nilai baru.

Pelajari selengkapnya

- `INCR`

  Menambah nilai bilangan bulat sebuah kunci sebanyak satu. Menggunakan 0 sebagai nilai awal jika kunci tidak ada.

  Pelajari selengkapnya

- `INCRBY`

  Menambah nilai bilangan bulat sebuah kunci sebanyak satu angka. Menggunakan 0 sebagai nilai awal jika kunci tidak ada.

  Pelajari selengkapnya

- `INCRBYFLOAT`

  Menambah nilai titik ambang dari bidang dengan angka. Menggunakan 0 sebagai nilai awal jika kunci tidak ada.

  Pelajari selengkapnya

- `LCS`

  Menemukan substring umum terpanjang.

  Pelajari selengkapnya

- `MGET`

  Secara atom mengembalikan nilai string dari satu atau beberapa kunci.

  Pelajari selengkapnya

- `MSET`

  Secara atom membuat atau mengubah nilai string dari satu atau beberapa kunci.

  Pelajari selengkapnya

- `MSETNX`

  Secara atom mengubah nilai string dari satu atau beberapa kunci hanya ketika semua kunci tidak ada.

Pelajari selengkapnya

- PSETEX

  Menetapkan nilai string dan waktu kedaluwarsa kunci dalam milidetik. Kunci dibuat jika tidak ada.

  Pelajari selengkapnya

- SET

  Menetapkan nilai string kunci, mengabaikan jenisnya. Kunci dibuat jika tidak ada.

  Pelajari selengkapnya

- SETEX

  Menetapkan nilai string dan waktu kedaluwarsa kunci. Membuat kunci jika tidak ada.

  Pelajari selengkapnya

- SETNX

  Menetapkan nilai string kunci hanya jika kunci tidak ada.

  Pelajari selengkapnya

- SETRANGE

  Menimpa bagian dari nilai string dengan yang lain dengan offset. Membuat kunci jika tidak ada.

  Pelajari selengkapnya

- STRLEN

  Mengembalikan panjang nilai string.

  Pelajari selengkapnya

- SUBSTR

  Mengembalikan substring dari nilai string.

  Pelajari selengkapnya

Perintah Transaction

- `DISCARD`

  Membuang transaksi.

  Pelajari selengkapnya
- `EXEC`

  Mengeksekusi semua perintah dalam transaksi.

  Pelajari selengkapnya
- `MULTI`

  Memulai transaksi.

  Pelajari selengkapnya

Perintah Valkey dan Redis OSS yang dibatasi

Untuk memberikan pengalaman layanan terkelola, ElastiCache batasi akses ke perintah khusus mesin cache tertentu yang memerlukan hak istimewa lanjutan. Untuk cache yang menjalankan Redis OSS, perintah berikut tidak tersedia:

- `acl setuser`
- `acl load`
- `acl save`
- `acl deluser`
- `bgrewriteaof`
- `bgsave`
- `cluster addslot`
- `cluster addslotsrange`
- `cluster bumpepoch`
- `cluster delslot`
- `cluster delslotsrange`
- `cluster failover`
- `cluster flushslots`
- `cluster forget`

- `cluster links`
- `cluster meet`
- `cluster setslot`
- `config`
- `debug`
- `migrate`
- `psync`
- `replicaof`
- `save`
- `slaveof`
- `shutdown`
- `sync`

Selain itu, perintah berikut tidak tersedia untuk cache nirserver:

- `acl log`
- `client caching`
- `client getredir`
- `client id`
- `client info`
- `client kill`
- `client list`
- `client no-evict`
- `client pause`
- `client tracking`
- `client trackinginfo`
- `client unblock`
- `client unpause`
- `cluster count-failure-reports`
- `commandlog`

- commandlog get
- commandlog help
- commandlog len
- commandlog reset
- fcall
- fcall_ro
- function
- function delete
- function dump
- function flush
- function help
- function kill
- function list
- function load
- function restore
- function stats
- keys
- lastsave
- latency
- latency doctor
- latency graph
- latency help
- latency histogram
- latency history
- latency latest
- latency reset
- memory
- memory doctor
- memory help
- memory malloc-stats

- memory purge
- memory stats
- memory usage
- monitor
- move
- object
- object encoding
- object freq
- object help
- object idletime
- object refcount
- pfdebug
- pfselftest
- psubscribe
- pubsub numpat
- punsubscribe
- script kill
- slowlog
- slowlog get
- slowlog help
- slowlog len
- slowlog reset
- swapdb
- unwatch
- wait
- watch

Perintah Memcached yang didukung

ElastiCache Tanpa server untuk Memcached mendukung semua perintah memcached di sumber terbuka memcached 1.6 kecuali untuk yang berikut ini:

- Koneksi klien memerlukan TLS, akibatnya protokol UDP tidak didukung.
- Protokol biner tidak didukung karena ini sudah secara resmi dihentikan di memcached 1.6.
- Perintah `GET/GETS` dibatasi hingga 16KB untuk menghindari potensi serangan DoS ke server dengan mengambil sejumlah besar kunci.
- Perintah `flush_all` yang tertunda akan ditolak dengan `CLIENT_ERROR`.
- Perintah yang mengonfigurasi mesin atau mengungkapkan informasi internal tentang status mesin atau log tidak didukung, seperti:
  - Untuk perintah `STATS`, hanya `stats` dan `stats reset` yang didukung. Variasi lain akan mengembalikan `ERROR`
  - `lru / lru_crawler` - perubahan untuk pengaturan perayap LRU dan LRU
  - `watch` - memantau log server memcached
  - `verbosity` - mengonfigurasi tingkat log server
  - `me`- perintah meta debug (me) tidak didukung

Konfigurasi dan batas Valkey dan Redis OSS

Mesin Valkey dan Redis OSS masing-masing menyediakan sejumlah parameter konfigurasi, beberapa di antaranya dapat dimodifikasi ElastiCache untuk Redis OSS dan beberapa di antaranya tidak dapat dimodifikasi untuk memberikan kinerja dan keandalan yang stabil.

Cache nirserver

Untuk cache tanpa server, grup parameter tidak digunakan dan semua konfigurasi Valkey atau Redis OSS tidak dapat dimodifikasi. Parameter Valkey atau Redis OSS berikut tersedia:

Nama	Detail	Deskripsi
`acl-pubsub-default`	`allchannels`	Izin saluran pubsub default untuk pengguna ACL di cache.

`client-output-buffer-limit`	`normal 0 0 0`
	
	`pubsub 32mb 8mb 60`
		Klien normal tidak memiliki batas buffer. PUB/SUB klien akan terputus jika mereka melanggar backlog 32MiB, atau melanggar backlog 8MiB selama 60-an.

Nama	Detail	Deskripsi
`client-query-buffer-limit`	1 GiB	Ukuran maksimum buffer kueri klien tunggal. Selain itu, klien tidak dapat mengeluarkan permintaan dengan lebih dari 3.999 argumen.
`cluster-allow-pubsubshard-when-down`	`yes`	Hal ini memungkinkan cache melayani lalu lintas pubsub saat sebagian cache tidak aktif.
`cluster-allow-reads-when-down`	`yes`	Hal ini memungkinkan cache untuk melayani lalu lintas baca saat cache tidak aktif sebagian.
`cluster-enabled`	`yes`	Semua cache nirserver menggunakan mode klaster diaktifkan, yang memungkinkan cache untuk secara transparan mempartisi data di beberapa serpihan (shard) backend. Semua slot muncul bagi klien sebagai slot yang dimiliki oleh satu simpul virtual.
`cluster-require-full-coverage`	no	Ketika ruang kunci tidak aktif sebagian (yaitu setidaknya satu slot hash tidak dapat diakses), cache akan terus menerima kueri untuk bagian ruang kunci yang masih tercakup. Seluruh ruang kunci akan selalu "dicakup" oleh satu simpul virtual di `cluster slots`.

Nama	Detail	Deskripsi
`lua-time-limit`	5000	Waktu eksekusi maksimum untuk skrip Lua, dalam milidetik, sebelum ElastiCache mengambil tindakan untuk menghentikan skrip. Jika *`lua-time-limit`* terlampaui, semua perintah Valkey atau Redis OSS dapat mengembalikan kesalahan bentuk ____-BUSY . Karena keadaan ini dapat menyebabkan gangguan dengan banyak operasi Valkey atau Redis OSS penting, pertama-tama ElastiCac he akan mengeluarkan perintah SCRIPT KILL. Jika ini tidak berhasil, ElastiCache akan secara paksa restart Valkey atau Redis OSS.
`maxclients`	65000	Jumlah klien maksimum yang dapat dihubungk an ke cache sekaligus. Koneksi lebih lanjut yang dibuat mungkin berhasil dan mungkin gagal.
`maxmemory-policy`	`volatile-lru`	Item dengan set TTL diusir mengikuti estimasi least-recently-used (LRU) saat batas memori cache tercapai.
`notify-keyspace-events`	(string kosong)	Peristiwa ruang kunci saat ini tidak didukung pada cache nirserver.
`port`	Port primer: 6379 Port baca: 6380	Cache nirserver menyatakan dua port dengan nama host yang sama. Port primer memungkin kan operasi tulis dan baca, sedangkan port baca memungkinkan operasi baca latensi rendah yang pada akhirnya konsisten menggunakan perintah `READONLY`.

Nama	Detail	Deskripsi
`proto-max-bulk-len`	512 MiB	Ukuran maksimum dari permintaan elemen tunggal.
`timeout`	0	Koneksi klien tidak terputus secara paksa pada waktu idle tertentu, tetapi koneksi klien mungkin terputus selama kondisi stabil untuk tujuan penyeimbangan beban.

Selain itu, batasan berikut ini berlaku:

Nama	Detail	Deskripsi
Ukuran per cache	5.000 GiB	Jumlah maksimum data yang dapat disimpan per cache tanpa server.
Ukuran per slot	32 GiB	Ukuran maksimum slot hash Valkey atau Redis OSS tunggal. Klien yang mencoba mengatur lebih banyak data daripada ini pada satu slot Valkey atau Redis OSS akan memicu kebijakan penggusuran pada slot, dan jika tidak ada kunci yang dapat diusir, akan menerima kesalahan kehabisan memori (). `OOM`
ECPU per cache	15.000.000 ECPU/detik	ElastiCache Metrik Unit Pemrosesan (ECPU). Jumlah yang ECPUs dikonsumsi oleh permintaan Anda tergantung pada waktu vCPU yang dibutuhkan dan jumlah data yang ditransfer.
ECPU per slot	30K - 90K ECPU/detik	Maksimal 30K ECPUs/second per slot atau 90K ECPUs/second saat menggunakan Read from Replica menggunakan koneksi READONLY.

Nama	Detail	Deskripsi
Argumen per Permintaan	3,999	Jumlah maksimum argumen per permintaan. Klien yang mengirim lebih banyak argumen per permintaan akan menerima kesalahan.
Panjang nama kunci	4 KiB	Ukuran maksimum untuk satu tombol Valkey atau Redis OSS atau nama saluran. Klien yang mereferensikan kunci yang lebih besar dari ukuran ini akan menghasilkan kesalahan.
Ukuran skrip Lua	4 MiB	Ukuran maksimum skrip Valkey atau Redis OSS Lua tunggal. Percobaan memuat skrip Lua yang lebih besar dari ukuran ini akan menghasilkan kesalahan.

Klaster yang dirancang sendiri

Untuk klaster yang dirancang sendiri, lihat Parameter Valkey dan Redis OSS untuk nilai default parameter konfigurasi dan mana yang dapat dikonfigurasi. Nilai default umumnya direkomendasikan kecuali jika Anda memiliki kasus penggunaan khusus yang mengharuskan nilai default ini diganti.

IPv6 contoh klien untuk Valkey, Memcached, dan Redis OSS

ElastiCache kompatibel dengan Valkey, Memcached, dan Redis OSS. Ini berarti bahwa klien yang mendukung IPv6 koneksi harus dapat terhubung ke IPv6 diaktifkan ElastiCache untuk cluster Memcached. Ada beberapa peringatan yang perlu diperhatikan saat berinteraksi dengan sumber daya yang IPv6 diaktifkan.

Anda dapat melihat Praktik terbaik untuk posting blog klien Valkey dan Redis di Blog AWS Database untuk rekomendasi tentang mengkonfigurasi klien Valkey dan Redis OSS untuk sumber daya. ElastiCache

Berikut ini adalah praktik terbaik untuk berinteraksi dengan ElastiCache sumber daya yang IPv6 diaktifkan dengan pustaka klien sumber terbuka yang umum digunakan.

Klien yang divalidasi dengan Valkey dan Redis OSS

ElastiCache kompatibel dengan Valkey dan Redis OSS open-source. Ini berarti bahwa klien Valkey dan open source Redis OSS yang mendukung IPv6 koneksi harus dapat terhubung ke IPv6 diaktifkan ElastiCache untuk cluster Redis OSS. Selain itu, beberapa klien Python dan Java yang paling populer telah diuji dan divalidasi secara khusus untuk bekerja dengan semua konfigurasi tipe jaringan yang didukung (IPv4 hanya, IPv6 hanya, dan Dual Stack)

Klien berikut secara khusus telah divalidasi untuk bekerja dengan semua konfigurasi jenis jaringan yang didukung untuk Valkey dan Redis OSS.

Klien yang Divalidasi:

- Redis Py () – 4.1.2
- Lettuce – Versi: 6.1.6.RELEASE
- Jedis - Versi: 3.6.0

Praktik terbaik untuk klien (Valkey dan Redis OSS)

Pelajari praktik terbaik untuk skenario umum dan ikuti contoh kode dari beberapa pustaka klien Valkey dan Redis OSS open source paling populer (redis-py,, dan Lettuce) PHPRedis, serta praktik terbaik untuk berinteraksi dengan ElastiCache sumber daya dengan pustaka klien Memcached open-source yang umum digunakan.

Topik

- Sejumlah besar koneksi (Valkey dan Redis OSS)
- Penemuan klien cluster dan backoff eksponensial (Valkey dan Redis OSS)
- Konfigurasikan batas waktu sisi klien (Valkey dan Redis OSS)
- Konfigurasikan batas waktu idle sisi server (Valkey dan Redis OSS)
- Skrip Lua
- Menyimpan item komposit besar (Valkey dan Redis OSS)
- Konfigurasi klien selada (Valkey dan Redis OSS)
- Mengkonfigurasi protokol pilihan untuk cluster tumpukan ganda (Valkey dan Redis OSS)

Sejumlah besar koneksi (Valkey dan Redis OSS)

Cache tanpa server dan individual ElastiCache untuk node Redis OSS mendukung hingga 65.000 koneksi klien bersamaan. Namun, untuk mengoptimalkan performa, kami menyarankan agar aplikasi klien tidak terus-menerus beroperasi pada tingkat koneksi tersebut. Valkey dan Redis OSS masing-masing memiliki proses single-threaded berdasarkan loop peristiwa di mana permintaan klien yang masuk ditangani secara berurutan. Artinya, waktu respons klien tertentu menjadi lebih lama seiring jumlah klien yang terhubung meningkat.

Anda dapat mengambil serangkaian tindakan berikut untuk menghindari hambatan koneksi pada server Valkey atau Redis OSS:

- Lakukan operasi baca dari replika baca. Ini dapat dilakukan dengan menggunakan endpoint ElastiCache pembaca dalam mode cluster dinonaktifkan atau dengan menggunakan replika untuk membaca dalam mode cluster diaktifkan, termasuk cache tanpa server.
- Distribusikan lalu lintas penulisan ke beberapa simpul primer. Anda dapat melakukannya dengan dua cara: Anda dapat menggunakan cluster Valkey atau Redis OSS multi-sharded dengan klien yang mampu mode cluster. Anda juga dapat menulis ke beberapa simpul primer dalam mode klaster dinonaktifkan dengan sharding sisi klien. Hal ini dilakukan secara otomatis dalam cache nirserver.
- Gunakan pool koneksi jika tersedia di pustaka klien Anda.

Secara umum, membuat koneksi TCP adalah operasi komputasi yang mahal dibandingkan dengan perintah Valkey atau Redis OSS yang khas. Misalnya, menangani SET/GET permintaan adalah urutan besarnya lebih cepat saat menggunakan kembali koneksi yang ada. Menggunakan pool koneksi klien dengan ukuran terbatas akan mengurangi overhead manajemen koneksi. Hal ini juga membatasi jumlah koneksi masuk konkuren dari aplikasi klien.

Contoh kode berikut PHPRedis menunjukkan bahwa koneksi baru dibuat untuk setiap permintaan pengguna baru:

```
$redis = new Redis();
if ($redis->connect($HOST, $PORT) != TRUE) {
 //ERROR: connection failed
 return;
}
$redis->set($key, $value);
unset($redis);
```

```
$redis = NULL;
```

Kami membandingkan kode ini dalam satu loop pada instance Amazon Elastic Compute Cloud EC2 (Amazon) yang terhubung ke Graviton2 (m6g.2xlarge) untuk node Redis OSS. ElastiCache Kita menempatkan klien dan server di Zona Ketersediaan yang sama. Latensi rata-rata seluruh operasi adalah 2,82 milidetik.

Saat kami memperbarui kode serta menggunakan koneksi persisten dan pool koneksi, latensi rata-rata seluruh operasi adalah 0,21 milidetik:

```
$redis = new Redis();
if ($redis->pconnect($HOST, $PORT) != TRUE) {
 // ERROR: connection failed
 return;
}
$redis->set($key, $value);
unset($redis);
$redis = NULL;
```

Konfigurasi redis.ini yang diperlukan:

- `redis.pconnect.pooling_enabled=1`
- `redis.pconnect.connection_limit=10`

Kode berikut adalah contoh pool koneksi Redis-Py:

```
conn = Redis(connection_pool=redis.BlockingConnectionPool(host=HOST,
 max_connections=10))
conn.set(key, value)
```

Kode berikut adalah contoh pool koneksi Lettuce:

```
RedisClient client = RedisClient.create(RedisURI.create(HOST, PORT));
GenericObjectPool<StatefulRedisConnection> pool =
 ConnectionPoolSupport.createGenericObjectPool(() -> client.connect(), new
 GenericObjectPoolConfig());
pool.setMaxTotal(10); // Configure max connections to 10
try (StatefulRedisConnection connection = pool.borrowObject()) {
 RedisCommands syncCommands = connection.sync();
```

```
 syncCommands.set(key, value);
}
```

Penemuan klien cluster dan backoff eksponensial (Valkey dan Redis OSS)

Saat menghubungkan ke cluster ElastiCache Valkey atau Redis OSS dalam mode cluster diaktifkan, pustaka klien yang sesuai harus sadar cluster. Klien harus mendapatkan peta slot hash ke simpul yang sesuai di klaster untuk mengirim permintaan ke simpul yang tepat dan menghindari overhead performa penanganan pengalihan klaster. Akibatnya, klien harus menemukan daftar lengkap slot dan simpul yang dipetakan dalam dua situasi berbeda:

- Klien diinisialisasi dan harus mengisi konfigurasi slot awal
- Pengalihan MOVED diterima dari server, seperti dalam situasi failover ketika semua slot yang dilayani oleh simpul primer sebelumnya diambil alih oleh replika, atau melakukan resharding ketika slot dipindahkan dari simpul primer sumber ke simpul primer target

Penemuan klien biasanya dilakukan dengan mengeluarkan perintah CLUSTER SLOT atau CLUSTER NODE ke server Valkey atau Redis OSS. Sebaiknya gunakan metode CLUSTER SLOT karena metode ini mengembalikan set rentang slot dan simpul primer dan replika terkait ke klien. Metode ini tidak memerlukan parsing tambahan dari klien dan lebih efisien.

Tergantung topologi klaster, ukuran respons untuk perintah CLUSTER SLOT dapat bervariasi berdasarkan ukuran klaster. Klaster yang lebih besar dengan lebih banyak simpul menghasilkan respons yang lebih besar. Oleh karena itu, penting untuk memastikan bahwa jumlah klien yang melakukan penemuan topologi klaster tidak bertambah tanpa batas. Misalnya, ketika aplikasi klien diaktifkan atau kehilangan koneksi dari server dan harus melakukan penemuan klaster, satu kesalahan umumnya adalah bahwa aplikasi klien memicu beberapa permintaan koneksi ulang dan penemuan tanpa menambahkan backoff eksponensial saat mencoba lagi. Ini dapat membuat server Valkey atau Redis OSS tidak responsif untuk jangka waktu yang lama, dengan pemanfaatan CPU 100%. Pemadaman akan lebih lama jika setiap perintah CLUSTER SLOT harus memproses sejumlah besar simpul di bus klaster. Kami telah mengamati beberapa pemadaman klien di masa lalu karena perilaku ini di sejumlah bahasa yang berbeda termasuk Python redis-py-cluster () dan Java (Lettuce dan Redisson).

Dalam cache nirserver, banyak masalah secara otomatis dikurangi karena topologi klaster yang dinyatakan bersifat statis dan terdiri dari dua entri: titik akhir tulis dan titik akhir baca. Penemuan klaster juga secara otomatis tersebar di beberapa simpul saat menggunakan titik akhir cache. Namun, rekomendasi berikut masih berguna.

Untuk mengurangi dampak yang disebabkan oleh masuknya permintaan koneksi dan penemuan secara tiba-tiba, kami merekomendasikan hal berikut:

- Menerapkan pool koneksi klien dengan ukuran terbatas untuk membatasi jumlah koneksi masuk konkuren dari aplikasi klien.
- Ketika klien terputus dari server karena waktu habis, coba lagi dengan backoff eksponensial dengan jitter. Hal ini membantu menghindari banyak klien membebani server pada waktu yang sama.
- Gunakan panduan dalam Menemukan titik akhir koneksi di ElastiCache guna menemukan titik akhir klaster untuk melakukan penemuan klaster. Dengan begitu, Anda menyebarkan beban penemuan ke semua simpul di klaster (hingga 90), bukan memanfaatkan hanya beberapa simpul seed hardcode di klaster.

Berikut ini adalah beberapa contoh kode untuk logika percobaan ulang backoff eksponensial di redis-py,, dan Lettuce. PHPRedis

Contoh logika backoff 1: redis-py

redis-py memiliki mekanisme percobaan ulang bawaan yang mencoba ulang satu kali segera setelah kegagalan. Mekanisme ini dapat diaktifkan melalui `retry_on_timeout` argumen yang diberikan saat membuat objek Redis OSS. Di sini kita mendemonstrasikan mekanisme percobaan ulang kustom dengan backoff eksponensial dan jitter. Kami telah mengirimkan permintaan tarik untuk mengimplementasikan backoff eksponensial secara native di redis-py (#1494). Di masa depan, hal tersebut mungkin tidak perlu diimplementasikan secara manual.

```
def run_with_backoff(function, retries=5):
base_backoff = 0.1 # base 100ms backoff
max_backoff = 10 # sleep for maximum 10 seconds
tries = 0
while True:
try:
  return function()
except (ConnectionError, TimeoutError):
  if tries >= retries:
 raise
  backoff = min(max_backoff, base_backoff * (pow(2, tries) + random.random()))
  print(f"sleeping for {backoff:.2f}s")
  sleep(backoff)
  tries += 1
```

Anda kemudian dapat menggunakan kode berikut untuk menetapkan nilai:

```
client = redis.Redis(connection_pool=redis.BlockingConnectionPool(host=HOST,
 max_connections=10))
res = run_with_backoff(lambda: client.set("key", "value"))
print(res)
```

Bergantung pada beban kerja Anda, Anda sebaiknya mengubah nilai backoff dasar dari 1 detik menjadi puluhan atau ratusan milidetik untuk beban kerja yang sensitif terhadap latensi.

Contoh logika backoff 2: PHPRedis

PHPRedis memiliki mekanisme coba ulang bawaan yang mencoba ulang maksimum (tidak dapat dikonfigurasi) sebanyak 10 kali. Ada penundaan yang dapat dikonfigurasi di antara percobaan (dengan jitter dari percobaan ulang kedua dan seterusnya). Untuk informasi selengkapnya, lihat kode sampel berikut. Kami telah mengirimkan permintaan tarik untuk menerapkan backoff eksponensial secara native di PHPredis (#1986) yang telah digabungkan dan didokumentasikan. Bagi mereka yang berada di rilis terbaru PHPRedis, tidak perlu mengimplementasikan secara manual tetapi kami telah menyertakan referensi di sini untuk mereka yang ada di versi sebelumnya. Untuk saat ini, berikut ini adalah contoh kode yang mengonfigurasi penundaan mekanisme percobaan ulang:

```
$timeout = 0.1; // 100 millisecond connection timeout
$retry_interval = 100; // 100 millisecond retry interval
$client = new Redis();
if($client->pconnect($HOST, $PORT, $timeout, NULL, $retry_interval) != TRUE) {
 return; // ERROR: connection failed
}
$client->set($key, $value);
```

Contoh logika backoff 3: Lettuce

Lettuce memiliki mekanisme percobaan ulang bawaan berdasarkan strategi backoff eksponensial yang dijelaskan dalam postingan Backoff Eksponensial dan Jitter. Berikut ini adalah kutipan kode yang menunjukkan pendekatan jitter penuh:

```
public static void main(String[] args)
{
 ClientResources resources = null;
 RedisClient client = null;
```

```
 try {
  resources = DefaultClientResources.builder()
    .reconnectDelay(Delay.fullJitter(
   Duration.ofMillis(100),     // minimum 100 millisecond delay
   Duration.ofSeconds(5),      // maximum 5 second delay
   100, TimeUnit.MILLISECONDS) // 100 millisecond base
  ).build();

  client = RedisClient.create(resources, RedisURI.create(HOST, PORT));
  client.setOptions(ClientOptions.builder()
 .socketOptions(SocketOptions.builder().connectTimeout(Duration.ofMillis(100)).build()) //
 100 millisecond connection timeout
 .timeoutOptions(TimeoutOptions.builder().fixedTimeout(Duration.ofSeconds(5)).build()) //
 5 second command timeout
 .build());

     // use the connection pool from above example
 } finally {
  if (connection != null) {
   connection.close();
  }

  if (client != null){
   client.shutdown();
  }

  if (resources != null){
   resources.shutdown();
  }

 }
}
```

Konfigurasikan batas waktu sisi klien (Valkey dan Redis OSS)

Mengkonfigurasi batas waktu sisi klien

Konfigurasikan waktu habis sisi klien dengan tepat agar server memiliki cukup waktu untuk memproses permintaan dan menghasilkan respons. Hal ini juga membantunya melakukan gagal cepat (fail fast) jika koneksi ke server tidak dapat dibuat. Perintah Valkey atau Redis OSS tertentu bisa lebih mahal secara komputasi daripada yang lain. Misalnya, skrip Lua atau MULTI/EXEC transaksi yang berisi beberapa perintah yang harus dijalankan secara atom. Secara umum, waktu

habis sisi klien yang lebih tinggi disarankan untuk menghindari waktu habis klien sebelum respons diterima dari server, termasuk yang berikut:

- Menjalankan perintah di beberapa kunci
- Menjalankan MULTI/EXEC transaksi atau skrip Lua yang terdiri dari beberapa perintah Valkey atau Redis OSS individual
- Membaca nilai besar
- Melakukan operasi pemblokiran seperti BLPOP

Dalam kasus operasi pemblokiran seperti BLPOP, praktik terbaiknya adalah dengan mengatur waktu habis perintah ke jumlah yang lebih rendah dari waktu habis soket.

Berikut ini adalah contoh kode untuk menerapkan batas waktu sisi klien di redis-py,, dan Lettuce. PHPRedis

Contoh konfigurasi waktu habis 1: redis-py

Berikut ini adalah contoh kode dengan redis-py:

```
# connect to Redis server with a 100 millisecond timeout
# give every Redis command a 2 second timeout
client = redis.Redis(connection_pool=redis.BlockingConnectionPool(host=HOST,
 max_connections=10,socket_connect_timeout=0.1,socket_timeout=2))

res = client.set("key", "value") # will timeout after 2 seconds
print(res)                       # if there is a connection error

res = client.blpop("list", timeout=1) # will timeout after 1 second
                                      # less than the 2 second socket timeout
print(res)
```

Contoh konfigurasi batas waktu 2: PHPRedis

Berikut ini adalah contoh kode dengan PHPRedis:

```
// connect to Redis server with a 100ms timeout
// give every Redis command a 2s timeout
$client = new Redis();
$timeout = 0.1; // 100 millisecond connection timeout
$retry_interval = 100; // 100 millisecond retry interval
$client = new Redis();
```

```
if($client->pconnect($HOST, $PORT, 0.1, NULL, 100, $read_timeout=2) != TRUE){
 return; // ERROR: connection failed
}
$client->set($key, $value);

$res = $client->set("key", "value"); // will timeout after 2 seconds
print "$res\n";                      // if there is a connection error

$res = $client->blpop("list", 1); // will timeout after 1 second
print "$res\n";                   // less than the 2 second socket timeout
```

Contoh konfigurasi waktu habis 3: Lettuce

Berikut ini adalah contoh kode dengan Lettuce:

```
// connect to Redis server and give every command a 2 second timeout
public static void main(String[] args)
{
 RedisClient client = null;
 StatefulRedisConnection<String, String> connection = null;
 try {
  client = RedisClient.create(RedisURI.create(HOST, PORT));
  client.setOptions(ClientOptions.builder()
 .socketOptions(SocketOptions.builder().connectTimeout(Duration.ofMillis(100)).build()) //
 100 millisecond connection timeout
 .timeoutOptions(TimeoutOptions.builder().fixedTimeout(Duration.ofSeconds(2)).build()) //
 2 second command timeout
 .build());

  // use the connection pool from above example

  commands.set("key", "value"); // will timeout after 2 seconds
  commands.blpop(1, "list"); // BLPOP with 1 second timeout
 } finally {
  if (connection != null) {
   connection.close();
  }

  if (client != null){
   client.shutdown();
  }
 }
}
```

Konfigurasikan batas waktu idle sisi server (Valkey dan Redis OSS)

Kami telah mengamati kasus ketika aplikasi pelanggan memiliki klien idle terhubung dalam jumlah yang tinggi, tetapi tidak secara aktif mengirim perintah. Dalam skenario tersebut, Anda dapat menghabiskan total 65.000 koneksi dengan jumlah klien idle yang tinggi. Untuk menghindari skenario tersebut, konfigurasikan pengaturan waktu habis dengan sesuai di server melalui Parameter Valkey dan Redis OSS. Hal ini memastikan bahwa server secara aktif memutus klien idle untuk menghindari peningkatan jumlah koneksi. Pengaturan ini tidak tersedia di cache nirserver.

Skrip Lua

Valkey dan Redis OSS mendukung lebih dari 200 perintah, termasuk yang menjalankan skrip Lua. Namun, ketika datang ke skrip Lua, ada beberapa jebakan yang dapat mempengaruhi memori dan ketersediaan Valkey atau Redis OSS.

Skrip Lua yang tidak diparameterisasi

Setiap skrip Lua di-cache di server Valkey atau Redis OSS sebelum berjalan. Skrip Lua yang tidak diparameterisasi adalah unik, yang dapat menyebabkan server Valkey atau Redis OSS menyimpan sejumlah besar skrip Lua dan menghabiskan lebih banyak memori. Untuk memitigasi hal ini, pastikan bahwa semua skrip Lua diparameterisasi dan secara teratur melakukan SCRIPT FLUSH untuk membersihkan skrip Lua yang di-cache jika diperlukan.

Ketahuilah juga bahwa kunci harus disediakan. Jika nilai untuk parameter KEY tidak disediakan, skrip akan gagal. Misalnya, ini tidak akan berhasil:

```
serverless-test-lst4hg.serverless.use1.cache.amazonaws.com:6379> eval 'return "Hello
 World"' 0
(error) ERR Lua scripts without any input keys are not supported.
```

Ini akan berhasil:

```
serverless-test-lst4hg.serverless.use1.cache.amazonaws.com:6379> eval 'return
 redis.call("get", KEYS[1])' 1 mykey-2
"myvalue-2"
```

Contoh berikut menunjukkan cara menggunakan skrip yang diparameterisasi. Pertama, kita memiliki contoh pendekatan tanpa parameterisasi yang menghasilkan tiga skrip Lua yang di-cache yang berbeda dan tidak disarankan:

```
eval "return redis.call('set','key1','1')" 0
eval "return redis.call('set','key2','2')" 0
eval "return redis.call('set','key3','3')" 0
```

Sebagai gantinya, gunakan pola berikut untuk membuat skrip tunggal yang dapat menerima parameter yang diteruskan:

```
eval "return redis.call('set',KEYS[1],ARGV[1])" 1 key1 1
eval "return redis.call('set',KEYS[1],ARGV[1])" 1 key2 2
eval "return redis.call('set',KEYS[1],ARGV[1])" 1 key3 3
```

Skrip Lua yang berjalan lama

Skrip Lua dapat menjalankan beberapa perintah secara atom, sehingga bisa memakan waktu lebih lama untuk diselesaikan daripada perintah Valkey atau Redis OSS biasa. Jika skrip Lua hanya menjalankan operasi hanya-baca, Anda dapat menghentikannya di tengah proses. Namun, segera setelah skrip Lua melakukan operasi penulisan, prosesnya tidak dapat dihentikan dan harus dijalankan hingga selesai. Skrip Lua yang berjalan lama yang bermutasi dapat menyebabkan server Valkey atau Redis OSS tidak responsif untuk waktu yang lama. Untuk memitigasi masalah ini, hindari skrip Lua yang berjalan lama dan uji skrip di lingkungan praproduksi.

Skrip Lua dengan penulisan stealth

Ada beberapa cara skrip Lua dapat terus menulis data baru ke Valkey atau Redis OSS bahkan ketika Valkey atau Redis OSS selesai: `maxmemory`

- Script dimulai ketika server Valkey atau Redis OSS di bawah ini`maxmemory`, dan berisi beberapa operasi tulis di dalamnya
- Perintah tulis pertama skrip tidak menghabiskan memori (seperti DEL), yang diikuti oleh lebih banyak operasi tulis yang menghabiskan memori
- Anda dapat mengurangi masalah ini dengan mengonfigurasi kebijakan penggusuran yang tepat di server Valkey atau Redis OSS selain. `noeviction` Hal ini memungkinkan Redis OSS untuk mengusir item dan membebaskan memori di antara skrip Lua.

Menyimpan item komposit besar (Valkey dan Redis OSS)

Dalam beberapa skenario, aplikasi dapat menyimpan item komposit besar di Valkey atau Redis OSS (seperti dataset hash multi-GB). Ini bukan praktik yang direkomendasikan karena sering

menyebabkan masalah kinerja di Valkey atau Redis OSS. Misalnya, klien dapat melakukan perintah HGETALL untuk mengambil seluruh koleksi hash multi GB. Hal ini dapat menghasilkan tekanan memori yang signifikan ke server Valkey atau Redis OSS buffering item besar dalam buffer output klien. Selain itu, untuk migrasi slot dalam mode cluster, ElastiCache tidak memigrasikan slot yang berisi item dengan ukuran serial yang lebih besar dari 256 MB.

Untuk mengatasi masalah item besar, kami memiliki rekomendasi berikut:

- Pecah item komposit besar menjadi beberapa item yang lebih kecil. Misalnya, pisahkan koleksi hash besar menjadi bidang nilai kunci individual dengan skema nama kunci yang mencerminkan koleksi dengan tepat, seperti menggunakan awalan umum dalam nama kunci untuk mengidentifikasi kumpulan item. Jika Anda harus mengakses beberapa bidang dalam koleksi yang sama secara atomik, Anda dapat menggunakan perintah MGET untuk mengambil beberapa nilai kunci dalam perintah yang sama.
- Jika Anda mengevaluasi semua opsi dan masih tidak dapat memecah set data koleksi besar, coba gunakan perintah yang beroperasi di subset data dalam koleksi, bukan seluruh koleksi. Hindari kasus penggunaan yang mengharuskan Anda mengambil seluruh koleksi multi-GB secara atomik dalam perintah yang sama. Salah satu contohnya adalah menggunakan perintah HGET atau HMGET, bukan HGETALL di koleksi hash.

Konfigurasi klien selada (Valkey dan Redis OSS)

Bagian ini menjelaskan opsi konfigurasi Java dan Lettuce yang direkomendasikan, dan bagaimana penerapannya pada ElastiCache cluster.

Rekomendasi di bagian ini diuji dengan Lettuce versi 6.2.2.

Topik

- Contoh: Konfigurasi selada untuk mode cluster, TLS diaktifkan
- Contoh: Konfigurasi selada untuk mode cluster dinonaktifkan, TLS diaktifkan

TTL cache DNS Java

Mesin virtual Java (JVM) menyimpan cache pencarian nama DNS. Ketika JVM menyelesaikan nama host ke alamat IP, itu menyimpan alamat IP untuk jangka waktu tertentu, yang dikenal sebagai (TTL). time-to-live

Pilihan nilai TTL merupakan kompromi antara latensi dan responsivitas terhadap perubahan. Dengan lebih pendek TTLs, penyelesai DNS melihat pembaruan di DNS cluster lebih cepat. Hal ini dapat membuat aplikasi Anda lebih cepat dalam merespons penggantian atau alur kerja lain yang dialami klaster Anda. Namun, jika TTL terlalu rendah, hal ini meningkatkan volume kueri, yang dapat meningkatkan latensi aplikasi Anda. Meskipun tidak ada nilai TTL yang benar, sebaiknya pertimbangkan lama waktu yang sesuai bagi Anda untuk menunggu perubahan berlaku saat menetapkan nilai TTL Anda.

Karena ElastiCache node menggunakan entri nama DNS yang mungkin berubah, kami sarankan Anda mengonfigurasi JVM Anda dengan TTL rendah 5 hingga 10 detik. Hal ini dapat memastikan bahwa ketika alamat IP simpul berubah, aplikasi Anda dapat menerima dan menggunakan alamat IP baru sumber daya dengan mengueri ulang DNS.

Pada beberapa konfigurasi Java, TTL default JVM diatur untuk tidak pernah menyegarkan entri DNS hingga JVM dimulai ulang.

Untuk detail tentang cara mengatur TTL JVM Anda, lihat How to set the JVM TTL.

Versi Lettuce

Sebaiknya gunakan Lettuce versi 6.2.2 atau versi yang lebih baru.

Titik akhir

Saat Anda menggunakan klaster dengan mode klaster diaktifkan, atur `redisUri` ke titik akhir konfigurasi klaster. Pencarian DNS untuk URI ini menampilkan daftar semua simpul yang tersedia di klaster, dan diresolusi secara acak ke salah satu simpul selama inisialisasi klaster. Untuk detail selengkapnya tentang cara kerja penyegaran topologi, lihat dynamicRefreshResourcesnanti dalam topik ini.

SocketOption

Aktifkan KeepAlive. Mengaktifkan opsi ini akan mengurangi kebutuhan untuk menangani koneksi yang gagal selama runtime perintah.

Pastikan Anda menetapkan Waktu habis koneksi berdasarkan persyaratan aplikasi dan beban kerja Anda. Untuk informasi selengkapnya, lihat bagian Waktu habis dalam topik ini.

ClusterClientOption: Mode Cluster Opsi klien yang diaktifkan

Aktifkan AutoReconnectsaat koneksi terputus.

Atur CommandTimeout. Untuk detail selengkapnya, lihat bagian Waktu habis dalam topik ini.

Atur nodeFilter untuk memfilter simpul yang gagal dari topologi. Lettuce menyimpan semua simpul yang ditemukan di output 'simpul klaster' (termasuk simpul dengan status PFAIL/FAIL) di 'partisi' klien (juga dikenal sebagai serpihan). Selama proses pembuatan topologi klaster, Lettuce mencoba terhubung ke semua simpul partisi. Perilaku Lettuce yang menambahkan simpul yang gagal ini dapat menyebabkan kesalahan koneksi (atau peringatan) ketika simpul diganti karena alasan apa pun.

Misalnya, setelah failover selesai dan klaster memulai proses pemulihan, sementara clusterTopology sedang diperbarui, peta simpul bus klaster memiliki periode waktu singkat ketika simpul yang nonaktif dicantumkan sebagai simpul FAIL, sebelum sepenuhnya dihapus dari topologi. Selama periode ini, klien Selada menganggapnya sebagai simpul yang sehat dan terus terhubung dengannya. Hal ini menyebabkan kegagalan setelah percobaan kembali habis.

Misalnya:

```
final ClusterClientOptions clusterClientOptions =
    ClusterClientOptions.builder()
    ... // other options
    .nodeFilter(it ->
        ! (it.is(RedisClusterNode.NodeFlag.FAIL)
        || it.is(RedisClusterNode.NodeFlag.EVENTUAL_FAIL)
        || it.is(RedisClusterNode.NodeFlag.HANDSHAKE)
        || it.is(RedisClusterNode.NodeFlag.NOADDR)))
    .validateClusterNodeMembership(false)
    .build();
redisClusterClient.setOptions(clusterClientOptions);
```

> **Note**
>
> Pemfilteran node paling baik digunakan dengan DynamicRefreshSources set ke true. Sebaliknya, jika tampilan topologi diambil dari satu simpul seed yang bermasalah, yang melihat simpul primer dari serpihan tertentu sebagai simpul yang mengalami kegagalan, pemfilteran tersebut akan memfilter simpul primer ini, sehingga akan mengakibatkan slot tidak tercakup. Memiliki beberapa node benih (kapan DynamicRefreshSources benar) mengurangi kemungkinan masalah ini, karena setidaknya beberapa node benih harus memiliki tampilan topologi yang diperbarui setelah failover dengan primer yang baru dipromosikan.

ClusterTopologyRefreshOptions: Opsi untuk mengontrol penyegaran topologi cluster dari klien Cluster Mode Enabled

> **Note**
>
> Klaster dengan mode klaster dinonaktifkan tidak mendukung perintah penemuan klaster dan tidak kompatibel dengan semua fungsionalitas penemuan topologi dinamis klien.
> Mode cluster dinonaktifkan dengan ElastiCache tidak kompatibel dengan Lettuce. `MasterSlaveTopologyRefresh` Sebagai gantinya, untuk mode klaster dinonaktifkan, Anda dapat mengonfigurasi `StaticMasterReplicaTopologyProvider` dan menyediakan titik akhir baca dan tulis klaster.
> Untuk informasi selengkapnya tentang menghubungkan ke klaster dengan mode klaster dinonaktifkan, lihat Menemukan Titik Akhir Cluster Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol).
> Jika Anda ingin menggunakan fungsionalitas penemuan topologi dinamis Lettuce, Anda dapat membuat klaster dengan mode klaster diaktifkan dengan konfigurasi serpihan yang sama dengan klaster yang ada. Namun, untuk klaster dengan mode klaster diaktifkan, sebaiknya konfigurasi setidaknya 3 serpihan dengan minimal 1 replika untuk mendukung failover cepat.

Aktifkan enablePeriodicRefresh. Opsi ini memungkinkan pembaruan topologi klaster berkala sehingga klien memperbarui topologi klaster dalam interval refreshPeriod (default: 60 detik). Ketika opsi ini dinonaktifkan, klien memperbarui topologi klaster hanya ketika kesalahan terjadi ketika klien tersebut mencoba menjalankan perintah terhadap klaster.

Dengan mengaktifkan opsi ini, Anda dapat mengurangi latensi yang terkait dengan penyegaran topologi klaster dengan menambahkan pekerjaan ini ke tugas latar belakang. Saat penyegaran topologi dilakukan dalam pekerjaan latar belakang, prosesnya bisa agak lambat untuk klaster yang memiliki banyak simpul. Hal ini karena semua simpul sedang dikueri untuk mendapatkan tampilan klaster yang paling terbaru. Jika Anda menjalankan klaster dalam jumlah besar, Anda sebaiknya menambah periodenya.

Aktifkan enableAllAdaptiveRefreshTriggers. Hal ini memungkinkan penyegaran topologi adaptif yang menggunakan semua pemicu: MOVED_REDIRECT, ASK_REDIRECT, PERSISTENT_RECONNECTS, UNCOVERED_SLOT, UNKNOWN_NODE. Pemicu penyegaran adaptif memulai pembaruan tampilan topologi berdasarkan peristiwa yang terjadi selama operasi klaster Valkey atau Redis OSS. Mengaktifkan opsi ini akan membuat penyegaran topologi langsung

dilakukan ketika salah satu pemicu sebelumnya terjadi. Penyegaran yang dipicu secara adaptif dibatasi lajunya menggunakan waktu habis karena peristiwa dapat terjadi dalam skala besar (waktu habis default di antara pembaruan: 30).

Aktifkan closeStaleConnections. Hal ini memungkinkan penutupan koneksi yang sudah tidak berlaku saat menyegarkan topologi klaster. Itu hanya berlaku jika ClusterTopologyRefreshOptions.isPeriodicRefreshEnabled () benar. Ketika diaktifkan, klien dapat menutup koneksi yang tidak berlaku dan membuat yang baru di latar belakang. Hal ini mengurangi kebutuhan untuk menangani koneksi yang gagal selama runtime perintah.

Aktifkan dynamicRefreshResources. Kami merekomendasikan dynamicRefreshResources untuk mengaktifkan cluster kecil, dan menonaktifkannya untuk cluster besar. dynamicRefreshResourcesmemungkinkan menemukan node cluster dari node benih yang disediakan (misalnya, titik akhir konfigurasi cluster). Ia menggunakan semua simpul yang ditemukan sebagai sumber untuk menyegarkan topologi klaster.

Menggunakan dynamic refresh akan mengueri semua simpul yang ditemukan untuk topologi klaster dan mencoba memilih tampilan klaster yang paling akurat. Jika diatur ke false, hanya simpul seed awal yang digunakan sebagai sumber untuk penemuan topologi, dan jumlah klien diperoleh hanya untuk simpul seed awal. Saat dinonaktifkan, jika titik akhir konfigurasi klaster diresolusi ke simpul yang gagal, percobaan untuk menyegarkan tampilan klaster akan gagal dan menyebabkan pengecualian. Skenario ini dapat terjadi karena dibutuhkan beberapa waktu hingga entri simpul yang gagal dihapus dari titik akhir konfigurasi klaster. Oleh karena itu, titik akhir konfigurasi masih dapat diresolusi secara acak ke simpul yang gagal untuk waktu yang singkat.

Namun, ketika diaktifkan, kita menggunakan semua simpul klaster yang diterima dari tampilan klaster untuk mengueri tampilannya saat ini. Karena kita memfilter simpul yang gagal dari tampilan tersebut, penyegaran topologi akan berhasil. Namun, kapan dynamicRefreshSources benar, Lettuce menanyakan semua node untuk mendapatkan tampilan cluster, dan kemudian membandingkan hasilnya. Jadi, klaster dengan banyak simpul bisa menghabiskan banyak daya komputasi. Sebaiknya nonaktifkan fitur ini untuk klaster dengan banyak simpul.

```
final ClusterTopologyRefreshOptions topologyOptions =
    ClusterTopologyRefreshOptions.builder()
    .enableAllAdaptiveRefreshTriggers()
    .enablePeriodicRefresh()
    .dynamicRefreshSources(true)
    .build();
```

ClientResources

Konfigurasikan DnsResolverdengan DirContextDnsResolver. DNS resolver didasarkan pada com.sun.jndi.dns Java. DnsContextFactory.

Konfigurasikan reconnectDelay dengan backoff eksponensial dan jitter penuh. Lettuce memiliki mekanisme percobaan ulang bawaan berdasarkan strategi backoff eksponensial. Untuk detailnya, lihat Exponential Backoff dan Jitter di Blog Arsitektur. AWS Untuk informasi lebih lanjut tentang pentingnya memiliki strategi backoff coba lagi, lihat bagian logika backoff dari posting blog Praktik Terbaik di Blog Database. AWS

```
ClientResources clientResources = DefaultClientResources.builder()
   .dnsResolver(new DirContextDnsResolver())
    .reconnectDelay(
        Delay.fullJitter(
            Duration.ofMillis(100),     // minimum 100 millisecond delay
            Duration.ofSeconds(10),      // maximum 10 second delay
            100, TimeUnit.MILLISECONDS)) // 100 millisecond base
    .build();
```

Batas Waktu

Gunakan nilai waktu habis koneksi yang lebih rendah daripada waktu habis perintah Anda. Lettuce menggunakan pembuatan lazy connection. Jadi, jika waktu habis koneksi lebih tinggi dari waktu habis perintah, Anda dapat mengalami periode kegagalan persisten setelah penyegaran topologi jika Lettuce mencoba terhubung ke simpul yang tidak berkondisi baik dan waktu habis perintah selalu terlampaui.

Gunakan waktu habis perintah dinamis untuk perintah yang berbeda-beda. Sebaiknya tetapkan waktu habis perintah berdasarkan durasi perintah yang diharapkan. Misalnya, gunakan waktu habis yang lebih lama untuk perintah yang mengulangi beberapa kunci, seperti skrip FLUSHDB, FLUSHALL, KEYS, SMEMBERS, atau Lua. Gunakan waktu habis yang lebih pendek untuk perintah kunci tunggal, seperti SET, GET, dan HSET.

> **Note**
>
> Waktu habis yang dikonfigurasi dalam contoh berikut adalah untuk pengujian yang menjalankan perintah SET/GET dengan kunci dan nilai hingga 20 byte. Waktu pemrosesan bisa lebih lama jika perintahnya kompleks atau kunci dan nilainya lebih besar. Anda harus mengatur waktu habis berdasarkan kasus penggunaan aplikasi Anda.

```
private static final Duration META_COMMAND_TIMEOUT = Duration.ofMillis(1000);
private static final Duration DEFAULT_COMMAND_TIMEOUT = Duration.ofMillis(250);
// Socket connect timeout should be lower than command timeout for Lettuce
private static final Duration CONNECT_TIMEOUT = Duration.ofMillis(100);

SocketOptions socketOptions = SocketOptions.builder()
    .connectTimeout(CONNECT_TIMEOUT)
    .build();


class DynamicClusterTimeout extends TimeoutSource {
     private static final Set<ProtocolKeyword> META_COMMAND_TYPES =
 ImmutableSet.<ProtocolKeyword>builder()
          .add(CommandType.FLUSHDB)
          .add(CommandType.FLUSHALL)
          .add(CommandType.CLUSTER)
          .add(CommandType.INFO)
          .add(CommandType.KEYS)
          .build();

    private final Duration defaultCommandTimeout;
    private final Duration metaCommandTimeout;

    DynamicClusterTimeout(Duration defaultTimeout, Duration metaTimeout)
    {
        defaultCommandTimeout = defaultTimeout;
        metaCommandTimeout = metaTimeout;
    }

    @Override
    public long getTimeout(RedisCommand<?, ?, ?> command) {
        if (META_COMMAND_TYPES.contains(command.getType())) {
            return metaCommandTimeout.toMillis();
        }
        return defaultCommandTimeout.toMillis();
    }
}

// Use a dynamic timeout for commands, to avoid timeouts during
// cluster management and slow operations.
TimeoutOptions timeoutOptions = TimeoutOptions.builder()
.timeoutSource(
    new DynamicClusterTimeout(DEFAULT_COMMAND_TIMEOUT, META_COMMAND_TIMEOUT))
```

```
.build();
```

Contoh: Konfigurasi selada untuk mode cluster, TLS diaktifkan

> **Note**
>
> Timeout dalam contoh berikut adalah untuk tes yang menjalankan SET/GET perintah dengan kunci dan nilai hingga 20 byte. Waktu pemrosesan bisa lebih lama jika perintahnya kompleks atau kunci dan nilainya lebih besar. Anda harus mengatur waktu habis berdasarkan kasus penggunaan aplikasi Anda.

```
// Set DNS cache TTL
public void setJVMProperties() {
    java.security.Security.setProperty("networkaddress.cache.ttl", "10");
}

private static final Duration META_COMMAND_TIMEOUT = Duration.ofMillis(1000);
private static final Duration DEFAULT_COMMAND_TIMEOUT = Duration.ofMillis(250);
// Socket connect timeout should be lower than command timeout for Lettuce
private static final Duration CONNECT_TIMEOUT = Duration.ofMillis(100);

// Create RedisURI from the cluster configuration endpoint
clusterConfigurationEndpoint = <cluster-configuration-endpoint> // TODO: add your
 cluster configuration endpoint
final RedisURI redisUriCluster =
    RedisURI.Builder.redis(clusterConfigurationEndpoint)
        .withPort(6379)
        .withSsl(true)
        .build();

// Configure the client's resources
ClientResources clientResources = DefaultClientResources.builder()
    .reconnectDelay(
        Delay.fullJitter(
            Duration.ofMillis(100),     // minimum 100 millisecond delay
            Duration.ofSeconds(10),      // maximum 10 second delay
            100, TimeUnit.MILLISECONDS)) // 100 millisecond base
    .dnsResolver(new DirContextDnsResolver())
    .build();

// Create a cluster client instance with the URI and resources
```

```
RedisClusterClient redisClusterClient =
    RedisClusterClient.create(clientResources, redisUriCluster);

// Use a dynamic timeout for commands, to avoid timeouts during
// cluster management and slow operations.
class DynamicClusterTimeout extends TimeoutSource {
     private static final Set<ProtocolKeyword> META_COMMAND_TYPES =
 ImmutableSet.<ProtocolKeyword>builder()
          .add(CommandType.FLUSHDB)
          .add(CommandType.FLUSHALL)
          .add(CommandType.CLUSTER)
          .add(CommandType.INFO)
          .add(CommandType.KEYS)
          .build();

    private final Duration metaCommandTimeout;
    private final Duration defaultCommandTimeout;

    DynamicClusterTimeout(Duration defaultTimeout, Duration metaTimeout)
    {
        defaultCommandTimeout = defaultTimeout;
        metaCommandTimeout = metaTimeout;
    }

    @Override
    public long getTimeout(RedisCommand<?, ?, ?> command) {
        if (META_COMMAND_TYPES.contains(command.getType())) {
            return metaCommandTimeout.toMillis();
        }
        return defaultCommandTimeout.toMillis();
    }
}

TimeoutOptions timeoutOptions = TimeoutOptions.builder()
    .timeoutSource(new DynamicClusterTimeout(DEFAULT_COMMAND_TIMEOUT,
 META_COMMAND_TIMEOUT))
     .build();

// Configure the topology refreshment options
final ClusterTopologyRefreshOptions topologyOptions =
    ClusterTopologyRefreshOptions.builder()
    .enableAllAdaptiveRefreshTriggers()
    .enablePeriodicRefresh()
    .dynamicRefreshSources(true)
```

```
    .build();

// Configure the socket options
final SocketOptions socketOptions =
    SocketOptions.builder()
    .connectTimeout(CONNECT_TIMEOUT)
    .keepAlive(true)
    .build();

// Configure the client's options
final ClusterClientOptions clusterClientOptions =
    ClusterClientOptions.builder()
    .topologyRefreshOptions(topologyOptions)
    .socketOptions(socketOptions)
    .autoReconnect(true)
    .timeoutOptions(timeoutOptions)
    .nodeFilter(it ->
        ! (it.is(RedisClusterNode.NodeFlag.FAIL)
        || it.is(RedisClusterNode.NodeFlag.EVENTUAL_FAIL)
        || it.is(RedisClusterNode.NodeFlag.NOADDR)))
    .validateClusterNodeMembership(false)
    .build();

redisClusterClient.setOptions(clusterClientOptions);

// Get a connection
final StatefulRedisClusterConnection<String, String> connection =
    redisClusterClient.connect();

// Get cluster sync/async commands
RedisAdvancedClusterCommands<String, String> sync = connection.sync();
RedisAdvancedClusterAsyncCommands<String, String> async = connection.async();
```

Contoh: Konfigurasi selada untuk mode cluster dinonaktifkan, TLS diaktifkan

> **Note**
>
> Timeout dalam contoh berikut adalah untuk tes yang menjalankan SET/GET perintah dengan kunci dan nilai hingga 20 byte. Waktu pemrosesan bisa lebih lama jika perintahnya kompleks atau kunci dan nilainya lebih besar. Anda harus mengatur waktu habis berdasarkan kasus penggunaan aplikasi Anda.

```
// Set DNS cache TTL
public void setJVMProperties() {
    java.security.Security.setProperty("networkaddress.cache.ttl", "10");
}

private static final Duration META_COMMAND_TIMEOUT = Duration.ofMillis(1000);
private static final Duration DEFAULT_COMMAND_TIMEOUT = Duration.ofMillis(250);
// Socket connect timeout should be lower than command timeout for Lettuce
private static final Duration CONNECT_TIMEOUT = Duration.ofMillis(100);

// Create RedisURI from the primary/reader endpoint
clusterEndpoint = <primary/reader-endpoint> // TODO: add your node endpoint
RedisURI redisUriStandalone =

 RedisURI.Builder.redis(clusterEndpoint).withPort(6379).withSsl(true).withDatabase(0).build();

ClientResources clientResources =
    DefaultClientResources.builder()
        .dnsResolver(new DirContextDnsResolver())
        .reconnectDelay(
            Delay.fullJitter(
                Duration.ofMillis(100), // minimum 100 millisecond delay
                Duration.ofSeconds(10), // maximum 10 second delay
                100,
                TimeUnit.MILLISECONDS)) // 100 millisecond base
        .build();

// Use a dynamic timeout for commands, to avoid timeouts during
// slow operations.
class DynamicTimeout extends TimeoutSource {
     private static final Set<ProtocolKeyword> META_COMMAND_TYPES =
 ImmutableSet.<ProtocolKeyword>builder()
          .add(CommandType.FLUSHDB)
          .add(CommandType.FLUSHALL)
          .add(CommandType.INFO)
          .add(CommandType.KEYS)
          .build();

    private final Duration metaCommandTimeout;
    private final Duration defaultCommandTimeout;

    DynamicTimeout(Duration defaultTimeout, Duration metaTimeout)
    {
```

```
        defaultCommandTimeout = defaultTimeout;
        metaCommandTimeout = metaTimeout;
    }

    @Override
    public long getTimeout(RedisCommand<?, ?, ?> command) {
        if (META_COMMAND_TYPES.contains(command.getType())) {
            return metaCommandTimeout.toMillis();
        }
        return defaultCommandTimeout.toMillis();
    }
}

TimeoutOptions timeoutOptions = TimeoutOptions.builder()
    .timeoutSource(new DynamicTimeout(DEFAULT_COMMAND_TIMEOUT, META_COMMAND_TIMEOUT))
     .build();

final SocketOptions socketOptions =
    SocketOptions.builder().connectTimeout(CONNECT_TIMEOUT).keepAlive(true).build();

ClientOptions clientOptions =

 ClientOptions.builder().timeoutOptions(timeoutOptions).socketOptions(socketOptions).build();

RedisClient redisClient = RedisClient.create(clientResources, redisUriStandalone);
redisClient.setOptions(clientOptions);
```

Mengkonfigurasi protokol pilihan untuk cluster tumpukan ganda (Valkey dan Redis OSS)

Untuk mode cluster diaktifkan Valkey atau Redis OSS cluster, Anda dapat mengontrol klien protokol yang akan digunakan untuk terhubung ke node di cluster dengan parameter IP Discovery. Parameter IP Discovery dapat diatur ke salah satu IPv4 atau IPv6.

Untuk cluster Valkey atau Redis OSS, parameter penemuan IP menetapkan protokol IP yang digunakan dalam slot cluster (), pecahan cluster (), dan node cluster () output. Perintah tersebut digunakan oleh klien untuk menemukan topologi klaster. Klien menggunakan perintah IPs in theses untuk terhubung ke node lain di cluster.

Perubahan pada Penemuan IP tidak akan mengakibatkan waktu henti untuk klien yang terhubung. Namun, perubahan ini akan memakan waktu untuk disebarkan. Untuk menentukan kapan perubahan telah sepenuhnya disebarkan untuk Cluster Valkey atau Redis OSS, pantau output dari. `cluster`

`slots` Setelah semua node dikembalikan oleh laporan perintah slot cluster IPs dengan protokol baru, perubahan telah selesai menyebar.

Contoh dengan Redis-Py:

```
cluster = RedisCluster(host="xxxx", port=6379)
target_type = IPv6Address # Or IPv4Address if changing to IPv4

nodes = set()
while len(nodes) == 0 or not all((type(ip_address(host)) is target_type) for host in
 nodes):
    nodes = set()

   # This refreshes the cluster topology and will discovery any node updates.
   # Under the hood it calls cluster slots
    cluster.nodes_manager.initialize()
    for node in cluster.get_nodes():
        nodes.add(node.host)
    self.logger.info(nodes)

    time.sleep(1)
```

Contoh dengan Lettuce:

```
RedisClusterClient clusterClient = RedisClusterClient.create(RedisURI.create("xxxx",
 6379));

Class targetProtocolType = Inet6Address.class; // Or Inet4Address.class if you're
 switching to IPv4

Set<String> nodes;

do {
   // Check for any changes in the cluster topology.
   // Under the hood this calls cluster slots
    clusterClient.refreshPartitions();
    Set<String> nodes = new HashSet<>();

    for (RedisClusterNode node : clusterClient.getPartitions().getPartitions()) {
        nodes.add(node.getUri().getHost());
    }

    Thread.sleep(1000);
```

```
} while (!nodes.stream().allMatch(node -> {
            try {
                return finalTargetProtocolType.isInstance(InetAddress.getByName(node));
            } catch (UnknownHostException ignored) {}
            return false;
}));
```

Praktik terbaik untuk klien (Memcached)

Mengkonfigurasi ElastiCache klien Anda untuk penyeimbangan beban yang efisien (Memcached)

> **Note**
>
> Bagian ini berlaku untuk klaster Memcached multisimpul yang dirancang sendiri.

Untuk secara efektif menggunakan beberapa node ElastiCache Memcached, Anda harus dapat menyebarkan kunci cache Anda di seluruh node. Cara sederhana untuk memuat keseimbangan cluster dengan n node adalah dengan menghitung hash dari kunci objek dan mod hasilnya dengan n:`hash(key) mod n`. Nilai yang dihasilkan (0 sampai n–1) adalah jumlah simpul tempat Anda menempatkan objek.

Pendekatan ini sederhana dan berfungsi dengan baik selama jumlah simpul (n) adalah konstan. Namun, setiap kali Anda menambahkan atau menghapus simpul dari klaster, jumlah kunci yang perlu dipindahkan adalah (n - 1)/n (dengan n adalah jumlah baru simpul). Jadi, pendekatan ini menghasilkan sejumlah besar kunci yang dipindahkan, yang berarti ada sejumlah besar cache miss awal, terutama seiring bertambahnya jumlah simpul. Dalam kasus terbaik, penskalaan dari 1 ke 2 simpul menghasilkan pemindahan kunci (2-1) / 2 (50 persen). Penskalaan dari 9 ke 10 simpul menghasilkan pemindahan kunci (10–1)/10 (90 persen). Jika Anda menaikkan skala karena lonjakan lalu lintas, Anda tidak ingin memiliki banyak cache miss. Banyaknya jumlah cache miss akan menghasilkan hit pada basis data, yang sudah kelebihan beban karena lonjakan lalu lintas.

Solusi untuk dilema ini adalah melakukan hashing secara konsisten. Hashing konsisten menggunakan algoritma sehingga setiap kali simpul ditambahkan atau dihapus dari klaster, jumlah kunci yang harus dipindahkan adalah kira-kira 1 / n (dengan n adalah jumlah baru simpul). Dalam kasus terburuk, penskalaan dari 1 ke 2 simpul menghasilkan pemindahan kunci 1/2 (50 persen). Penskalaan dari 9 ke 10 simpul menghasilkan pemindahan kunci 1/10 (10 persen).

Sebagai pengguna, Anda mengontrol jenis algoritma hashing yang digunakan untuk klaster multisimpul. Sebaiknya konfigurasikan klien Anda untuk menggunakan hashing konsisten. Untungnya, ada banyak pustaka klien Memcached dalam bahasa yang paling populer yang menerapkan hashing konsisten. Periksa dokumentasi untuk pustaka yang Anda gunakan untuk melihat apakah mendukung hashing konsisten dan cara menerapkannya.

Jika Anda bekerja di Java, PHP, atau .NET, kami sarankan Anda menggunakan salah satu pustaka ElastiCache klien Amazon.

Hashing konsisten Menggunakan Java

Klien Java ElastiCache Memcached didasarkan pada klien Java spymemcached open-source, yang memiliki kemampuan hashing yang konsisten bawaan. Pustaka menyertakan KetamaConnectionFactory kelas yang mengimplementasikan hashing yang konsisten. Secara default, hashing konsisten dinonaktifkan pada spymemcached.

Untuk informasi lebih lanjut, lihat KetamaConnectionFactory dokumentasi di KetamaConnectionFactory.

Hashing konsisten menggunakan PHP dengan Memcached

Klien PHP ElastiCache Memcached adalah pembungkus di sekitar perpustakaan PHP Memcached bawaan. Secara default, hashing konsisten dinonaktifkan oleh pustaka PHP Memcached.

Gunakan kode berikut untuk mengaktifkan hashing konsisten.

```
$m = new Memcached();
$m->setOption(Memcached::OPT_DISTRIBUTION, Memcached::DISTRIBUTION_CONSISTENT);
```

Selain kode sebelumnya, sebaiknya Anda juga mengaktifkan `memcached.sess_consistent_hash` pada file php.ini Anda.

Untuk informasi selengkapnya, lihat dokumentasi konfigurasi run-time untuk PHP Memcached di http://php. net/manual/en/memcached.configuration.php. Perhatikan secara khusus parameter `memcached.sess_consistent_hash`.

Hashing konsisten menggunakan.NET dengan Memcached

Klien ElastiCache Memcached .NET adalah pembungkus di sekitar Enyim Memcached. Secara default, hashing konsisten diaktifkan oleh klien Enyim Memcached.

Untuk informasi selengkapnya, lihat `memcached/locator` dokumentasi di https://github.com/enyim/EnyimMemcached/wiki/MemcachedClient-Configuration# user-content-memcachedlocator.

Klien yang divalidasi dengan Memcached

Klien berikut secara khusus telah divalidasi untuk bekerja dengan semua konfigurasi jenis jaringan yang didukung untuk Memcached.

Klien yang Divalidasi:

- AWS ElastiCache Klien Cluster Memcached untuk Php - Versi* 3.6.2
- AWS ElastiCache Cluster Client Memcached untuk Java - Master terbaru di Github

Mengkonfigurasi protokol pilihan untuk cluster tumpukan ganda (Memcached)

Untuk klaster Memcached, Anda dapat mengontrol protokol yang akan digunakan klien untuk terhubung ke simpul di klaster dengan parameter Penemuan IP. Parameter IP Discovery dapat diatur ke salah satu IPv4 atau IPv6.

Parameter Penemuan IP mengontrol protokol IP yang digunakan dalam output config get cluster. Yang pada gilirannya akan menentukan protokol IP yang digunakan oleh klien yang mendukung penemuan otomatis ElastiCache untuk klaster Memcached.

Perubahan pada Penemuan IP tidak akan mengakibatkan waktu henti untuk klien yang terhubung. Namun, perubahan ini akan memakan waktu untuk disebarkan.

Pantau output `getAvailableNodeEndPoints` untuk Java, sementara untuk Php, pantau output dari `getServerList`. Setelah output dari fungsi-fungsi ini melaporkan IPs semua node di cluster yang menggunakan protokol yang diperbarui, perubahan telah selesai menyebar.

Contoh Java:

```
MemcachedClient client = new MemcachedClient(new InetSocketAddress("xxxx", 11211));

Class targetProtocolType = Inet6Address.class; // Or Inet4Address.class if you're
 switching to IPv4

Set<String> nodes;

do {
```

```
    nodes =
 client.getAvailableNodeEndPoints().stream().map(NodeEndPoint::getIpAddress).collect(Collectors

    Thread.sleep(1000);
} while (!nodes.stream().allMatch(node -> {
            try {
                return finalTargetProtocolType.isInstance(InetAddress.getByName(node));
            } catch (UnknownHostException ignored) {}
            return false;
        }));
```

Contoh Php:

```
$client = new Memcached;
$client->setOption(Memcached::OPT_CLIENT_MODE, Memcached::DYNAMIC_CLIENT_MODE);
$client->addServer("xxxx", 11211);

$nodes = [];
$target_ips_count = 0;
do {
    # The PHP memcached client only updates the server list if the polling interval has
 expired and a
    # command is sent
    $client->get('test');

    $nodes = $client->getServerList();

    sleep(1);
    $target_ips_count = 0;

    // For IPv4 use FILTER_FLAG_IPV4
    $target_ips_count = count(array_filter($nodes, function($node) { return
 filter_var($node["ipaddress"], FILTER_VALIDATE_IP, FILTER_FLAG_IPV6); }));

} while (count($nodes) !== $target_ips_count);
```

Setiap koneksi klien yang ada yang dibuat sebelum Penemuan IP diperbarui akan tetap terhubung menggunakan protokol lama. Semua klien yang divalidasi akan secara otomatis terhubung kembali ke klaster menggunakan protokol IP baru setelah perubahan terdeteksi dalam output perintah penemuan klaster. Namun, hal ini tergantung pada implementasi klien.

TLS mengaktifkan cluster tumpukan ElastiCache ganda

Ketika TLS diaktifkan untuk cluster fungsi penemuan cluster (`cluster slots`,`cluster shards`, dan `cluster nodes` untuk Redis) atau `config get cluster` untuk Memcached mengembalikan ElastiCache nama host alih-alih. IPs Nama host kemudian digunakan sebagai pengganti IPs untuk terhubung ke ElastiCache cluster dan melakukan jabat tangan TLS. Hal ini berarti bahwa klien tidak akan terpengaruh oleh parameter Penemuan IP. Untuk klaster dengan TLS diaktifkan, parameter Penemuan IP tidak berpengaruh pada protokol IP pilihan. Sebagai gantinya, protokol IP yang digunakan akan ditentukan berdasarkan protokol IP mana yang lebih dipilih klien saat meresolusi nama host DNS.

Klien Java

Saat menghubungkan dari lingkungan Java yang mendukung keduanya IPv4 dan IPv6, Java secara default akan IPv4 lebih memilih IPv6 untuk kompatibilitas mundur. Namun, preferensi protokol IP dapat dikonfigurasi melalui argumen JVM. Untuk memilih IPv4, JVM menerima `-Djava.net.preferIPv4Stack=true` dan memilih set. IPv6 `-Djava.net.preferIPv6Stack=true` Pengaturan `-Djava.net.preferIPv4Stack=true` berarti bahwa JVM tidak akan lagi membuat koneksi apa pun IPv6 . Untuk Valkey atau Redis OSS, ini termasuk aplikasi OSS non-Valkey dan non-Redis lainnya.

Preferensi Tingkat Host

Secara umum, jika klien atau runtime klien tidak menyediakan opsi konfigurasi untuk mengatur preferensi protokol IP, saat melakukan resolusi DNS, protokol IP akan bergantung pada konfigurasi host. Secara default, sebagian besar host IPv6 lebih suka daripada IPv4 tetapi preferensi ini dapat dikonfigurasi di tingkat host. Ini akan memengaruhi semua permintaan DNS dari host itu, bukan hanya permintaan ke ElastiCache cluster.

Host Linux

Untuk Linux, preferensi protokol IP dapat dikonfigurasi dengan mengubah file `gai.conf`. File `gai.conf` dapat ditemukan dalam `/etc/gai.conf`. Jika tidak ada `gai.conf` yang ditentukan, maka contohnya akan tersedia di `/usr/share/doc/glibc-common-x.xx/gai.conf` yang dapat disalin ke `/etc/gai.conf` lalu konfigurasi default-nya harus di-uncommenting. Untuk memperbarui konfigurasi agar lebih disukai IPv4 saat menghubungkan ke ElastiCache kluster, perbarui prioritas untuk rentang CIDR yang mencakup cluster berada di atas prioritas IPs untuk koneksi default. IPv6 Secara default IPv6 koneksi memiliki prioritas 40. Misalnya, dengan asumsi cluster terletak di subnet

dengan CIDR 172.31.0. 0:0 /16, konfigurasi di bawah ini akan menyebabkan klien lebih memilih koneksi ke cluster itu. IPv4

```
label ::1/128       0
label ::/0          1
label 2002::/16     2
label ::/96         3
label ::ffff:0:0/96 4
label fec0::/10     5
label fc00::/7      6
label 2001:0::/32   7
label ::ffff:172.31.0.0/112 8
#
#    This default differs from the tables given in RFC 3484 by handling
#    (now obsolete) site-local IPv6 addresses and Unique Local Addresses.
#    The reason for this difference is that these addresses are never
#    NATed while IPv4 site-local addresses most probably are.  Given
#    the precedence of IPv6 over IPv4 (see below) on machines having only
#    site-local IPv4 and IPv6 addresses a lookup for a global address would
#    see the IPv6 be preferred.  The result is a long delay because the
#    site-local IPv6 addresses cannot be used while the IPv4 address is
#    (at least for the foreseeable future) NATed.  We also treat Teredo
#    tunnels special.
#
# precedence  <mask>   <value>
#    Add another rule to the RFC 3484 precedence table.  See section 2.1
#    and 10.3 in RFC 3484.  The default is:
#
precedence  ::1/128       50
precedence  ::/0          40
precedence  2002::/16     30
precedence ::/96          20
precedence ::ffff:0:0/96  10
precedence ::ffff:172.31.0.0/112 100
```

Rincian lebih lanjut tentang `gai.conf` tersedia di halaman manual Linux

Host Windows

Proses untuk host Windows juga serupa. Untuk host Windows, Anda dapat menjalankan `netsh interface ipv6 set prefix CIDR_CONTAINING_CLUSTER_IPS PRECEDENCE LABEL`. Hal ini memiliki efek yang sama seperti mengubah file `gai.conf` pada host Linux.

Ini akan memperbarui kebijakan preferensi untuk memilih IPv4 koneksi daripada IPv6 koneksi untuk rentang CIDR yang ditentukan. Misalnya, dengan asumsi bahwa cluster berada dalam subnet dengan 172.31.0. 0:0 /16 CIDR mengeksekusi `netsh interface ipv6 set prefix ::ffff:172.31.0.0:0/112 100 15` akan menghasilkan tabel prioritas berikut yang akan menyebabkan klien lebih suka saat menghubungkan ke cluster. IPv4

```
C:\Users\Administrator>netsh interface ipv6 show prefixpolicies
Querying active state...

Precedence Label Prefix
---------- ----- --------------------------------
100 15 ::ffff:172.31.0.0:0/112
20 4 ::ffff:0:0/96
50 0 ::1/128
40 1 ::/0
30 2 2002::/16
5 5 2001::/32
3 13 fc00::/7
1 11 fec0::/10
1 12 3ffe::/16
1 3 ::/96
```

Mengelola memori cadangan untuk Valkey dan Redis OSS

Memori cadangan adalah memori yang disisihkan untuk penggunaan nondata. Saat melakukan pencadangan atau failover, Valkey dan Redis OSS menggunakan memori yang tersedia untuk merekam operasi tulis ke cluster Anda saat data cluster sedang ditulis ke file.rdb. Jika tidak tersedia memori yang cukup untuk semua operasi tulis, proses tersebut akan gagal. Berikut ini, Anda dapat menemukan informasi tentang opsi untuk mengelola memori cadangan ElastiCache untuk Redis OSS dan cara menerapkan opsi tersebut.

Topik

Berapa Banyak Memori Cadangan yang Anda Butuhkan?

Jika Anda menjalankan versi Redis OSS sebelum 2.8.22, cadangan lebih banyak memori untuk backup dan failovers daripada jika Anda menjalankan Redis OSS 2.8.22 atau yang lebih baru. Persyaratan ini disebabkan oleh berbagai cara yang ElastiCache untuk Redis OSS mengimplementasikan proses pencadangan. Aturan praktisnya adalah untuk mencadangkan setengah dari `maxmemory` nilai tipe node untuk overhead Redis OSS untuk versi sebelum 2.8.22, dan seperempat untuk Redis OSS versi 2.8.22 dan yang lebih baru.

Karena berbagai cara yang ElastiCache mengimplementasikan proses pencadangan dan replikasi, aturan praktisnya adalah mencadangkan 25% dari `maxmemory` nilai tipe node dengan menggunakan parameter. `reserved-memory-percent` Ini adalah nilai default dan direkomendasikan untuk sebagian besar kasus.

Ketika tipe instans mikro dan kecil burstable beroperasi mendekati `maxmemory` batas, mereka mungkin mengalami penggunaan swap. Untuk meningkatkan keandalan operasional pada jenis instans ini selama pencadangan, replikasi, dan lalu lintas tinggi, kami sarankan untuk meningkatkan nilai `reserved-memory-percent` parameter hingga 30% pada jenis instans kecil, dan hingga 50% pada jenis instans mikro.

Untuk beban kerja berat tulis pada ElastiCache cluster dengan tiering data, kami sarankan untuk meningkatkan `reserved-memory-percent` hingga 50% dari memori node yang tersedia.

Untuk informasi lain, lihat hal berikut:

- Memastikan Anda memiliki cukup memori untuk membuat snapshot Valkey atau Redis OSS
- Cara penerapan sinkronisasi dan pencadangan
- Tingkatan data di ElastiCache

Parameter untuk Mengelola Memori Cadangan

Mulai 16 Maret 2017, Amazon ElastiCache menyediakan dua parameter yang saling eksklusif untuk mengelola memori Valkey atau Redis OSS Anda, dan. `reserved-memory` `reserved-memory-percent` Tak satu pun dari parameter ini merupakan bagian dari distribusi Valkey atau Redis OSS.

Bergantung pada kapan Anda menjadi ElastiCache pelanggan, satu atau yang lain dari parameter ini adalah parameter manajemen memori default. Parameter ini berlaku ketika Anda membuat cluster Valkey atau Redis OSS baru atau grup replikasi dan menggunakan grup parameter default.

- Untuk pelanggan yang memulai sebelum 16 Maret 2017 - Saat Anda membuat cluster Redis OSS atau grup replikasi menggunakan grup parameter default, parameter manajemen memori Anda adalah. `reserved-memory` Dalam hal ini, memori cadangan adalah nol (0) byte.
- Untuk pelanggan yang memulai pada atau setelah 16 Maret 2017 - Saat Anda membuat cluster Valkey atau Redis OSS atau grup replikasi menggunakan grup parameter default, parameter manajemen memori Anda adalah. `reserved-memory-percent` Dalam hal ini, 25 persen dari nilai `maxmemory` dari simpul Anda dicadangkan untuk tujuan nondata.

Setelah membaca tentang dua parameter manajemen memori Valkey atau Redis OSS, Anda mungkin lebih suka menggunakan salah satu yang bukan default Anda atau dengan nilai nondefault. Jika demikian, Anda dapat mengubah ke parameter manajemen memori cadangan yang lain.

Untuk mengubah nilai parameter tersebut, Anda dapat membuat grup parameter kustom dan mengubahnya untuk menggunakan parameter dan nilai manajemen memori pilihan Anda. Anda kemudian dapat menggunakan grup parameter kustom setiap kali Anda membuat cluster Valkey atau Redis OSS baru atau grup replikasi. Untuk klaster atau grup replikasi yang sudah ada, Anda dapat mengubahnya untuk menggunakan grup parameter kustom.

Untuk informasi selengkapnya, lihat berikut ini:

- Menentukan Parameter Manajemen Memori Cadangan
- Membuat grup ElastiCache parameter
- Memodifikasi grup ElastiCache parameter

- Memodifikasi cluster ElastiCache
- Mengubah grup replikasi

Parameter reserved-memory

Sebelum 16 Maret 2017, semua ElastiCache untuk manajemen memori cadangan Redis OSS dilakukan dengan menggunakan parameter. `reserved-memory` Nilai default `reserved-memory` adalah 0. Default ini tidak menyimpan memori untuk overhead Valkey atau Redis OSS dan memungkinkan Valkey atau Redis OSS untuk mengkonsumsi semua memori node dengan data.

Mengubah `reserved-memory` agar Anda memiliki cukup memori tersedia untuk pencadangan dan failover mengharuskan Anda membuat grup parameter kustom. Dalam grup parameter kustom ini, Anda menyetel `reserved-memory` ke nilai yang sesuai untuk versi Valkey atau Redis OSS yang berjalan pada tipe node cluster dan cluster Anda. Untuk informasi selengkapnya, lihat Berapa Banyak Memori Cadangan yang Anda Butuhkan?

Parameter `reserved-memory` khusus untuk ElastiCache dan bukan bagian dari distribusi Redis OSS umum.

Prosedur berikut menunjukkan cara menggunakan `reserved-memory` untuk mengelola memori pada cluster Valkey atau Redis OSS Anda.

Untuk mencadangkan memori menggunakan reserved-memory

1. Buat grup parameter kustom yang menentukan keluarga grup parameter yang cocok dengan versi mesin yang Anda jalankan—misalnya, menentukan keluarga grup parameter `redis2.8`. Untuk informasi selengkapnya, lihat Membuat grup ElastiCache parameter.

   ```
   aws elasticache create-cache-parameter-group \
      --cache-parameter-group-name redis6x-m3xl \
      --description "Redis OSS 2.8.x for m3.xlarge node type" \
      --cache-parameter-group-family redis6.x
   ```

2. Hitung berapa byte memori yang akan dicadangkan untuk overhead Valkey atau Redis OSS. Anda dapat menemukan nilai `maxmemory` untuk jenis simpul Anda di Parameter spesifik tipe node Redis OSS.
3. Ubah grup parameter kustom agar parameter `reserved-memory` adalah jumlah byte yang Anda hitung pada langkah sebelumnya. AWS CLI Contoh berikut mengasumsikan Anda

menjalankan versi Redis OSS sebelum 2.8.22 dan perlu memesan setengah dari node. `maxmemory` Untuk informasi selengkapnya, lihat Memodifikasi grup ElastiCache parameter.

```
aws elasticache modify-cache-parameter-group \
   --cache-parameter-group-name redis28-m3xl \
   --parameter-name-values "ParameterName=reserved-memory,
 ParameterValue=7130316800"
```

Anda memerlukan grup parameter kustom terpisah untuk setiap jenis simpul yang Anda gunakan, karena setiap jenis simpul memiliki nilai `maxmemory` yang berbeda. Dengan demikian, setiap jenis simpul membutuhkan nilai yang berbeda untuk `reserved-memory`.

4. Ubah cluster Redis OSS atau grup replikasi Anda untuk menggunakan grup parameter kustom Anda.

   Contoh CLI berikut mengubah klaster `my-redis-cluster` untuk menggunakan grup parameter kustom `redis28-m3xl` yang dimulai segera. Untuk informasi selengkapnya, lihat Memodifikasi cluster ElastiCache .

   ```
   aws elasticache modify-cache-cluster \
      --cache-cluster-id my-redis-cluster \
      --cache-parameter-group-name redis28-m3xl \
      --apply-immediately
   ```

   Contoh CLI berikut mengubah grup replikasi `my-redis-repl-grp` untuk menggunakan grup parameter kustom `redis28-m3xl` yang dimulai segera. Untuk informasi selengkapnya, lihat Mengubah grup replikasi.

   ```
   aws elasticache modify-replication-group \
      --replication-group-id my-redis-repl-grp \
      --cache-parameter-group-name redis28-m3xl \
      --apply-immediately
   ```

reserved-memory-percentParameternya

Pada 16 Maret 2017, Amazon ElastiCache memperkenalkan parameter `reserved-memory-percent` dan membuatnya tersedia di semua versi ElastiCache untuk Redis OSS. Tujuan dari `reserved-memory-percent` adalah untuk menyederhanakan manajemen memori cadangan di semua klaster Anda. Hal ini dilakukan dengan memungkinkan Anda memiliki satu grup parameter

tunggal untuk setiap keluarga grup parameter (seperti `redis2.8`) untuk mengelola memori cadangan dari klaster Anda, terlepas dari jenis simpulnya. Nilai default untuk `reserved-memory-percent` adalah 25 (25 persen).

Parameter `reserved-memory-percent` khusus untuk ElastiCache dan bukan bagian dari distribusi Redis OSS umum.

Jika klaster Anda menggunakan jenis simpul dari keluarga r6gd dan penggunaan memori Anda mencapai 75 persen, tingkatan data akan dipicu secara otomatis. Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.

Untuk menyimpan memori menggunakan reserved-memory-percent

Untuk menggunakan `reserved-memory-percent` untuk mengelola memori pada cluster ElastiCache untuk Redis OSS Anda, lakukan salah satu hal berikut:

- Jika Anda menjalankan Redis OSS 2.8.22 atau yang lebih baru, tetapkan grup parameter default ke cluster Anda. Default 25 persen seharusnya sudah memadai. Jika tidak, lakukan langkah yang dijelaskan berikut untuk mengubah nilai.
- Jika Anda menjalankan versi Redis OSS sebelum 2.8.22, Anda mungkin perlu memesan lebih banyak memori daripada `reserved-memory-percent` default 25 persen. Untuk melakukannya, gunakan prosedur berikut.

Untuk mengubah nilai persen reserved-memory-percent

1. Buat grup parameter kustom yang menentukan keluarga grup parameter yang cocok dengan versi mesin yang Anda jalankan—misalnya, menentukan keluarga grup parameter `redis2.8`. Grup parameter kustom diperlukan karena Anda tidak dapat mengubah grup parameter default. Untuk informasi selengkapnya, lihat Membuat grup ElastiCache parameter.

   ```
   aws elasticache create-cache-parameter-group \
      --cache-parameter-group-name redis28-50 \
      --description "Redis OSS 2.8.x 50% reserved" \
      --cache-parameter-group-family redis2.8
   ```

   Karena `reserved-memory-percent` mencadangkan memori sebagai persentase `maxmemory` simpul, Anda tidak memerlukan grup parameter kustom untuk setiap jenis simpul.
2. Ubah grup parameter kustom agar `reserved-memory-percent` menjadi 50 (50 persen). Untuk informasi selengkapnya, lihat Memodifikasi grup ElastiCache parameter.

```
aws elasticache modify-cache-parameter-group \
   --cache-parameter-group-name redis28-50 \
   --parameter-name-values "ParameterName=reserved-memory-percent,
 ParameterValue=50"
```

3. Gunakan grup parameter kustom ini untuk setiap cluster Redis OSS atau grup replikasi yang menjalankan versi Redis OSS yang lebih tua dari 2.8.22.

   Contoh CLI berikut memodifikasi cluster Redis OSS `my-redis-cluster` untuk menggunakan kelompok parameter kustom dimulai segera. `redis28-50` Untuk informasi selengkapnya, lihat Memodifikasi cluster ElastiCache .

   ```
   aws elasticache modify-cache-cluster \
      --cache-cluster-id my-redis-cluster \
      --cache-parameter-group-name redis28-50 \
      --apply-immediately
   ```

   Contoh CLI berikut memodifikasi kelompok replikasi Redis OSS `my-redis-repl-grp` untuk menggunakan kelompok parameter kustom dimulai segera. `redis28-50` Untuk informasi selengkapnya, lihat Mengubah grup replikasi.

   ```
   aws elasticache modify-replication-group \
      --replication-group-id my-redis-repl-grp \
      --cache-parameter-group-name redis28-50 \
      --apply-immediately
   ```

Menentukan Parameter Manajemen Memori Cadangan

Jika Anda adalah ElastiCache pelanggan saat ini pada 16 Maret 2017, parameter manajemen memori cadangan default Anda adalah `reserved-memory` dengan nol (0) byte memori cadangan. Jika Anda menjadi ElastiCache pelanggan setelah 16 Maret 2017, parameter manajemen memori cadangan default Anda adalah `reserved-memory-percent` dengan 25 persen dari memori node yang dicadangkan. Ini benar tidak masalah ketika Anda ElastiCache membuat cluster atau grup replikasi Redis OSS Anda. Namun, Anda dapat mengubah parameter manajemen memori cadangan menggunakan ElastiCache API AWS CLI atau API.

Parameter `reserved-memory` dan `reserved-memory-percent` keduanya bersifat saling eksklusif. Grup parameter selalu memiliki salah satu dari keduanya tetapi tidak pernah keduanya.

Anda dapat mengubah parameter yang digunakan grup parameter untuk manajemen memori cadangan dengan mengubah grup parameter. Grup parameter harus berupa grup parameter kustom, karena Anda tidak dapat mengubah grup parameter default. Untuk informasi selengkapnya, lihat Membuat grup ElastiCache parameter.

Untuk menentukan reserved-memory-percent

Untuk menggunakan `reserved-memory-percent` sebagai parameter manajemen memori cadangan, ubah grup parameter kustom menggunakan perintah `modify-cache-parameter-group`. Gunakan parameter `parameter-name-values` untuk menentukan `reserved-memory-percent` dan memberikan nilai untuknya.

Contoh CLI berikut mengubah grup parameter kustom `redis32-cluster-on` sehingga menggunakan `reserved-memory-percent` untuk mengelola memori cadangan. Nilai harus diberikan ke `ParameterValue` agar grup parameter dapat menggunakan parameter `ParameterName` untuk manajemen memori cadangan. Untuk informasi selengkapnya, lihat Memodifikasi grup ElastiCache parameter.

```
aws elasticache modify-cache-parameter-group \
   --cache-parameter-group-name redis32-cluster-on \
   --parameter-name-values "ParameterName=reserved-memory-percent, ParameterValue=25"
```

Untuk menentukan reserved-memory

Untuk menggunakan `reserved-memory` sebagai parameter manajemen memori cadangan, ubah grup parameter kustom menggunakan perintah `modify-cache-parameter-group`. Gunakan parameter `parameter-name-values` untuk menentukan `reserved-memory` dan memberikan nilai untuknya.

Contoh CLI berikut mengubah grup parameter kustom `redis32-m3xl` sehingga menggunakan `reserved-memory` untuk mengelola memori cadangan. Nilai harus diberikan ke `ParameterValue` agar grup parameter dapat menggunakan parameter `ParameterName` untuk manajemen memori cadangan. Karena versi mesin lebih baru dari 2.8.22, kami menetapkan nilainya menjadi `3565158400`, yaitu 25 persen `maxmemory` dari `cache.m3.xlarge`. Lihat informasi yang lebih lengkap di Memodifikasi grup ElastiCache parameter.

```
aws elasticache modify-cache-parameter-group \
   --cache-parameter-group-name redis32-m3xl \
   --parameter-name-values "ParameterName=reserved-memory, ParameterValue=3565158400"
```

Praktik terbaik saat bekerja dengan cluster yang dirancang sendiri oleh Valkey dan Redis OSS

Penggunaan multi-AZ, memiliki memori yang cukup, mengubah ukuran cluster, dan meminimalkan waktu henti adalah semua konsep yang berguna untuk diingat saat bekerja dengan cluster yang dirancang sendiri di Valkey atau Redis OSS. Sebaiknya tinjau dan ikuti praktik terbaik ini.

Topik

Meminimalkan waktu henti dengan Multi-AZ

Ada sejumlah contoh di mana ElastiCache Valkey atau Redis OSS mungkin perlu mengganti node primer; ini termasuk jenis pemeliharaan terencana tertentu dan kejadian yang tidak mungkin dari node primer atau kegagalan Availability Zone.

Penggantian ini mengakibatkan waktu henti pada klaster, tetapi jika Multi-AZ diaktifkan, waktu henti dapat dikurangi. Peran simpul primer akan secara otomatis melakukan failover ke salah satu replika baca. Tidak perlu membuat dan menyediakan simpul utama baru, karena ElastiCache akan menangani ini secara transparan. Failover dan promosi replika ini memastikan Anda dapat melanjutkan penulisan ke primer baru segera setelah promosi selesai.

LihatMeminimalkan downtime ElastiCache dengan menggunakan Multi-AZ dengan Valkey dan Redis OSS, untuk mempelajari lebih lanjut tentang Multi-AZ dan meminimalkan waktu henti.

Memastikan Anda memiliki cukup memori untuk membuat snapshot Valkey atau Redis OSS

Snapshot dan sinkronisasi di Valkey 7.2 dan yang lebih baru, dan Redis OSS versi 2.8.22 dan yang lebih baru

Valkey memiliki dukungan default untuk snapshot dan sinkronisasi. Redis OSS 2.8.22 memperkenalkan proses penyimpanan tanpa garpu yang memungkinkan Anda mengalokasikan lebih banyak memori untuk penggunaan aplikasi Anda tanpa menimbulkan peningkatan penggunaan

swap selama sinkronisasi dan penyimpanan. Untuk informasi selengkapnya, lihat Cara penerapan sinkronisasi dan pencadangan.

Redis OSS snapshot dan sinkronisasi sebelum versi 2.8.22

Ketika Anda bekerja dengan ElastiCache untuk Redis OSS, Redis OSS memanggil perintah penulisan latar belakang dalam sejumlah kasus:

- Saat membuat snapshot untuk cadangan.
- Saat menyinkronkan replika dengan primer dalam grup replikasi.
- Saat mengaktifkan fitur file append-only (AOF) untuk Redis OSS.
- Saat mempromosikan replika menjadi primer (yang menyebabkan sinkronisasi primer/replika).

Setiap kali Redis OSS mengeksekusi proses penulisan latar belakang, Anda harus memiliki memori yang cukup tersedia untuk mengakomodasi overhead proses. Jika memori yang tersedia tidak cukup, proses akan gagal. Karena itu, penting untuk memilih jenis instance node yang memiliki memori yang cukup saat membuat cluster Redis OSS Anda.

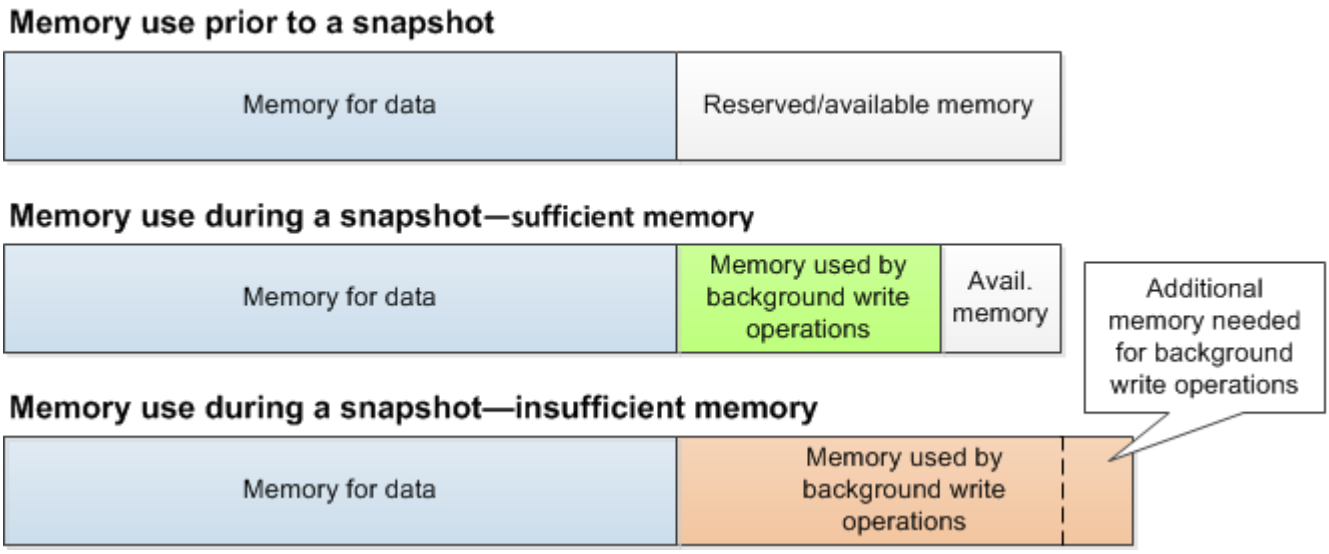
Proses Penulisan Latar Belakang dan Penggunaan Memori dengan Valkey dan Redis OSS

Setiap kali proses penulisan latar belakang disebut, Valkey dan Redis OSS memotong prosesnya (ingat, mesin ini berulir tunggal). Satu garpu menyimpan data Anda ke disk dalam file snapshot Redis OSS.rdb. Fork lainnya melayani semua operasi baca dan tulis. Untuk memastikan bahwa snapshot Anda adalah point-in-time snapshot, semua pembaruan dan penambahan data ditulis ke area memori yang tersedia terpisah dari area data.

Asalkan Anda memiliki cukup memori yang tersedia untuk mencatat semua operasi tulis saat data sedang dipersistensi ke disk, Anda tidak akan mengalami masalah kekurangan memori. Anda mungkin mengalami masalah kekurangan memori jika salah satu hal berikut berlaku:

- Aplikasi Anda melakukan banyak operasi tulis, sehingga membutuhkan sejumlah besar memori yang tersedia untuk menerima data yang baru atau diperbarui.
- Anda memiliki sedikit memori yang tersedia untuk menulis data yang baru atau diperbarui.
- Anda memiliki set data besar yang membutuhkan waktu lama untuk dipersistensi ke disk, sehingga memerlukan sejumlah besar operasi tulis.

Diagram berikut menggambarkan penggunaan memori saat menjalankan operasi tulis di latar belakang.



Untuk informasi tentang dampak melakukan pencadangan pada performa, lihat Dampak performa pencadangan klaster yang dirancang sendiri.

Untuk informasi lebih lanjut tentang bagaimana Valkey dan Redis OSS melakukan snapshot, lihat http://valkey.io.

Untuk informasi selengkapnya tentang wilayah dan Zona Ketersediaan, lihat Memilih wilayah dan zona ketersediaan untuk ElastiCache.

Menghindari kehabisan memori saat menjalankan operasi tulis di latar belakang

Setiap kali proses penulisan latar belakang seperti `BGSAVE` atau `BGREWRITEAOF` dipanggil, untuk menjaga proses agar tidak gagal, Anda harus memiliki lebih banyak memori yang tersedia daripada yang akan dikonsumsi oleh operasi penulisan selama proses. Skenario terburuk adalah bahwa selama operasi penulisan latar belakang setiap catatan diperbarui dan beberapa catatan baru ditambahkan ke cache. Karena itu, kami menyarankan Anda mengatur `reserved-memory-percent` ke 50 (50 persen) untuk versi Redis OSS sebelum 2.8.22, atau 25 (25 persen) untuk Valkey dan semua Redis OSS versi 2.8.22 dan yang lebih baru.

Nilai `maxmemory` menunjukkan memori yang tersedia untuk overhead data dan operasional. Karena Anda tidak dapat mengubah parameter `reserved-memory` dalam grup parameter default, Anda harus membuat grup parameter kustom untuk klaster. Nilai default untuk `reserved-memory` adalah 0, yang memungkinkan Redis OSS untuk mengkonsumsi semua maxmemory dengan data, berpotensi meninggalkan terlalu sedikit memori untuk kegunaan lain, seperti proses penulisan latar belakang. Untuk nilai `maxmemory` berdasarkan jenis instans simpul, lihat Parameter spesifik tipe node Redis OSS.

Anda juga dapat menggunakan `reserved-memory` parameter untuk mengurangi jumlah memori yang digunakan pada kotak.

Untuk informasi lebih lanjut tentang parameter khusus Valkey dan REDIS di ElastiCache, lihat. Parameter Valkey dan Redis OSS

Untuk informasi tentang cara membuat dan mengubah grup parameter DB, lihat Membuat grup ElastiCache parameter dan Memodifikasi grup ElastiCache parameter.

Perubahan ukuran klaster online

Resharding melibatkan proses menambahkan dan menghapus serpihan atau simpul ke klaster Anda dan mendistribusikan ulang ruang kunci. Oleh karena itu, ada beberapa hal yang memiliki dampak pada operasi resharding, seperti beban pada klaster, pemanfaatan memori, dan ukuran keseluruhan data. Untuk pengalaman terbaik, sebaiknya ikuti praktik terbaik klaster secara keseluruhan untuk distribusi pola beban kerja seragam. Selain itu, sebaiknya lakukan langkah-langkah berikut.

Sebelum memulai resharding, sebaiknya lakukan hal berikut:

- Uji aplikasi Anda – Uji perilaku aplikasi Anda selama resharding di lingkungan staging jika memungkinkan.
- Dapatkan notifikasi awal untuk masalah penskalaan – Resharding adalah operasi sarat komputasi. Karena itu, kami merekomendasikan untuk menjaga pemanfaatan CPU di bawah 80 persen pada instance multicore dan kurang dari 50 persen pada instance inti tunggal selama resharding. Pantau ElastiCache metrik Redis OSS dan mulai resharding sebelum aplikasi Anda mulai mengamati masalah penskalaan. Metrik yang berguna untuk dilacak adalah `CPUUtilization`, `NetworkBytesIn`, `NetworkBytesOut`, `CurrConnections`, `NewConnections`, `FreeableMemory`, `SwapUsage`, dan `BytesUsedForCacheItems`.
- Pastikan untuk menyediakan memori yang cukup sebelum penskalaan ke dalam – Jika Anda melakukan penskalaan ke dalam, pastikan memori pada serpihan yang akan dipertahankan tersedia setidaknya 1,5 kali dari memori yang akan digunakan pada serpihan yang akan dihapus.
- Mulai resharding di luar jam puncak – Praktik ini membantu mengurangi dampak latensi dan throughput pada klien selama operasi resharding. Hal ini juga membantu untuk menyelesaikan resharding lebih cepat karena lebih banyak sumber daya dapat digunakan untuk distribusi ulang slot.
- Tinjau perilaku waktu habis klien – Beberapa klien mungkin mengalami latensi yang lebih tinggi selama perubahan ukuran klaster online. Anda dapat mengonfigurasi pustaka klien dengan waktu habis yang lebih tinggi agar sistem memiliki waktu untuk terhubung bahkan dalam kondisi beban

yang lebih tinggi pada server. Dalam beberapa kasus, mungkin sebaiknya buka sejumlah besar koneksi ke server. Dalam kasus ini, pertimbangkan untuk menambahkan backoff eksponensial ke logika koneksi ulang. Melakukan hal ini dapat membantu mencegah lonjakan koneksi baru di server pada waktu yang sama.

- Muat Functions Anda di setiap shard — Saat menskalakan klaster Anda, ElastiCache akan secara otomatis mereplikasi Fungsi yang dimuat di salah satu node yang ada (dipilih secara acak) ke node baru. Jika klaster Anda memiliki Valkey 7.2 ke atas, atau Redis OSS 7.0 atau lebih tinggi, dan aplikasi Anda menggunakan Fungsi, kami sarankan memuat semua fungsi Anda ke semua pecahan sebelum penskalaan sehingga cluster Anda tidak berakhir dengan fungsi yang berbeda pada pecahan yang berbeda.

Setelah resharding, perhatikan hal berikut:

- Penskalaan ke dalam mungkin berhasil sebagian jika memori tidak cukup pada serpihan target. Jika hal tersebut terjadi, tinjau memori yang tersedia dan coba lagi operasi, jika perlu. Data pada serpihan target tidak akan dihapus.
- Slot dengan item besar tidak dimigrasikan. Secara khusus, slot dengan item yang lebih besar dari 256 MB pasca-serialisasi tidak dimigrasikan.
- Perintah `FLUSHALL` dan `FLUSHDB` tidak didukung dalam skrip Lua selama operasi resharding. Sebelum Redis OSS 6, `BRPOPLPUSH` perintah tidak didukung jika beroperasi pada slot yang sedang dimigrasikan.

Meminimalkan waktu henti selama pemeliharaan

Konfigurasi mode klaster memiliki ketersediaan terbaik selama operasi terkelola atau tidak terkelola. Sebaiknya gunakan klien yang mendukung mode klaster yang menghubungkan ke titik akhir penemuan klaster. Untuk mode klaster dinonaktifkan, sebaiknya gunakan titik akhir primer untuk semua operasi tulis.

Untuk aktivitas baca, aplikasi juga dapat menghubungkan ke simpul mana pun di klaster. Tidak seperti titik akhir primer, titik akhir simpul diresolusi ke titik akhir tertentu. Jika Anda membuat perubahan dalam klaster Anda, seperti menambahkan atau menghapus replika, Anda harus memperbarui titik akhir simpul di aplikasi Anda. Dengan demikian, untuk mode klaster dinonaktifkan, sebaiknya gunakan titik akhir pembaca untuk aktivitas baca.

Jika AutoFailover diaktifkan di cluster, simpul utama mungkin berubah. Oleh karena itu, aplikasi harus mengonfirmasi peran simpul dan memperbarui semua titik akhir baca. Tindakan ini akan membantu

memastikan bahwa Anda tidak menyebabkan beban besar pada primer. Dengan AutoFailover dinonaktifkan, peran node tidak berubah. Namun, downtime dalam operasi terkelola atau tidak terkelola lebih tinggi dibandingkan dengan cluster yang diaktifkan. AutoFailover

Hindari mengarahkan permintaan baca ke satu simpul replika baca karena ketidaktersediaannya dapat menyebabkan pemadaman operasi baca. Lakukan fallback untuk membaca dari primer, atau pastikan bahwa Anda memiliki setidaknya dua replika baca untuk menghindari gangguan operasi baca selama pemeliharaan.

Strategi cache untuk Memcached

Dalam topik berikut, Anda dapat menemukan strategi untuk mengisi dan memelihara cache Memcached Anda.

Strategi yang akan diterapkan untuk mengisi dan memelihara cache Anda akan bergantung pada data apa yang disimpan ke cache dan pola akses ke data tersebut. Misalnya, Anda kemungkinan tidak ingin menggunakan strategi yang sama untuk papan peringkat 10 teratas di situs game dan berita yang sedang tren. Selanjutnya, kita akan membahas strategi pemeliharaan cache umum beserta kelebihan dan kekurangannya.

Topik

Replika baca

Anda sering dapat secara signifikan meningkatkan kinerja untuk cache ElastiCache tanpa server dengan membuat replika dan membaca dari mereka alih-alih node cache utama. Untuk mengetahui informasi selengkapnya, lihat Praktik Terbaik untuk menggunakan Read Replicas.

Lazy loading

Seperti namanya, lazy loading adalah strategi pembuatan cache yang memuat data ke dalam cache hanya jika diperlukan. Cara kerjanya adalah seperti dijelaskan berikut.

Amazon ElastiCache adalah penyimpanan nilai kunci dalam memori yang berada di antara aplikasi Anda dan penyimpanan data (database) yang diaksesnya. Setiap kali aplikasi Anda meminta data, pertama kali membuat permintaan ke ElastiCache cache. Jika data terdapat di dalam cache dan data itu masih baru, maka ElastiCache mengembalikan data itu ke aplikasi Anda. Jika data tidak ada di dalam cache atau telah kedaluwarsa, maka aplikasi Anda akan meminta data dari penyimpanan data Anda. Penyimpanan data Anda kemudian mengembalikan data ke aplikasi Anda. Aplikasi Anda selanjutnya menulis data yang diterima dari penyimpanan ke cache. Dengan cara ini, data dapat lebih cepat diambil jika diminta lagi pada waktu berikutnya.

Cache hit terjadi ketika data ada dalam cache dan tidak habis masa berlakunya:

1. Aplikasi Anda meminta data dari cache.
2. Cache mengembalikan data ke aplikasi.

Cache miss terjadi ketika data tidak ada dalam cache atau data sudah tidak berlaku:

1. Aplikasi Anda meminta data dari cache.
2. Cache tidak memiliki data yang diminta, sehingga mengembalikan `null`.
3. Aplikasi Anda meminta dan menerima data dari basis data.
4. Aplikasi Anda memperbarui cache dengan data baru.

Kelebihan dan kekurangan dari lazy loading

Kelebihan dari lazy loading adalah sebagai berikut:

- Hanya data yang diminta yang disimpan ke cache.

  Karena sebagian besar data tidak pernah diminta, lazy loading menghindari mengisi cache dengan data yang tidak diminta.
- Kegagalan simpul tidak menjadi fatal untuk aplikasi Anda.

  Ketika simpul gagal dan digantikan oleh simpul baru yang kosong, aplikasi Anda terus berfungsi, meskipun dengan peningkatan latensi. Karena permintaan dibuat ke simpul baru, setiap cache miss menghasilkan kueri ke basis data. Pada saat yang sama, salinan data ditambahkan ke cache sehingga permintaan berikutnya diambil dari cache tersebut.

Kekurangan dari lazy loading adalah sebagai berikut:

- Terjadi kerugian akibat cache miss. Setiap cache miss menyebabkan tiga perjalanan:
  1. Permintaan awal untuk data dari cache
  2. Kueri basis data untuk data
  3. Menulis data ke cache

  Miss ini dapat menyebabkan penundaan yang terasa pada data yang masuk ke aplikasi.
- Data yang usang.

Jika data ditulis ke cache hanya jika ada cache miss, maka data dalam cache dapat menjadi usang. Hal ini terjadi karena tidak ada pembaruan pada cache ketika data berubah dalam basis data. Untuk mengatasi masalah ini, Anda dapat menggunakan strategi Write-through dan Menambahkan TTL.

Contoh kode semu lazy loading

Berikut adalah contoh kode semu logika lazy loading.

```
// ****************************************
// function that returns a customer's record.
// Attempts to retrieve the record from the cache.
// If it is retrieved, the record is returned to the application.
// If the record is not retrieved from the cache, it is
//    retrieved from the database,
//    added to the cache, and
//    returned to the application
// ****************************************
get_customer(customer_id)

    customer_record = cache.get(customer_id)
    if (customer_record == null)

        customer_record = db.query("SELECT * FROM Customers WHERE id = {0}",
 customer_id)
        cache.set(customer_id, customer_record)

    return customer_record
```

Untuk contoh ini, kode aplikasi yang mengambil data adalah sebagai berikut.

```
customer_record = get_customer(12345)
```

Write-through

Strategi write-through menambahkan data atau pembaruan data ke dalam cache setiap kali data ditulis ke basis data.

Kelebihan dan kekurangan dari write-through

Kelebihan dari write-through adalah sebagai berikut:

- Data dalam cache tidak pernah usang.

  Karena data dalam cache diperbarui setiap kali data itu ditulis ke basis data, data dalam cache selalu yang terbaru.
- Kerugian operasi tulis vs kerugian operasi baca.

  Setiap operasi tulis melibatkan dua proses:

  1. Operasi tulis ke cache
  2. Operasi tulis ke basis data

  Yang menambahkan latensi pada proses. Namun, pengguna akhir umumnya lebih toleran terhadap latensi saat memperbarui data daripada saat mengambil data. Ada perasaan yang melekat bahwa memperbarui membutuhkan upaya lebih banyak dan karena itu memakan waktu lebih lama.

Kekurangan dari write-through adalah sebagai berikut:

- Data yang hilang.

  Jika Anda membuat simpul baru, dengan alasan kegagalan simpul atau untuk penskalaan ke luar, maka akan ada data yang hilang. Data ini tetap hilang hingga ditambahkan atau diperbarui pada basis data. Anda dapat meminimalkan ini dengan menerapkan lazy loading dengan write-through.
- Churn Cache.

  Sebagian besar data tidak pernah dibaca, yang merupakan pemborosan sumber daya. Dengan menambahkan nilai time to live (TTL), Anda dapat meminimalkan ruang penyimpanan yang terbuang.

Contoh kode semu Write-through

Berikut adalah contoh kode semu dari logika write-through.

```
// ****************************************
// function that saves a customer's record.
// ****************************************
save_customer(customer_id, values)

    customer_record = db.query("UPDATE Customers WHERE id = {0}", customer_id, values)
    cache.set(customer_id, customer_record)
```

```
    return success
```

Untuk contoh ini, kode aplikasi yang mengambil data adalah sebagai berikut.

```
save_customer(12345,{"address":"123 Main"})
```

Menambahkan TTL

Lazy loading memungkinkan data menjadi usang tetapi tidak gagal dengan simpul kosong. Write-through memastikan data selalu segar, tetapi dapat gagal dengan simpul kosong dan dapat mengisi cache dengan data berlebihan yang tidak berguna. Dengan menambahkan nilai time to live (TTL) untuk setiap operasi tulis, Anda dapat memanfaatkan kelebihan dari setiap strategi. Pada saat yang sama, Anda dapat secara signifikan menghindari cache menjadi penuh karena data tambahan.

Time to live (TTL) adalah nilai integer yang menentukan jumlah detik sampai kunci berakhir. Valkey atau Redis OSS dapat menentukan detik atau milidetik untuk nilai ini. Memcached menentukan nilai ini dalam detik. Ketika sebuah aplikasi mencoba membaca kunci yang kedaluwarsa, maka perlakuannya adalah seolah-olah kunci itu tidak ditemukan. Terjadi kueri ke basis data untuk meminta kunci dan cache diperbarui. Pendekatan ini tidak menjamin sebuah nilai tidak menjadi usang. Namun, hal ini menjaga data agar tidak terlalu usang dan mensyaratkan nilai di dalam cache agar terkadang disegarkan kembali dari basis data.

Untuk informasi lebih lanjut, lihat perintah Valkey dan Redis OSS atau perintah Memcached. `set`

Contoh kode semu TTL

Berikut adalah contoh kode semu dari logika write-through dengan TTL.

```
// ****************************************
// function that saves a customer's record.
// The TTL value of 300 means that the record expires
//    300 seconds (5 minutes) after the set command
//    and future reads will have to query the database.
// ****************************************
save_customer(customer_id, values)

    customer_record = db.query("UPDATE Customers WHERE id = {0}", customer_id, values)
    cache.set(customer_id, customer_record, 300)

    return success
```

Berikut adalah contoh kode semu logika lazy loading dengan TTL.

```
// ****************************************
// function that returns a customer's record.
// Attempts to retrieve the record from the cache.
// If it is retrieved, the record is returned to the application.
// If the record is not retrieved from the cache, it is
//    retrieved from the database,
//    added to the cache, and
//    returned to the application.
// The TTL value of 300 means that the record expires
//    300 seconds (5 minutes) after the set command
//    and subsequent reads will have to query the database.
// ****************************************
get_customer(customer_id)

    customer_record = cache.get(customer_id)

    if (customer_record != null)
        if (customer_record.TTL < 300)
            return customer_record        // return the record and exit function

    // do this only if the record did not exist in the cache OR
    //    the TTL was >= 300, i.e., the record in the cache had expired.
    customer_record = db.query("SELECT * FROM Customers WHERE id = {0}", customer_id)
    cache.set(customer_id, customer_record, 300)  // update the cache
    return customer_record                // return the newly retrieved record and exit
 function
```

Untuk contoh ini, kode aplikasi yang mengambil data adalah sebagai berikut.

```
save_customer(12345,{"address":"123 Main"})
```

```
customer_record = get_customer(12345)
```

Topik terkait

- Penyimpanan Data Dalam Memori
- Memilih mesin dan versi
- Penskalaan ElastiCache

Mengelola cluster yang dirancang sendiri di ElastiCache

ElastiCache menawarkan dua opsi penerapan, caching tanpa server dan cluster yang dirancang sendiri. Masing-masing memiliki kemampuan dan persyaratannya sendiri.

Bagian ini berisi topik untuk membantu Anda mengelola cluster yang dirancang sendiri.

> **Note**
>
> Topik-topik ini tidak berlaku untuk Tanpa ElastiCache Server.

Topik

Auto Scaling Valkey dan Redis OSS cluster

Prasyarat

ElastiCache Auto Scaling terbatas pada hal-hal berikut:

- Valkey atau Redis OSS (mode cluster diaktifkan) cluster yang menjalankan Valkey 7.2 dan seterusnya, atau menjalankan Redis OSS 6.0 dan seterusnya
- Tiering data (mode cluster diaktifkan) cluster yang menjalankan Valkey 7.2 dan seterusnya, atau menjalankan Redis OSS 7.0.7 dan seterusnya
- Ukuran instans - Besar, XLarge, 2 XLarge
- Kelompok tipe instans - R7g, R6g, R6gd, R5, M7g, M6g, M5, C7gn
- Auto Scaling in ElastiCache tidak didukung untuk kluster yang berjalan di Datastores Global, Outposts, atau Local Zones.

Mengelola Kapasitas Secara Otomatis dengan ElastiCache Auto Scaling dengan Valkey atau Redis OSS

ElastiCache auto scaling dengan Valkey atau Redis OSS adalah kemampuan untuk menambah atau mengurangi pecahan atau replika yang diinginkan dalam layanan Anda secara otomatis. ElastiCache ElastiCache memanfaatkan layanan Application Auto Scaling untuk menyediakan fungsionalitas ini. Untuk informasi selengkapnya, lihat Application Auto Scaling. Untuk menggunakan penskalaan otomatis, Anda menentukan dan menerapkan kebijakan penskalaan yang menggunakan CloudWatch metrik dan nilai target yang Anda tetapkan. ElastiCache auto scaling menggunakan kebijakan untuk menambah atau mengurangi jumlah instance sebagai respons terhadap beban kerja aktual.

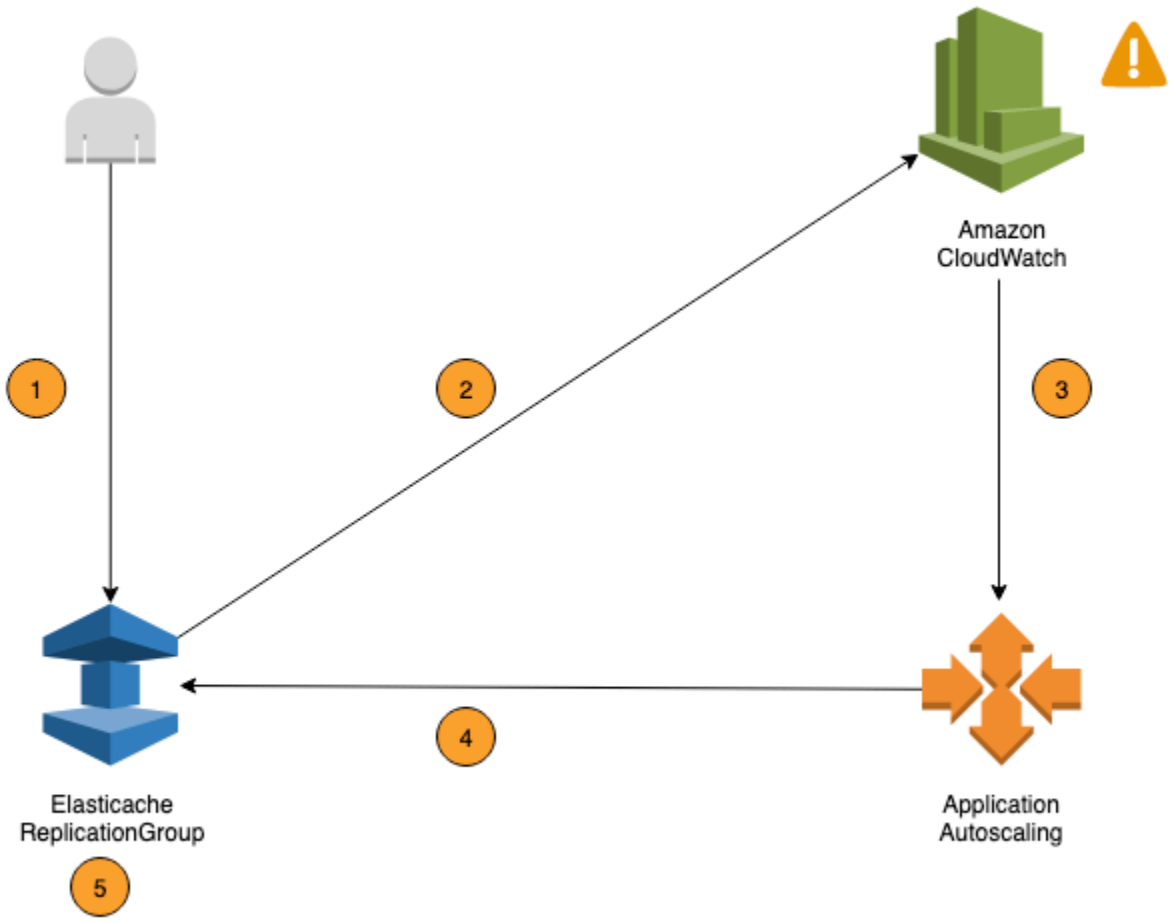
Anda dapat menggunakan AWS Management Console untuk menerapkan kebijakan penskalaan berdasarkan metrik yang telah ditentukan sebelumnya. Sebuah `predefined metric` didefinisikan dalam penghitungan sehingga Anda dapat menentukannya dengan nama dalam kode atau menggunakannya dalam AWS Management Console. Metrik kustom tidak tersedia untuk pilihan menggunakan AWS Management Console. Atau, Anda dapat menggunakan Application Auto Scaling API AWS CLI atau Application Auto Scaling untuk menerapkan kebijakan penskalaan berdasarkan metrik yang telah ditentukan atau kustom.

ElastiCache untuk Valkey dan Redis OSS mendukung penskalaan untuk dimensi berikut:

- Serpihan – Secara otomatis menambahkan/menghapus serpihan di klaster yang mirip dengan resharding online secara manual. Dalam hal ini, penskalaan ElastiCache otomatis memicu penskalaan atas nama Anda.
- Replika - add/remove replicas in the cluster similar to manual Increase/Decrease replica operations. ElastiCache auto scaling for Valkey and Redis OSS adds/removes Replika secara otomatis secara seragam di semua pecahan di cluster.

ElastiCache untuk Valkey dan Redis OSS mendukung jenis kebijakan penskalaan otomatis berikut:

- Kebijakan penskalaan pelacakan target— Menambah atau mengurangi jumlah shards/replicas layanan Anda berjalan berdasarkan nilai target untuk metrik tertentu. Hal ini serupa dengan cara termostat mempertahankan suhu rumah Anda. Anda memilih suhu dan termostat melakukan sisanya.
- Penskalaan terjadwal untuk aplikasi Anda. — ElastiCache untuk Valkey dan Redis OSS auto scaling dapat menambah atau mengurangi shards/replicas jumlah layanan Anda berjalan berdasarkan tanggal dan waktu.



Langkah-langkah berikut merangkum proses ElastiCache penskalaan otomatis Valkey dan Redis OSS seperti yang ditunjukkan pada diagram sebelumnya:

1. Anda membuat kebijakan penskalaan ElastiCache otomatis untuk Grup Replikasi Anda.
2. ElastiCache auto scaling menciptakan sepasang CloudWatch alarm atas nama Anda. Setiap pasangan merepresentasikan batas atas dan bawah Anda untuk metrik. CloudWatch Alarm ini dipicu ketika penggunaan aktual cluster menyimpang dari penggunaan target Anda untuk jangka waktu yang berkelanjutan. Anda dapat melihat alarm di konsol.
3. Jika nilai metrik yang dikonfigurasi melebihi pemanfaatan target Anda (atau berada di bawah target) untuk jangka waktu tertentu, CloudWatch memicu alarm yang memanggil penskalaan otomatis untuk mengevaluasi kebijakan penskalaan Anda.
4. ElastiCache auto scaling mengeluarkan permintaan Modify untuk menyesuaikan kapasitas cluster Anda.

5. ElastiCache memproses permintaan Modify, secara dinamis meningkatkan (atau mengurangi) Shards/Replicas kapasitas cluster sehingga mendekati pemanfaatan target Anda.

Untuk memahami cara kerja ElastiCache Auto Scaling, misalkan Anda memiliki cluster bernama. `UsersCluster` Dengan memantau CloudWatch metrik`UsersCluster`, Anda menentukan pecahan Max yang dibutuhkan cluster saat lalu lintas berada di puncaknya dan Min Shards saat lalu lintas berada pada titik terendah. Anda juga menentukan nilai target untuk pemanfaatan CPU untuk `UsersCluster` cluster. ElastiCache auto scaling menggunakan algoritma pelacakan targetnya untuk memastikan bahwa pecahan yang `UsersCluster` disediakan disesuaikan sesuai kebutuhan sehingga pemanfaatan tetap pada atau mendekati nilai target.

> **Note**
>
> Penskalaan mungkin membutuhkan waktu yang nyata dan akan membutuhkan sumber daya klaster tambahan agar pecahan dapat diseimbangkan kembali. ElastiCache Auto Scaling memodifikasi pengaturan sumber daya hanya jika beban kerja sebenarnya tetap meningkat (atau tertekan) selama beberapa menit. Algoritma pelacakan target penskalaan otomatis berupaya menjaga pemanfaatan target pada atau mendekati nilai yang Anda pilih dalam jangka panjang.

Kebijakan Auto Scaling

Kebijakan penskalaan memiliki komponen berikut:

- Metrik target — CloudWatch Metrik yang digunakan ElastiCache untuk Valkey dan Redis OSS Auto Scaling untuk menentukan kapan dan berapa banyak skala.
- Kapasitas minimum dan maksimum – Jumlah serpihan atau replika minimum dan maksimum yang digunakan untuk penskalaan.

> **Important**
>
> Saat membuat kebijakan penskalaan Otomatis, jika kapasitas saat ini lebih tinggi dari kapasitas maksimal yang dikonfigurasi, kami menskaleIn MaxCapacity selama pembuatan kebijakan. Demikian pula jika kapasitas saat ini lebih rendah dari kapasitas min yang dikonfigurasi, kami ScaleOut ke. MinCapacity

- Periode pendinginan – Jumlah waktu, dalam detik, setelah aktivitas penskalaan ke dalam atau penskalaan ke luar selesai sebelum aktivitas penskalaan ke luar lainnya dapat dimulai.
- Peran terkait layanan — Peran AWS Identitas dan Manajemen Akses (IAM) and Access Management (IAM) yang ditautkan ke layanan tertentu. AWS Peran terkait layanan mencakup semua izin yang diperlukan layanan untuk memanggil AWS layanan lain atas nama Anda. ElastiCache Auto Scaling secara otomatis menghasilkan peran ini,`AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG`, untuk Anda.
- Aktifkan atau nonaktifkan aktivitas penskalaan ke dalam - Kemampuan untuk mengaktifkan atau menonaktifkan aktivitas penskalaan ke dalam untuk suatu kebijakan.

Topik

Metrik target untuk Auto Scaling

Dalam jenis kebijakan ini, metrik yang telah ditentukan atau kustom dan nilai target untuk metrik ditentukan dalam konfigurasi kebijakan penskalaan pelacakan target. ElastiCache untuk Valkey dan Redis OSS Auto Scaling membuat dan mengelola CloudWatch alarm yang memicu kebijakan penskalaan dan menghitung penyesuaian penskalaan berdasarkan metrik dan nilai target. Kebijakan penskalaan menambahkan atau menghapus shards/replicas sesuai kebutuhan untuk menjaga metrik pada, atau mendekati, nilai target yang ditentukan. Selain menjaga metrik tetap dekat dengan nilai target, kebijakan penskalaan pelacakan target juga disesuaikan menurut fluktuasi metrik karena pola beban kerja yang berubah. Kebijakan semacam itu juga meminimalkan fluktuasi cepat dalam jumlah yang tersedia shards/replicas untuk klaster Anda.

Misalnya, pertimbangkan kebijakan penskalaan yang menggunakan metrik `ElastiCachePrimaryEngineCPUUtilization` rata-rata standar. Kebijakan tersebut dapat menjaga pemanfaatan CPU pada, atau mendekati, persentase pemanfaatan tertentu, seperti 70 persen.

> **Note**
>
> Untuk setiap klaster, Anda hanya dapat membuat satu kebijakan Auto Scaling untuk setiap metrik target.

Kapasitas minimum dan maksimum

Serpihan

Anda dapat menentukan jumlah pecahan maksimum yang dapat diskalakan oleh ElastiCache untuk penskalaan otomatis Valkey dan Redis OSS. Nilai ini harus kurang dari atau sama dengan 250 dengan minimum 1. Anda juga dapat menentukan jumlah minimum pecahan yang akan dikelola dengan penskalaan otomatis. Nilai ini harus minimal 1, dan sama dengan atau kurang dari nilai yang ditentukan untuk serpihan maksimum yaitu 250.

Replika

Anda dapat menentukan jumlah maksimum replika yang akan dikelola oleh ElastiCache untuk penskalaan otomatis Valkey dan Redis OSS. Nilai ini harus kurang dari atau sama dengan 5. Anda juga dapat menentukan jumlah minimum replika yang akan dikelola dengan penskalaan otomatis. Nilai ini harus minimal 1, dan sama dengan atau kurang dari nilai yang ditentukan untuk replika maksimum yaitu 5.

Untuk menentukan jumlah minimum dan maksimum shards/replicas yang Anda butuhkan untuk lalu lintas tipikal, uji konfigurasi Auto Scaling Anda dengan tingkat lalu lintas yang diharapkan ke model Anda.

> **Note**
>
> ElastiCache Kebijakan penskalaan otomatis meningkatkan kapasitas klaster hingga mencapai ukuran maksimum yang Anda tentukan atau hingga batas layanan berlaku. Untuk meminta penambahan batas, lihat Batas Layanan AWS dan pilih jenis batas Simpul per klaster per jenis instans.

> **Important**
>
> Penskalaan ke dalam terjadi ketika tidak ada lalu lintas. Jika lalu lintas varian menjadi nol, ElastiCache secara otomatis menskalakan ke jumlah minimum instance yang ditentukan.

Periode pendinginan

Anda dapat menyesuaikan daya respons kebijakan penskalaan pelacakan target dengan menambahkan periode pendinginan yang memengaruhi penskalaan klaster Anda. Periode pendinginan memblokir permintaan penskalaan ke dalam atau ke luar berikutnya hingga periode ini berakhir. Ini memperlambat penghapusan di cluster Valkey dan Redis OSS Anda ElastiCache untuk permintaan scale-in, dan pembuatan untuk permintaan scale-out. shards/replicas shards/replicas Anda dapat menentukan periode pendinginan berikut:

- Aktivitas scale-in mengurangi jumlah di klaster shards/replicas Anda. Periode pendinginan penskalaan ke dalam menentukan jumlah waktu, dalam detik, setelah aktivitas penskalaan ke dalam selesai sebelum aktivitas penskalaan ke dalam lainnya dapat dimulai.
- Aktivitas scale-out meningkatkan jumlah di klaster shards/replicas Anda. Periode pendinginan penskalaan ke luar menentukan jumlah waktu, dalam detik, setelah aktivitas penskalaan ke luar selesai sebelum aktivitas penskalaan ke luar lainnya dapat dimulai.

Ketika periode pendinginan penskalaan ke dalam atau penskalaan ke luar tidak ditentukan, nilai default untuk penskalaan ke luar adalah 600 detik dan untuk penskalaan ke dalam adalah 900 detik.

Mengaktifkan atau menonaktifkan aktivitas penskalaan ke dalam

Anda dapat mengaktifkan atau menonaktifkan aktivitas penskalaan ke dalam untuk sebuah kebijakan. Mengaktifkan aktivitas scale-in memungkinkan kebijakan penskalaan dihapus. shards/replicas. When scale-in activities are enabled, the scale-in cooldown period in the scaling policy applies to scale-in activities. Disabling scale-in activities prevents the scaling policy from deleting shards/replicas

> **Note**
>
> Aktivitas scale-out selalu diaktifkan sehingga kebijakan penskalaan dapat membuat ElastiCache pecahan atau replika sesuai kebutuhan.

Izin IAM Diperlukan untuk Auto Scaling

ElastiCache untuk Valkey dan Redis OSS Auto Scaling dimungkinkan oleh kombinasi,, dan Application CloudWatch Auto ElastiCache Scaling. APIs Cluster dibuat dan diperbarui dengan ElastiCache, alarm dibuat dengan CloudWatch, dan kebijakan penskalaan dibuat dengan Application Auto Scaling. Selain izin IAM standar untuk membuat dan memperbarui cluster, pengguna IAM yang mengakses pengaturan Auto ElastiCache Scaling harus memiliki izin yang sesuai untuk layanan yang mendukung penskalaan dinamis. Dalam kebijakan terbaru ini kami telah menambahkan dukungan untuk penskalaan vertikal Memcached, dengan tindakan. `elasticache:ModifyCacheCluster` Pengguna IAM harus memiliki izin untuk menggunakan tindakan yang ditunjukkan dalam contoh kebijakan berikut:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:*",
                "elasticache:DescribeReplicationGroups",
                "elasticache:ModifyReplicationGroupShardConfiguration",
                "elasticache:IncreaseReplicaCount",
                "elasticache:DecreaseReplicaCount",
                "elasticache:DescribeCacheClusters",
                "elasticache:DescribeCacheParameters",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarmHistory",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:DescribeAlarmsForMetric",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DisableAlarmActions",
                "cloudwatch:EnableAlarmActions",
                "iam:CreateServiceLinkedRole",
                "sns:CreateTopic",
                "sns:Subscribe",
                "sns:Get*",
                "sns:List*"
```

```
            ],
            "Resource": "arn:aws:iam::123456789012:role/autoscaling-roles-for-
cluster"
        }
    ]
}
```

Peran terkait layanan

Layanan penskalaan otomatis ElastiCache untuk Valkey dan Redis OSS juga memerlukan izin untuk menggambarkan cluster CloudWatch dan alarm Anda, dan izin untuk mengubah ElastiCache kapasitas target Anda atas nama Anda. Jika Anda mengaktifkan Auto Scaling untuk klaster Anda, itu akan membuat peran terkait layanan bernama. `AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG` Peran terkait layanan ini memberikan izin penskalaan ElastiCache otomatis untuk menjelaskan alarm kebijakan Anda, untuk memantau kapasitas armada saat ini, dan untuk memodifikasi kapasitas armada. Peran terkait layanan adalah peran default untuk penskalaan ElastiCache otomatis. Untuk informasi selengkapnya, lihat Peran terkait layanan ElastiCache untuk penskalaan otomatis Redis OSS di Panduan Pengguna Application Auto Scaling.

Praktik Terbaik Auto Scaling

Sebelum mendaftar Auto Scaling, kami merekomendasikan hal berikut:

1. Gunakan hanya satu metrik pelacakan – Identifikasi apakah klaster Anda memiliki beban kerja yang sarat CPU atau sarat data dan gunakan metrik standar yang sesuai untuk menentukan Kebijakan Penskalaan.
   - CPU mesin: `ElastiCachePrimaryEngineCPUUtilization` (dimensi serpihan) atau `ElastiCacheReplicaEngineCPUUtilization` (dimensi replika)
   - Penggunaan basis data: `ElastiCacheDatabaseCapacityUsageCountedForEvictPercentage` Kebijakan penskalaan ini berfungsi paling baik dengan maxmemory-policy yang ditetapkan ke noeviction di klaster.

   Kami menyarankan Anda menghindari beberapa kebijakan per dimensi pada klaster. ElastiCache untuk Valkey dan Redis OSS Penskalaan otomatis akan menskalakan target yang dapat diskalakan jika ada kebijakan pelacakan target yang siap untuk diskalakan, tetapi akan menskalakan hanya jika semua kebijakan pelacakan target (dengan bagian penskalaan diaktifkan)

siap untuk diskalakan. Jika beberapa kebijakan menginstruksikan target yang dapat diskalakan untuk menskalakan ke luar atau ke dalam pada saat yang sama, target akan diskalakan berdasarkan kebijakan yang memberikan kapasitas terbesar untuk penskalaan ke dalam dan penskalaan ke luar.

2. Metrik yang Disesuaikan untuk Pelacakan Target – Berhati-hatilah saat menggunakan metrik yang disesuaikan untuk Pelacakan Target karena Penskalaan otomatis paling cocok untuk penskalaan ke luar/ke dalam secara berbanding lurus dengan perubahan metrik yang dipilih untuk kebijakan. Jika metrik tersebut tidak berubah secara proporsional dengan tindakan penskalaan yang digunakan untuk pembuatan kebijakan, hal tersebut dapat menyebabkan tindakan penskalaan ke luar atau penskalaan ke dalam berkelanjutan yang dapat memengaruhi ketersediaan atau biaya.

   Untuk klaster tingkatan data (jenis instans keluarga r6gd), hindari penggunaan metrik berbasis memori untuk penskalaan.
3. Penskalaan Terjadwal - Jika Anda mengidentifikasi bahwa beban kerja Anda bersifat deterministik (mencapai high/low pada waktu tertentu), sebaiknya gunakan Penskalaan Terjadwal dan konfigurasikan kapasitas target Anda sesuai dengan kebutuhan. Pelacakan Target paling cocok untuk beban kerja non-deterministik dan agar klaster beroperasi pada metrik target yang diperlukan dengan penskalaan ke luar saat Anda membutuhkan lebih banyak sumber daya dan penskalaan ke dalam saat Anda membutuhkan lebih sedikit sumber daya.
4. Nonaktifkan Scale-In — Penskalaan otomatis pada Pelacakan Target paling cocok untuk cluster dengan beban kerja bertahap karena spikes/dip dalam metrik dapat memicu increase/decrease penskalaan berurut/dalam osilasi. Untuk menghindari pergantian tersebut, Anda dapat memulai dengan menonaktifkan penskalaan ke luar, dan melakukan penskalaan ke dalam secara manual sesuai kebutuhan Anda kapan saja.
5. Uji aplikasi Anda — Kami menyarankan Anda menguji aplikasi Anda dengan perkiraan Min/Max beban kerja Anda untuk menentukan Min, Max absolut yang shards/replicas diperlukan untuk klaster sambil membuat kebijakan Penskalaan untuk menghindari masalah ketersediaan. Penskalaan otomatis dapat menskalakan ke luar hingga Maks dan menskalakan ke dalam hingga ambang batas Min yang dikonfigurasi untuk target.
6. Mendefinisikan Nilai Target — Anda dapat menganalisis CloudWatch metrik yang sesuai untuk pemanfaatan klaster selama periode empat minggu untuk menentukan ambang nilai target. Jika Anda masih tidak yakin nilai apa yang harus dipilih, sebaiknya mulai dengan nilai metrik standar minimum yang didukung.
7. AutoScaling pada Pelacakan Target paling cocok untuk cluster dengan distribusi beban kerja yang seragam di seluruh shards/replicas dimensi. Memiliki distribusi yang tidak seragam dapat menyebabkan:

- Penskalaan saat tidak diperlukan karena beban kerja spike/dip pada beberapa pecahan/replika panas.
- Tidak ada penskalaan saat diperlukan karena nilai rata-rata keseluruhan mendekati target meskipun memiliki serpihan/replika "hot".

> **Note**
>
> Saat menskalakan cluster Anda, secara otomatis ElastiCache akan mereplikasi Fungsi yang dimuat di salah satu node yang ada (dipilih secara acak) ke node baru. Jika klaster Anda memiliki Valkey atau Redis OSS 7.0 atau lebih tinggi dan aplikasi Anda menggunakan Fungsi, kami sarankan memuat semua fungsi Anda ke semua pecahan sebelum penskalaan sehingga cluster Anda tidak berakhir dengan fungsi yang berbeda pada pecahan yang berbeda.

Setelah mendaftar AutoScaling, perhatikan hal berikut:

- Ada batasan pada Konfigurasi yang Didukung Auto Scaling. Jadi, kami menyarankan Anda untuk tidak mengubah konfigurasi grup replikasi yang terdaftar untuk Auto Scaling. Berikut ini adalah beberapa contohnya:
  - Mengubah jenis Instans secara manual ke jenis yang tidak didukung.
  - Mengaitkan grup replikasi ke penyimpanan data Global.
  - Mengubah parameter `ReservedMemoryPercent`.
  - increasing/decreasing shards/replicas beyond the Min/MaxKapasitas secara manual dikonfigurasi selama pembuatan kebijakan.

Menggunakan Auto Scaling dengan serpihan

Dengan ElastiCache's AutoScaling Anda dapat menggunakan kebijakan pelacakan dan terjadwal dengan mesin Valkey atau Redis OSS Anda.

Berikut ini memberikan rincian tentang pelacakan target dan kebijakan terjadwal dan bagaimana menerapkannya menggunakan AWS Management Console AWS CLI dan APIs.

Topik

- Kebijakan penskalaan pelacakan target

- Menambahkan kebijakan penskalaan
- Mendaftarkan Target yang Dapat Diskalakan
- Menetapkan kebijakan penskalaan
- Menonaktifkan aktivitas penskalaan ke dalam
- Menerapkan kebijakan penskalaan
- Mengedit kebijakan penskalaan
- Menghapus kebijakan penskalaan
- Gunakan AWS CloudFormation untuk kebijakan Auto Scaling
- Penskalaan terjadwal

Kebijakan penskalaan pelacakan target

Dengan kebijakan penskalaan pelacakan target, Anda memilih metrik dan menetapkan nilai target. ElastiCache untuk Valkey dan Redis OSS Auto Scaling membuat dan mengelola CloudWatch alarm yang memicu kebijakan penskalaan dan menghitung penyesuaian penskalaan berdasarkan metrik dan nilai target. Kebijakan penskalaan menambahkan atau menghapus serpihan yang diperlukan untuk menjaga metrik pada, atau mendekati, nilai target yang ditentukan. Selain menjaga metrik agar mendekati nilai target, kebijakan penskalaan pelacakan target juga menyesuaikan dengan fluktuasi metrik karena fluktuasi pola beban dan meminimalkan fluktuasi cepat dalam kapasitas armada.

Misalnya, pertimbangkan kebijakan penskalaan yang menggunakan metrik `ElastiCachePrimaryEngineCPUUtilization` rata-rata standar dengan nilai target yang dikonfigurasi. Kebijakan tersebut dapat menjaga pemanfaatan CPU pada, atau mendekati, nilai target yang ditentukan.

Metrik standar

Metrik yang telah ditentukan adalah struktur yang mengacu pada nama, dimensi, dan statistik tertentu (`average`) dari CloudWatch metrik tertentu. Kebijakan Auto Scaling menentukan salah satu metrik standar di bawah ini untuk klaster Anda:

Nama Metrik Standar	CloudWatch Nama Metrik	CloudWatch Dimensi Metrik	Jenis Instans yang Tidak Memenuhi Syarat
ElastiCachePrimaryEngineCPUUtilization	EngineCPUUtilization	ReplicationGroupId, Peran = Primer	Tidak ada
ElastiCacheDatabaseCapacityUsageCountedForEvictPercentage	DatabaseCapacityUsageCountedForEvictPercentage	Metrik Grup Replikasi Valkey atau Redis OSS	Tidak ada
ElastiCacheDatabaseMemoryUsageCountedForEvictPercentage	DatabaseMemoryUsageCountedForEvictPercentage	Metrik Grup Replikasi Valkey atau Redis OSS	R6gd

Jenis instans bertingkatan data tidak dapat menggunakan `ElastiCacheDatabaseMemoryUsageCountedForEvictPercentage`, karena jenis instans ini menyimpan data di memori dan SSD. Kasus penggunaan yang diharapkan untuk instans bertingkatan data adalah memiliki 100 persen penggunaan memori dan mengisi SSD sesuai kebutuhan.

Kriteria Auto Scaling untuk serpihan

Ketika layanan mendeteksi bahwa metrik standar Anda sama dengan atau lebih besar dari pengaturan Target, itu akan meningkatkan kapasitas pecahan Anda secara otomatis. ElastiCache untuk Valkey dan Redis OSS menskalakan pecahan cluster Anda dengan hitungan yang sama

dengan yang lebih besar dari dua angka: Persen variasi dari Target dan 20 persen pecahan saat ini. Untuk scale-in, ElastiCache tidak akan auto scale-in kecuali nilai metrik keseluruhan di bawah 75 persen dari Target yang Anda tentukan.

Untuk contoh penskalaan ke luar, jika Anda memiliki 50 serpihan dan

- jika Target Anda melanggar 30 persen, ElastiCache skala sebesar 30 persen, yang menghasilkan 65 pecahan per cluster.
- jika Target Anda melanggar 10 persen, ElastiCache skala secara default Minimum 20 persen, yang menghasilkan 60 pecahan per cluster.

Untuk contoh scale-in, jika Anda telah memilih nilai Target 60 persen, ElastiCache tidak akan auto scale-in sampai metrik kurang dari atau sama dengan 45 persen (25 persen di bawah Target 60 persen).

Pertimbangan untuk Auto Scaling

Perhatikan sejumlah pertimbangan berikut:

- Kebijakan penskalaan pelacakan target mengasumsikan bahwa penskalaan ke luar harus dilakukan saat metrik yang ditentukan berada di atas nilai target. Anda tidak dapat menggunakan kebijakan penskalaan pelacakan target untuk memperkecil skala ketika metrik yang ditentukan berada di bawah nilai target. ElastiCache untuk Valkey dan Redis OSS skala pecahan dengan minimal 20 persen deviasi target pecahan yang ada di cluster.
- Kebijakan penskalaan pelacakan target tidak melakukan penskalaan saat metrik yang ditentukan tidak memiliki data yang mencukupi. Kebijakan penskalaan pelacakan target tidak melakukan penskalaan ke dalam karena data yang tidak mencukupi tidak ditafsirkan sebagai pemanfaatan yang rendah.
- Anda mungkin melihat kesenjangan antara nilai target dan titik data metrik aktual. Ini karena ElastiCache Auto Scaling selalu bertindak konservatif dengan membulatkan ke atas atau ke bawah ketika menentukan berapa banyak kapasitas untuk menambah atau menghapus. Hal ini mencegah layanan menambahkan kapasitas yang tidak mencukupi atau menghapus kapasitas terlalu banyak.
- Untuk memastikan ketersediaan aplikasi, layanan menskalakan ke luar secara proporsional berdasarkan metrik secepat mungkin, tetapi menskalakan ke dalam secara lebih konservatif.
- Anda dapat memiliki beberapa kebijakan penskalaan pelacakan target ElastiCache untuk klaster Valkey dan Redis OSS, asalkan masing-masing menggunakan metrik yang berbeda. Tujuan ElastiCache Auto Scaling adalah untuk selalu memprioritaskan ketersediaan, sehingga perilakunya

berbeda tergantung pada apakah kebijakan pelacakan target siap untuk skala atau skala. Fitur ini akan menskalakan ke luar layanan jika salah satu kebijakan pelacakan target siap untuk diskalakan ke luar. Namun, penskalaan ke dalam akan dilakukan hanya jika semua kebijakan pelacakan target (dengan porsi penskalaan ke dalam diaktifkan) siap untuk diskalakan ke dalam.

- Jangan mengedit atau menghapus CloudWatch alarm yang dikelola ElastiCache Auto Scaling untuk kebijakan penskalaan pelacakan target. ElastiCache Auto Scaling menghapus alarm secara otomatis saat Anda menghapus kebijakan penskalaan.
- ElastiCache Auto Scaling tidak mencegah Anda memodifikasi pecahan cluster secara manual. Penyesuaian manual ini tidak memengaruhi CloudWatch alarm yang ada yang melekat pada kebijakan penskalaan, tetapi dapat memengaruhi metrik yang dapat memicu alarm ini. CloudWatch
- CloudWatch Alarm yang dikelola oleh Auto Scaling ini ditentukan melalui metrik AVG di semua pecahan di cluster. Jadi, serpihan dengan lalu lintas tertinggi dapat menghasilkan salah satu skenario berikut:
  - penskalaan saat tidak diperlukan karena memuat beberapa pecahan panas yang memicu alarm CloudWatch
  - tidak melakukan penskalaan saat diperlukan karena AVG agregat di semua serpihan yang memengaruhi alarm tidak terlampaui.
- ElastiCache batas default pada Node per cluster masih berlaku. Jadi, saat memilih Auto Scaling dan jika Anda mengharapkan simpul maksimum lebih dari batas default, minta peningkatan batas di Batas Layanan AWS dan pilih jenis batas Simpul per klaster per jenis instans.
- Pastikan Anda memiliki cukup ENIs (Antarmuka Jaringan Elastis) yang tersedia di VPC Anda, yang diperlukan selama penskalaan. Untuk informasi selengkapnya, lihat Antarmuka Jaringan Elastis.
- Jika tidak ada kapasitas yang cukup EC2, ElastiCache Auto Scaling tidak akan menskalakan dan ditunda sampai kapasitas tersedia.
- ElastiCache untuk Redis OSS Auto Scaling selama scale-in tidak akan menghapus pecahan dengan slot yang memiliki ukuran item lebih besar dari 256 MB pasca-serialisasi.
- Selama penskalaan ke dalam, serpihan tidak akan dihapus jika memori yang tersedia tidak mencukupi pada konfigurasi serpihan yang dihasilkan.

Menambahkan kebijakan penskalaan

Anda dapat menambahkan kebijakan penskalaan menggunakan. AWS Management Console

Untuk menambahkan kebijakan Auto Scaling ke klaster Valkey dan Redis OSS ElastiCache

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Valkey atau Redis OSS.
3. Pilih klaster yang ingin Anda tambahi kebijakan (pilih nama klaster, bukan tombol di sebelah kirinya).
4. Pilih tab Kebijakan Auto Scaling.
5. Pilih Tambahkan penskalaan dinamis.
6. Untuk Nama kebijakan, masukkan nama kebijakan.
7. Untuk Dimensi yang Dapat Diskalakan pilih serpihan.
8. Untuk metrik target, pilih salah satu dari berikut:

   - Pemanfaatan CPU Primer untuk membuat kebijakan berdasarkan pemanfaatan CPU rata-rata.
   - Memori untuk membuat kebijakan berdasarkan memori basis data rata-rata.
   - Kapasitas untuk membuat kebijakan berdasarkan penggunaan kapasitas basis data rata-rata. Metrik Kapasitas mencakup pemanfaatan memori dan SSD untuk instans bertingkatan data, dan pemanfaatan memori untuk semua jenis instans lainnya.
9. Untuk nilai target, pilih nilai yang lebih besar dari atau sama dengan 35 dan kurang dari atau sama dengan 70. Penskalaan otomatis akan mempertahankan nilai ini untuk metrik target yang dipilih di seluruh ElastiCache pecahan Anda:

   - Pemanfaatan CPU Primer: mempertahankan nilai target untuk metrik `EngineCPUUtilization` di simpul primer.
   - Memori: mempertahankan nilai target untuk metrik `DatabaseMemoryUsageCountedForEvictPercentage`
   - Kapasitas mempertahankan nilai target untuk metrik `DatabaseCapacityUsageCountedForEvictPercentage`,

   Serpihan klaster ditambahkan atau dihapus untuk menjaga metrik tetap mendekati nilai yang ditentukan.
10. (Opsional) Periode pendinginan penskalaan ke dalam atau penskalaan ke luar tidak didukung dari konsol. Gunakan AWS CLI untuk memodifikasi nilai cooldown.

11. Untuk kapasitas Minimum, ketikkan jumlah pecahan minimum yang harus dipertahankan oleh kebijakan ElastiCache Auto Scaling.
12. Untuk kapasitas Maksimum, ketikkan jumlah pecahan maksimum yang harus dipertahankan oleh kebijakan ElastiCache Auto Scaling. Nilai ini harus kurang dari atau sama dengan 250.
13. Pilih Buat.

Mendaftarkan Target yang Dapat Diskalakan

Sebelum Anda dapat menggunakan Auto Scaling dengan cluster ElastiCache for Valkey dan Redis OSS, Anda mendaftarkan cluster Anda dengan auto scaling. ElastiCache Anda melakukannya untuk menentukan dimensi dan batas penskalaan yang akan diterapkan ke cluster itu. ElastiCache auto scaling secara dinamis menskalakan cluster di sepanjang dimensi yang `elasticache:replication-group:NodeGroups` dapat diskalakan, yang mewakili jumlah pecahan cluster.

Menggunakan AWS CLI

Untuk mendaftarkan cluster Valkey dan Redis OSS Anda ElastiCache , gunakan register-scalable-targetperintah dengan parameter berikut:

- `--service-namespace` – Atur nilai ini ke `elasticache`.
- `--resource-id`— Pengidentifikasi sumber daya untuk cluster. Untuk parameter ini, tipe sumber daya adalah `ReplicationGroup` dan pengidentifikasi unik adalah nama cluster, misalnya`replication-group/myscalablecluster`.
- `--scalable-dimension` – Atur nilai ini ke `elasticache:replication-group:NodeGroups`.
- `--max-capacity` — Jumlah pecahan maksimum yang akan dikelola oleh penskalaan ElastiCache otomatis. Untuk informasi tentang hubungan antara `--min-capacity`, `--max-capacity`, dan jumlah serpihan dalam klaster Anda, lihat Kapasitas minimum dan maksimum.
- `--min-capacity` — Jumlah minimum pecahan yang akan dikelola dengan penskalaan ElastiCache otomatis. Untuk informasi tentang hubungan antara `--min-capacity`, `--max-capacity`, dan jumlah serpihan dalam klaster Anda, lihat Kapasitas minimum dan maksimum.

Example

Dalam contoh berikut, Anda mendaftarkan sebuah ElastiCache cluster bernama`myscalablecluster`. Pendaftaran ini menunjukkan bahwa klaster harus diskalakan secara dinamis agar memiliki satu hingga sepuluh serpihan.

Untuk Linux, macOS, atau Unix:

```
aws application-autoscaling register-scalable-target \
    --service-namespace elasticache \
    --resource-id replication-group/myscalablecluster \
    --scalable-dimension elasticache:replication-group:NodeGroups \
    --min-capacity 1 \
    --max-capacity 10 \
```

Untuk Windows:

```
aws application-autoscaling register-scalable-target ^
    --service-namespace elasticache ^
    --resource-id replication-group/myscalablecluster ^
    --scalable-dimension elasticache:replication-group:NodeGroups ^
    --min-capacity 1 ^
    --max-capacity 10 ^
```

Menggunakan API

Untuk mendaftarkan ElastiCache cluster Anda, gunakan register-scalable-targetperintah dengan parameter berikut:

- ServiceNamespace — Tetapkan nilai ini ke elasticache.
- ResourceId — Pengenal sumber daya untuk cluster. ElastiCache Untuk parameter ini, tipe sumber daya adalah ReplicationGroup dan pengidentifikasi unik adalah nama cluster, misalnya`replication-group/myscalablecluster`.
- ScalableDimension — Tetapkan nilai ini ke`elasticache:replication-group:NodeGroups`.
- MinCapacity — Jumlah minimum pecahan yang akan dikelola dengan penskalaan ElastiCache otomatis. Untuk informasi tentang hubungan antara --min-capacity, --max-capacity, dan jumlah replika dalam klaster Anda, lihat Kapasitas minimum dan maksimum.
- MaxCapacity — Jumlah pecahan maksimum yang akan dikelola oleh penskalaan ElastiCache otomatis. Untuk informasi tentang hubungan antara --min-capacity, --max-capacity, dan jumlah replika dalam klaster Anda, lihat Kapasitas minimum dan maksimum.

Example

Dalam contoh berikut, Anda mendaftarkan sebuah ElastiCache cluster bernama `myscalablecluster` dengan Application Auto Scaling API. Pendaftaran ini menunjukkan bahwa klaster harus diskalakan secara dinamis agar memiliki satu hingga 5 replika.

```
POST / HTTP/1.1
Host: autoscaling.us-east-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 219
X-Amz-Target: AnyScaleFrontendService.RegisterScalableTarget
X-Amz-Date: 20160506T182145Z
User-Agent: aws-cli/1.10.23 Python/2.7.11 Darwin/15.4.0 botocore/1.4.8
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS
{
    "ServiceNamespace": "elasticache",
    "ResourceId": "replication-group/myscalablecluster",
    "ScalableDimension": "elasticache:replication-group:NodeGroups",
    "MinCapacity": 1,
    "MaxCapacity": 5
}
```

Menetapkan kebijakan penskalaan

Konfigurasi kebijakan penskalaan pelacakan target direpresentasikan oleh blok JSON yang digunakan untuk mendefinisikan metrik dan nilai target. Anda dapat menyimpan konfigurasi kebijakan penskalaan sebagai blok JSON dalam file teks. Anda menggunakan file teks tersebut saat menjalankan AWS CLI atau Application Auto Scaling API. Untuk informasi selengkapnya tentang sintaks konfigurasi kebijakan, lihat TargetTrackingScalingPolicyConfigurationdi Referensi API Application Auto Scaling.

Opsi berikut tersedia untuk menetapkan konfigurasi kebijakan penskalaan pelacakan target:

Topik

- Menggunakan metrik standar
- Menggunakan metrik kustom
- Menggunakan periode pendinginan

Menggunakan metrik standar

Dengan menggunakan metrik yang telah ditentukan sebelumnya, Anda dapat dengan cepat menentukan kebijakan penskalaan pelacakan target untuk klaster Valkey dan Redis OSS yang ElastiCache berfungsi dengan pelacakan target di Auto Scaling. ElastiCache

Saat ini, ElastiCache mendukung metrik standar berikut di Auto NodeGroup Scaling:

- ElastiCachePrimaryEngineCPUUtilization— Nilai rata-rata `EngineCPUUtilization` metrik di CloudWatch seluruh node primer di cluster.
- ElastiCacheDatabaseMemoryUsageCountedForEvictPercentage— Nilai rata-rata `DatabaseMemoryUsageCountedForEvictPercentage` metrik di CloudWatch seluruh node primer di cluster.
- ElastiCacheDatabaseCapacityUsageCountedForEvictPercentage— Nilai rata-rata `ElastiCacheDatabaseCapacityUsageCountedForEvictPercentage` metrik di CloudWatch seluruh node primer di cluster.

Untuk informasi selengkapnya tentang metrik `EngineCPUUtilization`, `DatabaseMemoryUsageCountedForEvictPercentage` dan `DatabaseCapacityUsageCountedForEvictPercentage`, lihat Pemantauan penggunaan dengan CloudWatch Metrik. Untuk menggunakan metrik standar dalam kebijakan penskalaan, Anda mengonfigurasi pelacakan target untuk kebijakan penskalaan Anda. Konfigurasi ini harus menyertakan `PredefinedMetricSpecification` untuk metrik standar dan TargetValue untuk nilai target metrik tersebut.

Example

Contoh berikut menjelaskan konfigurasi kebijakan tipikal untuk penskalaan pelacakan target untuk klaster Valkey dan Redis OSS. ElastiCache Dalam konfigurasi ini, metrik yang `ElastiCachePrimaryEngineCPUUtilization` telah ditentukan digunakan untuk menyesuaikan cluster berdasarkan pemanfaatan CPU rata-rata 40 persen di semua node primer di cluster.

```
{
    "TargetValue": 40.0,
    "PredefinedMetricSpecification":
    {
        "PredefinedMetricType": "ElastiCachePrimaryEngineCPUUtilization"
    }
}
```

Menggunakan metrik kustom

Dengan metrik kustom, Anda dapat menentukan kebijakan penskalaan pelacakan target yang memenuhi persyaratan kustom Anda. Anda dapat menentukan metrik kustom berdasarkan ElastiCache metrik apa pun yang berubah sebanding dengan penskalaan. Tidak semua ElastiCache metrik berfungsi untuk pelacakan target. Metrik harus berupa metrik pemanfaatan yang valid dan menjelaskan seberapa sibuk suatu instans. Nilai metrik harus meningkat atau menurun secara berbanding lurus dengan jumlah Serpihan dalam klaster. Peningkatan atau penurunan proporsional ini diperlukan untuk menggunakan data metrik untuk menskalakan jumlah serpihan ke luar atau ke dalam secara proporsional.

Example

Contoh berikut menjelaskan konfigurasi pelacakan target untuk kebijakan penskalaan. Dalam konfigurasi ini, metrik kustom menyesuaikan cluster ElastiCache untuk Redis OSS berdasarkan pemanfaatan CPU rata-rata 50 persen di semua pecahan dalam cluster bernama. `my-db-cluster`

```
{
    "TargetValue": 50,
    "CustomizedMetricSpecification":
    {
        "MetricName": "EngineCPUUtilization",
        "Namespace": "AWS/ElastiCache",
        "Dimensions": [
            {
                "Name": "ReplicationGroup","Value": "my-db-cluster"
            },
            {
                "Name": "Role","Value": "PRIMARY"
            }
        ],
        "Statistic": "Average",
        "Unit": "Percent"
    }
}
```

Menggunakan periode pendinginan

Anda dapat menentukan nilai, dalam detik, untuk `ScaleOutCooldown` guna menambahkan periode pendinginan untuk menskalakan klaster Anda ke luar. Demikian pula, Anda dapat menentukan nilai, dalam detik, untuk `ScaleInCooldown` guna menambahkan periode

pendinginan untuk menskalakan klaster Anda ke dalam. Untuk informasi selengkapnya, lihat TargetTrackingScalingPolicyConfigurationdi Referensi API Application Auto Scaling.

Contoh berikut menjelaskan konfigurasi pelacakan target untuk kebijakan penskalaan. Dalam konfigurasi ini, metrik yang `ElastiCachePrimaryEngineCPUUtilization` telah ditentukan digunakan untuk menyesuaikan cluster ElastiCache for Redis OSS berdasarkan pemanfaatan CPU rata-rata 40 persen di semua node utama di cluster tersebut. Konfigurasi ini menyediakan periode pendinginan penskalaan ke dalam selama 10 menit dan periode pendinginan penskalaan ke luar selama 5 menit.

```
{
    "TargetValue": 40.0,
    "PredefinedMetricSpecification":
    {
        "PredefinedMetricType": "ElastiCachePrimaryEngineCPUUtilization"
    },
    "ScaleInCooldown": 600,
    "ScaleOutCooldown": 300
}
```

Menonaktifkan aktivitas penskalaan ke dalam

Anda dapat mencegah konfigurasi kebijakan penskalaan pelacakan target dari penskalaan di klaster Anda dengan menonaktifkan aktivitas penskalaan. Menonaktifkan aktivitas penskalaan ke dalam mencegah kebijakan penskalaan menghapus serpihan, namun masih memungkinkan kebijakan penskalaan untuk membuatnya sesuai kebutuhan.

Anda dapat menentukan nilai Boolean untuk `DisableScaleIn` guna mengaktifkan atau menonaktifkan aktivitas penskalaan ke dalam untuk klaster Anda. Untuk informasi selengkapnya, lihat TargetTrackingScalingPolicyConfigurationdi Referensi API Application Auto Scaling.

Contoh berikut menjelaskan konfigurasi pelacakan target untuk kebijakan penskalaan. Dalam konfigurasi ini, metrik yang `ElastiCachePrimaryEngineCPUUtilization` telah ditentukan menyesuaikan cluster ElastiCache untuk Valkey dan Redis OSS berdasarkan pemanfaatan CPU rata-rata 40 persen di semua node utama di cluster itu. Konfigurasi ini menonaktifkan aktivitas penskalaan ke dalam untuk kebijakan penskalaan.

```
{
    "TargetValue": 40.0,
    "PredefinedMetricSpecification":
```

```
    {
        "PredefinedMetricType": "ElastiCachePrimaryEngineCPUUtilization"
    },
    "DisableScaleIn": true
}
```

Menerapkan kebijakan penskalaan

Setelah mendaftarkan klaster Anda dengan ElastiCache penskalaan otomatis Valkey dan Redis OSS dan menentukan kebijakan penskalaan, Anda menerapkan kebijakan penskalaan ke klaster terdaftar. Untuk menerapkan kebijakan penskalaan ke klaster Redis OSS, Anda dapat menggunakan Application Auto Scaling API AWS CLI atau Application Auto Scaling. ElastiCache

Menerapkan kebijakan penskalaan menggunakan AWS CLI

Untuk menerapkan kebijakan penskalaan ke klaster Valkey dan Redis OSS Anda ElastiCache , gunakan put-scaling-policyperintah dengan parameter berikut:

- --policy-name – Nama kebijakan penskalaan.
- --policy-name – Atur nilai ini ke `TargetTrackingScaling`.
- --resource-id — Pengidentifikasi sumber daya. Untuk parameter ini, tipe sumber daya adalah `ReplicationGroup` dan pengidentifikasi unik adalah nama cluster, misalnya`replication-group/myscalablecluster`.
- --service-namespace – Atur nilai ini ke `elasticache`.
- --scalable-dimension – Atur nilai ini ke `elasticache:replication-group:NodeGroups`.
- -- target-tracking-scaling-policy -configuration - Konfigurasi kebijakan penskalaan pelacakan target yang akan digunakan untuk klaster.

Dalam contoh berikut, Anda menerapkan kebijakan penskalaan pelacakan target yang diberi nama `myscalablepolicy` untuk klaster Valkey dan Redis OSS yang ElastiCache diberi nama dengan penskalaan otomatis. `myscalablecluster` ElastiCache Untuk melakukannya, Anda menggunakan konfigurasi kebijakan yang disimpan dalam file bernama `config.json`.

Untuk Linux, macOS, atau Unix:

```
aws application-autoscaling put-scaling-policy \
    --policy-name myscalablepolicy \
```

```
    --policy-type TargetTrackingScaling \
    --resource-id replication-group/myscalablecluster \
    --service-namespace elasticache \
    --scalable-dimension elasticache:replication-group:NodeGroups \
    --target-tracking-scaling-policy-configuration file://config.json
```

Untuk Windows:

```
aws application-autoscaling put-scaling-policy ^
    --policy-name myscalablepolicy ^
    --policy-type TargetTrackingScaling ^
    --resource-id replication-group/myscalablecluster ^
    --service-namespace elasticache ^
    --scalable-dimension elasticache:replication-group:NodeGroups ^
    --target-tracking-scaling-policy-configuration file://config.json
```

Menerapkan kebijakan penskalaan menggunakan API

Untuk menerapkan kebijakan penskalaan ke klaster Valkey dan Redis OSS Anda ElastiCache , gunakan PutScalingPolicy AWS CLI perintah dengan parameter berikut:

- --policy-name – Nama kebijakan penskalaan.
- --resource-id — Pengidentifikasi sumber daya. Untuk parameter ini, tipe sumber daya adalah `ReplicationGroup` dan pengidentifikasi unik adalah nama cluster, misalnya`replication-group/myscalablecluster`.
- --service-namespace – Atur nilai ini ke `elasticache`.
- --scalable-dimension – Atur nilai ini ke `elasticache:replication-group:NodeGroups`.
- -- target-tracking-scaling-policy -configuration - Konfigurasi kebijakan penskalaan pelacakan target yang akan digunakan untuk klaster.

Dalam contoh berikut, Anda menerapkan kebijakan penskalaan pelacakan target yang diberi nama `myscalablepolicy` ke ElastiCache kluster yang diberi nama dengan penskalaan `myscalablecluster` otomatis ElastiCache . Anda menggunakan konfigurasi kebijakan berdasarkan pada metrik standar `ElastiCachePrimaryEngineCPUUtilization`.

```
POST / HTTP/1.1
Host: autoscaling.us-east-2.amazonaws.com
```

```
Accept-Encoding: identity
Content-Length: 219
X-Amz-Target: AnyScaleFrontendService.PutScalingPolicy
X-Amz-Date: 20160506T182145Z
User-Agent: aws-cli/1.10.23 Python/2.7.11 Darwin/15.4.0 botocore/1.4.8
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS
{
    "PolicyName": "myscalablepolicy",
    "ServiceNamespace": "elasticache",
    "ResourceId": "replication-group/myscalablecluster",
    "ScalableDimension": "elasticache:replication-group:NodeGroups",
    "PolicyType": "TargetTrackingScaling",
    "TargetTrackingScalingPolicyConfiguration": {
        "TargetValue": 40.0,
        "PredefinedMetricSpecification":
        {
            "PredefinedMetricType": "ElastiCachePrimaryEngineCPUUtilization"
        }
    }
}
```

Mengedit kebijakan penskalaan

Anda dapat mengedit kebijakan penskalaan menggunakan API AWS Management Console, Application Auto Scaling AWS CLI, atau Application Auto Scaling.

Mengedit kebijakan penskalaan menggunakan AWS Management Console

Untuk mengedit kebijakan Auto Scaling untuk klaster Valkey dan Redis OSS ElastiCache

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih mesin yang sesuai.
3. Pilih klaster yang ingin Anda tambahi kebijakan (pilih nama klaster, bukan tombol di sebelah kirinya).
4. Pilih tab Kebijakan Auto Scaling.
5. Di bagian Kebijakan penskalaan, pilih tombol di samping kebijakan Auto Scaling yang ingin Anda ubah, lalu pilih Ubah.
6. Buat perubahan yang diperlukan pada kebijakan.

7. Pilih Ubah.

Mengedit kebijakan penskalaan menggunakan AWS CLI dan API

Anda dapat menggunakan Application Auto Scaling API AWS CLI atau Application Auto Scaling untuk mengedit kebijakan penskalaan dengan cara yang sama seperti Anda menerapkan kebijakan penskalaan:

- Saat menggunakan AWS CLI, tentukan nama kebijakan yang ingin Anda edit di `--policy-name` parameter. Tentukan nilai baru untuk parameter yang ingin Anda ubah.
- Saat menggunakan API Application Auto Scaling, tentukan nama kebijakan yang ingin Anda edit dalam parameter `PolicyName`. Tentukan nilai baru untuk parameter yang ingin Anda ubah.

Lihat informasi yang lebih lengkap di Menerapkan kebijakan penskalaan.

Menghapus kebijakan penskalaan

Anda dapat menghapus kebijakan penskalaan menggunakan API AWS Management Console, Application Auto Scaling AWS CLI, atau Application Auto Scaling.

Menghapus kebijakan penskalaan menggunakan AWS Management Console

Untuk menghapus kebijakan Auto Scaling untuk klaster Redis ElastiCache OSS

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Valkey atau Redis OSS.
3. Pilih klaster yang kebijakan Auto Scaling-nya ingin Anda edit (pilih nama klaster, bukan tombol di sebelah kiri).
4. Pilih tab Kebijakan Auto Scaling.
5. Di bagian Kebijakan penskalaan, pilih kebijakan Auto Scaling, lalu pilih Hapus.

Menghapus kebijakan penskalaan menggunakan AWS CLI

Untuk menghapus kebijakan penskalaan ke klaster Valkey dan Redis OSS Anda ElastiCache , gunakan delete-scaling-policy AWS CLI perintah dengan parameter berikut:

- --policy-name – Nama kebijakan penskalaan.

- --resource-id — Pengidentifikasi sumber daya. Untuk parameter ini, tipe sumber daya adalah `ReplicationGroup` dan pengidentifikasi unik adalah nama cluster, misalnya`replication-group/myscalablecluster`.
- --service-namespace – Atur nilai ini ke `elasticache`.
- --scalable-dimension – Atur nilai ini ke `elasticache:replication-group:NodeGroups`.

Dalam contoh berikut, Anda menghapus kebijakan penskalaan pelacakan target yang dinamai `myscalablepolicy` dari kluster bernama. `myscalablecluster`

Untuk Linux, macOS, atau Unix:

```
aws application-autoscaling delete-scaling-policy \
    --policy-name myscalablepolicy \
    --resource-id replication-group/myscalablecluster \
    --service-namespace elasticache \
    --scalable-dimension elasticache:replication-group:NodeGroups
```

Untuk Windows:

```
aws application-autoscaling delete-scaling-policy ^
    --policy-name myscalablepolicy ^
    --resource-id replication-group/myscalablecluster ^
    --service-namespace elasticache ^
    --scalable-dimension elasticache:replication-group:NodeGroups
```

Menghapus kebijakan penskalaan menggunakan API

Untuk menghapus kebijakan penskalaan ke klaster Valkey dan Redis OSS Anda ElastiCache , gunakan DeleteScalingPolicy AWS CLI perintah dengan parameter berikut:

- --policy-name – Nama kebijakan penskalaan.
- --resource-id — Pengidentifikasi sumber daya. Untuk parameter ini, tipe sumber daya adalah `ReplicationGroup` dan pengidentifikasi unik adalah nama cluster, misalnya`replication-group/myscalablecluster`.
- --service-namespace – Atur nilai ini ke `elasticache`.
- --scalable-dimension – Atur nilai ini ke `elasticache:replication-group:NodeGroups`.

Dalam contoh berikut, Anda menghapus kebijakan penskalaan pelacakan target yang dinamai `myscalablepolicy` dari kluster bernama. `myscalablecluster`

```
POST / HTTP/1.1
Host: autoscaling.us-east-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 219
X-Amz-Target: AnyScaleFrontendService.DeleteScalingPolicy
X-Amz-Date: 20160506T182145Z
User-Agent: aws-cli/1.10.23 Python/2.7.11 Darwin/15.4.0 botocore/1.4.8
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS
{
    "PolicyName": "myscalablepolicy",
    "ServiceNamespace": "elasticache",
    "ResourceId": "replication-group/myscalablecluster",
    "ScalableDimension": "elasticache:replication-group:NodeGroups"
}
```

Gunakan AWS CloudFormation untuk kebijakan Auto Scaling

Cuplikan ini menunjukkan cara membuat kebijakan pelacakan target dan menerapkannya ke AWS::ElastiCache::ReplicationGroupsumber daya menggunakan sumber daya. AWS::ApplicationAutoScaling::ScalableTarget Tindakan tersebut menggunakan fungsi intrinsik Fn::Join dan Ref untuk membangun properti `ResourceId` dengan nama logis sumber daya `AWS::ElastiCache::ReplicationGroup` yang ditentukan dalam templat yang sama.

```
ScalingTarget:
   Type: 'AWS::ApplicationAutoScaling::ScalableTarget'
   Properties:
     MaxCapacity: 3
     MinCapacity: 1
     ResourceId: !Sub replication-group/${logicalName}
     ScalableDimension: 'elasticache:replication-group:NodeGroups'
     ServiceNamespace: elasticache
     RoleARN: !Sub "arn:aws:iam::${AWS::AccountId}:role/aws-
service-role/elasticache.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG"

  ScalingPolicy:
    Type: "AWS::ApplicationAutoScaling::ScalingPolicy"
    Properties:
```

```
      ScalingTargetId: !Ref ScalingTarget
      ServiceNamespace: elasticache
      PolicyName: testpolicy
      PolicyType: TargetTrackingScaling
      ScalableDimension: 'elasticache:replication-group:NodeGroups'
      TargetTrackingScalingPolicyConfiguration:
        PredefinedMetricSpecification:
          PredefinedMetricType: ElastiCachePrimaryEngineCPUUtilization
        TargetValue: 40
```

Penskalaan terjadwal

Penskalaan berdasarkan jadwal memungkinkan Anda menskalakan aplikasi sebagai respons terhadap perubahan permintaan yang dapat diprediksi. Untuk menggunakan penskalaan terjadwal, Anda membuat tindakan terjadwal, yang memberi tahu Valkey dan Redis OSS ElastiCache untuk melakukan aktivitas penskalaan pada waktu tertentu. Saat membuat tindakan terjadwal, Anda menentukan klaster yang ada, kapan aktivitas penskalaan harus terjadi, kapasitas minimum, dan kapasitas maksimum. Anda dapat membuat tindakan terjadwal yang menskalakan satu kali saja atau menskalakan berdasarkan jadwal berulang.

Anda hanya dapat membuat tindakan terjadwal untuk cluster yang sudah ada. Anda tidak dapat membuat tindakan terjadwal pada saat yang sama saat Anda membuat klaster.

Untuk informasi selengkapnya tentang terminologi untuk pembuatan, manajemen, dan penghapusan tindakan terjadwal, lihat Perintah yang umum digunakan untuk pembuatan, manajemen, dan penghapusan tindakan terjadwal

Untuk membuat jadwal berulang:

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Valkey atau Redis OSS.
3. Pilih klaster yang ingin Anda tambahkan kebijakannya.
4. Pilih Kelola kebijakan Auto Scaling dari drop-down Tindakan.
5. Pilih tab Kebijakan Auto Scaling.
6. Di bagian Kebijakan Auto Scaling, kotak dialog Tambahkan kebijakan penskalaan akan muncul. Pilih Penskalaan terjadwal.
7. Untuk Nama Kebijakan, masukkan nama kebijakan.

8. Untuk Dimensi yang Dapat Diskalakan, pilih Serpihan.
9. Untuk Serpihan Target, pilih nilai.
10. Untuk Perulangan, pilih Berulang.
11. Untuk Frekuensi, pilih nilai masing-masing.
12. Untuk Tanggal Mulai dan Waktu mulai, pilih waktu dari kapan kebijakan akan berlaku.
13. Pilih Tambahkan kebijakan.

Untuk membuat tindakan terjadwal satu kali:

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Valkey atau Redis OSS.
3. Pilih klaster yang ingin Anda tambahkan kebijakannya.
4. Pilih Kelola kebijakan Auto Scaling dari drop-down Tindakan.
5. Pilih tab Kebijakan Auto Scaling.
6. Di bagian Kebijakan Auto Scaling, kotak dialog Tambahkan kebijakan penskalaan akan muncul. Pilih Penskalaan terjadwal.
7. Untuk Nama Kebijakan, masukkan nama kebijakan.
8. Untuk Dimensi yang Dapat Diskalakan, pilih Serpihan.
9. Untuk Serpihan Target, pilih nilai.
10. Untuk Perulangan, pilih Satu Kali.
11. Untuk Tanggal Mulai dan Waktu mulai, pilih waktu dari kapan kebijakan akan berlaku.
12. Untuk Tanggal Berakhir pilih sampai tanggal kapan kebijakan akan berlaku.
13. Pilih Tambahkan kebijakan.

Untuk menghapus tindakan terjadwal

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Valkey atau Redis OSS.
3. Pilih klaster yang ingin Anda tambahkan kebijakannya.
4. Pilih Kelola kebijakan Auto Scaling dari drop-down Tindakan.

5. Pilih tab Kebijakan Auto Scaling.
6. Pada bagian Kebijakan Auto Scaling, pilih kebijakan Auto Scaling, lalu pilih Hapus dari dialog Tindakan.

Untuk mengelola penskalaan terjadwal menggunakan AWS CLI

Gunakan APIs aplikasi-autoscaling berikut:

- put-scheduled-action
- describe-scheduled-actions
- delete-scheduled-action

Gunakan AWS CloudFormation untuk membuat tindakan terjadwal

Cuplikan ini menunjukkan cara membuat kebijakan pelacakan target dan menerapkannya ke AWS::ElastiCache::ReplicationGroupsumber daya menggunakan sumber daya. AWS::ApplicationAutoScaling::ScalableTarget Tindakan tersebut menggunakan fungsi intrinsik Fn::Join dan Ref untuk membangun properti `ResourceId` dengan nama logis sumber daya `AWS::ElastiCache::ReplicationGroup` yang ditentukan dalam templat yang sama.

```
ScalingTarget:
   Type: 'AWS::ApplicationAutoScaling::ScalableTarget'
   Properties:
     MaxCapacity: 3
     MinCapacity: 1
     ResourceId: !Sub replication-group/${logicalName}
     ScalableDimension: 'elasticache:replication-group:NodeGroups'
     ServiceNamespace: elasticache
     RoleARN: !Sub "arn:aws:iam::${AWS::AccountId}:role/aws-
service-role/elasticache.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG"
     ScheduledActions:
       - EndTime: '2020-12-31T12:00:00.000Z'
         ScalableTargetAction:
           MaxCapacity: '5'
           MinCapacity: '2'
         ScheduledActionName: First
         Schedule: 'cron(0 18 * * ? *)'
```

Menggunakan Auto Scaling dengan replika

Grup ElastiCache replikasi dapat mengatur satu atau lebih cache untuk bekerja sebagai simpul logis tunggal.

Berikut ini memberikan rincian tentang pelacakan target dan kebijakan terjadwal dan bagaimana menerapkannya menggunakan AWS Management Console AWS CLI dan APIs.

Kebijakan penskalaan pelacakan target

Dengan kebijakan penskalaan pelacakan target, Anda memilih metrik dan menetapkan nilai target. ElastiCache untuk Valkey dan Redis OSS AutoScaling membuat dan mengelola CloudWatch alarm yang memicu kebijakan penskalaan dan menghitung penyesuaian penskalaan berdasarkan metrik dan nilai target. Kebijakan penskalaan menambahkan atau menghapus replika secara seragam di semua serpihan yang diperlukan untuk menjaga metrik pada, atau mendekati, nilai target yang ditentukan. Selain menjaga metrik agar mendekati nilai target, kebijakan penskalaan pelacakan target juga menyesuaikan dengan fluktuasi metrik karena fluktuasi pola beban dan meminimalkan fluktuasi cepat dalam kapasitas armada.

Kriteria Auto Scaling untuk replika

Kebijakan Auto Scaling menentukan metrik standar berikut untuk klaster Anda:

`ElastiCacheReplicaEngineCPUUtilization`: Ambang batas pemanfaatan AVG EngineCPU digabungkan di semua replika yang digunakan ElastiCache untuk memicu operasi auto-scaling. Anda dapat menetapkan target pemanfaatan antara 35 persen dan 70 persen.

Ketika layanan mendeteksi bahwa `ElastiCacheReplicaEngineCPUUtilization` metrik Anda sama dengan atau lebih besar dari pengaturan Target, itu akan meningkatkan replika di seluruh pecahan Anda secara otomatis. ElastiCache skala replika cluster Anda dengan hitungan yang sama dengan yang lebih besar dari dua angka: Variasi persen dari Target dan satu replika. Untuk scale-in, ElastiCache tidak akan auto scale-in kecuali nilai metrik keseluruhan di bawah 75 persen dari Target yang Anda tentukan.

Sebagai contoh penskalaan ke luar, jika Anda memiliki 5 serpihan dan 1 replika masing-masing:

Jika Target Anda melanggar 30 persen, ElastiCache untuk Valkey dan Redis OSS skala sebesar 1 replika (maks (0,3, default 1)) di semua pecahan. yang menghasilkan 5 pecahan dengan masing-masing 2 replika,

Untuk contoh scale-in, jika Anda telah memilih nilai Target 60 persen, ElastiCache untuk Valkey dan Redis OSS tidak akan auto scale-in sampai metrik kurang dari atau sama dengan 45 persen (25 persen di bawah Target 60 persen).

Pertimbangan untuk Auto Scaling

Perhatikan sejumlah pertimbangan berikut:

- Kebijakan penskalaan pelacakan target mengasumsikan bahwa penskalaan ke luar harus dilakukan saat metrik yang ditentukan berada di atas nilai target. Anda tidak dapat menggunakan kebijakan penskalaan pelacakan target untuk memperkecil skala ketika metrik yang ditentukan berada di bawah nilai target. ElastiCache untuk Valkey dan Redis OSS menskalakan replika dengan maksimum (% deviasi dibulatkan dari Target, default 1) dari replika yang ada di semua pecahan di cluster.
- Kebijakan penskalaan pelacakan target tidak melakukan penskalaan saat metrik yang ditentukan tidak memiliki data yang mencukupi. Kebijakan penskalaan pelacakan target tidak melakukan penskalaan ke dalam karena data yang tidak mencukupi tidak ditafsirkan sebagai pemanfaatan yang rendah.
- Anda mungkin melihat kesenjangan antara nilai target dan titik data metrik yang aktual. Ini karena ElastiCache Auto Scaling selalu bertindak konservatif dengan membulatkan ke atas atau ke bawah ketika menentukan berapa banyak kapasitas untuk menambah atau menghapus. Hal ini mencegah layanan menambahkan kapasitas yang tidak mencukupi atau menghapus kapasitas terlalu banyak.
- Untuk memastikan ketersediaan aplikasi, layanan menskalakan ke luar secara proporsional berdasarkan metrik secepat mungkin, tetapi menskalakan ke dalam secara lebih bertahap dengan penskalaan ke dalam maksimal 1 replika di seluruh serpihan di klaster.
- Anda dapat memiliki beberapa kebijakan penskalaan pelacakan target ElastiCache untuk klaster Valkey dan Redis OSS, asalkan masing-masing menggunakan metrik yang berbeda. Tujuan Auto Scaling adalah untuk selalu memprioritaskan ketersediaan, sehingga perilakunya berbeda tergantung pada apakah kebijakan pelacakan target siap untuk skala atau skala. Fitur ini akan menskalakan ke luar layanan jika salah satu kebijakan pelacakan target siap untuk diskalakan ke luar. Namun, penskalaan ke dalam akan dilakukan hanya jika semua kebijakan pelacakan target (dengan porsi penskalaan ke dalam diaktifkan) siap untuk diskalakan ke dalam.
- Jangan mengedit atau menghapus CloudWatch alarm yang dikelola ElastiCache Auto Scaling untuk kebijakan penskalaan pelacakan target. Auto Scaling menghapus alarm secara otomatis saat Anda menghapus kebijakan penskalaan atau menghapus klaster.
- ElastiCache Auto Scaling tidak mencegah Anda memodifikasi replika secara manual di seluruh pecahan. Penyesuaian manual ini tidak memengaruhi CloudWatch alarm yang ada yang melekat

pada kebijakan penskalaan, tetapi dapat memengaruhi metrik yang dapat memicu alarm ini. CloudWatch

- CloudWatch Alarm yang dikelola oleh Auto Scaling ini ditentukan melalui metrik AVG di semua pecahan di cluster. Jadi, serpihan dengan lalu lintas tertinggi dapat menghasilkan salah satu skenario berikut:
  - penskalaan saat tidak diperlukan karena beban pada beberapa pecahan panas yang memicu alarm CloudWatch
  - tidak melakukan penskalaan saat diperlukan karena AVG agregat di semua serpihan yang memengaruhi alarm tidak terlampaui.
- ElastiCache batas default pada node per cluster masih berlaku. Jadi, saat memilih Auto Scaling dan jika Anda mengharapkan simpul maksimum lebih dari batas default, minta peningkatan batas di Batas Layanan AWS dan pilih jenis batas Simpul per klaster per jenis instans.
- Pastikan Anda memiliki cukup ENIs (Antarmuka Jaringan Elastis) yang tersedia di VPC Anda, yang diperlukan selama penskalaan. Untuk informasi selengkapnya, lihat Antarmuka Jaringan Elastis.
- Jika kapasitas tidak cukup tersedia EC2, ElastiCache Auto Scaling tidak akan ditingkatkan hingga kapasitas tersedia, atau jika Anda memodifikasi cluster secara manual ke jenis instans yang memiliki kapasitas cukup.
- ElastiCache Auto Scaling tidak mendukung penskalaan replika dengan cluster yang memiliki `ReservedMemoryPercent` kurang dari 25 persen. Lihat informasi yang lebih lengkap di Mengelola memori cadangan untuk Valkey dan Redis OSS.

Menambahkan kebijakan penskalaan

Anda dapat menambahkan kebijakan penskalaan menggunakan. AWS Management Console

Menambahkan kebijakan penskalaan menggunakan AWS Management Console

Untuk menambahkan kebijakan penskalaan otomatis ElastiCache untuk Valkey dan Redis OSS

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Valkey atau Redis OSS.
3. Pilih klaster yang ingin Anda tambahi kebijakan (pilih nama klaster, bukan tombol di sebelah kirinya).
4. Pilih tab Kebijakan Auto Scaling.
5. Pilih Tambahkan penskalaan dinamis.

6. Di bagian Kebijakan penskalaan, pilih Tambahkan penskalaan dinamis.
7. Untuk Nama Kebijakan, masukkan nama kebijakan.
8. Untuk Dimensi yang Dapat Diskalakan, pilih Replika dari kotak dialog.
9. Untuk nilai target, ketikkan persentase Avg dari pemanfaatan CPU yang ingin Anda pertahankan di ElastiCache Replicas. Nilai ini harus >=35 dan <=70. Replika klaster ditambahkan atau dihapus untuk menjaga metrik tetap mendekati nilai yang ditentukan.
10. (Opsional) Periode pendinginan penskalaan ke dalam atau penskalaan ke luar tidak didukung dari Konsol. Gunakan AWS CLI untuk memodifikasi nilai pendinginan.
11. Untuk kapasitas Minimum, ketikkan jumlah replika minimum yang harus dipertahankan oleh kebijakan ElastiCache Auto Scaling.
12. Untuk kapasitas Maksimum, ketikkan jumlah maksimum replika yang harus dipertahankan oleh kebijakan ElastiCache Auto Scaling. Nilai ini harus >=5.
13. Pilih Buat.

Mendaftarkan Target yang Dapat Diskalakan

Anda dapat menerapkan kebijakan penskalaan berdasarkan metrik standar atau kustom. Untuk melakukannya, Anda dapat menggunakan Application Auto Scaling AWS CLI API atau Application Auto Scaling. Langkah pertama adalah mendaftarkan grup replikasi Valkey dan Redis OSS Anda ElastiCache dengan Auto Scaling.

Sebelum Anda dapat menggunakan ElastiCache auto scaling dengan cluster, Anda harus mendaftarkan cluster Anda dengan ElastiCache auto scaling. Anda melakukannya untuk menentukan dimensi dan batas penskalaan yang akan diterapkan ke cluster itu. ElastiCache auto scaling secara dinamis menskalakan cluster di sepanjang dimensi yang `elasticache:replication-group:Replicas` dapat diskalakan, yang mewakili jumlah replika cluster per shard.

Menggunakan CLI

Untuk mendaftarkan ElastiCache cluster Anda, gunakan register-scalable-targetperintah dengan parameter berikut:

- --service-namespace – Atur nilai ini ke elasticache.
- --resource-id — Pengidentifikasi sumber daya untuk cluster. ElastiCache Untuk parameter ini, tipe sumber daya adalah ReplicationGroup dan pengidentifikasi unik adalah nama cluster, misalnya`replication-group/myscalablecluster`.
- --scalable-dimension – Atur nilai ini ke `elasticache:replication-group:Replicas`.

- --min-capacity — Jumlah minimum replika yang akan dikelola dengan penskalaan otomatis. ElastiCache Untuk informasi tentang hubungan antara --min-capacity, --max-capacity, dan jumlah replika dalam klaster Anda, lihat Kapasitas minimum dan maksimum.
- --max-capacity — Jumlah maksimum replika yang akan dikelola dengan penskalaan otomatis. ElastiCache Untuk informasi tentang hubungan antara --min-capacity, --max-capacity, dan jumlah replika dalam klaster Anda, lihat Kapasitas minimum dan maksimum.

Example

Dalam contoh berikut, Anda mendaftarkan sebuah ElastiCache cluster bernama`myscalablecluster`. Pendaftaran ini menunjukkan bahwa klaster harus diskalakan secara dinamis agar memiliki satu hingga 5 replika.

Untuk Linux, macOS, atau Unix:

```
aws application-autoscaling register-scalable-target \
    --service-namespace elasticache \
    --resource-id replication-group/myscalablecluster \
    --scalable-dimension elasticache:replication-group:Replicas \
    --min-capacity 1 \
    --max-capacity 5 \
```

Untuk Windows:

```
aws application-autoscaling register-scalable-target ^
    --service-namespace elasticache ^
    --resource-id replication-group/myscalablecluster ^
    --scalable-dimension elasticache:replication-group:Replicas ^
    --min-capacity 1 ^
    --max-capacity 5 ^
```

Menggunakan API

Untuk mendaftarkan ElastiCache cluster Anda, gunakan register-scalable-targetperintah dengan parameter berikut:

- ServiceNamespace — Tetapkan nilai ini ke elasticache.
- ResourceId — Pengenal sumber daya untuk cluster. ElastiCache Untuk parameter ini, tipe sumber daya adalah ReplicationGroup dan pengidentifikasi unik adalah nama cluster, misalnya`replication-group/myscalablecluster`.

- ScalableDimension — Tetapkan nilai ini ke`elasticache:replication-group:Replicas`.
- MinCapacity — Jumlah minimum replika yang akan dikelola dengan penskalaan ElastiCache otomatis. Untuk informasi tentang hubungan antara --min-capacity, --max-capacity, dan jumlah replika dalam klaster Anda, lihat Kapasitas minimum dan maksimum.
- MaxCapacity — Jumlah maksimum replika yang akan dikelola oleh penskalaan ElastiCache otomatis. Untuk informasi tentang hubungan antara --min-capacity, --max-capacity, dan jumlah replika dalam klaster Anda, lihat Kapasitas minimum dan maksimum.

Example

Dalam contoh berikut, Anda mendaftarkan klaster bernama `myscalablecluster` Application Auto Scaling API. Pendaftaran ini menunjukkan bahwa klaster harus diskalakan secara dinamis agar memiliki satu hingga 5 replika.

```
POST / HTTP/1.1
Host: autoscaling.us-east-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 219
X-Amz-Target: AnyScaleFrontendService.RegisterScalableTarget
X-Amz-Date: 20160506T182145Z
User-Agent: aws-cli/1.10.23 Python/2.7.11 Darwin/15.4.0 botocore/1.4.8
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS
{
    "ServiceNamespace": "elasticache",
    "ResourceId": "replication-group/myscalablecluster",
    "ScalableDimension": "elasticache:replication-group:Replicas",
    "MinCapacity": 1,
    "MaxCapacity": 5
}
```

Menetapkan kebijakan penskalaan

Konfigurasi kebijakan penskalaan pelacakan target direpresentasikan oleh blok JSON yang digunakan untuk mendefinisikan metrik dan nilai target. Anda dapat menyimpan konfigurasi kebijakan penskalaan sebagai blok JSON dalam file teks. Anda menggunakan file teks tersebut saat menjalankan AWS CLI atau Application Auto Scaling API. Untuk informasi selengkapnya tentang sintaksis konfigurasi kebijakan, lihat TargetTrackingScalingPolicyConfiguration dalam Referensi API Application Auto Scaling.

Opsi berikut tersedia untuk menetapkan konfigurasi kebijakan penskalaan pelacakan target:

Topik

Menggunakan metrik standar

Konfigurasi kebijakan penskalaan pelacakan target direpresentasikan oleh blok JSON yang digunakan untuk mendefinisikan metrik dan nilai target. Anda dapat menyimpan konfigurasi kebijakan penskalaan sebagai blok JSON dalam file teks. Anda menggunakan file teks tersebut saat menjalankan AWS CLI atau Application Auto Scaling API. Untuk informasi selengkapnya tentang sintaksis konfigurasi kebijakan, lihat TargetTrackingScalingPolicyConfiguration dalam Referensi API Application Auto Scaling.

Opsi berikut tersedia untuk menetapkan konfigurasi kebijakan penskalaan pelacakan target:

Topik

Menggunakan metrik standar

Dengan menggunakan metrik yang telah ditentukan sebelumnya, Anda dapat dengan cepat menentukan kebijakan penskalaan pelacakan target untuk klaster Valkey dan Redis OSS yang ElastiCache berfungsi dengan pelacakan target di Auto Scaling. ElastiCache Saat ini, ElastiCache mendukung metrik standar berikut di Auto Scaling ElastiCache Replika:

`ElastiCacheReplicaEngineCPUUtilization`— Nilai rata-rata CPUUtilization metrik Engine CloudWatch di semua replika di cluster. Anda dapat menemukan nilai metrik agregat di CloudWatch

bawah ElastiCache `ReplicationGroupId, Role` untuk Replica Required ReplicationGroupId dan Role Replica.

Untuk menggunakan metrik standar dalam kebijakan penskalaan, Anda membuat konfigurasi pelacakan target untuk kebijakan penskalaan Anda. Konfigurasi ini harus menyertakan `PredefinedMetricSpecification` untuk metrik standar dan `TargetValue` untuk nilai target metrik tersebut.

Menggunakan metrik kustom

Dengan menggunakan metrik kustom, Anda dapat menentukan kebijakan penskalaan pelacakan target yang memenuhi persyaratan kustom Anda. Anda dapat menentukan metrik kustom berdasarkan metrik Valkey dan Redis OSS apa pun ElastiCache yang berubah sebanding dengan penskalaan. Tidak semua ElastiCache metrik berfungsi untuk pelacakan target. Metrik harus berupa metrik pemanfaatan yang valid dan menjelaskan seberapa sibuk suatu instans. Nilai metrik harus meningkat atau menurun proporsinya terhadap jumlah replika dalam klaster. Peningkatan atau penurunan proporsional ini diperlukan untuk menggunakan data metrik untuk meningkatkan atau menurunkan jumlah replika secara proporsional.

Example

Contoh berikut menjelaskan konfigurasi pelacakan target untuk kebijakan penskalaan. Dalam konfigurasi ini, metrik kustom menyesuaikan cluster berdasarkan pemanfaatan CPU rata-rata 50 persen di semua replika dalam sebuah cluster bernama. `my-db-cluster`

```
{"TargetValue": 50,
    "CustomizedMetricSpecification":
    {"MetricName": "EngineCPUUtilization",
        "Namespace": "AWS/ElastiCache",
        "Dimensions": [
            {"Name": "ReplicationGroup","Value": "my-db-cluster"},
            {"Name": "Role","Value": "REPLICA"}
        ],
        "Statistic": "Average",
        "Unit": "Percent"
    }
}
```

Menggunakan periode pendinginan

Anda dapat menentukan nilai, dalam detik, untuk `ScaleOutCooldown` guna menambahkan periode pendinginan untuk menskalakan klaster Anda ke luar. Demikian pula, Anda dapat menentukan nilai, dalam detik, untuk `ScaleInCooldown` guna menambahkan periode pendinginan untuk menskalakan klaster Anda ke dalam. Untuk informasi selengkapnya tentang `ScaleInCooldown` dan `ScaleOutCooldown`, lihat TargetTrackingScalingPolicyConfiguration dalam Referensi API Application Auto Scaling. Contoh berikut menjelaskan konfigurasi pelacakan target untuk kebijakan penskalaan. Dalam konfigurasi ini, metrik yang `ElastiCacheReplicaEngineCPUUtilization` telah ditentukan digunakan untuk menyesuaikan cluster berdasarkan pemanfaatan CPU rata-rata 40 persen di semua replika di cluster itu. Konfigurasi ini menyediakan periode pendinginan penskalaan ke dalam selama 10 menit dan periode pendinginan penskalaan ke luar selama 5 menit.

```
{"TargetValue": 40.0,
    "PredefinedMetricSpecification":
    {"PredefinedMetricType": "ElastiCacheReplicaEngineCPUUtilization"
    },
    "ScaleInCooldown": 600,
    "ScaleOutCooldown": 300
}
```

Menonaktifkan aktivitas penskalaan ke dalam

Anda dapat mencegah konfigurasi kebijakan penskalaan pelacakan target dari penskalaan di klaster Valkey dan Redis ElastiCache OSS Anda dengan menonaktifkan aktivitas penskalaan. Menonaktifkan aktivitas penskalaan ke dalam mencegah kebijakan penskalaan menghapus replika, namun masih memungkinkan kebijakan penskalaan untuk menambahkannya sesuai kebutuhan.

Anda dapat menentukan nilai Boolean untuk `DisableScaleIn` guna mengaktifkan atau menonaktifkan aktivitas penskalaan ke dalam untuk klaster Anda. Untuk informasi selengkapnya tentang `DisableScaleIn`, lihat TargetTrackingScalingPolicyConfiguration dalam Referensi API Application Auto Scaling.

Example

Contoh berikut menjelaskan konfigurasi pelacakan target untuk kebijakan penskalaan. Dalam konfigurasi ini, metrik yang `ElastiCacheReplicaEngineCPUUtilization` telah ditentukan menyesuaikan cluster berdasarkan pemanfaatan CPU rata-rata 40 persen di semua replika di cluster itu. Konfigurasi ini menonaktifkan aktivitas penskalaan ke dalam untuk kebijakan penskalaan.

```
{"TargetValue": 40.0,
    "PredefinedMetricSpecification":
    {"PredefinedMetricType": "ElastiCacheReplicaEngineCPUUtilization"
    },
    "DisableScaleIn": true
}
```

Menerapkan kebijakan penskalaan ke klaster ElastiCache Valkey dan Redis OSS

Setelah mendaftarkan klaster Anda dengan ElastiCache penskalaan otomatis Valkey dan Redis OSS dan menentukan kebijakan penskalaan, Anda menerapkan kebijakan penskalaan ke klaster terdaftar. Untuk menerapkan kebijakan penskalaan ke klaster Valkey dan Redis OSS, Anda dapat menggunakan Application Auto Scaling API AWS CLI atau Application Auto Scaling. ElastiCache

Menggunakan AWS CLI

Untuk menerapkan kebijakan penskalaan ke klaster Valkey dan Redis OSS Anda ElastiCache , gunakan put-scaling-policyperintah dengan parameter berikut:

- --policy-name – Nama kebijakan penskalaan.
- --policy-name – Atur nilai ini ke `TargetTrackingScaling`.
- --resource-id — Pengidentifikasi sumber daya untuk cluster. Untuk parameter ini, tipe sumber daya adalah ReplicationGroup dan pengidentifikasi unik adalah nama cluster, misalnya`replication-group/myscalablecluster`.
- --service-namespace – Atur nilai ini ke elasticache.
- --scalable-dimension – Atur nilai ini ke `elasticache:replication-group:Replicas`.
- -- target-tracking-scaling-policy -configuration - Konfigurasi kebijakan penskalaan pelacakan target yang akan digunakan untuk klaster.

Example

Dalam contoh berikut, Anda menerapkan kebijakan penskalaan pelacakan target yang diberi nama `myscalablepolicy` ke kluster yang diberi nama dengan penskalaan `myscalablecluster` otomatis ElastiCache . Untuk melakukannya, Anda menggunakan konfigurasi kebijakan yang disimpan dalam file bernama `config.json`.

Untuk Linux, macOS, atau Unix:

```
aws application-autoscaling put-scaling-policy \
    --policy-name myscalablepolicy \
    --policy-type TargetTrackingScaling \
    --resource-id replication-group/myscalablecluster \
    --service-namespace elasticache \
    --scalable-dimension elasticache:replication-group:Replicas \
    --target-tracking-scaling-policy-configuration file://config.json
```

```
{"TargetValue": 40.0,
    "PredefinedMetricSpecification":
    {"PredefinedMetricType": "ElastiCacheReplicaEngineCPUUtilization"
    },
    "DisableScaleIn": true
}
```

Untuk Windows:

```
aws application-autoscaling put-scaling-policy ^
    --policy-name myscalablepolicy ^
    --policy-type TargetTrackingScaling ^
    --resource-id replication-group/myscalablecluster ^
    --service-namespace elasticache ^
    --scalable-dimension elasticache:replication-group:Replicas ^
    --target-tracking-scaling-policy-configuration file://config.json
```

Menggunakan API

Untuk menerapkan kebijakan penskalaan ke ElastiCache klaster Anda dengan Application Auto Scaling API, gunakan PutScalingPolicyoperasi Application Auto Scaling API dengan parameter berikut:

- PolicyName — Nama kebijakan penskalaan.
- PolicyType — Tetapkan nilai ini ke`TargetTrackingScaling`.
- ResourceId — Pengidentifikasi sumber daya untuk cluster. Untuk parameter ini, tipe sumber daya adalah ReplicationGroup dan pengidentifikasi unik adalah nama cluster ElastiCache untuk Redis OSS, misalnya. `replication-group/myscalablecluster`
- ServiceNamespace — Tetapkan nilai ini ke elasticache.

- ScalableDimension — Tetapkan nilai ini ke`elasticache:replication-group:Replicas`.
- TargetTrackingScalingPolicyConfiguration — Konfigurasi kebijakan penskalaan pelacakan target yang akan digunakan untuk klaster.

Example

Dalam contoh berikut, Anda menerapkan kebijakan penskalaan pelacakan target yang diberi nama `scalablepolicy` ke kluster yang diberi nama dengan penskalaan `myscalablecluster` otomatis ElastiCache . Anda menggunakan konfigurasi kebijakan berdasarkan pada metrik standar `ElastiCacheReplicaEngineCPUUtilization`.

```
POST / HTTP/1.1
Host: autoscaling.us-east-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 219
X-Amz-Target: AnyScaleFrontendService.PutScalingPolicy
X-Amz-Date: 20160506T182145Z
User-Agent: aws-cli/1.10.23 Python/2.7.11 Darwin/15.4.0 botocore/1.4.8
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS
{
    "PolicyName": "myscalablepolicy",
    "ServiceNamespace": "elasticache",
    "ResourceId": "replication-group/myscalablecluster",
    "ScalableDimension": "elasticache:replication-group:Replicas",
    "PolicyType": "TargetTrackingScaling",
    "TargetTrackingScalingPolicyConfiguration": {
        "TargetValue": 40.0,
        "PredefinedMetricSpecification":
        {
            "PredefinedMetricType": "ElastiCacheReplicaEngineCPUUtilization"
        }
    }
}
```

Mengedit kebijakan penskalaan

Anda dapat mengedit kebijakan penskalaan menggunakan API AWS Management Console, Application Auto Scaling AWS CLI, atau Application Auto Scaling.

Mengedit kebijakan penskalaan menggunakan AWS Management Console

Anda hanya dapat mengedit kebijakan dengan jenis Metrik standar menggunakan AWS Management Console

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Valkey atau Redis OSS
3. Pilih klaster yang ingin Anda tambahi kebijakan (pilih nama klaster, bukan tombol di sebelah kirinya).
4. Pilih tab Kebijakan Auto Scaling.
5. Di bagian Kebijakan penskalaan, pilih tombol di samping kebijakan Auto Scaling yang ingin Anda ubah, lalu pilih Ubah.
6. Buat perubahan yang diperlukan pada kebijakan.
7. Pilih Ubah.
8. Buat perubahan pada kebijakan.
9. Pilih Ubah.

Mengedit kebijakan penskalaan menggunakan AWS CLI atau Application Auto Scaling API

Anda dapat menggunakan Application Auto Scaling API AWS CLI atau Application Auto Scaling untuk mengedit kebijakan penskalaan dengan cara yang sama seperti Anda menerapkan kebijakan penskalaan:

- Saat menggunakan API Application Auto Scaling, tentukan nama kebijakan yang ingin Anda edit dalam parameter `PolicyName`. Tentukan nilai baru untuk parameter yang ingin Anda ubah.

Lihat informasi yang lebih lengkap di Menerapkan kebijakan penskalaan ke klaster ElastiCache Valkey dan Redis OSS.

Menghapus kebijakan penskalaan

Anda dapat menghapus kebijakan penskalaan menggunakan AWS Management Console, Application Auto Scaling API AWS CLI atau Application Auto Scaling

Menghapus kebijakan penskalaan menggunakan AWS Management Console

Anda hanya dapat mengedit kebijakan dengan jenis Metrik standar menggunakan AWS Management Console

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Valkey atau Redis OSS
3. Pilih klaster yang kebijakan Auto Scaling-nya ingin Anda hapus.
4. Pilih tab Kebijakan Auto Scaling.
5. Di bagian Kebijakan penskalaan, pilih kebijakan Auto Scaling, lalu pilih Hapus.

Menghapus kebijakan penskalaan menggunakan AWS CLI atau Application Auto Scaling API

Anda dapat menggunakan Application Auto Scaling API AWS CLI atau Application Auto Scaling untuk menghapus kebijakan penskalaan dari klaster. ElastiCache

CLI

Untuk menghapus kebijakan penskalaan dari klaster Valkey dan Redis OSS Anda ElastiCache , gunakan delete-scaling-policyperintah dengan parameter berikut:

- --policy-name – Nama kebijakan penskalaan.
- --resource-id — Pengidentifikasi sumber daya untuk cluster. Untuk parameter ini, tipe sumber daya adalah ReplicationGroup dan pengidentifikasi unik adalah nama cluster, misalnya`replication-group/myscalablecluster`.
- --service-namespace – Atur nilai ini ke elasticache.
- --scalable-dimension – Atur nilai ini ke `elasticache:replication-group:Replicas`.

Example

Dalam contoh berikut, Anda menghapus kebijakan penskalaan pelacakan target yang disebut `myscalablepolicy` dari ElastiCache for Redis bernama `myscalablecluster`.

Untuk Linux, macOS, atau Unix:

```
aws application-autoscaling delete-scaling-policy \
    --policy-name myscalablepolicy \
    --resource-id replication-group/myscalablecluster \
    --service-namespace elasticache \
    --scalable-dimension elasticache:replication-group:Replicas \
```

Untuk Windows:

```
aws application-autoscaling delete-scaling-policy ^
    --policy-name myscalablepolicy ^
    --resource-id replication-group/myscalablecluster ^
    --service-namespace elasticache ^
    --scalable-dimension elasticache:replication-group:Replicas ^
```

API

Untuk menghapus kebijakan penskalaan dari klaster Valkey dan Redis OSS Anda ElastiCache , gunakan operasi DeleteScalingPolicyApplication Auto Scaling API dengan parameter berikut:

- PolicyName — Nama kebijakan penskalaan.
- ResourceId — Pengidentifikasi sumber daya untuk cluster. Untuk parameter ini, tipe sumber daya adalah ReplicationGroup dan pengidentifikasi unik adalah nama cluster, misalnya`replication-group/myscalablecluster`.
- ServiceNamespace — Tetapkan nilai ini ke elasticache.
- ScalableDimension — Tetapkan nilai ini ke`elasticache:replication-group:Replicas`.

Dalam contoh berikut, Anda menghapus kebijakan penskalaan pelacakan target yang diberi nama `myscalablepolicy` dari klaster bernama `myscalablecluster` dengan Application Auto Scaling API.

```
POST / HTTP/1.1
>>>>>>> mainline
Host: autoscaling.us-east-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 219
X-Amz-Target: AnyScaleFrontendService.DeleteScalingPolicy
X-Amz-Date: 20160506T182145Z
User-Agent: aws-cli/1.10.23 Python/2.7.11 Darwin/15.4.0 botocore/1.4.8
```

```
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS
{
    "PolicyName": "myscalablepolicy",
    "ServiceNamespace": "elasticache",
    "ResourceId": "replication-group/myscalablecluster",
    "ScalableDimension": "elasticache:replication-group:Replicas"
}
```

Gunakan AWS CloudFormation untuk kebijakan Auto Scaling

Cuplikan ini menunjukkan cara membuat tindakan terjadwal dan menerapkannya ke AWS::ElastiCache::ReplicationGroupsumber daya menggunakan sumber daya. AWS::ApplicationAutoScaling::ScalableTarget Tindakan tersebut menggunakan fungsi intrinsik Fn::Join dan Ref untuk membangun properti `ResourceId` dengan nama logis sumber daya `AWS::ElastiCache::ReplicationGroup` yang ditentukan dalam templat yang sama.

```
ScalingTarget:
   Type: 'AWS::ApplicationAutoScaling::ScalableTarget'
   Properties:
     MaxCapacity: 0
     MinCapacity: 0
     ResourceId: !Sub replication-group/${logicalName}
     ScalableDimension: 'elasticache:replication-group:Replicas'
     ServiceNamespace: elasticache
     RoleARN: !Sub "arn:aws:iam::${AWS::AccountId}:role/aws-
service-role/elasticache.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG"

  ScalingPolicy:
    Type: "AWS::ApplicationAutoScaling::ScalingPolicy"
    Properties:
      ScalingTargetId: !Ref ScalingTarget
      ServiceNamespace: elasticache
      PolicyName: testpolicy
      PolicyType: TargetTrackingScaling
      ScalableDimension: 'elasticache:replication-group:Replicas'
      TargetTrackingScalingPolicyConfiguration:
        PredefinedMetricSpecification:
          PredefinedMetricType: ElastiCacheReplicaEngineCPUUtilization
        TargetValue: 40
```

Penskalaan terjadwal

Penskalaan berdasarkan jadwal memungkinkan Anda menskalakan aplikasi sebagai respons terhadap perubahan permintaan yang dapat diprediksi. Untuk menggunakan penskalaan terjadwal, Anda membuat tindakan terjadwal, yang memberi tahu Valkey dan Redis OSS ElastiCache untuk melakukan aktivitas penskalaan pada waktu tertentu. Saat membuat tindakan terjadwal, Anda menentukan ElastiCache klaster yang ada, kapan aktivitas penskalaan harus terjadi, kapasitas minimum, dan kapasitas maksimum. Anda dapat membuat tindakan terjadwal yang menskalakan satu kali saja atau menskalakan berdasarkan jadwal berulang.

Anda hanya dapat membuat tindakan terjadwal untuk ElastiCache cluster yang sudah ada. Anda tidak dapat membuat tindakan terjadwal pada saat yang sama saat Anda membuat klaster.

Untuk informasi selengkapnya tentang terminologi untuk pembuatan, manajemen, dan penghapusan tindakan terjadwal, lihat Perintah yang umum digunakan untuk pembuatan, manajemen, dan penghapusan tindakan terjadwal

Untuk membuat tindakan terjadwal satu kali:

Mirip dengan dimensi Serpihan. Lihat Penskalaan terjadwal .

Untuk menghapus tindakan terjadwal

Mirip dengan dimensi Serpihan. Lihat Penskalaan terjadwal .

Untuk mengelola penskalaan terjadwal menggunakan AWS CLI

Gunakan APIs aplikasi-autoscaling berikut:

- put-scheduled-action
- describe-scheduled-actions
- delete-scheduled-action

Gunakan AWS CloudFormation untuk membuat kebijakan Auto Scaling

Cuplikan ini menunjukkan cara membuat tindakan terjadwal dan menerapkannya ke AWS::ElastiCache::ReplicationGroupsumber daya menggunakan sumber daya. AWS::ApplicationAutoScaling::ScalableTarget Tindakan tersebut menggunakan fungsi intrinsik Fn::Join dan Ref untuk membangun properti `ResourceId` dengan nama logis sumber daya `AWS::ElastiCache::ReplicationGroup` yang ditentukan dalam templat yang sama.

```
ScalingTarget:
   Type: 'AWS::ApplicationAutoScaling::ScalableTarget'
   Properties:
     MaxCapacity: 0
     MinCapacity: 0
     ResourceId: !Sub replication-group/${logicalName}
     ScalableDimension: 'elasticache:replication-group:Replicas'
     ServiceNamespace: elasticache
     RoleARN: !Sub "arn:aws:iam::${AWS::AccountId}:role/aws-
service-role/elasticache.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG"
     ScheduledActions:
       - EndTime: '2020-12-31T12:00:00.000Z'
         ScalableTargetAction:
           MaxCapacity: '5'
           MinCapacity: '2'
         ScheduledActionName: First
         Schedule: 'cron(0 18 * * ? *)'
```

Mengubah mode klaster

Valkey dan Redis OSS adalah database dalam memori terdistribusi yang mendukung sharding dan replikasi. ElastiCache Cluster Valkey dan Redis OSS adalah implementasi terdistribusi yang memungkinkan data dipartisi di beberapa node. Cluster ElastiCache untuk Redis OSS memiliki dua mode operasi, mode Cluster diaktifkan (CME) dan mode cluster dinonaktifkan (CMD). Di CME, mesin Valkey dan Redis OSS bekerja sebagai database terdistribusi dengan beberapa pecahan dan node, sedangkan di CMD, Valkey dan Redis OSS bekerja sebagai node tunggal.

Sebelum bermigrasi dari CMD ke CME, kondisi berikut harus dipenuhi:

> **Important**
>
> Konfigurasi mode klaster hanya dapat diubah dari mode klaster dinonaktifkan ke mode klaster diaktifkan. Konfigurasi ini tidak dapat dikembalikan.

- Klaster mungkin hanya memiliki kunci dalam basis data 0 saja.

- Aplikasi harus menggunakan klien Valkey atau Redis OSS yang mampu menggunakan protokol Cluster dan menggunakan titik akhir konfigurasi.
- Failover otomatis harus diaktifkan pada klaster dengan minimal 1 replika.
- Versi mesin minimum yang diperlukan untuk migrasi adalah Valkey 7.2 ke atas, atau Redis OSS 7.0 ke atas.

Untuk bermigrasi dari CMD ke CME, konfigurasi mode klaster harus diubah dari mode klaster dinonaktifkan ke mode klaster diaktifkan. Ini adalah prosedur dua langkah yang memastikan ketersediaan klaster selama proses migrasi.

> **Note**
>
> Anda perlu menyediakan grup parameter dengan konfigurasi klaster diaktifkan, yaitu parameter klaster diaktifkan diatur ke `yes`. Jika Anda menggunakan grup parameter default, ElastiCache untuk Redis OSS akan secara otomatis memilih grup parameter default yang sesuai dengan konfigurasi cluster-enabled. Nilai parameter yang didukung klaster diatur ke no untuk klaster CMD. Saat klaster beralih ke mode yang kompatibel, nilai parameter klaster diaktifkan akan diperbarui ke `yes` sebagai bagian dari tindakan perubahan.
> Untuk informasi selengkapnya, lihat Mengkonfigurasi parameter mesin menggunakan grup ElastiCache parameter

1. Siapkan — Buat cluster CME uji dan pastikan tumpukan Anda siap untuk bekerja dengannya. ElastiCache untuk Redis OSS tidak memiliki cara untuk memverifikasi kesiapan Anda. Untuk informasi selengkapnya, lihat Membuat cluster untuk Valkey atau Redis OSS.
2. Ubah Konfigurasi CMD Cluster yang ada ke mode cluster yang kompatibel - Dalam mode ini, akan ada pecahan tunggal yang digunakan, dan ElastiCache untuk Redis OSS akan berfungsi sebagai node tunggal tetapi juga sebagai cluster shard tunggal. Mode yang kompatibel berarti aplikasi klien dapat menggunakan kedua protokol untuk berkomunikasi dengan klaster. Dalam mode ini, aplikasi harus dikonfigurasi ulang untuk mulai menggunakan protokol Valkey atau Redis OSS Cluster dan titik akhir konfigurasi. Untuk mengubah mode cluster Valkey atau Redis OSS ke mode cluster yang kompatibel, ikuti langkah-langkah di bawah ini:

> **Note**
>
> Dalam mode kompatibel, operasi perubahan lainnya seperti penskalaan dan versi mesin tidak diizinkan untuk klaster. Selain itu, parameter (tidak termasuk`cacheParameterGroupName`) tidak dapat dimodifikasi saat menentukan parameter mode cluster dalam permintaan. ModifyReplicationGroup

a. Menggunakan AWS Management Console, melihat Mengubah grup replikasi dan mengatur mode cluster ke Kompatibel

b. Menggunakan API, lihat ModifyReplicationGroupdan perbarui `ClusterMode` parameter `kecompatible`.

c. Menggunakan AWS CLI, lihat modify-replication-groupdan perbarui `cluster-mode` parameter `kecompatible`.

Setelah mengubah mode cluster Valkey atau Redis OSS ke mode cluster yang kompatibel, DescribeReplicationGroupsAPI akan mengembalikan titik akhir konfigurasi cluster Redis OSS ElastiCache untuk. Titik akhir konfigurasi klaster adalah titik akhir tunggal yang dapat digunakan oleh aplikasi untuk terhubung ke klaster. Untuk informasi selengkapnya, lihat Menemukan titik akhir koneksi di ElastiCache.

3. Mengubah Konfigurasi Klaster ke mode klaster diaktifkan – Setelah mode klaster diatur ke mode klaster kompatibel, langkah kedua adalah mengubah konfigurasi klaster ke mode klaster diaktifkan. Dalam mode ini, serpihan tunggal sedang berjalan, dan pelanggan sekarang dapat menskalakan klaster mereka atau mengubah konfigurasi klaster lainnya.

   Untuk mengubah mode klaster ke diaktifkan, ikuti langkah di bawah:

   Sebelum memulai, pastikan klien Valkey atau Redis OSS Anda telah bermigrasi menggunakan protokol cluster dan titik akhir konfigurasi cluster tidak digunakan.

   a. Menggunakan AWS Management Console, lihat Mengubah grup replikasi dan atur mode cluster ke Enabled.

   b. Menggunakan API, lihat ModifyReplicationGroupdan perbarui `ClusterMode` parameter `keenabled`.

c. Menggunakan AWS CLI, lihat modify-replication-groupdan perbarui `cluster-mode` parameter ke`enabled`.

Setelah mengubah mode cluster menjadi diaktifkan, titik akhir akan dikonfigurasi sesuai spesifikasi cluster Valkey atau Redis OSS. DescribeReplicationGroupsAPI akan mengembalikan parameter mode cluster sebagai `enabled` dan titik akhir cluster yang sekarang tersedia untuk digunakan oleh aplikasi untuk terhubung ke cluster.

Perhatikan bahwa titik akhir klaster akan berubah setelah mode klaster diubah menjadi diaktifkan. Pastikan untuk memperbarui aplikasi Anda dengan titik akhir baru.

Anda juga dapat memilih untuk kembali ke mode klaster dinonaktifkan (CMD) dari mode klaster yang kompatibel dan mempertahankan konfigurasi asli.

Mengubah Konfigurasi Klaster ke mode klaster dinonaktifkan dari mode klaster kompatibel

1. Menggunakan AWS Management Console, melihat Mengubah grup replikasi dan mengatur mode cluster ke Dinonaktifkan
2. Menggunakan API, lihat ModifyReplicationGroupdan perbarui `ClusterMode` parameter ke`disabled`.
3. Menggunakan AWS CLI, lihat modify-replication-groupdan perbarui `cluster-mode` parameter ke`disabled`.

Setelah mengubah mode cluster menjadi dinonaktifkan, DescribeReplicationGroupsAPI akan mengembalikan parameter mode cluster sebagai`disabled`.

Replikasi lintas AWS Wilayah menggunakan datastores global

> **Note**
>
> Penyimpanan Data Global saat ini tersedia hanya untuk klaster yang dirancang sendiri.

Dengan menggunakan fitur Global Datastore, Anda dapat bekerja dengan replikasi klaster Valkey atau Redis OSS yang dikelola sepenuhnya, cepat, andal, dan aman di seluruh Wilayah. AWS Dengan menggunakan fitur ini, Anda dapat membuat klaster replika baca lintas wilayah untuk mengaktifkan pembacaan latensi rendah dan pemulihan bencana di seluruh Wilayah. AWS

Di bagian berikutnya, Anda dapat menemukan deskripsi tentang cara menggunakan penyimpanan data global.

Topik

Gambaran Umum

Setiap penyimpanan data global adalah kumpulan dari satu klaster atau lebih yang mereplikasi satu sama lain.

Penyimpanan data global mencakup klaster berikut:

- Klaster primer (aktif) – Klaster primer menerima operasi tulis yang direplikasi ke semua klaster dalam penyimpanan data global. Klaster primer juga menerima permintaan baca.
- Klaster sekunder (pasif) – Klaster sekunder hanya menerima permintaan baca dan mereplikasi pembaruan data dari klaster primer. Cluster sekunder harus berada di AWS Wilayah yang berbeda dari cluster primer.

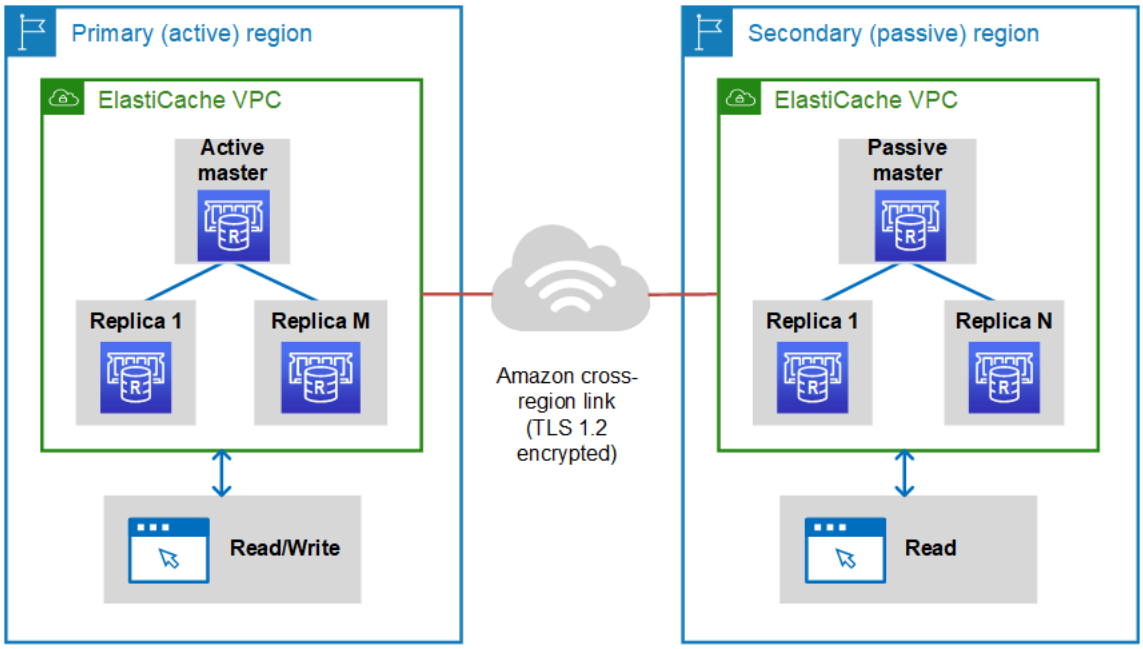
Saat Anda membuat datastore global ElastiCache untuk Valkey atau Redis OSS, data tersebut secara otomatis mereplikasi data Anda dari cluster primer ke cluster sekunder. Anda memilih AWS Wilayah di mana data Valkey atau Redis OSS harus direplikasi dan kemudian membuat cluster sekunder di Wilayah itu. AWS ElastiCache kemudian mengatur dan mengelola replikasi data asinkron otomatis antara dua cluster.

Menggunakan datastore global untuk Valkey atau Redis OSS memberikan keuntungan sebagai berikut:

- Kinerja geolokal — Dengan menyiapkan klaster replika jarak jauh di AWS Wilayah tambahan dan menyinkronkan data Anda di antara mereka, Anda dapat mengurangi latensi akses data di Wilayah tersebut. AWS Datastore global dapat membantu meningkatkan daya tanggap aplikasi Anda dengan menyajikan pembacaan geolokal latensi rendah di seluruh Wilayah. AWS

- Pemulihan bencana – Jika klaster primer di penyimpanan data global mengalami penurunan, Anda dapat mempromosikan klaster sekunder sebagai klaster primer baru Anda. Anda dapat melakukannya dengan menghubungkan ke AWS Wilayah mana pun yang berisi cluster sekunder.

Diagram berikut menunjukkan cara kerja penyimpanan data global.



Prasyarat dan batasan

Sebelum mulai menggunakan penyimpanan data global, perhatikan sejumlah hal berikut:

- Datastores global didukung di Wilayah berikut: AWS
  - Afrika - Cape Town
  - Asia Pasifik - Hong Kong, Hyderabad, Jakarta, Malaysia, Melbourne, Mumbai, Osaka, Seoul, Singapura, Sydney, Thailand, dan Tokyo
  - Kanada - Kanada Tengah dan Kanada Barat (Calgary)
  - Tiongkok - Beijing dan Ningxia
  - Eropa - Frankfurt, London, Irlandia, Milan, Paris, Spanyol, Stockholm, dan Zurich
  - AWS GovCloud-AS-Barat dan AS-Timur
  - Israel - Tel Aviv
  - Timur Tengah - Bahrain dan UEA
  - AS - Timur (Virginia N. dan Ohio) dan AS Barat (California N. dan Oregon)

  - Amerika Selatan - Meksiko (Tengah) dan São Paulo
- Semua klaster—primer dan sekunder—di penyimpanan data global Anda harus memiliki kesamaan dalam hal jumlah simpul primer, jenis simpul, versi mesin, dan jumlah serpihan (dalam kasus mode klaster diaktifkan). Setiap klaster di penyimpanan data global Anda dapat memiliki jumlah replika baca yang berbeda untuk mengakomodasi lalu lintas baca lokal untuk klaster tersebut.

  Replikasi harus diaktifkan jika Anda berencana menggunakan klaster simpul tunggal yang sudah ada.
- Datastores global didukung pada contoh ukuran besar dan di atas.
- Anda dapat mengatur replikasi untuk klaster utama dari satu AWS Wilayah ke cluster sekunder hingga dua AWS Wilayah lainnya.

  > **Note**
  >
  > Pengecualian berlaku pada Wilayah Tiongkok (Beijing) dan Tiongkok (Ningxia), di mana replikasi hanya dapat terjadi di antara kedua wilayah itu.
- Anda dapat menggunakan penyimpanan data global hanya dalam klaster VPC. Untuk informasi selengkapnya, lihat Pola Akses untuk Mengakses ElastiCache Cache di VPC Amazon. Datastores global tidak didukung saat Anda menggunakan -Classic. EC2 Untuk informasi selengkapnya, lihat EC2-Klasik di Panduan EC2 Pengguna Amazon.

  > **Note**
  >
  > Untuk saat ini, Anda tidak dapat menggunakan penyimpanan data global di Menggunakan zona lokal dengan ElastiCache.
- ElastiCache tidak mendukung autofailover dari satu AWS Wilayah ke Wilayah lainnya. Jika diperlukan, Anda dapat mempromosikan klaster sekunder secara manual. Sebagai contoh, lihat Mempromosikan klaster sekunder menjadi primer.
- Untuk melakukan bootstrap dari data yang sudah ada, gunakan klaster yang sudah ada sebagai klaster primer untuk membuat penyimpanan data global. Sebaiknya jangan menambahkan klaster yang sudah ada sebagai klaster sekunder. Proses penambahan klaster sebagai klaster sekunder akan menghapus data, yang dapat mengakibatkan hilangnya data.
- Pembaruan parameter diterapkan untuk semua klaster saat Anda mengubah grup parameter lokal dari klaster yang termasuk dalam penyimpanan data global.

- Anda dapat menskalakan klaster regional baik secara vertikal (menaikkan dan menurunkan skala) maupun horizontal (menskalakan ke dalam dan ke luar). Anda dapat menskalakan klaster dengan mengubah penyimpanan data global. Selanjutnya, semua klaster regional pada penyimpanan data global akan diskalakan tanpa gangguan. Untuk informasi selengkapnya, lihat Penskalaan ElastiCache.
- Datastores global mendukung enkripsi saat istirahat, enkripsi dalam perjalanan, dan AUTH.
- Datastores global tidak mendukung Internet Protocol versi 6 (). IPv6
- Kunci dukungan datastores global. AWS KMS Untuk informasi selengkapnya, lihat AWS Key Management Service dalam Panduan Developer AWS Key Management Service .

> **Note**
>
> Penyimpanan data global mendukung pesan pub/sub dengan ketentuan sebagai berikut:
>
> - Untuk mode cluster dinonaktifkan, didukung pub/sub sepenuhnya. Peristiwa yang diterbitkan di cluster utama AWS Wilayah primer disebarkan ke AWS Wilayah sekunder.
> - Untuk mode klaster diaktifkan, hal berikut berlaku:
>   - Untuk acara yang dipublikasikan yang tidak berada di ruang kunci, hanya pelanggan di AWS Wilayah yang sama yang menerima acara tersebut.
>   - Untuk acara keyspace yang dipublikasikan, pelanggan di semua AWS Wilayah menerima acara.

Menggunakan penyimpanan data global (konsol)

Untuk membuat penyimpanan data global menggunakan konsol, ikuti proses dua langkah ini:

1. Buat klaster primer, baik menggunakan klaster yang sudah ada atau membuat klaster baru. Mesin harus Valkey 7.2 atau lebih baru, atau Redis OSS 5.0.6 atau lebih baru.
2. Tambahkan hingga dua cluster sekunder di AWS Wilayah yang berbeda, sekali lagi menggunakan Valkey 7.2 atau yang lebih baru, atau Redis OSS 5.0.6 atau yang lebih baru.

Prosedur berikut memandu Anda tentang cara membuat datastore global untuk Valkey atau Redis OSS dan melakukan operasi lain menggunakan konsol. ElastiCache

Topik

Membuat penyimpanan data global menggunakan klaster yang sudah ada

Dalam skenario ini, Anda menggunakan klaster yang sudah ada agar berfungsi sebagai klaster primer penyimpanan data global yang baru. Anda kemudian membuat klaster sekunder hanya baca di Wilayah AWS yang berbeda. Klaster sekunder ini menerima pembaruan otomatis dan asinkron dari klaster primer.

> **Important**
>
> Cluster yang ada harus menggunakan mesin yaitu Valkey 7.2 atau yang lebih baru atau Redis OSS 5.0.6 atau yang lebih baru.

Untuk membuat penyimpanan data global menggunakan klaster yang sudah ada

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pada panel navigasi, pilih Global Datastores dan kemudian pilih Create global datastore.
3. Pada halaman pengaturan klaster utama, lakukan hal berikut:
   - Di bidang info Datastore Global, masukkan nama untuk datastore global baru.
   - (Opsional) Masukkan nilai Deskripsi.
4. Di bawah Cluster Regional, pilih Gunakan klaster regional yang ada.
5. Di bawah Cluster yang ada, pilih cluster yang ada yang ingin Anda gunakan.
6. Jangan ubah opsi berikut. Opsi tersebut sudah diisi otomatis agar sesuai dengan konfigurasi klaster primer dan Anda tidak dapat mengubahnya.

- Versi mesin
- Jenis simpul
- Grup parameter

> **Note**
>
> ElastiCache membuat otomatis grup parameter baru dari nilai grup parameter yang disediakan dan menerapkan grup parameter baru ke cluster. Gunakan grup parameter baru ini untuk mengubah parameter pada penyimpanan data global. Setiap grup parameter yang dihasilkan otomatis dikaitkan dengan hanya satu klaster dan, karena itu, hanya satu penyimpanan data global.

- Jumlah serpihan
- Enkripsi diam – Mengaktifkan enkripsi pada data yang disimpan di disk. Untuk informasi selengkapnya, lihat Enkripsi diam.

> **Note**
>
> Anda dapat menyediakan kunci enkripsi yang berbeda dengan memilih kunci AWS KMS yang Dikelola Pelanggan dan memilih kunci. Untuk informasi selengkapnya, lihat Menggunakan kunci AWS KMS yang Dikelola Pelanggan.

- Enkripsi bergerak – Mengaktifkan enkripsi data selama pengiriman. Untuk informasi selengkapnya, lihat Enkripsi bergerak. Untuk Valkey 7.2 dan seterusnya dan Redis OSS 6.0 dan seterusnya, jika Anda mengaktifkan enkripsi dalam perjalanan, Anda diminta untuk menentukan salah satu opsi Kontrol Akses berikut:
  - Tidak Ada Kontrol Akses – Ini adalah pengaturan default. Hal ini menunjukkan tidak ada batasan.
  - Daftar Kontrol Akses Grup Pengguna – Pilih grup pengguna dengan sejumlah pengguna yang ditentukan dan izin untuk operasi yang tersedia. Untuk informasi selengkapnya, lihat Mengelola Grup Pengguna dengan Konsol dan CLI.
  - AUTH Default User - Mekanisme otentikasi untuk server Valkey atau Redis OSS. Untuk informasi lebih lanjut, lihat AUTH.

7. (Opsional) Jika diperlukan, perbarui pengaturan klaster sekunder yang tersisa. Pengaturan ini sudah diisi otomatis dengan nilai yang sama dengan klaster primer, tetapi Anda dapat memperbarui nilainya untuk memenuhi persyaratan khusus untuk klaster tersebut.

   - Port
   - Jumlah replika
   - Grup subnet
   - Zona Ketersediaan Pilihan
   - Grup keamanan
   - Dikelola Pelanggan (kunci AWS KMS)
   - Token AUTH
   - Aktifkan pencadangan otomatis
   - Periode retensi cadangan
   - Periode pencadangan
   - Periode pemeliharaan
   - Topik untuk notifikasi SNS

8. Pilih Buat. Tindakan ini mengubah status penyimpanan data global menjadi Membuat. Status berubah menjadi Mengubah setelah klaster primer dikaitkan dengan penyimpanan data global dan klaster sekunder dalam status Mengaitkan.

   Setelah klaster primer dan klaster sekunder dikaitkan dengan penyimpanan data global, status berubah menjadi Tersedia. Pada tahap ini, Anda memiliki klaster primer yang menerima operasi baca dan tulis serta klaster sekunder yang menerima operasi baca yang direplikasi dari klaster primer.

   Halaman diperbarui untuk menunjukkan apakah klaster merupakan bagian dari datastore global, termasuk:

   - Penyimpanan Data Global – Nama penyimpanan data global yang menjadi asal klaster.
   - Peran Penyimpanan Data Global – Peran klaster, primer atau sekunder.

Anda dapat menambahkan hingga satu cluster sekunder tambahan di AWS Wilayah yang berbeda. Untuk informasi selengkapnya, lihat Menambahkan Wilayah ke penyimpanan data global.

Membuat penyimpanan data global baru menggunakan klaster primer baru

Jika Anda memilih untuk membuat penyimpanan data global dengan klaster baru, gunakan prosedur berikut.

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pada panel navigasi, pilih Global Datastores dan kemudian pilih Create global datastore.
3. Pada Pengaturan klaster primer, lakukan hal berikut:
    a. Untuk Mode klaster, pilih Diaktifkan atau Dinonaktifkan.
    b. Untuk info Global Datastore masukkan nilai untuk Nama. ElastiCache menggunakan akhiran untuk menghasilkan nama unik untuk datastore global. Anda dapat mencari penyimpanan data global menggunakan akhiran yang Anda tentukan di sini.
    c. (Opsional) Masukkan nilai untuk Deskripsi Penyimpanan Data Global.
4. Di Klaster regional:
    a. Untuk Wilayah, pilih AWS Wilayah yang tersedia.
    b. Pilih Buat klaster regional baru atau Gunakan klaster regional yang ada
    c. Jika Anda memilih Buat klaster regional baru, di Info klaster, masukkan nama dan deskripsi opsional klaster.
    d. Untuk Lokasi, sebaiknya Anda menerima pengaturan default untuk Multi-AZ dan Failover Otomatis.
5. Di Pengaturan klaster
    a. Untuk Versi mesin, pilih versi yang tersedia, yaitu 5.0.6 atau yang lebih baru.
    b. Untuk Port, gunakan port default, 6379. Jika Anda memiliki alasan untuk menggunakan port lain, masukkan nomor port tersebut.
    c. Untuk Grup parameter, pilih grup parameter atau buat yang baru. Grup parameter mengontrol parameter runtime dari klaster Anda. Untuk informasi selengkapnya tentang grup parameter, lihat Parameter Valkey dan Redis OSS dan Membuat grup ElastiCache parameter.

> **Note**
>
> Saat Anda memilih grup parameter untuk menetapkan nilai konfigurasi mesin, grup parameter tersebut diterapkan ke semua klaster di penyimpanan data global. Pada halaman Grup Parameter, atribut Global ya/tidak menunjukkan apakah grup parameter adalah bagian dari penyimpanan data global.

d. Untuk Jenis simpul, pilih panah bawah
( ).
Pada kotak dialog Ubah jenis simpul, pilih nilai untuk Keluarga instans untuk jenis simpul yang Anda inginkan. Kemudian pilih jenis simpul yang ingin Anda gunakan untuk klaster ini, lalu pilih Simpan.

   Untuk informasi selengkapnya, lihat Memilih ukuran simpul Anda.

   Jika Anda memilih jenis simpul r6gd, tingkatan data akan diaktifkan secara otomatis. Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.

e. Jika Anda membuat cluster Valkey atau Redis OSS (mode cluster dinonaktifkan):

   Untuk Jumlah replika, pilih jumlah replika yang Anda inginkan untuk klaster ini.

f. Jika Anda membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan):

   i. Untuk Jumlah pecahan, pilih jumlah pecahan (partisi/grup simpul) yang Anda inginkan untuk cluster Valkey atau Redis OSS (mode cluster diaktifkan) ini.

      Untuk beberapa versi Valkey atau Redis OSS (mode cluster diaktifkan), Anda dapat mengubah jumlah pecahan di cluster Anda secara dinamis:

      - Redis OSS 3.2.10 dan yang lebih baru - Jika cluster Anda menjalankan Redis OSS 3.2.10 atau versi yang lebih baru, Anda dapat mengubah jumlah pecahan di cluster Anda secara dinamis. Untuk informasi selengkapnya, lihat Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan).
      - Versi Redis OSS lainnya - Jika cluster Anda menjalankan versi Redis OSS sebelum versi 3.2.10, ada pendekatan lain. Untuk mengubah jumlah serpihan di klaster Anda, buat klaster baru dengan jumlah serpihan baru. Untuk informasi selengkapnya, lihat Melakukan pemulihan dari cadangan ke dalam cache baru.

ii. Untuk Replika per serpihan, pilih jumlah simpul replika baca yang Anda inginkan dalam setiap serpihan.

Pembatasan berikut ada untuk Valkey atau Redis OSS (mode cluster diaktifkan).

- Jika Multi-AZ diaktifkan, pastikan bahwa Anda memiliki setidaknya satu replika per serpihan.
- Replika akan berjumlah sama untuk setiap serpihan saat membuat klaster menggunakan konsol.
- Jumlah replika baca per serpihan bersifat tetap dan tidak dapat diubah. Jika Anda membutuhkan lebih banyak atau lebih sedikit replika per serpihan (API/CLI: grup simpul), Anda harus membuat klaster baru dengan jumlah replika yang baru. Untuk informasi selengkapnya, lihat Tutorial: Menyemai cluster baru yang dirancang sendiri dengan cadangan yang dibuat secara eksternal.

6. Untuk pengaturan grup Subnet, pilih subnet yang ingin Anda terapkan ke cluster ini. ElastiCache menyediakan grup IPv4 subnet default atau Anda dapat memilih untuk membuat yang baru. Untuk IPv6, Anda perlu membuat grup subnet dengan blok IPv6 CIDR. Jika Anda memilih tumpukan ganda, Anda harus memilih jenis IP Discovery, salah satu IPv6 atau IPv4.

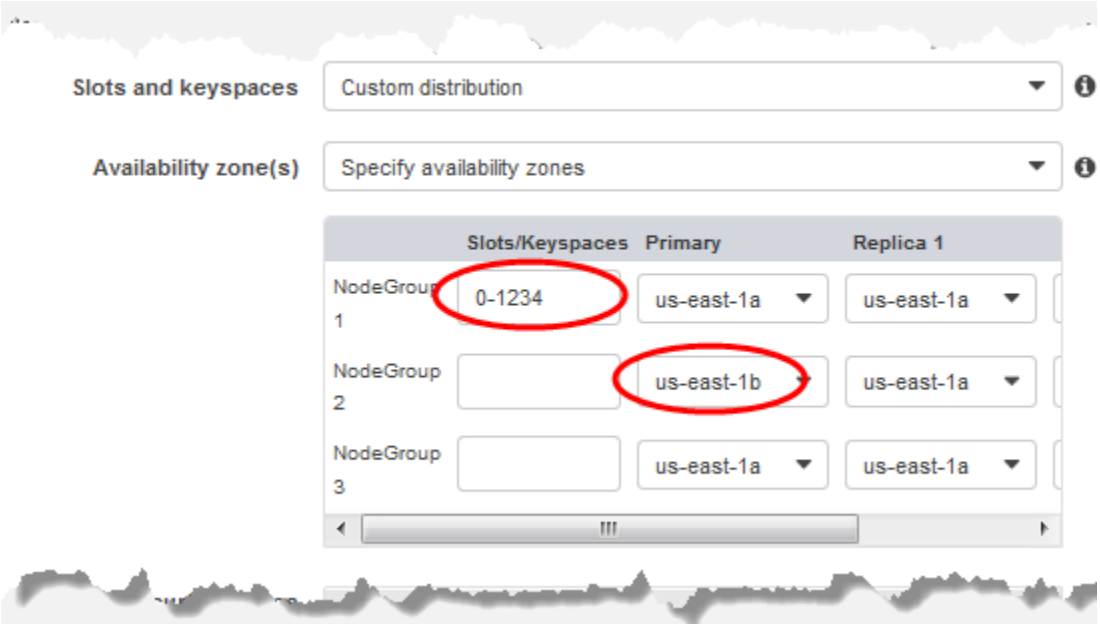
   Untuk informasi selengkapnya, lihat Membuat subnet di VPC Anda.

7. Untuk Penempatan zona ketersediaan, Anda memiliki dua opsi:

   - Tidak ada preferensi — ElastiCache memilih Availability Zone.
   - Tentukan zona ketersediaan – Anda menentukan Zona Ketersediaan untuk setiap klaster.

     Jika Anda memilih untuk menentukan Zona Ketersediaan, untuk setiap klaster di setiap serpihan, pilih Zona Ketersediaan dari daftar.

   Untuk informasi selengkapnya, lihat Memilih wilayah dan zona ketersediaan untuk ElastiCache.



Menentukan Ruang Kunci dan Zona Ketersediaan

8. Pilih Berikutnya
9. Di bawah pengaturan Advanced Valkey dan Redis OSS

   - Untuk Keamanan:

     i. Untuk mengenkripsi data Anda, Anda memiliki opsi berikut:

     - Enkripsi diam – Mengaktifkan enkripsi pada data yang disimpan di disk. Untuk informasi selengkapnya, lihat Enkripsi Diam.

       > **Note**
       >
       > Anda memiliki opsi untuk menyediakan kunci enkripsi yang berbeda dengan memilih kunci AWS KMS yang Dikelola Pelanggan dan memilih kunci. Untuk informasi selengkapnya, lihat Menggunakan kunci yang dikelola pelanggan dari AWS KMS.

     - Enkripsi bergerak – Mengaktifkan enkripsi data selama pengiriman. Untuk informasi selengkapnya, lihat Enkripsi bergerak. Untuk Valkey 7.2 dan di atasnya dan Redis OSS 6.0 ke atas, jika Anda mengaktifkan Enkripsi dalam perjalanan, Anda akan diminta untuk menentukan salah satu opsi Kontrol Akses berikut:

- Tanpa Kontrol Akses – Ini adalah pengaturan default. Opsi ini menunjukkan bahwa tidak ada batasan akses pengguna ke klaster.
- Daftar Kontrol Akses Grup Pengguna – Pilih grup pengguna dengan kumpulan pengguna tertentu yang dapat mengakses klaster. Untuk informasi selengkapnya, lihat Mengelola Grup Pengguna dengan Konsol dan CLI.
- AUTH Default User - Mekanisme otentikasi untuk server Valkey atau Redis OSS. Untuk informasi lebih lanjut, lihat AUTH.

- AUTH — Mekanisme otentikasi untuk server Valkey atau Redis OSS. Untuk informasi lebih lanjut, lihat AUTH.

> **Note**
>
> Untuk versi Redis OSS antara 3.2.6 dan seterusnya, tidak termasuk versi 3.2.10, AUTH adalah satu-satunya pilihan.

ii. Untuk Grup keamanan, pilih grup keamanan yang Anda inginkan untuk klaster ini. Grup keamanan bertindak sebagai firewall untuk mengontrol akses jaringan ke klaster Anda. Anda dapat menggunakan grup keamanan default untuk VPC Anda atau membuat yang baru.

Untuk informasi selengkapnya tentang grup keamanan, lihat Grup Keamanan untuk VPC Anda dalam Panduan Pengguna Amazon VPC.

10. Untuk pencadangan otomatis terjadwal secara berkala, pilih Aktifkan pencadangan otomatis, lalu masukkan jumlah hari yang diinginkan untuk mempertahankan cadangan otomatis sebelum dihapus secara otomatis. Jika Anda tidak ingin melakukan pencadangan otomatis terjadwal secara berkala, hapus kotak centang Aktifkan pencadangan otomatis. Apa pun pilihannya, Anda dapat membuat pencadangan secara manual kapan saja.

    Untuk informasi selengkapnya tentang pencadangan dan pemulihan, lihatMelakukan snapshot dan pemulihan.

11. (Opsional) Tentukan periode pemeliharaan. Jendela pemeliharaan adalah waktu yang biasanya satu jam setiap minggu saat ElastiCache menjadwalkan pemeliharaan sistem untuk klaster Anda. Anda dapat mengizinkan ElastiCache untuk memilih hari dan waktu untuk jendela pemeliharaan Anda (Tidak ada preferensi), atau Anda dapat memilih hari, waktu, dan durasi sendiri (Tentukan jendela pemeliharaan). Jika Anda memilih Tentukan periode pemeliharaan

dari daftar, pilih Hari mulai, Waktu mulai, dan Durasi (dalam jam) untuk periode pemeliharaan. Semua waktu menggunakan zona waktu UTC.

Untuk informasi selengkapnya, lihat Mengelola pemeliharaan ElastiCache cluster.

12. (Opsional) Untuk Log:
    - Di bagian Format log, pilih Teks atau JSON.
    - Di bawah Jenis Tujuan, pilih CloudWatch Log atau Kinesis Firehose.
    - Di bawah Tujuan log, pilih Buat baru dan masukkan nama grup CloudWatch log Log atau nama aliran Firehose Anda, atau pilih Pilih yang ada, lalu pilih nama grup CloudWatch log Log atau nama aliran Firehose Anda,
13. Untuk Tag, untuk membantu mengelola cluster dan ElastiCache sumber daya lainnya, Anda dapat menetapkan metadata Anda sendiri ke setiap sumber daya dalam bentuk tag. Untuk informasi selengkapnya, lihat Menandai sumber daya Anda ElastiCache .
14. Tinjau semua entri dan pilihan Anda, lalu lakukan koreksi yang diperlukan. Saat Anda siap, pilih Berikutnya.
15. Setelah Anda mengonfigurasi klaster di langkah sebelumnya, Anda sekarang harus mengonfigurasi detail klaster sekunder Anda.
16. Di bawah kluster Regional, pilih AWS Wilayah tempat klaster berada.
17. Di Info klaster, masukkan nama dan deskripsi opsional klaster.
18. Opsi berikut sudah diisi sebelumnya untuk menyesuaikan konfigurasi klaster primer dan tidak dapat diubah:
    - Lokasi
    - Versi mesin
    - Jenis instans
    - Jenis simpul
    - Jumlah serpihan
    - Grup parameter

> **Note**
>
> ElastiCache membuat otomatis grup parameter baru dari nilai grup parameter yang disediakan dan menerapkan grup parameter baru ke cluster. Gunakan grup parameter baru ini untuk mengubah parameter pada penyimpanan data global. Setiap grup

parameter yang dihasilkan otomatis dikaitkan dengan hanya satu klaster dan, karena itu, hanya satu penyimpanan data global.

- Enkripsi diam – Mengaktifkan enkripsi pada data yang disimpan di disk. Untuk informasi selengkapnya, lihat Enkripsi diam.

  > **Note**
  >
  > Anda dapat menyediakan kunci enkripsi yang berbeda dengan memilih kunci AWS KMS yang Dikelola Pelanggan dan memilih kunci. Untuk informasi selengkapnya, lihat Menggunakan kunci AWS KMS yang Dikelola Pelanggan.

- Enkripsi bergerak – Mengaktifkan enkripsi data selama pengiriman. Untuk informasi selengkapnya, lihat Enkripsi bergerak. Untuk Valkey 7.2 dan di atasnya dan Redis OSS 6.4 ke atas, jika Anda mengaktifkan enkripsi dalam perjalanan, Anda diminta untuk menentukan salah satu opsi Kontrol Akses berikut:
  - Tanpa Kontrol Akses – Ini adalah pengaturan default. Opsi ini menunjukkan bahwa tidak ada batasan akses pengguna ke klaster.
  - Daftar Kontrol Akses Grup Pengguna – Pilih grup pengguna dengan kelompok pengguna tertentu yang dapat mengakses klaster. Untuk informasi selengkapnya, lihat Mengelola Grup Pengguna dengan Konsol dan CLI.
  - AUTH Default User - Mekanisme otentikasi untuk server Valkey atau Redis OSS. Untuk informasi lebih lanjut, lihat AUTH.

> **Note**
>
> Untuk versi Redis OSS antara 4.0.2, saat Enkripsi dalam transit pertama kali didukung, dan 6.0.4, AUTH adalah satu-satunya pilihan.

Pengaturan klaster sekunder yang tersisa diisi otomatis dengan nilai yang sama dengan klaster primer, tetapi daftar berikut ini dapat diperbarui untuk memenuhi persyaratan khusus untuk klaster tersebut:

- Port
- Jumlah replika

- Grup subnet
- Zona Ketersediaan Pilihan
- Grup keamanan
- Dikelola Pelanggan (kunci AWS KMS)
- Token AUTH
- Aktifkan pencadangan otomatis
- Periode retensi cadangan
- Periode pencadangan
- Periode pemeliharaan
- Topik untuk notifikasi SNS

19. Pilih Buat. Tindakan ini menetapkan status penyimpanan data global menjadi Membuat. Setelah klaster primer dan klaster sekunder dikaitkan dengan penyimpanan data global, status berubah menjadi Tersedia. Anda memiliki klaster primer yang menerima operasi baca dan tulis serta klaster sekunder yang menerima operasi baca yang direplikasi dari klaster primer.

    Halaman ini juga diperbarui untuk menunjukkan apakah klaster merupakan bagian dari datastore global, termasuk yang berikut ini:

    - Penyimpanan Data Global – Nama penyimpanan data global yang menjadi asal klaster.
    - Peran Penyimpanan Data Global – Peran klaster, primer atau sekunder.

Anda dapat menambahkan hingga satu cluster sekunder tambahan di AWS Wilayah yang berbeda. Untuk informasi selengkapnya, lihat Menambahkan Wilayah ke penyimpanan data global.

Melihat detail penyimpanan data global

Anda dapat melihat detail datastores global yang ada dan juga memodifikasinya di halaman Global Datastores.

Untuk melihat detail penyimpanan data global

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pada panel navigasi, pilih Datastores Global dan kemudian pilih datastore global yang tersedia.

Anda kemudian dapat memeriksa properti penyimpanan data global berikut:

- Nama Penyimpanan Data Global: Nama penyimpanan data global
- Deskripsi: Deskripsi tentang penyimpanan data global
- Status: Opsinya meliputi:
  - Membuat
  - Mengubah
  - Tersedia
  - Menghapus
  - Hanya Primer - Status ini menunjukkan bahwa penyimpanan data global hanya berisi klaster primer. Kemungkinan semua klaster sekunder dihapus atau tidak berhasil dibuat.
- Mode Klaster: Diaktifkan atau dinonaktifkan
- Versi Mesin: Versi mesin Valkey atau Redis OSS yang menjalankan datastore global
- Jenis Simpul Instans: Jenis simpul yang digunakan untuk penyimpanan data global
- Enkripsi saat tidak digunakan: Diaktifkan atau dinonaktifkan
- Enkripsi bergerak: Diaktifkan atau dinonaktifkan
- AUTH: Entah diaktifkan atau dinonaktifkan

Anda dapat membuat perubahan berikut pada penyimpanan data global:

- Menambahkan Wilayah ke penyimpanan data global
- Menghapus Wilayah dari penyimpanan data global
- Mempromosikan klaster sekunder menjadi primer
- Mengubah penyimpanan data global

Halaman Penyimpanan Data Global juga mencantumkan klaster individual yang membentuk penyimpanan data global dan properti berikut untuk masing-masing:

- Wilayah - AWS Wilayah tempat cluster disimpan
- Peran - Primer atau sekunder
- Nama klaster: Nama klaster
- Status - Opsinya meliputi:
  - Mengaitkan - Klaster sedang dalam proses dikaitkan dengan penyimpanan data global

- Dikaitkan - Klaster sudah dikaitkan dengan penyimpanan data global
- Tidak mengaitkan - Proses menghapus klaster sekunder dari penyimpanan data global menggunakan nama penyimpanan data global. Setelah ini, cluster sekunder tidak lagi menerima pembaruan dari cluster primer tetapi tetap sebagai cluster mandiri di AWS Wilayah itu.
- Terpisah - Klaster sekunder telah dihapus dari penyimpanan data global dan sekarang menjadi klaster mandiri di Wilayah AWS .

- Kelambatan Replika Datastore Global - Menunjukkan satu nilai per AWS Wilayah sekunder di datastore global. Ini adalah lag antara simpul primer dari Wilayah sekunder dan simpul primer dari Wilayah primer. Untuk mode cluster yang diaktifkan Valkey atau Redis OSS, lag menunjukkan penundaan maksimum dalam hitungan detik di antara pecahan.

Menambahkan Wilayah ke penyimpanan data global

Anda dapat menambahkan hingga satu AWS Wilayah tambahan ke datastore global yang ada. Dalam skenario ini, Anda membuat klaster hanya-baca di AWS Wilayah terpisah yang menerima pembaruan otomatis dan asinkron dari cluster utama.

Untuk menambahkan AWS Region ke datastore global

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pada panel navigasi, pilih Datastores Global, lalu pilih datastore global yang ada.
3. Pilih Add Regional cluster, dan pilih AWS Region tempat klaster sekunder berada.
4. Di bawah Info Cluster, masukkan nilai untuk Nama dan, secara opsional, untuk Deskripsi untuk klaster.
5. Jangan ubah opsi berikut. Opsi diisi otomatis agar sesuai dengan konfigurasi klaster primer dan Anda tidak dapat mengubahnya.
    - Versi mesin
    - Jenis instans
    - Jenis simpul
    - Jumlah serpihan
    - Grup parameter

> **Note**
>
> ElastiCache membuat otomatis grup parameter baru dari nilai grup parameter yang disediakan dan menerapkan grup parameter baru ke cluster. Gunakan grup parameter baru ini untuk mengubah parameter pada penyimpanan data global. Setiap grup parameter yang dihasilkan otomatis dikaitkan dengan hanya satu klaster dan, karena itu, hanya satu penyimpanan data global.

- Enkripsi diam

> **Note**
>
> Anda dapat menyediakan kunci enkripsi yang berbeda dengan memilih kunci AWS KMS yang Dikelola Pelanggan dan memilih kunci.

- Enkripsi bergerak
- AUTENTIKASI

6. (Opsional) Perbarui pengaturan klaster sekunder yang tersisa. Pengaturan ini sudah diisi otomatis dengan nilai yang sama dengan klaster primer, tetapi Anda dapat memperbarui nilainya untuk memenuhi persyaratan khusus untuk klaster tersebut.

   - Port
   - Jumlah replika
   - Grup subnet
   - Zona Ketersediaan Pilihan
   - Grup keamanan
   - Kunci AWS KMS yang Dikelola Pelanggan)
   - Token AUTH
   - Aktifkan pencadangan otomatis
   - Periode retensi cadangan
   - Periode pencadangan
   - Periode pemeliharaan
   - Topik untuk notifikasi SNS

7. Pilih Tambahkan.

Mengubah penyimpanan data global

Anda dapat mengubah properti dari klaster regional. Hanya satu operasi perubahan dapat berlangsung pada penyimpanan data global, dengan pengecualian pada operasi mempromosikan klaster sekunder menjadi primer. Untuk informasi selengkapnya, lihat Mempromosikan klaster sekunder menjadi primer.

Untuk mengubah penyimpanan data global

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pada panel navigasi, pilih Datastores Global, dan kemudian untuk Nama Datastore Global, pilih datastore global.
3. Pilih Ubah dan pilih salah satu opsi berikut:

   - Ubah deskripsi – Memperbarui deskripsi penyimpanan data global
   - Ubah versi mesin - Hanya Valkey 7.2 dan yang lebih baru atau Redis OSS 5.0.6 dan yang lebih baru tersedia.
   - Ubah jenis simpul – Menskalakan klaster regional baik secara vertikal (menaikkan dan menurunkan skala) dan secara horizontal (menskalakan ke dalam dan ke luar). Opsi mencakup keluarga simpul R5 dan M5. Untuk informasi selengkapnya tentang jenis simpul, lihat Jenis simpul yang didukung.
   - Ubah Failover Otomatis – Aktifkan atau nonaktifkan Failover Otomatis. Saat Anda mengaktifkan failover dan node primer di cluster regional ditutup secara tak terduga, ElastiCache gagal ke salah satu replika regional. Untuk informasi selengkapnya, lihat Failover otomatis.

   Untuk cluster Valkey atau Redis OSS dengan mode cluster diaktifkan:

   - Tambahkan serpihan – Masukkan jumlah serpihan untuk ditambahkan dan, jika perlu, tentukan satu atau beberapa Zona Ketersediaan.
   - Hapus pecahan — Pilih pecahan yang akan dihapus di setiap AWS Wilayah.
   - Seimbangkan ulang serpihan – Menyeimbangkan ulang distribusi slot untuk memastikan distribusi yang merata di seluruh serpihan yang ada dalam klaster.

Untuk memodifikasi parameter datastore global, modifikasi grup parameter dari setiap cluster anggota untuk datastore global. ElastiCache menerapkan perubahan ini ke semua cluster dalam datastore global itu secara otomatis. Untuk memodifikasi grup parameter cluster tersebut, gunakan konsol Valkey atau Redis OSS atau operasi API. ModifyCacheCluster Untuk informasi selengkapnya, lihat Memodifikasi grup ElastiCache parameter. Saat Anda mengubah grup parameter dari klaster yang tercakup dalam penyimpanan data global, tindakan ini akan diterapkan ke semua klaster dalam penyimpanan data global.

Untuk mengatur ulang seluruh grup parameter atau parameter tertentu, gunakan operasi ResetCacheParameterGroupAPI.

Mempromosikan klaster sekunder menjadi primer

Jika klaster utama atau AWS Wilayah menjadi tidak tersedia atau mengalami masalah kinerja, Anda dapat mempromosikan klaster sekunder ke primer. Promosi boleh dilakukan setiap saat, bahkan saat perubahan lain sedang berlangsung. Anda juga dapat menjalankan beberapa promosi secara paralel dan penyimpanan data global pada akhirnya tetap akan diresolusi ke satu primer. Jika Anda mempromosikan beberapa cluster sekunder secara bersamaan, ElastiCache tidak menjamin mana yang akhirnya diselesaikan menjadi primer.

Untuk mempromosikan klaster sekunder menjadi primer

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pada panel navigasi, pilih Global Datastores.
3. Pilih nama penyimpanan data global untuk melihat detailnya.
4. Pilih klaster Sekunder.
5. Pilih Promosikan menjadi primer.

   Anda kemudian diminta untuk mengonfirmasi keputusan Anda dengan peringatan berikut:

   ```
   Promoting a region to primary will make the cluster in this region as
   read/writable. Are you sure you want to promote the secondary cluster
   to primary?

   The current primary cluster in primary region will become secondary
   and will stop accepting writes after this operation completes. Please
   ensure you update your application stack to direct traffic to the new
   primary region.
   ```

6. Pilih Konfirmasi jika Anda ingin melanjutkan promosi atau Batalkan jika tidak.

Jika Anda memilih untuk mengonfirmasi, penyimpanan data global Anda akan berubah status menjadi Mengubah dan tidak tersedia sampai promosi selesai.

Menghapus Wilayah dari penyimpanan data global

Anda dapat menghapus AWS Region dari datastore global dengan menggunakan prosedur berikut.

Untuk menghapus AWS Region dari datastore global

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pada panel navigasi, pilih Global Datastores.
3. Pilih penyimpanan data global.
4. Pilih Wilayah yang ingin dihapus.
5. Pilih Hapus wilayah.

   > **Note**
   >
   > Opsi ini hanya tersedia untuk klaster sekunder.

   Anda kemudian diminta untuk mengonfirmasi keputusan Anda dengan peringatan berikut: `Removing the region will remove your only available cross region replica for the primary cluster. Your primary cluster will no longer be set up for disaster recovery and improved read latency in remote region. Are you sure you want to remove the selected region from the global datastore?`
6. Pilih Konfirmasi jika Anda ingin melanjutkan promosi atau Batalkan jika tidak.

Jika Anda memilih konfirmasi, AWS Wilayah akan dihapus dan klaster sekunder tidak lagi menerima pembaruan replikasi.

Menghapus penyimpanan data global

Untuk menghapus penyimpanan data global, hapus semua klaster sekunder terlebih dahulu. Untuk informasi selengkapnya, lihat Menghapus Wilayah dari penyimpanan data global. Tindakan ini membuat penyimpanan data global berstatus hanya primer.

Untuk menghapus penyimpanan data global

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pada panel navigasi, pilih Global Datastores.
3. Di bagian Nama Penyimpanan Data Global, pilih penyimpanan data global yang ingin dihapus, lalu pilih Hapus.

   Anda kemudian diminta untuk mengonfirmasi keputusan Anda dengan peringatan berikut: `Are you sure you want to delete this Global Datastore?`
4. Pilih Hapus.

Penyimpanan data global berubah status menjadi Menghapus.

Menggunakan penyimpanan data global (CLI)

Anda dapat menggunakan AWS Command Line Interface (AWS CLI) untuk mengontrol beberapa AWS layanan dari baris perintah dan mengotomatiskannya melalui skrip. Anda dapat menggunakan AWS CLI untuk operasi ad hoc (satu kali).

Mengunduh dan mengonfigurasi AWS CLI

AWS CLI Berjalan di Windows, macOS, atau Linux. Gunakan prosedur berikut untuk mengunduh dan mengonfigurasinya.

Untuk mengunduh, menginstal, dan mengonfigurasi CLI

1. Unduh AWS CLI di halaman web antarmuka AWS baris perintah.
2. Ikuti petunjuk untuk Menginstal AWS CLI dan Mengkonfigurasi AWS CLI di Panduan Pengguna.AWS Command Line Interface

Menggunakan AWS CLI dengan datastores global

Gunakan operasi CLI berikut untuk menangani penyimpanan data global:

- create-global-replication-group

```
aws elasticache create-global-replication-group \
   --global-replication-group-id-suffix my global datastore  \
   --primary-replication-group-id sample-repl-group  \
   --global-replication-group-description an optional description of the global
 datastore
```

Amazon ElastiCache secara otomatis menerapkan awalan ke ID datastore global saat dibuat. Setiap AWS Wilayah memiliki awalan sendiri. Sebagai contoh, ID penyimpanan data global yang dibuat di Wilayah AS Barat (California Utara) dimulai dengan "virxk" dan diikuti nama akhiran yang Anda berikan. Akhiran tersebut dikombinasikan dengan awalan yang dihasilkan secara otomatis, menjamin keunikan nama penyimpanan data global di beberapa Wilayah.

Tabel berikut mencantumkan setiap AWS Wilayah dan awalan ID datastore globalnya.

Nama Wilayah/Wilayah	Awalan
Wilayah AS Timur (Ohio) `us-east-2`	`fpkhr`
Wilayah AS Timur (Virginia Utara) `us-east-1`	`ldgnf`
Wilayah AS Barat (California Utara) `us-west-1`	`virxk`
Wilayah AS Barat (Oregon) `us-west-2`	`sgaui`
Wilayah Kanada (Pusat)	`bxodz`

Nama Wilayah/Wilayah	Awalan
`ca-central-1`	
Wilayah Asia Pasifik (Mumbai) `ap-south-1`	`erpgt`
Wilayah Asia Pasifik (Tokyo) `ap-northeast-1`	`quwsw`
Wilayah Asia Pasifik (Seoul) `ap-northeast-2`	`lfqnh`
Wilayah Asia Pasifik (Osaka) `ap-northeast-3`	`nlapn`
Wilayah Asia Pasifik (Singapura) `ap-southeast-1`	`vlqxn`
Wilayah Asia Pasifik (Sydney) `ap-southeast-2`	`vbgxd`
Wilayah Eropa (Frankfurt) `eu-central-1`	`iudkw`
Wilayah Eropa (Irlandia) `eu-west-1`	`gxeiz`
Wilayah Eropa (London) `eu-west-2`	`okuqm`

Nama Wilayah/Wilayah	Awalan
Wilayah Eropa (Paris) `eu-west-3`	`fgjhi`
Wilayah Amerika Selatan (Sao Paulo) `sa-east-1`	`juxlw`
Wilayah Tiongkok (Beijing) `cn-north-1`	`emvgo`
Wilayah China (Ningxia) `cn-northwest-1`	`ckbem`
Wilayah Asia Pasifik (Hong Kong) `ap-east-1`	`knjmp`
AWS GovCloud (AS-Barat) `us-gov-west-1`	`sgwui`

- create-replication-group— Gunakan operasi ini untuk membuat cluster sekunder untuk datastore global dengan memasok nama datastore global ke parameter. `--global-replication-group-id`

```
aws elasticache create-replication-group \
  --replication-group-id secondary replication group name \
  --replication-group-description “Replication group description" \
  --global-replication-group-id global datastore name
```

Saat memanggil operasi ini dan meneruskan `--global-replication-group-id` nilai, ElastiCache akan menyimpulkan nilai dari grup replikasi utama grup replikasi global untuk parameter berikut. Jangan memasukkan nilai untuk parameter ini:

`"PrimaryClusterId",`

```
  "AutomaticFailoverEnabled",

  "NumNodeGroups",

  "CacheParameterGroupName",

  "CacheNodeType",

  "Engine",

  "EngineVersion",

  "CacheSecurityGroupNames",

  "EnableTransitEncryption",

  "AtRestEncryptionEnabled",

  "SnapshotArns",

  "SnapshotName"
```

- describe-global-replication-groups

```
aws elasticache describe-global-replication-groups \
   --global-replication-group-id my global datastore  \
   --show-member-info an optional parameter that returns a list of the primary and
 secondary clusters that make up the global datastore
```

- modify-global-replication-group

```
aws elasticache modify-global-replication-group \
   --global-replication-group-id my global datastore  \
   --automatic-failover-enabled \
   --cache-node-type node type \
   --cache-parameter-group-name parameter group name \
   --engine-version engine version \
   --–apply-immediately \
   --global-replication-group-description description
```

Redis ke OSS Valkey cross-engine upgrade untuk ElastiCache GlobalDataStore

Anda dapat memutakhirkan grup replikasi global Redis OSS yang ada ke Valkey menggunakan Konsol, API, atau CLI.

Jika Anda memiliki grup replikasi global Redis OSS yang ada, Anda dapat meningkatkan ke Valkey dengan menentukan versi mesin dan mesin baru dengan API. modify-global-replication-group

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-global-replication-group \
   --global-replication-group-id myGlobalReplGroup \
   --engine valkey \
   --apply-immediately \
   --engine-version 8.0
```

Untuk Windows:

```
aws elasticache modify-global-replication-group ^
   --global-replication-group-id myGlobalReplGroup ^
   --engine valkey ^
   --apply-immediately ^
   --engine-version 8.0
```

Jika Anda memiliki grup parameter cache khusus yang diterapkan ke grup replikasi global Redis OSS yang ingin Anda tingkatkan, Anda juga harus melewati grup parameter cache Valkey kustom dalam permintaan. Grup parameter kustom Valkey input harus memiliki nilai parameter statis Redis OSS yang sama dengan grup parameter kustom Redis OSS yang ada.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-global-replication-group \
   --global-replication-group-id myGlobalReplGroup \
   --engine valkey \
   --engine-version 8.0 \
   --apply-immediately \
   --cache-parameter-group-name myParamGroup
```

Untuk Windows:

```
aws elasticache modify-global-replication-group ^
```

```
   --global-replication-group-id myGlobalReplGroup ^
   --engine valkey ^
   --engine-version 8.0 ^
   --apply-immediately ^
   --cache-parameter-group-name myParamGroup
```

- delete-global-replication-group

```
aws elasticache delete-global-replication-group \
   --global-replication-group-id my global datastore  \
   --retain-primary-replication-group defaults to true
```

- disassociate-global-replication-group

```
aws elasticache disassociate-global-replication-group \
   --global-replication-group-id my global datastore  \
   --replication-group-id my secondary cluster  \
   --replication-group-region the AWS Region in which the secondary cluster resides
```

- failover-global-replication-group

```
aws elasticache failover-replication-group \
   --global-replication-group-id my global datastore \
   --primary-region The AWS Region of the primary cluster \
   --primary-replication-group-id  The name of the global datastore, including the
 suffix.
```

- increase-node-groups-in-global-replication-group

```
aws elasticache increase-node-groups-in-global-replication-group \
   --apply-immediately yes \
   --global-replication-group-id global-replication-group-name \
   --node-group-count 3
```

- decrease-node-groups-in-global-replication-group

```
aws elasticache decrease-node-groups-in-global-replication-group \
   --apply-immediately yes \
   --global-replication-group-id global-replication-group-name \
   --node-group-count 3
```

- rebalance-shards-in-global-replikasi-kelompok

```
aws elasticache rebalance-shards-in-global-replication-group \
   --apply-immediately yes \
   --global-replication-group-id global-replication-group-name
```

Gunakan bantuan untuk membuat daftar semua perintah yang tersedia ElastiCache untuk Valkey atau Redis OSS.

```
aws elasticache help
```

Anda juga dapat menggunakan bantuan untuk mengetahui penjelasan perintah tertentu dan mempelajari selengkapnya penggunaannya:

```
aws elasticache create-global-replication-group help
```

Ketersediaan tinggi menggunakan grup replikasi

Cluster Amazon ElastiCache Valkey dan Redis OSS simpul tunggal adalah entitas dalam memori dengan layanan perlindungan data terbatas (AOF). Jika klaster Anda gagal karena alasan apa pun, Anda akan kehilangan semua data klaster. Namun, jika Anda menjalankan mesin Valkey atau Redis OSS, Anda dapat mengelompokkan 2 hingga 6 node ke dalam cluster dengan replika di mana 1 hingga 5 node read-only berisi data replikasi dari node primer tunggal grup. read/write Dalam skenario ini, jika salah satu simpul gagal karena alasan apa pun, Anda tidak kehilangan semua data karena data direplikasikan pada satu atau beberapa simpul lainnya. Karena latensi replikasi, beberapa data mungkin hilang jika itu adalah read/write node utama yang gagal.

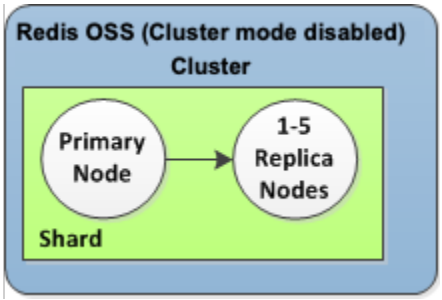
Seperti yang terlihat pada grafik berikut, struktur replikasi terkandung dalam pecahan (disebut kelompok simpul dalam API/CLI) yang terkandung dalam cluster Valkey atau Redis OSS. Cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) selalu memiliki satu pecahan. Cluster Valkey atau Redis OSS (mode cluster enabled) dapat memiliki hingga 500 pecahan dengan data cluster yang dipartisi di seluruh pecahan. Anda dapat membuat klaster dengan jumlah serpihan lebih banyak dan jumlah replika lebih sedikit dengan jumlah total hingga 90 simpul per klaster. Konfigurasi klaster ini dapat berkisar dari 90 serpihan dan 0 replika hingga 15 serpihan dan 5 replika, yang merupakan jumlah replika maksimum yang diizinkan.

Batas node atau shard dapat ditingkatkan hingga maksimum 500 per cluster dengan Valkey, dan dengan ElastiCache versi 5.0.6 atau lebih tinggi ElastiCache untuk Redis OSS. Sebagai contoh,

Anda dapat memilih untuk mengonfigurasi sebuah klaster dengan 500 simpul yang berkisar antara 83 serpihan (satu primer dan 5 replika per serpihan) dan 500 serpihan (satu primer dan tanpa replika). Pastikan alamat IP yang tersedia mencukupi untuk mengakomodasi peningkatan tersebut. Kesalahan umumnya termasuk subnet dalam grup subnet memiliki rentang CIDR yang terlalu kecil atau subnet dibagikan dan banyak digunakan oleh klaster lainnya. Untuk informasi selengkapnya, lihat Membuat grup subnet.

Untuk versi di bawah 5.0.6, batasnya adalah 250 per klaster.

Untuk meminta penambahan batas, lihat Batas Layanan AWS dan pilih jenis batas Simpul per klaster per jenis instans.



Cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) memiliki satu pecahan dan 0 hingga 5 node replika

Jika klaster dengan replika mengaktifkan Multi-AZ dan simpul primer gagal, simpul primer akan melakukan failover ke replika baca. Karena data diperbarui pada simpul replika secara asinkron, mungkin akan terjadi sejumlah kehilangan data karena latensi dalam pembaruan simpul replika. Lihat informasi yang lebih lengkap di Mengurangi Kegagalan saat Menjalankan Valkey atau Redis OSS.

Topik

- Memahami replikasi Valkey dan Redis OSS
- Replikasi: Mode Cluster Valkey dan Redis OSS Dinonaktifkan vs Diaktifkan
- Meminimalkan downtime ElastiCache dengan menggunakan Multi-AZ dengan Valkey dan Redis OSS
- Cara penerapan sinkronisasi dan pencadangan
- Membuat grup replikasi Valkey atau Redis OSS
- Melihat detail grup replikasi
- Menemukan titik akhir grup replikasi
- Mengubah grup replikasi

- Menghapus grup replikasi
- Mengubah jumlah replika
- Mempromosikan replika baca ke primer, untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan)

Memahami replikasi Valkey dan Redis OSS

Redis OSS mengimplementasikan replikasi dalam dua cara:

- Dengan pecahan tunggal yang berisi semua data cluster di setiap node—Valkey atau Redis OSS (mode cluster dinonaktifkan)
- Dengan data yang dipartisi hingga 500 pecahan — Valkey atau Redis OSS (mode cluster diaktifkan)

Setiap pecahan dalam grup replikasi memiliki simpul read/write primer tunggal dan hingga 5 node replika hanya-baca. Anda dapat membuat klaster dengan jumlah serpihan lebih banyak dan jumlah replika lebih sedikit dengan jumlah total hingga 90 simpul per klaster. Konfigurasi klaster ini dapat berkisar dari 90 serpihan dan 0 replika hingga 15 serpihan dan 5 replika, yang merupakan jumlah replika maksimum yang diizinkan.

Batas node atau shard dapat ditingkatkan hingga maksimum 500 per cluster jika versi mesin Redis OSS adalah 5.0.6 atau lebih tinggi. Sebagai contoh, Anda dapat memilih untuk mengonfigurasi sebuah klaster dengan 500 simpul yang berkisar antara 83 serpihan (satu primer dan 5 replika per serpihan) dan 500 serpihan (satu primer dan tanpa replika). Pastikan alamat IP yang tersedia mencukupi untuk mengakomodasi peningkatan tersebut. Kesalahan umumnya termasuk subnet dalam grup subnet memiliki rentang CIDR yang terlalu kecil atau subnet dibagikan dan banyak digunakan oleh klaster lainnya. Untuk informasi selengkapnya, lihat Membuat grup subnet.

Untuk versi di bawah 5.0.6, batasnya adalah 250 per klaster.

Untuk meminta penambahan batas, lihat Batas Layanan AWS dan pilih jenis batas Simpul per klaster per jenis instans.

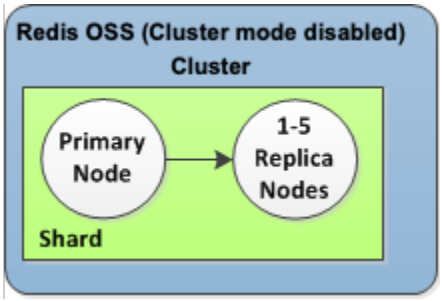
Topik

- Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)
- Valkey atau Redis OSS (mode cluster diaktifkan)

Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)

Cluster Valkey atau Redis OSS (mode cluster disabled) memiliki pecahan tunggal, di dalamnya terdapat kumpulan node; satu node primer dan hingga lima read/write node replika read-only sekunder. Setiap replika baca memelihara salinan data dari simpul primer klaster. Mekanisme

replikasi asinkron digunakan untuk menjaga sinkronisasi replika baca dengan primer. Aplikasi dapat membaca dari simpul apa pun dalam klaster. Aplikasi hanya dapat menulis ke simpul primer. Replika baca meningkatkan throughput baca dan menjaga agar data tidak hilang jika terjadi kegagalan simpul.



Valkey atau Redis OSS (mode cluster dinonaktifkan) cluster dengan pecahan tunggal dan node replika

Anda dapat menggunakan klaster Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan node replika untuk menskalakan solusi Anda ElastiCache untuk menangani aplikasi yang intensif baca atau untuk mendukung sejumlah besar klien yang secara bersamaan membaca dari cluster yang sama.

Semua node dalam cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) harus berada di wilayah yang sama.

Saat Anda menambahkan replika baca ke klaster, semua data dari simpul primer akan disalin ke simpul baru. Setelah tindakan tersebut, setiap kali data ditulis ke simpul primer, perubahan akan disebarkan secara asinkron ke semua replika baca.

Untuk meningkatkan toleransi kesalahan dan mengurangi waktu henti tulis, aktifkan Multi-AZ dengan Failover Otomatis untuk cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) Anda dengan replika. Untuk informasi selengkapnya, lihat Meminimalkan downtime ElastiCache dengan menggunakan Multi-AZ dengan Valkey dan Redis OSS.

Anda dapat mengubah peran node dalam klaster Valkey atau Redis OSS (mode cluster dinonaktifkan), dengan peran utama dan salah satu replika bertukar peran. Sebaiknya pertimbangkan untuk melakukan tindakan ini untuk menyesuaikan performa. Misalnya, dengan aplikasi web yang memiliki operasi tulis yang berat, Anda dapat memilih simpul dengan latensi jaringan terendah. Untuk informasi selengkapnya, lihat Mempromosikan replika baca ke primer, untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan).

Valkey atau Redis OSS (mode cluster diaktifkan)

Cluster Valkey atau Redis OSS (mode cluster enabled) terdiri dari 1 hingga 500 pecahan (API/CLI: grup node). Setiap serpihan memiliki satu simpul primer dan hingga lima simpul replika hanya baca. Konfigurasi dapat berkisar dari 90 serpihan dan 0 replika hingga 15 serpihan dan 5 replika, yang merupakan jumlah replika maksimum yang diizinkan.

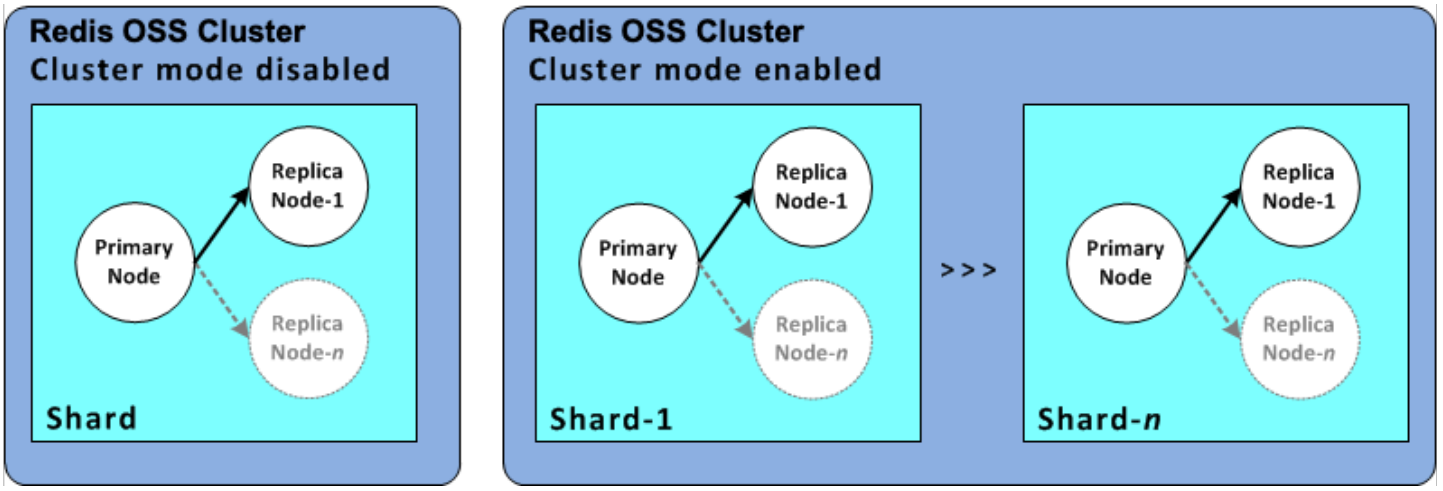
Batas node atau shard dapat ditingkatkan hingga maksimum 500 per cluster jika versi mesin Valkey 7.2 dan lebih tinggi, atau Redis OSS 5.0.6 dan lebih tinggi. Sebagai contoh, Anda dapat memilih untuk mengonfigurasi sebuah klaster dengan 500 simpul yang berkisar antara 83 serpihan (satu primer dan 5 replika per serpihan) dan 500 serpihan (satu primer dan tanpa replika). Pastikan alamat IP yang tersedia mencukupi untuk mengakomodasi peningkatan tersebut. Kesalahan umumnya termasuk subnet dalam grup subnet memiliki rentang CIDR yang terlalu kecil atau subnet dibagikan dan banyak digunakan oleh klaster lainnya. Untuk informasi selengkapnya, lihat Membuat grup subnet.

Untuk versi di bawah 5.0.6, batasnya adalah 250 per klaster.

Untuk meminta penambahan batas, lihat Batas Layanan AWS dan pilih jenis batas Simpul per klaster per jenis instans.

Setiap replika baca dalam serpihan mempertahankan salinan data dari primer pada serpihan ini. Mekanisme replikasi asinkron digunakan untuk menjaga sinkronisasi replika baca dengan primer. Aplikasi dapat membaca dari simpul apa pun dalam klaster. Aplikasi hanya dapat menulis ke simpul primer. Replika baca meningkatkan skalabilitas baca dan mencegah kehilangan data. Data dipartisi di seluruh pecahan dalam cluster Valkey atau Redis OSS (mode cluster enabled).

Aplikasi menggunakan titik akhir konfigurasi cluster Valkey atau Redis OSS (mode cluster enabled) untuk terhubung dengan node di cluster. Untuk informasi selengkapnya, lihat Menemukan titik akhir koneksi di ElastiCache.



Valkey atau Redis OSS (mode cluster diaktifkan) cluster dengan beberapa pecahan dan node replika

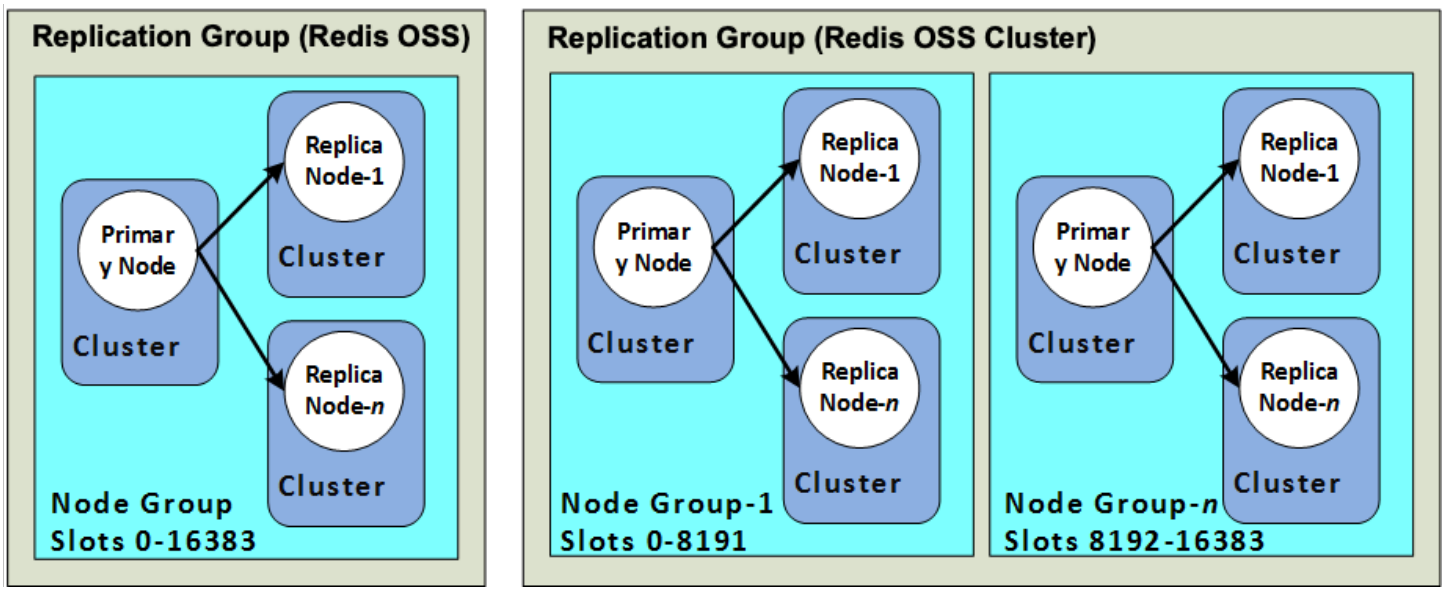
Semua node dalam cluster Valkey atau Redis OSS (mode cluster enabled) harus berada di wilayah yang sama. Untuk meningkatkan toleransi kesalahan, Anda dapat menyediakan replika primer dan replika baca pada beberapa Zona Ketersediaan dalam wilayah tersebut.

Saat ini, fitur Valkey atau Redis OSS (mode cluster enabled) memiliki beberapa keterbatasan.

- Anda tidak dapat secara manual mempromosikan salah satu simpul replika menjadi primer.

Replikasi: Mode Cluster Valkey dan Redis OSS Dinonaktifkan vs Diaktifkan

Dimulai dengan Valkey 7.2 dan Redis OSS versi 3.2, Anda memiliki kemampuan untuk membuat salah satu dari dua jenis cluster yang berbeda (API/CLI: replication groups). A Valkey or Redis OSS (cluster mode disabled) cluster always has a single shard (API/CLI: grup node) dengan hingga 5 node replika baca. Cluster Valkey atau Redis OSS (mode cluster enabled) memiliki hingga 500 pecahan dengan 1 hingga 5 node replika baca di masing-masing.



Valkey atau Redis OSS (mode cluster dinonaktifkan), dan Valkey atau Redis OSS (mode cluster diaktifkan) cluster

Tabel berikut merangkum perbedaan penting antara Valkey atau Redis OSS (mode cluster dinonaktifkan) dan Valkey atau Redis OSS (mode cluster diaktifkan) cluster.

Membandingkan Valkey atau Redis OSS (mode cluster dinonaktifkan) dan Valkey atau Redis OSS (mode cluster diaktifkan) Cluster

Fitur	Valkey atau Redis OSS (mode cluster dinonaktifkan)	Valkey atau Redis OSS (mode cluster diaktifkan)
Dapat diubah	Ya. Mendukung penambahan dan penghapusan simpul replika, dan peningkatan skala jenis simpul.	Terbatas. Untuk informasi selengkapnya, lihat Manajemen Versi untuk ElastiCache dan Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan).
Pembuatan Partisi Data	Tidak	Ya
Serpihan	1	1 hingga 500
Replika baca	0 hingga 5	0 hingga 5 per serpihan.

Fitur	Valkey atau Redis OSS (mode cluster dinonaktifkan)	Valkey atau Redis OSS (mode cluster diaktifkan)
	Important Jika replika tidak tersedia dan simpul gagal, Anda akan mengalami kehilangan data total.	**Important** Jika replika tidak tersedia dan satu simpul gagal, Anda akan mengalami kehilangan semua data di serpihan tersebut.
Multi-AZ	Ya, dengan setidaknya 1 replika. Tidak wajib. Aktif secara default.	Ya Opsional. Aktif secara default.
Snapshot (Cadangan)	Ya, membuat file .rdb tunggal.	Ya, membuat file .rdb yang unik untuk setiap serpihan.
Memulihkan	Ya, menggunakan satu file.rdb dari cluster Valkey atau Redis OSS (mode cluster dinonakti fkan).	Ya, menggunakan file.rdb dari Valkey atau Redis OSS (mode cluster dinonaktifkan) atau cluster Valkey atau Redis OSS (mode cluster diaktifkan).
Didukung oleh	Semua versi Valkey dan Redis OSS	Semua versi Valkey, dan Redis OSS 3.2 dan berikut
Mesin dapat ditingkatkan	Ya, dengan beberapa batasan. Untuk informasi selengkapnya, lihat Manajemen Versi untuk ElastiCache.	Ya, dengan beberapa batasan. Untuk informasi selengkapnya, lihat Manajemen Versi untuk ElastiCache.

Fitur	Valkey atau Redis OSS (mode cluster dinonaktifkan)	Valkey atau Redis OSS (mode cluster diaktifkan)
Enkripsi	Versi 3.2.6 (dijadwalkan untuk EOL, lihat jadwal akhir masa pakai versi Redis OSS) dan 4.0.10 dan yang lebih baru.	Versi 3.2.6 (dijadwalkan untuk EOL, lihat jadwal akhir masa pakai versi Redis OSS) dan 4.0.10 dan yang lebih baru.
Memenuhi Syarat HIPAA	Versi 3.2.6 (dijadwalkan untuk EOL, lihat jadwal akhir masa pakai versi Redis OSS) dan 4.0.10 dan yang lebih baru.	Versi 3.2.6 (dijadwalkan untuk EOL, lihat jadwal akhir masa pakai versi Redis OSS) dan 4.0.10 dan yang lebih baru.
Mematuhi Standar PCI DSS	Versi 3.2.6 (dijadwalkan untuk EOL, lihat jadwal akhir masa pakai versi Redis OSS) dan 4.0.10 dan yang lebih baru.	Versi 3.2.6 (dijadwalkan untuk EOL, lihat jadwal akhir masa pakai versi Redis OSS) dan 4.0.10 dan yang lebih baru.
Resharding online	N/A	Versi 3.2.10 (dijadwalkan untuk EOL, lihat versi Redis OSS jadwal akhir hidup) dan yang lebih baru.

Sebaiknya memilih yang mana?

Saat memilih antara Valkey atau Redis OSS (mode cluster dinonaktifkan) atau Valkey atau Redis OSS (mode cluster diaktifkan), pertimbangkan faktor-faktor berikut:

- Penskalaan vs. pembuatan partisi – Bisnis membutuhkan perubahan. Anda perlu menyediakan baik untuk permintaan puncak atau melakukan penskalaan mengikuti perubahan permintaan. Valkey atau Redis OSS (mode cluster dinonaktifkan) mendukung penskalaan. Anda dapat menskalakan kapasitas baca dengan menambahkan atau menghapus simpul replika, atau Anda dapat menskalakan kapasitas dengan menaikkan skala ke jenis simpul yang lebih besar. Kedua operasi ini membutuhkan waktu. Untuk informasi lebih lanjut, lihat Menskalakan node replika untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan).

Valkey atau Redis OSS (mode cluster diaktifkan) mendukung partisi data Anda di hingga 500 grup node. Anda dapat mengubah jumlah serpihan secara dinamis mengikuti perubahan kebutuhan bisnis Anda. Salah satu keuntungan dari pembuatan partisi adalah beban disebarkan ke lebih banyak titik akhir, sehingga mengurangi kemacetan akses selama permintaan puncak. Selain itu, Anda dapat mengakomodasi set data yang lebih besar karena data dapat disebarkan ke beberapa server. Untuk informasi tentang penskalaan partisi Anda, lihat. Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan)

- Ukuran node v. jumlah node — Karena cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) hanya memiliki satu pecahan, tipe node harus cukup besar untuk mengakomodasi semua data cluster ditambah overhead yang diperlukan. Di sisi lain, karena Anda dapat mempartisi data Anda di beberapa pecahan saat menggunakan cluster Valkey atau Redis OSS (mode cluster enabled), tipe node bisa lebih kecil, meskipun Anda membutuhkan lebih banyak dari mereka. Untuk informasi selengkapnya, lihat Memilih ukuran simpul Anda.

- Membaca v. menulis - Jika beban utama pada cluster Anda adalah aplikasi yang membaca data, Anda dapat menskalakan cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan menambahkan dan menghapus replika baca. Namun, terdapat jumlah maksimum sebanyak 5 replika baca. Jika beban pada cluster Anda berat tulis, Anda bisa mendapatkan keuntungan dari titik akhir penulisan tambahan dari cluster Valkey atau Redis OSS (mode cluster enabled) dengan beberapa pecahan.

Apa pun jenis klaster yang Anda pilih untuk diterapkan, pastikan untuk memilih jenis simpul yang memadai untuk kebutuhan Anda saat ini dan pada masa mendatang.

Meminimalkan downtime ElastiCache dengan menggunakan Multi-AZ dengan Valkey dan Redis OSS

Ada sejumlah contoh di mana ElastiCache untuk Valkey dan Redis OSS mungkin perlu mengganti node primer; ini termasuk jenis pemeliharaan terencana tertentu dan kejadian yang tidak mungkin dari node primer atau kegagalan Availability Zone.

Penggantian ini mengakibatkan waktu henti pada klaster, tetapi jika Multi-AZ diaktifkan, waktu henti dapat dikurangi. Peran simpul primer akan secara otomatis melakukan failover ke salah satu replika baca. Tidak perlu membuat dan menyediakan simpul utama baru, karena ElastiCache akan menangani ini secara transparan. Failover dan promosi replika ini memastikan Anda dapat melanjutkan penulisan ke primer baru segera setelah promosi selesai.

ElastiCache juga menyebarkan nama Domain Name Service (DNS) dari replika yang dipromosikan. Hal ini dilakukan karena jika aplikasi Anda menulis ke titik akhir primer, tidak perlu ada perubahan titik akhir dalam aplikasi Anda. Jika Anda membaca dari titik akhir individual, pastikan untuk mengubah titik akhir baca dari replika yang dipromosikan menjadi primer ke titik akhir dari replika yang baru.

Jika penggantian simpul terencana dimulai karena pembaruan pemeliharaan atau pembaruan mandiri, perhatikan hal berikut:

- Untuk cluster Valkey dan Redis OSS, penggantian node yang direncanakan selesai sementara cluster melayani permintaan tulis yang masuk.
- Untuk cluster yang dinonaktifkan mode cluster Valkey dan Redis OSS dengan multi-AZ diaktifkan yang berjalan pada mesin 5.0.6 atau yang lebih baru, penggantian node yang direncanakan selesai sementara cluster melayani permintaan tulis yang masuk.
- Untuk cluster yang dinonaktifkan mode cluster Valkey dan Redis OSS dengan multi-AZ diaktifkan yang berjalan pada mesin 4.0.10 atau sebelumnya, Anda mungkin melihat gangguan penulisan singkat yang terkait dengan pembaruan DNS. Gangguan ini mungkin memakan waktu hingga beberapa detik. Proses ini jauh lebih cepat daripada membuat dan menetapkan primer baru, yang akan terjadi jika Anda tidak mengaktifkan Multi-AZ.

Anda dapat mengaktifkan Multi-AZ menggunakan ElastiCache Management Console, the AWS CLI, atau ElastiCache API.

Mengaktifkan ElastiCache Multi-AZ di klaster Valkey atau Redis OSS Anda (di API dan CLI, grup replikasi) meningkatkan toleransi kesalahan Anda. Hal ini berlaku terutama dalam kasus di mana klaster read/write utama cluster Anda menjadi tidak dapat dijangkau atau gagal karena alasan apa

pun. Multi-AZ hanya didukung pada cluster Valkey dan Redis OSS dengan lebih dari satu node di setiap shard.

Topik

- Mengaktifkan Multi-AZ
- Skenario kegagalan dengan respons Multi-AZ
- Menguji failover otomatis
- Batasan pada Multi-AZ

Mengaktifkan Multi-AZ

Anda dapat mengaktifkan Multi-AZ saat membuat atau memodifikasi cluster (API atau CLI, grup replikasi) menggunakan ElastiCache konsol AWS CLI,, atau API. ElastiCache

Anda dapat mengaktifkan Multi-AZ hanya pada cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) yang memiliki setidaknya satu replika baca yang tersedia. Klaster tanpa replika baca tidak menyediakan ketersediaan tinggi atau toleransi kesalahan. Untuk informasi tentang cara membuat klaster dengan replikasi, lihat Membuat grup replikasi Valkey atau Redis OSS. Untuk informasi tentang cara menambahkan replika baca ke klaster dengan replikasi, lihat Menambahkan replika baca untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan).

Topik

- Mengaktifkan Multi-AZ (Konsol)
- Mengaktifkan Multi-AZ (AWS CLI)
- Mengaktifkan Multi-AZ (API) ElastiCache

Mengaktifkan Multi-AZ (Konsol)

Anda dapat mengaktifkan Multi-AZ menggunakan ElastiCache konsol saat Anda membuat cluster Valkey atau Redis OSS baru atau dengan memodifikasi cluster yang ada dengan replikasi.

Multi-AZ diaktifkan secara default pada cluster Valkey atau Redis OSS (mode cluster enabled).

> **Important**
>
> ElastiCache akan secara otomatis mengaktifkan Multi-AZ hanya jika cluster berisi setidaknya satu replika di Availability Zone yang berbeda dari yang utama di semua pecahan.

Mengaktifkan Multi-AZ saat membuat cluster menggunakan konsol ElastiCache

Lihat informasi selengkapnya tentang prosedur ini, lihat Membuat cluster Valkey (mode cluster dinonaktifkan) (Konsol). Pastikan untuk memiliki satu atau beberapa replika dan mengaktifkan Multi-AZ.

Mengaktifkan Multi-AZ pada klaster yang sudah ada (Konsol)

Untuk informasi selengkapnya tentang proses ini, lihat Mengubah Klaster Menggunakan ElastiCache AWS Management Console.

Mengaktifkan Multi-AZ (AWS CLI)

Contoh kode berikut menggunakan AWS CLI untuk mengaktifkan Multi-AZ untuk grup replikasi. `redis12`

> **Important**
>
> Replikasi grup `redis12` harus sudah ada dan memiliki setidaknya satu replika baca yang tersedia.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
    --replication-group-id redis12 \
    --automatic-failover-enabled \
    --multi-az-enabled \
    --apply-immediately
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
    --replication-group-id redis12 ^
    --automatic-failover-enabled ^
    --multi-az-enabled ^
    --apply-immediately
```

Output JSON dari perintah ini akan terlihat seperti berikut.

```
{
    "ReplicationGroup": {
```

```
        "Status": "modifying",
        "Description": "One shard, two nodes",
        "NodeGroups": [
            {
                "Status": "modifying",
                "NodeGroupMembers": [
                    {
                        "CurrentRole": "primary",
                        "PreferredAvailabilityZone": "us-west-2b",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Port": 6379,
                            "Address":
"redis12-001.v5r9dc.0001.usw2.cache.amazonaws.com"
                        },
                        "CacheClusterId": "redis12-001"
                    },
                    {
                        "CurrentRole": "replica",
                        "PreferredAvailabilityZone": "us-west-2a",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Port": 6379,
                            "Address":
"redis12-002.v5r9dc.0001.usw2.cache.amazonaws.com"
                        },
                        "CacheClusterId": "redis12-002"
                    }
                ],
                "NodeGroupId": "0001",
                "PrimaryEndpoint": {
                    "Port": 6379,
                    "Address": "redis12.v5r9dc.ng.0001.usw2.cache.amazonaws.com"
                }
            }
        ],
        "ReplicationGroupId": "redis12",
        "SnapshotRetentionLimit": 1,
        "AutomaticFailover": "enabling",
        "MultiAZ": "enabled",
        "SnapshotWindow": "07:00-08:00",
        "SnapshottingClusterId": "redis12-002",
        "MemberClusters": [
            "redis12-001",
```

```
            "redis12-002"
        ],
        "PendingModifiedValues": {}
    }
}
```

Untuk informasi selengkapnya, lihat topik ini dalam Referensi Perintah AWS CLI :

- create-cache-cluster
- create-replication-group
- modify-replication-groupdalam AWS CLI Command Reference.

Mengaktifkan Multi-AZ (API) ElastiCache

Contoh kode berikut menggunakan ElastiCache API untuk mengaktifkan Multi-AZ untuk grup replikasi. `redis12`

> **Note**
>
> Untuk menggunakan contoh ini, grup replikasi `redis12` harus sudah ada dan memiliki setidaknya satu replika baca dalam keadaan tersedia.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=ModifyReplicationGroup
    &ApplyImmediately=true
    &AutoFailover=true
    &MultiAZEnabled=true
    &ReplicationGroupId=redis12
    &Version=2015-02-02
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20140401T192317Z
    &X-Amz-Credential=<credential>
```

Untuk informasi lain, lihat topik ini di Referensi API ElastiCache :

- CreateCacheCluster
- CreateReplicationGroup

- ModifyReplicationGroup

Skenario kegagalan dengan respons Multi-AZ

Sebelum pengenalan Multi-AZ, ElastiCache mendeteksi dan mengganti node gagal cluster dengan membuat ulang dan reprovisioning node gagal. Jika Anda mengaktifkan Multi-AZ, simpul primer yang gagal akan melakukan failover ke replika dengan lag replikasi terkecil. Replika yang dipilih akan dipromosikan secara otomatis menjadi primer, yang jauh lebih cepat daripada membuat dan menetapkan kembali simpul primer baru. Proses ini biasanya memakan waktu hanya beberapa detik hingga Anda dapat menulis lagi ke klaster.

Ketika Multi-AZ diaktifkan, ElastiCache terus memonitor keadaan node utama. Jika simpul primer gagal, salah satu tindakan berikut akan dilakukan bergantung pada jenis kegagalan.

Topik

Skenario kegagalan ketika hanya simpul primer yang gagal

Jika hanya simpul primer yang gagal, replika baca dengan lag replikasi terkecil akan dipromosikan menjadi primer. Replika baca pengganti kemudian dibuat dan ditetapkan di Zona Ketersediaan yang sama dengan primer yang gagal.

Ketika hanya node utama yang gagal, ElastiCache Multi-AZ melakukan hal berikut:

1. Simpul primer yang gagal akan dibuat offline.
2. Replika baca dengan lag replikasi terkecil akan dipromosikan menjadi primer.

   Proses tulis dapat dilanjutkan segera setelah proses promosi selesai, biasanya hanya beberapa detik. Jika aplikasi Anda menulis ke titik akhir primer, Anda tidak perlu mengubah titik akhir untuk tulis atau baca. ElastiCache menyebarkan nama DNS dari replika yang dipromosikan.
3. Replika baca pengganti diluncurkan dan ditetapkan.

   Replika baca pengganti diluncurkan pada Zona Ketersediaan yang sama dengan simpul primer yang gagal sehingga distribusi simpul tetap terpelihara.
4. Replika melakukan sinkronisasi dengan simpul primer yang baru.

Setelah replika baru tersedia, perhatikan efeknya berikut ini:

- Titik akhir primer – Anda tidak perlu membuat perubahan apa pun pada aplikasi Anda, karena nama DNS dari simpul primer baru disebarkan ke titik akhir primer.
- Titik akhir baca – Titik akhir pembaca diperbarui secara otomatis untuk mengarah ke simpul replika yang baru.

Untuk informasi tentang cara menemukan titik akhir klaster, lihat topik berikut:

- Menemukan Titik Akhir Cluster Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol)
- Menemukan Titik Akhir untuk Grup Replikasi Valkey atau Redis OSS ()AWS CLI
- Menemukan Titik Akhir untuk Grup Replikasi Valkey atau Redis OSS (API) ElastiCache

Skenario kegagalan ketika simpul primer dan beberapa replika baca gagal

Jika primer dan setidaknya satu replika baca gagal, replika yang tersedia dengan lag replikasi terkecil akan dipromosikan menjadi klaster primer. Replika baca yang baru juga dibuat dan ditetapkan di Zona Ketersediaan yang sama dengan simpul yang gagal dan replika yang dipromosikan menjadi primer.

Ketika node utama dan beberapa replika baca gagal, ElastiCache Multi-AZ melakukan hal berikut:

1. Simpul primer yang gagal dan replika baca yang gagal akan dibuat offline.
2. Replika yang tersedia dengan lag replikasi terkecil akan dipromosikan menjadi simpul primer.

   Operasi tulis dapat dilanjutkan segera setelah proses promosi selesai, biasanya hanya beberapa detik. Jika aplikasi Anda menulis ke titik akhir utama, tidak perlu mengubah titik akhir untuk menulis. ElastiCache menyebarkan nama DNS dari replika yang dipromosikan.
3. Replika pengganti dibuat dan ditetapkan.

   Replika pengganti dibuat di Zona Ketersediaan dari simpul yang gagal sehingga distribusi simpul tetap terpelihara.
4. Semua klaster melakukan sinkronisasi dengan simpul primer baru.

Lakukan perubahan berikut pada aplikasi Anda setelah simpul yang baru tersedia:

- Titik akhir primer – Jangan membuat perubahan apa pun pada aplikasi Anda. Nama DNS dari simpul primer baru disebarkan ke titik akhir primer.
- Titik akhir baca – Titik akhir baca diperbarui secara otomatis untuk mengarah ke simpul replika yang baru.

Untuk informasi tentang menemukan titik akhir grup replikasi, lihat topik berikut:

- Menemukan Titik Akhir Cluster Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol)
- Menemukan Titik Akhir untuk Grup Replikasi Valkey atau Redis OSS ()AWS CLI
- Menemukan Titik Akhir untuk Grup Replikasi Valkey atau Redis OSS (API) ElastiCache

Skenario kegagalan ketika seluruh klaster gagal

Jika semuanya gagal, semua simpul dibuat kembali dan ditetapkan pada Zona Ketersediaan yang sama dengan simpul asli.

Dalam skenario ini, semua data dalam klaster akan hilang karena kegagalan dari setiap simpul dalam klaster. Kasus ini jarang terjadi.

Ketika seluruh cluster gagal, ElastiCache Multi-AZ melakukan hal berikut:

1. Simpul primer dan replika baca yang gagal akan dibuat offline.
2. Simpul primer pengganti dibuat dan ditetapkan.
3. Replika pengganti dibuat dan ditetapkan.

   Penggantinya dibuat di Zona Ketersediaan dari simpul yang gagal sehingga distribusi simpul tetap dipertahankan.

   Karena seluruh klaster gagal, data menjadi hilang dan semua simpul baru akan melakukan cold start.

Karena setiap simpul pengganti memiliki titik akhir yang sama dengan simpul yang digantikannya, Anda tidak perlu membuat perubahan titik akhir pada aplikasi Anda.

Untuk informasi tentang menemukan titik akhir grup replikasi, lihat topik berikut:

- Menemukan Titik Akhir Cluster Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol)
- Menemukan Titik Akhir untuk Grup Replikasi Valkey atau Redis OSS ()AWS CLI
- Menemukan Titik Akhir untuk Grup Replikasi Valkey atau Redis OSS (API) ElastiCache

Sebaiknya buat simpul primer dan replika baca di Zona Ketersediaan yang berbeda untuk meningkatkan tingkat toleransi kesalahan Anda.

Menguji failover otomatis

Setelah mengaktifkan failover otomatis, Anda dapat mengujinya menggunakan ElastiCache konsol, file AWS CLI, dan ElastiCache API.

Saat menguji, perhatikan hal berikut:

- Anda dapat menggunakan operasi ini untuk menguji failover otomatis hingga 15 pecahan (disebut grup simpul di ElastiCache API dan AWS CLI) dalam periode 24 jam bergulir apa pun.
- Jika Anda memanggil operasi ini pada serpihan di klaster yang berbeda (disebut grup replikasi pada API dan CLI), Anda dapat membuat panggilan secara konkuren.
- Dalam beberapa kasus, Anda mungkin memanggil operasi ini beberapa kali pada pecahan yang berbeda di grup replikasi Valkey atau Redis OSS (mode cluster enabled) yang sama. Dalam kasus tersebut, penggantian simpul pertama harus selesai sebelum panggilan berikutnya dapat dibuat.
- Untuk menentukan apakah penggantian node selesai, periksa peristiwa menggunakan ElastiCache konsol Amazon, API AWS CLI, atau ElastiCache API. Cari peristiwa yang berkaitan dengan failover otomatis, yang tercantum berikut ini dalam urutan kemungkinan kemunculannya:

  1. Pesan grup replikasi: `Test Failover API called for node group <node-group-id>`
  2. Pesan klaster cache: `Failover from primary node <primary-node-id> to replica node <node-id> completed`
  3. Pesan grup replikasi: `Failover from primary node <primary-node-id> to replica node <node-id> completed`
  4. Pesan klaster cache: `Recovering cache nodes <node-id>`
  5. Pesan klaster cache: `Finished recovery for cache nodes <node-id>`

  Untuk informasi lain, lihat hal berikut:

  - Melihat ElastiCache acara di Panduan Pengguna ElastiCache
  - DescribeEvents di Referensi API ElastiCache
  - describe-events dalam Referensi Perintah AWS CLI .
- API ini dirancang untuk menguji perilaku aplikasi Anda jika terjadi ElastiCache failover. Hal ini tidak dirancang untuk menjadi alat operasional untuk memulai failover guna mengatasi masalah dengan klaster. Selain itu, dalam kondisi tertentu seperti peristiwa operasional skala besar, AWS dapat memblokir API ini.

Topik

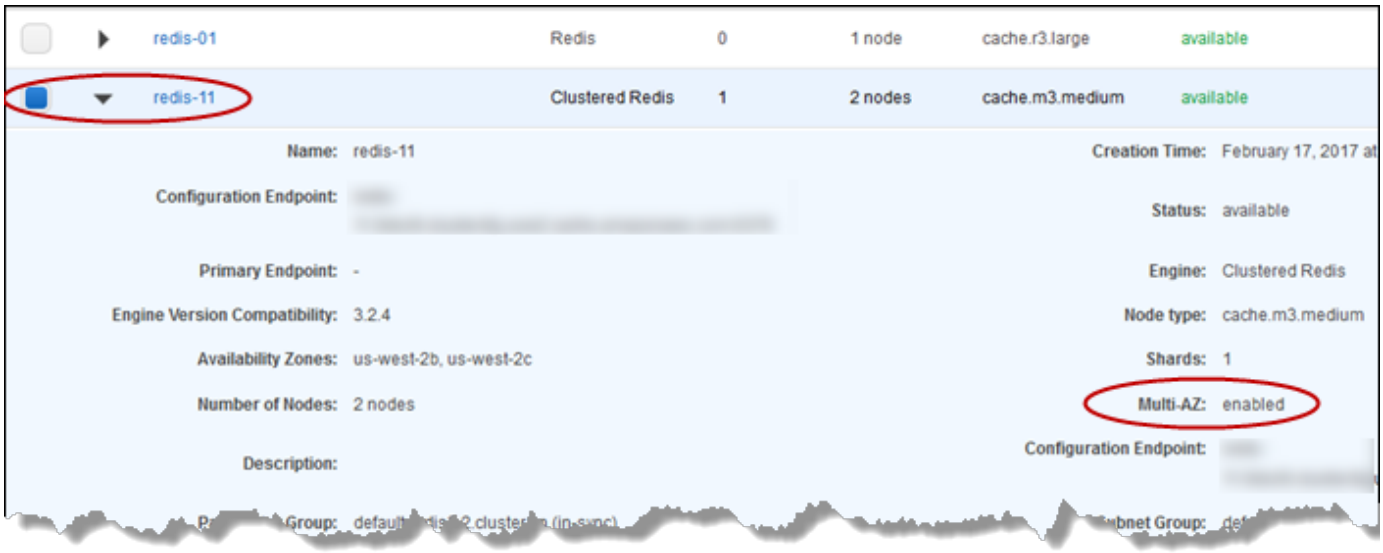
- Menguji failover otomatis menggunakan AWS Management Console
- Menguji failover otomatis menggunakan AWS CLI
- Menguji failover otomatis menggunakan API ElastiCache

Menguji failover otomatis menggunakan AWS Management Console

Gunakan prosedur berikut untuk menguji failover otomatis dengan konsol.

Untuk menguji failover otomatis

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Valkey atau Redis OSS.
3. Dari daftar cluster, pilih kotak di sebelah kiri cluster yang ingin Anda uji. Klaster ini harus memiliki setidaknya satu simpul replika baca.
4. Pada bagian Detail, lakukan konfirmasi bahwa klaster ini sudah mengaktifkan Multi-AZ. Jika klaster tidak memiliki Multi-AZ yang aktif, pilih klaster yang berbeda atau ubah klaster ini agar memiliki Multi-AZ yang aktif. Untuk informasi selengkapnya, lihat Menggunakan ElastiCache AWS Management Console.



5. Untuk Valkey atau Redis OSS (mode cluster dinonaktifkan), pilih nama cluster.

   Untuk Valkey atau Redis OSS (mode cluster diaktifkan), lakukan hal berikut:

   a. Pilih nama klaster.

b. Di halaman Serpihan, untuk serpihan (disebut grup simpul pada API dan CLI) yang ingin Anda uji failover-nya, pilih nama serpihan ini.

6. Di halaman Simpul, pilih Failover Primer.
7. Pilih Lanjutkan untuk melakukan failover primer, atau Batalkan untuk membatalkan failover simpul primer.

   Selama proses failover, konsol terus menunjukkan status simpul sebagai tersedia. Untuk memantau progres pengujian failover Anda, pilih Peristiwa dari panel navigasi konsol. Di tab Peristiwa, perhatikan peristiwa yang menunjukkan failover Anda telah dimulai (`Test Failover API called`) dan selesai (`Recovery completed`).

Menguji failover otomatis menggunakan AWS CLI

Anda dapat menguji failover otomatis pada cluster yang diaktifkan Multi-AZ menggunakan operasi. AWS CLI `test-failover`

Parameter

- `--replication-group-id` – Wajib. Grup replikasi (di konsol, klaster) yang akan diuji.
- `--node-group-id` – Wajib. Nama grup simpul yang ingin diuji failover otomatis. Anda dapat menguji maksimal 15 grup node dalam periode 24 jam bergulir.

Contoh berikut menggunakan AWS CLI untuk menguji failover otomatis pada grup `redis00-0003` node di Valkey atau Redis OSS (mode cluster enabled) cluster. `redis00`

Example Menguji failover otomatis

Untuk Linux, macOS, atau Unix:

```
aws elasticache test-failover \
   --replication-group-id redis00 \
   --node-group-id redis00-0003
```

Untuk Windows:

```
aws elasticache test-failover ^
```

```
    --replication-group-id redis00 ^
    --node-group-id redis00-0003
```

Output dari perintah sebelumnya akan terlihat seperti berikut.

```
{
    "ReplicationGroup": {
        "Status": "available",
        "Description": "1 shard, 3 nodes (1 + 2 replicas)",
        "NodeGroups": [
            {
                "Status": "available",
                "NodeGroupMembers": [
                    {
                        "CurrentRole": "primary",
                        "PreferredAvailabilityZone": "us-west-2c",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Port": 6379,
                            "Address":
 "redis1x3-001.7ekv3t.0001.usw2.cache.amazonaws.com"
                        },
                        "CacheClusterId": "redis1x3-001"
                    },
                    {
                        "CurrentRole": "replica",
                        "PreferredAvailabilityZone": "us-west-2a",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Port": 6379,
                            "Address":
 "redis1x3-002.7ekv3t.0001.usw2.cache.amazonaws.com"
                        },
                        "CacheClusterId": "redis1x3-002"
                    },
                    {
                        "CurrentRole": "replica",
                        "PreferredAvailabilityZone": "us-west-2b",
                        "CacheNodeId": "0001",
                        "ReadEndpoint": {
                            "Port": 6379,
                            "Address":
 "redis1x3-003.7ekv3t.0001.usw2.cache.amazonaws.com"
```

```
                    },
                    "CacheClusterId": "redis1x3-003"
                }
            ],
            "NodeGroupId": "0001",
            "PrimaryEndpoint": {
                "Port": 6379,
                "Address": "redis1x3.7ekv3t.ng.0001.usw2.cache.amazonaws.com"
            }
        }
    ],
    "ClusterEnabled": false,
    "ReplicationGroupId": "redis1x3",
    "SnapshotRetentionLimit": 1,
    "AutomaticFailover": "enabled",
    "MultiAZ": "enabled",
    "SnapshotWindow": "11:30-12:30",
    "SnapshottingClusterId": "redis1x3-002",
    "MemberClusters": [
        "redis1x3-001",
        "redis1x3-002",
        "redis1x3-003"
    ],
    "CacheNodeType": "cache.m3.medium",
    "DataTiering": "disabled",
    "PendingModifiedValues": {}
  }
}
```

Untuk melacak kemajuan failover Anda, gunakan AWS CLI `describe-events` operasi.

Untuk informasi selengkapnya, lihat berikut ini:

- test-failover dalam Referensi Perintah AWS CLI .
- describe-events dalam Referensi Perintah AWS CLI .

Menguji failover otomatis menggunakan API ElastiCache

Anda dapat menguji failover otomatis pada cluster apa pun yang diaktifkan dengan Multi-AZ menggunakan operasi ElastiCache API. `TestFailover`

Parameter

- `ReplicationGroupId` – Wajib. Grup replikasi (di konsol, klaster) yang akan diuji.
- `NodeGroupId` – Wajib. Nama grup simpul yang ingin diuji failover otomatis. Anda dapat menguji maksimal 15 grup node dalam periode 24 jam bergulir.

Contoh berikut menguji failover otomatis pada grup simpul `redis00-0003` pada grup replikasi (pada konsol, klaster) `redis00`.

Example Menguji failover otomatis

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=TestFailover
    &NodeGroupId=redis00-0003
    &ReplicationGroupId=redis00
    &Version=2015-02-02
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20140401T192317Z
    &X-Amz-Credential=<credential>
```

Untuk melacak kemajuan failover Anda, gunakan operasi ElastiCache `DescribeEvents` API.

Untuk informasi selengkapnya, lihat berikut ini:

- TestFailoverdi Referensi ElastiCache API
- DescribeEventsdi Referensi ElastiCache API

Batasan pada Multi-AZ

Waspadai batasan berikut untuk Multi-AZ:

- Multi-AZ didukung pada Valkey, dan pada Redis OSS versi 2.8.6 dan yang lebih baru.
- Multi-AZ tidak didukung pada tipe node T1.
- Replikasi Valkey dan Redis OSS tidak sinkron. Oleh karena itu, saat simpul primer melakukan failover ke replika, sejumlah kecil data mungkin hilang karena lag replikasi.

Saat memilih replika untuk dipromosikan ke primer, ElastiCache pilih replika dengan jeda replikasi paling sedikit. Dengan kata lain, yang dipilih adalah replika yang terkini. Hal ini membantu meminimalkan jumlah data yang hilang. Replika dengan lag replikasi terkecil dapat berada di Zona Ketersediaan yang sama atau berbeda dari simpul primer yang gagal.

- Saat Anda mempromosikan replika baca secara manual ke primer di cluster Valkey atau Redis OSS dengan mode cluster dinonaktifkan, Anda hanya dapat melakukannya ketika multi-AZ dan failover otomatis dinonaktifkan. Untuk mempromosikan replika baca menjadi primer, lakukan langkah berikut:

  1. Nonaktifkan Multi-AZ pada klaster.
  2. Nonaktifkan failover otomatis pada klaster. Anda dapat melakukan ini melalui konsol dengan membersihkan kotak centang Auto failover untuk grup replikasi. Anda juga dapat melakukan ini menggunakan AWS CLI dengan mengatur `AutomaticFailoverEnabled` properti `false` saat memanggil `ModifyReplicationGroup` operasi.
  3. Promosikan replika baca menjadi primer.
  4. Aktifkan kembali Multi-AZ.

- ElastiCache untuk Redis OSS Multi-AZ dan file append-only (AOF) saling eksklusif. Jika Anda mengaktifkan salah satunya, Anda tidak dapat mengaktifkan yang lain.
- Kegagalan simpul dapat disebabkan oleh peristiwa langka saat seluruh Zona Ketersediaan gagal. Dalam kasus ini, replika untuk menggantikan primer yang gagal akan dibuat hanya saat Zona Ketersediaan sudah dipulihkan. Sebagai contoh, misalkan grup replikasi dengan primer berada di AZ-a, sedangkan replika ada di AZ-b dan AZ-c. Jika primer gagal, replika dengan lag replikasi terkecil akan dipromosikan menjadi klaster primer. Kemudian, ElastiCache buat replika baru di AZ-a (di mana primer yang gagal berada) hanya ketika AZ-A di-back up dan tersedia.
- Boot ulang primer yang dilakukan oleh pelanggan tidak memicu failover otomatis. Boot ulang lain dan kegagalan akan memicu failover otomatis.
- Saat primer di-booting ulang, data dihapus dari primer saat primer kembali online. Saat replika baca melihat klaster primer yang bersih tanpa data, replika baca akan menghapus salinan datanya, yang menyebabkan hilangnya data.
- Setelah replika baca dipromosikan, replika lain melakukan sinkronisasi dengan primer yang baru. Setelah sinkronisasi awal, konten replikasi dihapus dan replika menyinkronkan data dari primer yang baru. Proses sinkronisasi ini menyebabkan gangguan singkat, sehingga replika tidak dapat diakses. Proses sinkronisasi juga menyebabkan peningkatan beban sementara pada primer pada saat melakukan sinkronisasi dengan replika. Perilaku ini asli Valkey dan Redis OSS dan tidak unik untuk Multi-AZ. ElastiCache Untuk detail tentang perilaku ini, lihat Replikasi di situs web Valkey.

Important

Untuk Valkey 7.2.6 dan yang lebih baru atau Redis OSS versi 2.8.22 dan yang lebih baru, Anda tidak dapat membuat replika eksternal.

Untuk versi Redis OSS sebelum 2.8.22, sebaiknya Anda tidak menghubungkan replika eksternal ke ElastiCache cluster yang diaktifkan Multi-AZ. Konfigurasi yang tidak didukung ini dapat membuat masalah yang ElastiCache mencegah terjadinya failover dan pemulihan dengan benar. Untuk menghubungkan replika eksternal ke ElastiCache cluster, pastikan bahwa Multi-AZ tidak diaktifkan sebelum Anda membuat koneksi.

Cara penerapan sinkronisasi dan pencadangan

Semua versi Valkey dan Redis OSS yang didukung mendukung pencadangan dan sinkronisasi antara node primer dan replika. Namun, cara pencadangan dan sinkronisasi diimplementasikan bervariasi tergantung pada versinya.

Redis OSS Versi 2.8.22 dan yang lebih baru

Redis OSS replikasi, dalam versi 2.8.22 dan yang lebih baru, pilih antara dua metode. Untuk informasi selengkapnya, lihat Versi Redis OSS Sebelum 2.8.22 dan Melakukan snapshot dan pemulihan.

Selama proses forkless, jika operasi tulis memiliki beban berat, operasi tulis ke klaster akan ditunda untuk memastikan bahwa Anda tidak menumpuk terlalu banyak perubahan sehingga menghalangi keberhasilan snapshot.

Versi Redis OSS Sebelum 2.8.22

Redis OSS backup dan sinkronisasi dalam versi sebelum 2.8.22 adalah proses tiga langkah.

1. Fork, dan dalam proses di latar belakang, menserialisasi data klaster ke disk. Ini menciptakan point-in-time snapshot.
2. Di latar depan, log perubahan pada buffer output klien diakumulasi.

   > **Important**
   >
   > Jika log perubahan melebihi ukuran buffer output klien, pencadangan atau sinkronisasi akan gagal. Untuk informasi selengkapnya, lihat Memastikan Anda memiliki cukup memori untuk membuat snapshot Valkey atau Redis OSS.

3. Sebagai langkah terakhir, data cache dan log perubahan ditransmisikan ke simpul replika.

Membuat grup replikasi Valkey atau Redis OSS

Anda memiliki opsi berikut untuk membuat klaster dengan simpul replika. Satu berlaku ketika Anda sudah memiliki cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) yang tersedia yang tidak terkait dengan cluster mana pun yang memiliki replika untuk digunakan sebagai simpul utama. Opsi lain berlaku jika Anda perlu membuat simpul primer dengan klaster dan replika baca. Saat ini, cluster Valkey atau Redis OSS (mode cluster enabled) harus dibuat dari awal.

Opsi 1: Membuat grup replikasi menggunakan klaster yang sudah ada

Gunakan opsi ini untuk memanfaatkan cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) yang ada. Anda menetapkan simpul yang sudah ada ini sebagai simpul primer dalam klaster baru, lalu secara individual menambahkan 1 hingga 5 replika baca untuk klaster ini. Jika klaster yang sudah ada ini aktif, replika baca melakukan sinkronisasi dengan klaster itu begitu replika dibuat. Lihat Membuat grup replikasi menggunakan klaster yang sudah ada.

> **Important**
>
> Anda tidak dapat membuat cluster Valkey atau Redis OSS (mode cluster enabled) menggunakan cluster yang ada. Untuk membuat cluster Valkey atau Redis OSS (mode cluster enabled) (API/CLI: grup replikasi) menggunakan konsol, lihat. ElastiCache Membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) (Konsol)

Opsi 2: Membuat grup replikasi Valkey atau Redis OSS dari awal

Gunakan opsi ini jika Anda belum memiliki cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) yang tersedia untuk digunakan sebagai simpul utama cluster, atau jika Anda ingin membuat cluster Valkey atau Redis OSS (mode cluster enabled). Lihat Membuat grup replikasi Valkey atau Redis OSS dari awal.

Membuat grup replikasi menggunakan klaster yang sudah ada

Prosedur berikut menambahkan grup replikasi ke cluster simpul tunggal Valkey atau Redis OSS (mode cluster dinonaktifkan) Anda, yang diperlukan untuk meningkatkan klaster Anda ke versi terbaru Valkey. Ini adalah prosedur di tempat yang melibatkan nol downtime dan nol kehilangan data. Saat Anda membuat grup replikasi untuk kluster simpul tunggal Anda, simpul klaster menjadi simpul utama di cluster baru. Jika Anda tidak memiliki cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) yang dapat Anda gunakan sebagai cluster utama klaster baru, lihat. Membuat grup replikasi Valkey atau Redis OSS dari awal

Cluster yang tersedia adalah cluster Valkey atau Redis OSS node tunggal yang ada. Saat ini, Valkey atau Redis OSS (mode cluster diaktifkan) tidak mendukung pembuatan cluster dengan replika menggunakan cluster simpul tunggal yang tersedia. Jika Anda ingin membuat cluster Valkey atau Redis OSS (mode cluster enabled), lihat. Membuat cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan) (Konsol)

Membuat grup replikasi menggunakan cluster yang ada (Console)

Lihat topik Menggunakan ElastiCache AWS Management Console.

Membuat grup replikasi menggunakan cluster cache Valkey atau Redis OSS yang tersedia ()AWS CLI

Ada dua langkah untuk membuat grup replikasi dengan replika baca saat menggunakan Valkey atau Redis OSS Cache Cluster yang tersedia untuk primer saat menggunakan. AWS CLI

Saat menggunakan AWS CLI Anda membuat grup replikasi yang menentukan node mandiri yang tersedia sebagai simpul utama cluster, `--primary-cluster-id` dan jumlah node yang Anda inginkan di cluster menggunakan perintah CLI,. `create-replication-group` Sertakan parameter berikut.

--replication-group-id

Nama grup replikasi yang Anda buat. Nilai parameter ini digunakan sebagai dasar untuk nama simpul yang ditambahkan dengan nomor 3 digit berurutan ditambahkan di akhir `--replication-group-id`. Misalnya, `sample-repl-group-001`.

Kendala penamaan grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) adalah sebagai berikut:

- Harus berisi 1–40 karakter alfanumerik atau tanda hubung.

- Harus diawali dengan huruf.
- Tidak boleh berisi dua tanda hubung berurutan.
- Tidak boleh diakhiri dengan tanda hubung.

--replication-group-description

Deskripsi grup replikasi.

--num-node-groups

Jumlah simpul yang Anda inginkan dalam klaster ini. Nilai ini mencakup simpul primer. Parameter ini memiliki nilai maksimum sebesar enam.

--primary-cluster-id

Nama node cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) yang tersedia yang Anda inginkan menjadi simpul utama dalam grup replikasi ini.

Perintah berikut membuat grup replikasi `sample-repl-group` menggunakan cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) yang tersedia `redis01` sebagai node utama grup replikasi. Tindakan ini membuat 2 simpul baru yang merupakan replika baca. Pengaturan dari `redis01` (yaitu grup parameter, grup keamanan, jenis simpul, versi mesin, dan seterusnya.) akan diterapkan untuk semua simpul dalam grup replikasi.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
   --replication-group-id sample-repl-group \
   --replication-group-description "demo cluster with replicas" \
   --num-cache-clusters 3 \
   --primary-cluster-id redis01
```

Untuk Windows:

```
aws elasticache create-replication-group ^
   --replication-group-id sample-repl-group ^
   --replication-group-description "demo cluster with replicas" ^
   --num-cache-clusters 3 ^
   --primary-cluster-id redis01
```

Untuk informasi dan parameter tambahan yang mungkin ingin Anda gunakan, lihat AWS CLI topiknyacreate-replication-group.

Selanjutnya, tambahkan replika baca ke grup replikasi

Setelah grup replikasi dibuat, tambahkan satu hingga lima replika baca ke grup untuk menggunakan perintah `create-cache-cluster`. Pastikan untuk menyertakan parameter berikut.

--cache-cluster-id

Nama klaster yang Anda tambahkan ke grup replikasi.

Batasan penamaan klaster adalah sebagai berikut:

- Harus berisi 1–40 karakter alfanumerik atau tanda hubung.
- Harus diawali dengan huruf.
- Tidak boleh berisi dua tanda hubung berurutan.
- Tidak boleh diakhiri dengan tanda hubung.

--replication-group-id

Nama grup replikasi yang dituju untuk menambahkan klaster cache ini.

Ulangi perintah ini untuk setiap replika baca yang ingin Anda tambahkan ke grup replikasi, dengan mengubah nilai dari parameter `--cache-cluster-id` saja.

> **Note**
>
> Ingat, grup replikasi tidak dapat memiliki lebih dari lima replika baca. Jika Anda mencoba menambahkan replika baca ke grup replikasi yang sudah memiliki lima replika baca, maka operasi ini akan gagal.

Kode berikut menambahkan replika baca `my-replica01` ke grup replikasi `sample-repl-group`. Pengaturan dari klaster primer–grup parameter, grup keamanan, jenis simpul, dan sebagainya–akan diterapkan ke simpul begitu simpul ditambahkan ke grup replikasi.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-cache-cluster \
   --cache-cluster-id my-replica01 \
   --replication-group-id sample-repl-group
```

Untuk Windows:

```
aws elasticache create-cache-cluster ^
   --cache-cluster-id my-replica01 ^
   --replication-group-id sample-repl-group
```

Output dari perintah ini akan terlihat seperti ini.

```
{
    "ReplicationGroup": {
        "Status": "creating",
        "Description": "demo cluster with replicas",
        "ClusterEnabled": false,
        "ReplicationGroupId": "sample-repl-group",
        "SnapshotRetentionLimit": 1,
        "AutomaticFailover": "disabled",
        "SnapshotWindow": "00:00-01:00",
        "SnapshottingClusterId": "redis01",
        "MemberClusters": [
            "sample-repl-group-001",
            "sample-repl-group-002",
            "redis01"
        ],
        "CacheNodeType": "cache.m4.large",
        "DataTiering": "disabled",
        "PendingModifiedValues": {}
    }
}
```

Untuk informasi tambahan, lihat AWS CLI topik:

- create-replication-group
- modify-replication-group

Menambahkan replika ke cluster Valkey atau Redis OSS (Cluster Mode Disabled) mandiri (API) ElastiCache

Saat menggunakan ElastiCache API, Anda membuat grup replikasi yang menentukan node mandiri yang tersedia sebagai simpul utama cluster, `PrimaryClusterId` dan jumlah node yang Anda inginkan di cluster menggunakan perintah CLI,. `CreateReplicationGroup` Sertakan parameter berikut.

ReplicationGroupId

Nama grup replikasi yang Anda buat. Nilai parameter ini digunakan sebagai dasar untuk nama simpul yang ditambahkan dengan nomor 3 digit berurutan ditambahkan di akhir `ReplicationGroupId`. Misalnya, `sample-repl-group-001`.

Kendala penamaan grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) adalah sebagai berikut:

- Harus berisi 1–40 karakter alfanumerik atau tanda hubung.
- Harus diawali dengan huruf.
- Tidak boleh berisi dua tanda hubung berurutan.
- Tidak boleh diakhiri dengan tanda hubung.

ReplicationGroupDescription

Deskripsi klaster dengan replika.

NumCacheClusters

Jumlah simpul yang Anda inginkan dalam klaster ini. Nilai ini mencakup simpul primer. Parameter ini memiliki nilai maksimum sebesar enam.

PrimaryClusterId

Nama klaster Valkey atau Redis OSS (mode cluster dinonaktifkan) yang tersedia yang Anda inginkan menjadi simpul utama di cluster ini.

Perintah berikut membuat cluster dengan replika `sample-repl-group` menggunakan cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) yang tersedia `redis01` sebagai node utama grup replikasi. Tindakan ini membuat 2 simpul baru yang merupakan replika baca. Pengaturan dari `redis01` (yaitu grup parameter, grup keamanan, jenis simpul, versi mesin, dan seterusnya.) akan diterapkan untuk semua simpul dalam grup replikasi.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=CreateReplicationGroup
   &Engine=redis
   &EngineVersion=6.0
   &ReplicationGroupDescription=Demo%20cluster%20with%20replicas
   &ReplicationGroupId=sample-repl-group
   &PrimaryClusterId=redis01
   &Version=2015-02-02
```

```
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &X-Amz-Credential=<credential>
```

Untuk informasi tambahan, lihat topik ElastiCache APL:

- CreateReplicationGroup
- ModifyReplicationGroup

Selanjutnya, tambahkan replika baca ke grup replikasi

Setelah grup replikasi dibuat, tambahkan satu hingga lima replika baca ke grup untuk menggunakan operasi `CreateCacheCluster`. Pastikan untuk menyertakan parameter berikut.

CacheClusterId

Nama klaster yang Anda tambahkan ke grup replikasi.

Batasan penamaan klaster adalah sebagai berikut:

- Harus berisi 1–40 karakter alfanumerik atau tanda hubung.
- Harus diawali dengan huruf.
- Tidak boleh berisi dua tanda hubung berurutan.
- Tidak boleh diakhiri dengan tanda hubung.

ReplicationGroupId

Nama grup replikasi yang dituju untuk menambahkan klaster cache ini.

Ulangi operasi ini untuk setiap replika baca yang ingin Anda tambahkan ke grup replikasi, dengan mengubah nilai dari parameter `CacheClusterId` saja.

Kode berikut menambahkan replika baca `myReplica01` ke grup replikasi `myReplGroup`. Pengaturan klaster primer–grup parameter, grup keamanan, jenis simpul, dan sebagainya–akan diterapkan ke simpul begitu simpul ditambahkan ke grup replikasi.

```
https://elasticache.us-west-2.amazonaws.com/
 ?Action=CreateCacheCluster
 &CacheClusterId=myReplica01
```

```
&ReplicationGroupId=myReplGroup
&SignatureMethod=HmacSHA256
&SignatureVersion=4
&Version=2015-02-02
&X-Amz-Algorithm=&AWS;4-HMAC-SHA256
&X-Amz-Credential=[your-access-key-id]/20150202/us-west-2/elasticache/aws4_request
&X-Amz-Date=20150202T170651Z
&X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date
&X-Amz-Signature=[signature-value]
```

Untuk informasi dan parameter tambahan yang mungkin ingin Anda gunakan, lihat topik ElastiCache APICreateCacheCluster.

Membuat grup replikasi Valkey atau Redis OSS dari awal

Berikut ini, Anda dapat menemukan cara membuat grup replikasi Valkey atau Redis OSS tanpa menggunakan cluster Valkey atau Redis OSS yang ada sebagai yang utama. Anda dapat membuat grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) atau Valkey atau Redis OSS (mode cluster enabled) dari awal menggunakan ElastiCache konsol, the, atau API. AWS CLI ElastiCache

Sebelum melanjutkan, putuskan apakah Anda ingin membuat grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) atau Valkey atau Redis OSS (mode cluster diaktifkan). Untuk panduan dalam memutuskan, lihat Replikasi: Mode Cluster Valkey dan Redis OSS Dinonaktifkan vs Diaktifkan.

Topik

- Membuat grup replikasi Valkey atau Redis OSS (Cluster Mode Disabled) dari awal
- Membuat grup replikasi di Valkey atau Redis OSS (Mode Cluster Diaktifkan) dari awal

Membuat grup replikasi Valkey atau Redis OSS (Cluster Mode Disabled) dari awal

Anda dapat membuat grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) dari awal menggunakan ElastiCache konsol, the AWS CLI, atau API. ElastiCache Grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) selalu memiliki satu grup node, cluster utama, dan hingga lima replika baca. Grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) tidak mendukung partisi data Anda.

> **Note**
>
> node/shard Batas dapat ditingkatkan hingga maksimum 500 per cluster. Untuk meminta penambahan batas, lihat Batas Layanan AWS dan sertakan jenis instans dalam permintaan yang diajukan.

Untuk membuat grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) dari awal, ambil salah satu pendekatan berikut:

Membuat grup replikasi Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) dari awal ()AWS CLI

Prosedur berikut membuat grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) menggunakan grup replikasi. AWS CLI

Saat Anda membuat grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) dari awal, Anda membuat grup replikasi dan semua node dengan satu panggilan ke perintah. AWS CLI `create-replication-group` Sertakan parameter berikut.

--replication-group-id

Nama grup replikasi yang Anda buat.

Kendala penamaan grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) adalah sebagai berikut:

- Harus berisi 1–40 karakter alfanumerik atau tanda hubung.
- Harus diawali dengan huruf.
- Tidak boleh berisi dua tanda hubung berurutan.
- Tidak boleh diakhiri dengan tanda hubung.

--replication-group-description

Deskripsi grup replikasi.

--num-cache-clusters

Jumlah simpul yang ingin dibuat dengan grup replikasi ini, digabungkan dengan replika primer dan replika baca.

Jika Anda mengaktifkan Multi-AZ (`--automatic-failover-enabled`), nilai dari `--num-cache-clusters` harus minimal 2.

--cache-node-type

Jenis simpul untuk setiap simpul dalam grup replikasi.

ElastiCache mendukung jenis node berikut. Secara umum, jenis generasi saat ini memberikan lebih banyak memori dan daya komputasi dengan biaya lebih rendah dibandingkan dengan jenis generasi sebelumnya yang setara.

Untuk informasi selengkapnya tentang detail performa untuk setiap jenis node, lihat Jenis EC2 Instance Amazon.

--data-tiering-enabled

Atur parameter ini jika Anda menggunakan jenis simpul r6gd. Jika Anda tidak ingin tingkatan data, atur `--no-data-tiering-enabled`. Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.

--cache-parameter-group

Tentukan grup parameter yang sesuai dengan versi mesin Anda. Jika Anda menjalankan Redis OSS 3.2.4 atau yang lebih baru, tentukan grup parameter atau grup `default.redis3.2` parameter yang berasal dari `default.redis3.2` untuk membuat grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan). Untuk informasi selengkapnya, lihat Parameter Valkey dan Redis OSS.

--network-type

Baik `ipv4`, `ipv6` atau `dual-stack`. Jika Anda memilih tumpukan ganda, Anda harus menetapkan parameter `--IpDiscovery` ke `ipv4` atau `ipv6`.

--engine

redis

--engine-version

Untuk mendapatkan kumpulan fitur terbanyak, pilih versi mesin terbaru.

Nama simpul akan diambil dari nama grup replikasi dengan tambahan `-00#` di belakang nama grup replikasi. Sebagai contoh, menggunakan nama grup replikasi `myReplGroup`, nama untuk replika primer menjadi `myReplGroup-001` dan replika baca menjadi `myReplGroup-002` hingga `myReplGroup-006`.

Jika Anda ingin mengaktifkan enkripsi bergerak atau diam pada grup replikasi ini, tambahkan salah satu atau kedua parameter `--transit-encryption-enabled` atau `--at-rest-encryption-enabled` dan penuhi kondisi berikut.

- Grup replikasi Anda harus menjalankan Redis OSS versi 3.2.6 atau 4.0.10.
- Grup replikasi harus dibuat dalam Amazon VPC.
- Anda juga harus menyertakan parameter `--cache-subnet-group`.
- Anda juga harus menyertakan parameter `--auth-token` dengan nilai string yang ditentukan oleh pelanggan untuk token (kata sandi) AUTH Anda yang diperlukan untuk melakukan operasi pada grup replikasi ini.

Operasi berikut membuat grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) `sample-repl-group` dengan tiga node, primer dan dua replika.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
   --replication-group-id sample-repl-group \
   --replication-group-description "Demo cluster with replicas" \
   --num-cache-clusters 3 \
   --cache-node-type cache.m4.large \
   --engine redis
```

Untuk Windows:

```
aws elasticache create-replication-group ^
   --replication-group-id sample-repl-group ^
   --replication-group-description "Demo cluster with replicas" ^
   --num-cache-clusters 3 ^
   --cache-node-type cache.m4.large ^
   --engine redis
```

Output dari perintah ini adalah seperti berikut.

```
{
    "ReplicationGroup": {
        "Status": "creating",
        "Description": "Demo cluster with replicas",
        "ClusterEnabled": false,
        "ReplicationGroupId": "sample-repl-group",
        "SnapshotRetentionLimit": 0,
        "AutomaticFailover": "disabled",
        "SnapshotWindow": "01:30-02:30",
        "MemberClusters": [
            "sample-repl-group-001",
            "sample-repl-group-002",
            "sample-repl-group-003"
        ],
        "CacheNodeType": "cache.m4.large",
        "DataTiering": "disabled",
        "PendingModifiedValues": {}
    }
}
```

Untuk informasi dan parameter tambahan yang mungkin ingin Anda gunakan, lihat AWS CLI topiknya create-replication-group.

Membuat grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) dari awal (API) ElastiCache

Prosedur berikut membuat grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) menggunakan API. ElastiCache

Saat Anda membuat grup replikasi Valkey atau Redis OSS (mode cluster disabled) dari awal, Anda membuat grup replikasi dan semua node dengan satu panggilan ke operasi API. ElastiCache `CreateReplicationGroup` Sertakan parameter berikut.

ReplicationGroupId

Nama grup replikasi yang Anda buat.

Kendala penamaan grup replikasi Valkey atau Redis OSS (mode cluster enabled) adalah sebagai berikut:

- Harus berisi 1–40 karakter alfanumerik atau tanda hubung.
- Harus diawali dengan huruf.

- Tidak boleh berisi dua tanda hubung berurutan.
- Tidak boleh diakhiri dengan tanda hubung.

ReplicationGroupDescription

Deskripsi Anda tentang grup replikasi.

NumCacheClusters

Total jumlah simpul yang ingin dibuat dengan grup replikasi ini, digabungkan dengan primer dan replika baca.

Jika Anda mengaktifkan Multi-AZ (`AutomaticFailoverEnabled=true`), nilai dari `NumCacheClusters` harus minimal 2.

CacheNodeType

Jenis simpul untuk setiap simpul dalam grup replikasi.

ElastiCache mendukung jenis node berikut. Secara umum, jenis generasi saat ini memberikan lebih banyak memori dan daya komputasi dengan biaya lebih rendah dibandingkan dengan jenis generasi sebelumnya yang setara.

Untuk informasi selengkapnya tentang detail performa untuk setiap jenis node, lihat Jenis EC2 Instance Amazon.

--data-tiering-enabled

Atur parameter ini jika Anda menggunakan jenis simpul r6gd. Jika Anda tidak ingin tingkatan data, atur `--no-data-tiering-enabled`. Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.

CacheParameterGroup

Tentukan grup parameter yang sesuai dengan versi mesin Anda. Jika Anda menjalankan Redis OSS 3.2.4 atau yang lebih baru, tentukan grup parameter atau grup `default.redis3.2` parameter yang berasal dari `default.redis3.2` untuk membuat grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan). Untuk informasi selengkapnya, lihat Parameter Valkey dan Redis OSS.

--network-type

Baik `ipv4`, `ipv` atau `dual-stack`. Jika Anda memilih tumpukan ganda, Anda harus mengatur parameter `--IpDiscovery` ke salah satu `ipv4` atau `ipv6`.

Engine

redis

EngineVersion

6.0

Nama simpul akan diambil dari nama grup replikasi dengan tambahan `-00#` di belakang nama grup replikasi. Sebagai contoh, menggunakan nama grup replikasi `myReplGroup`, nama untuk replika primer menjadi `myReplGroup-001` dan replika baca menjadi `myReplGroup-002` hingga `myReplGroup-006`.

Jika Anda ingin mengaktifkan enkripsi bergerak atau diam pada grup replikasi ini, tambahkan salah satu atau kedua parameter `TransitEncryptionEnabled=true` atau `AtRestEncryptionEnabled=true` dan penuhi kondisi berikut.

- Grup replikasi Anda harus menjalankan Redis OSS versi 3.2.6 atau 4.0.10.
- Grup replikasi harus dibuat dalam Amazon VPC.
- Anda juga harus menyertakan parameter `CacheSubnetGroup`.
- Anda juga harus menyertakan parameter `AuthToken` dengan nilai string yang ditentukan oleh pelanggan untuk token (kata sandi) AUTH Anda yang diperlukan untuk melakukan operasi pada grup replikasi ini.

Operasi berikut membuat grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) `myReplGroup` dengan tiga node, primer dan dua replika.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=CreateReplicationGroup
   &CacheNodeType=cache.m4.large
   &CacheParameterGroup=default.redis6.x
   &Engine=redis
   &EngineVersion=6.0
   &NumCacheClusters=3
   &ReplicationGroupDescription=test%20group
   &ReplicationGroupId=myReplGroup
   &Version=2015-02-02
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
```

```
    &X-Amz-Credential=<credential>
```

Untuk informasi dan parameter tambahan yang mungkin ingin Anda gunakan, lihat topik ElastiCache APICreateReplicationGroup.

Membuat grup replikasi di Valkey atau Redis OSS (Mode Cluster Diaktifkan) dari awal

Anda dapat membuat klaster Valkey atau Redis OSS (mode cluster enabled) (API/CLI: grup replikasi) menggunakan ElastiCache konsol, the, atau API. AWS CLI ElastiCache Grup replikasi Valkey atau Redis OSS (mode cluster enabled) memiliki 1 hingga 500 pecahan (API/CLI: grup simpul), simpul utama di setiap pecahan, dan hingga 5 replika baca di setiap pecahan. Anda dapat membuat klaster dengan jumlah serpihan lebih banyak dan jumlah replika lebih sedikit dengan jumlah total hingga 90 simpul per klaster. Konfigurasi klaster ini dapat berkisar dari 90 serpihan dan 0 replika hingga 15 serpihan dan 5 replika, yang merupakan jumlah replika maksimum yang diizinkan.

Batas node atau shard dapat ditingkatkan hingga maksimum 500 per cluster jika versi mesin Valkey atau Redis OSS adalah 5.0.6 atau lebih tinggi. Sebagai contoh, Anda dapat memilih untuk mengonfigurasi sebuah klaster dengan 500 simpul yang berkisar antara 83 serpihan (satu primer dan 5 replika per serpihan) dan 500 serpihan (satu primer dan tanpa replika). Pastikan alamat IP yang tersedia mencukupi untuk mengakomodasi peningkatan tersebut. Kesalahan umumnya termasuk subnet dalam grup subnet memiliki rentang CIDR yang terlalu kecil atau subnet dibagikan dan banyak digunakan oleh klaster lainnya. Untuk informasi selengkapnya, lihat Membuat grup subnet.

Untuk versi di bawah 5.0.6, batasnya adalah 250 per klaster.

Untuk meminta penambahan batas, lihat Batas Layanan AWS dan pilih jenis batas Simpul per klaster per jenis instans.

Membuat Cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan)

- Membuat cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan) (Konsol)
- Membuat grup replikasi Valkey atau Redis OSS (Cluster Mode Enabled) dari awal ()AWS CLI
- Membuat grup replikasi di Valkey atau Redis OSS (Mode Cluster Diaktifkan) dari awal (API) ElastiCache

Membuat cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan) (Konsol)

Untuk membuat cluster Valkey atau Redis OSS (mode cluster enabled), lihat. Membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) (Konsol) Pastikan untuk mengaktifkan mode klaster, Mode Klaster diaktifkan (Menskalakan Ke Luar), dan tentukan setidaknya dua serpihan dan satu simpul replika di setiap serpihan.

Membuat grup replikasi Valkey atau Redis OSS (Cluster Mode Enabled) dari awal ()AWS CLI

Prosedur berikut membuat grup replikasi Valkey atau Redis OSS (mode cluster enabled) menggunakan. AWS CLI

Saat Anda membuat grup replikasi Valkey atau Redis OSS (mode cluster enabled) dari awal, Anda membuat grup replikasi dan semua node dengan satu panggilan ke perintah. AWS CLI `create-replication-group` Sertakan parameter berikut.

--replication-group-id

Nama grup replikasi yang Anda buat.

Kendala penamaan grup replikasi Valkey atau Redis OSS (mode cluster enabled) adalah sebagai berikut:

- Harus berisi 1–40 karakter alfanumerik atau tanda hubung.
- Harus diawali dengan huruf.
- Tidak boleh berisi dua tanda hubung berurutan.
- Tidak boleh diakhiri dengan tanda hubung.

--replication-group-description

Deskripsi grup replikasi.

--cache-node-type

Jenis simpul untuk setiap simpul dalam grup replikasi.

ElastiCache mendukung jenis node berikut. Secara umum, jenis generasi saat ini memberikan lebih banyak memori dan daya komputasi dengan biaya lebih rendah dibandingkan dengan jenis generasi sebelumnya yang setara.

Untuk informasi selengkapnya tentang detail performa untuk setiap jenis node, lihat Jenis EC2 Instance Amazon.

--data-tiering-enabled

Atur parameter ini jika Anda menggunakan jenis simpul r6gd. Jika Anda tidak ingin tingkatan data, atur `--no-data-tiering-enabled`. Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.

--cache-parameter-group

Tentukan grup `default.redis6.x.cluster.on` parameter atau grup parameter yang berasal dari `default.redis6.x.cluster.on` untuk membuat grup replikasi Valkey atau Redis OSS (mode cluster diaktifkan). Untuk informasi selengkapnya, lihat Redis OSS 6.x perubahan parameter.

--engine

redis

--engine-version

3.2.4

--num-node-groups

Jumlah grup simpul dalam grup replikasi ini. Nilai yang valid adalah 1 sampai 500.

> **Note**
>
> node/shard Batas dapat ditingkatkan hingga maksimum 500 per cluster. Untuk meminta penambahan batas, lihat Batas Layanan AWS dan pilih jenis batas "Simpul per klaster per jenis instans".

--replicas-per-node-group

Jumlah simpul replika di setiap grup simpul. Nilai yang valid adalah 0 sampai 5.

--network-type

Baik `ipv4`, `ipv` atau `dual-stack`. Jika Anda memilih tumpukan ganda, Anda harus mengatur parameter `--IpDiscovery` ke salah satu `ipv4` atau `ipv6`.

Jika Anda ingin mengaktifkan enkripsi bergerak atau diam pada grup replikasi ini, tambahkan salah satu atau kedua parameter `--transit-encryption-enabled` atau `--at-rest-encryption-enabled` dan penuhi kondisi berikut.

- Grup replikasi Anda harus menjalankan Redis OSS versi 3.2.6 atau 4.0.10.
- Grup replikasi harus dibuat dalam Amazon VPC.

- Anda juga harus menyertakan parameter `--cache-subnet-group`.
- Anda juga harus menyertakan parameter `--auth-token` dengan nilai string yang ditentukan oleh pelanggan untuk token (kata sandi) AUTH Anda yang diperlukan untuk melakukan operasi pada grup replikasi ini.

Operasi berikut membuat grup replikasi Valkey atau Redis OSS (mode cluster enabled) `sample-repl-group` dengan tiga node groups/shards (--num-node-groups), masing-masing dengan tiga node, primer dan dua replika baca (--). replicas-per-node-group

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
   --replication-group-id sample-repl-group \
   --replication-group-description "Demo cluster with replicas" \
   --num-node-groups 3 \
   --replicas-per-node-group 2 \
   --cache-node-type cache.m4.large \
   --engine redis \
   --security-group-ids SECURITY_GROUP_ID \
   --cache-subnet-group-name SUBNET_GROUP_NAME>
```

Untuk Windows:

```
aws elasticache create-replication-group ^
   --replication-group-id sample-repl-group ^
   --replication-group-description "Demo cluster with replicas" ^
   --num-node-groups 3 ^
   --replicas-per-node-group 2 ^
   --cache-node-type cache.m4.large ^
   --engine redis ^
   --security-group-ids SECURITY_GROUP_ID ^
   --cache-subnet-group-name SUBNET_GROUP_NAME>
```

Perintah sebelumnya menghasilkan output berikut.

```
{
    "ReplicationGroup": {
        "Status": "creating",
```

```
        "Description": "Demo cluster with replicas",
        "ReplicationGroupId": "sample-repl-group",
        "SnapshotRetentionLimit": 0,
        "AutomaticFailover": "enabled",
        "SnapshotWindow": "05:30-06:30",
        "MemberClusters": [
            "sample-repl-group-0001-001",
            "sample-repl-group-0001-002",
            "sample-repl-group-0001-003",
            "sample-repl-group-0002-001",
            "sample-repl-group-0002-002",
            "sample-repl-group-0002-003",
            "sample-repl-group-0003-001",
            "sample-repl-group-0003-002",
            "sample-repl-group-0003-003"
        ],
        "PendingModifiedValues": {}
    }
}
```

Saat Anda membuat grup replikasi Valkey atau Redis OSS (mode cluster enabled) dari awal, Anda dapat mengonfigurasi setiap pecahan di cluster menggunakan `--node-group-configuration` parameter seperti yang ditunjukkan pada contoh berikut yang mengonfigurasi dua grup node (Console: shards). Serpihan pertama memiliki dua simpul, satu primer dan satu replika baca. Serpihan kedua memiliki tiga simpul, satu primer dan dua replika baca.

--node-group-configuration

Konfigurasi untuk setiap grup simpul. Parameter `--node-group-configuration` terdiri dari bidang berikut.

- `PrimaryAvailabilityZone` – Zona Ketersediaan yang menjadi lokasi dari simpul primer dari grup simpul ini. Jika parameter ini dihilangkan, ElastiCache pilih Availability Zone untuk node utama.

  Contoh: us-west-2a.

- `ReplicaAvailabilityZones` – Daftar yang dipisahkan koma untuk Zona Ketersediaan yang menjadi lokasi replika baca. Jumlah Zona Ketersediaan dalam daftar harus sesuai dengan nilai dari `ReplicaCount`. Jika parameter ini dihilangkan, ElastiCache pilih Availability Zones untuk node replika.

  Contoh: "us-west-2a,us-west-2b,us-west-2c"

- `ReplicaCount` – Jumlah simpul replika di setiap grup simpul.
- `Slots` – String yang menentukan ruang kunci untuk grup simpul. String menggunakan format `startKey-endKey`. Jika parameter ini dihilangkan, ElastiCache mengalokasikan kunci secara merata di antara kelompok node.

  Contoh: "0-4999"

Operasi berikut membuat grup replikasi Valkey atau Redis OSS (mode cluster enabled) `new-group` dengan dua grup simpul/pecahan (). `--num-node-groups` Tidak seperti contoh sebelumnya, setiap grup simpul dikonfigurasi secara berbeda dari grup simpul lainnya (`--node-group-configuration`).

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
  --replication-group-id new-group \
  --replication-group-description "Sharded replication group" \
  --engine redis \
  --snapshot-retention-limit 8 \
  --cache-node-type cache.m4.medium \
  --num-node-groups 2 \
  --node-group-configuration \
      "ReplicaCount=1,Slots=0-8999,PrimaryAvailabilityZone='us-
east-1c',ReplicaAvailabilityZones='us-east-1b'" \
      "ReplicaCount=2,Slots=9000-16383,PrimaryAvailabilityZone='us-
east-1a',ReplicaAvailabilityZones='us-east-1a','us-east-1c'"
```

Untuk Windows:

```
aws elasticache create-replication-group ^
  --replication-group-id new-group ^
  --replication-group-description "Sharded replication group" ^
  --engine redis ^
  --snapshot-retention-limit 8 ^
  --cache-node-type cache.m4.medium ^
  --num-node-groups 2 ^
  --node-group-configuration \
      "ReplicaCount=1,Slots=0-8999,PrimaryAvailabilityZone='us-
east-1c',ReplicaAvailabilityZones='us-east-1b'" \
```

```
      "ReplicaCount=2,Slots=9000-16383,PrimaryAvailabilityZone='us-
east-1a',ReplicaAvailabilityZones='us-east-1a','us-east-1c'"
```

Operasi sebelumnya menghasilkan output berikut.

```
{
    "ReplicationGroup": {
        "Status": "creating",
        "Description": "Sharded replication group",
        "ReplicationGroupId": "rc-rg",
        "SnapshotRetentionLimit": 8,
        "AutomaticFailover": "enabled",
        "SnapshotWindow": "10:00-11:00",
        "MemberClusters": [
            "rc-rg-0001-001",
            "rc-rg-0001-002",
            "rc-rg-0002-001",
            "rc-rg-0002-002",
            "rc-rg-0002-003"
        ],
        "PendingModifiedValues": {}
    }
}
```

Untuk informasi dan parameter tambahan yang mungkin ingin Anda gunakan, lihat AWS CLI topiknyacreate-replication-group.

Membuat grup replikasi di Valkey atau Redis OSS (Mode Cluster Diaktifkan) dari awal (API) ElastiCache

Prosedur berikut membuat grup replikasi Valkey atau Redis OSS (mode cluster enabled) menggunakan API. ElastiCache

Saat Anda membuat grup replikasi Valkey atau Redis OSS (mode cluster enabled) dari awal, Anda membuat grup replikasi dan semua node dengan satu panggilan ke operasi API. ElastiCache `CreateReplicationGroup` Sertakan parameter berikut.

ReplicationGroupId

Nama grup replikasi yang Anda buat.

Kendala penamaan grup replikasi Valkey atau Redis OSS (mode cluster enabled) adalah sebagai berikut:

- Harus berisi 1–40 karakter alfanumerik atau tanda hubung.
- Harus diawali dengan huruf.
- Tidak boleh berisi dua tanda hubung berurutan.
- Tidak boleh diakhiri dengan tanda hubung.

ReplicationGroupDescription

Deskripsi grup replikasi.

NumNodeGroups

Jumlah grup simpul yang ingin Anda buat dengan grup replikasi ini. Nilai yang valid adalah 1 sampai 500.

ReplicasPerNodeGroup

Jumlah simpul replika di setiap grup simpul. Nilai yang valid adalah 1 sampai 5.

NodeGroupConfiguration

Konfigurasi untuk setiap grup simpul. Parameter `NodeGroupConfiguration` terdiri dari bidang berikut.

- `PrimaryAvailabilityZone` – Zona Ketersediaan yang menjadi lokasi dari simpul primer dari grup simpul ini. Jika parameter ini dihilangkan, ElastiCache pilih Availability Zone untuk node utama.

  Contoh: us-west-2a.
- `ReplicaAvailabilityZones` – Daftar Zona Ketersediaan tempat replika baca berada. Jumlah Zona Ketersediaan dalam daftar harus sesuai dengan nilai dari `ReplicaCount`. Jika parameter ini dihilangkan, ElastiCache pilih Availability Zones untuk node replika.
- `ReplicaCount` – Jumlah simpul replika di setiap grup simpul.
- `Slots` – String yang menentukan ruang kunci untuk grup simpul. String menggunakan format `startKey-endKey`. Jika parameter ini dihilangkan, ElastiCache mengalokasikan kunci secara merata di antara kelompok node.

  Contoh: "0-4999"

CacheNodeType

Jenis simpul untuk setiap simpul dalam grup replikasi.

ElastiCache mendukung jenis node berikut. Secara umum, jenis generasi saat ini memberikan lebih banyak memori dan daya komputasi dengan biaya lebih rendah dibandingkan dengan jenis generasi sebelumnya yang setara.

Untuk informasi selengkapnya tentang detail performa untuk setiap jenis node, lihat Jenis EC2 Instance Amazon.

--data-tiering-enabled

Atur parameter ini jika Anda menggunakan jenis simpul r6gd. Jika Anda tidak ingin tingkatan data, atur `--no-data-tiering-enabled`. Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.

CacheParameterGroup

Tentukan grup `default.redis6.x.cluster.on` parameter atau grup parameter yang berasal dari `default.redis6.x.cluster.on` untuk membuat grup replikasi Valkey atau Redis OSS (mode cluster diaktifkan). Untuk informasi selengkapnya, lihat Redis OSS 6.x perubahan parameter.

--network-type

Baik `ipv4`, `ipv` atau `dual-stack`. Jika Anda memilih tumpukan ganda, Anda harus mengatur parameter `--IpDiscovery` ke salah satu `ipv4` atau `ipv6`.

Engine

redis

EngineVersion

6.0

Jika Anda ingin mengaktifkan enkripsi bergerak atau diam pada grup replikasi ini, tambahkan salah satu atau kedua parameter `TransitEncryptionEnabled=true` atau `AtRestEncryptionEnabled=true` dan penuhi kondisi berikut.

- Grup replikasi Anda harus menjalankan Redis OSS versi 3.2.6 atau 4.0.10.
- Grup replikasi harus dibuat dalam Amazon VPC.
- Anda juga harus menyertakan parameter `CacheSubnetGroup`.
- Anda juga harus menyertakan parameter `AuthToken` dengan nilai string yang ditentukan oleh pelanggan untuk token (kata sandi) AUTH Anda yang diperlukan untuk melakukan operasi pada grup replikasi ini.

Jeda baris ditambahkan agar dapat lebih mudah dibaca.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=CreateReplicationGroup
   &CacheNodeType=cache.m4.large
   &CacheParemeterGroup=default.redis6.xcluster.on
   &Engine=redis
   &EngineVersion=6.0
   &NumNodeGroups=3
   &ReplicasPerNodeGroup=2
   &ReplicationGroupDescription=test%20group
   &ReplicationGroupId=myReplGroup
   &Version=2015-02-02
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &X-Amz-Credential=<credential>
```

Untuk informasi dan parameter tambahan yang mungkin ingin Anda gunakan, lihat topik ElastiCache APICreateReplicationGroup.

Melihat detail grup replikasi

Terkadang Anda sebaiknya melihat detail grup replikasi. Anda dapat menggunakan ElastiCache konsol, AWS CLI for ElastiCache, atau ElastiCache API. Proses konsol berbeda untuk Valkey atau Redis OSS (mode cluster dinonaktifkan) dan Valkey atau Redis OSS (mode cluster diaktifkan).

Melihat Detail Grup Replikasi

- Melihat Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) dengan replika
  - Melihat Grup Replikasi Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol)
  - Melihat grup replikasi Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) ()AWS CLI
  - Melihat Grup Replikasi (API) Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) ElastiCache
- Melihat grup replikasi: Valkey atau Redis OSS (Mode Cluster Diaktifkan)
  - Melihat cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan) (Konsol)
  - Melihat cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan) ()AWS CLI
  - Melihat Cluster (API) Valkey atau Redis OSS (Mode Cluster Diaktifkan) ElastiCache
- Melihat detail grup replikasi (AWS CLI)
- Melihat detail grup replikasi (ElastiCache API)

Melihat Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) dengan replika

Anda dapat melihat detail cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan replika (API/CLI: grup replikasi) menggunakan ElastiCache konsol, for, atau API. AWS CLI ElastiCache ElastiCache

Melihat Detail Cluster Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)

- Melihat Grup Replikasi Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol)
- Melihat grup replikasi Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) ()AWS CLI
- Melihat Grup Replikasi (API) Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) ElastiCache

Melihat Grup Replikasi Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol)

Untuk melihat detail cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan replika menggunakan ElastiCache konsol, lihat topiknya. Melihat detail Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol)

Melihat grup replikasi Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) ()AWS CLI

Untuk AWS CLI contoh yang menampilkan detail grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan), lihat. Melihat detail grup replikasi (AWS CLI)

Melihat Grup Replikasi (API) Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) ElastiCache

Untuk contoh ElastiCache API yang menampilkan detail grup replikasi Valkey atau Redis OSS (mode klaster dinonaktifkan), lihat. Melihat detail grup replikasi (ElastiCache API)

Melihat grup replikasi: Valkey atau Redis OSS (Mode Cluster Diaktifkan)

Melihat cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan) (Konsol)

Untuk melihat detail cluster Valkey atau Redis OSS (mode cluster enabled) menggunakan ElastiCache konsol, lihat. Melihat detail untuk cluster Valkey atau Redis OSS (Cluster Mode Enabled) (Konsol)

Melihat cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan) ()AWS CLI

Untuk contoh ElastiCache CLI yang menampilkan detail grup replikasi Valkey atau Redis OSS (mode cluster enabled), lihat. Melihat detail grup replikasi (AWS CLI)

Melihat Cluster (API) Valkey atau Redis OSS (Mode Cluster Diaktifkan) ElastiCache

Untuk contoh ElastiCache API yang menampilkan detail grup replikasi Valkey atau Redis OSS (mode cluster enabled), lihat. Melihat detail grup replikasi (ElastiCache API)

Melihat detail grup replikasi (AWS CLI)

Anda dapat melihat detail untuk grup replikasi menggunakan AWS CLI `describe-replication-groups` perintah. Gunakan parameter opsional berikut untuk menyaring daftar. Menghilangkan parameter akan menampilkan detail hingga 100 grup replikasi.

Parameter Opsional

- `--replication-group-id` – Gunakan parameter ini untuk mencantumkan detail grup replikasi tertentu. Jika grup replikasi yang ditentukan memiliki lebih dari satu grup simpul, hasil yang ditampilkan akan dikelompokkan berdasarkan grup simpul.
- `--max-items` – Gunakan parameter ini untuk membatasi jumlah grup replikasi yang dicantumkan. Nilai dari `--max-items` tidak boleh kurang dari 20 atau lebih besar dari 100.

Example

Kode berikut menampilkan daftar detail hingga 100 grup replikasi.

```
aws elasticache describe-replication-groups
```

Kode berikut menampilkan daftar detail untuk `sample-repl-group`.

```
aws elasticache describe-replication-groups --replication-group-id sample-repl-group
```

Kode berikut menampilkan daftar detail untuk `sample-repl-group`.

```
aws elasticache describe-replication-groups --replication-group-id sample-repl-group
```

Kode berikut mencantumkan detail hingga 25 grup replikasi.

```
aws elasticache describe-replication-groups --max-items 25
```

Output dari operasi ini terlihat seperti berikut ini (format JSON).

```
{
```

```
    "ReplicationGroups": [
      {
        "Status": "available",
        "Description": "test",
        "NodeGroups": [
          {
             "Status": "available",
                "NodeGroupMembers": [
                   {
                      "CurrentRole": "primary",
                      "PreferredAvailabilityZone": "us-west-2a",
                      "CacheNodeId": "0001",
                      "ReadEndpoint": {
                         "Port": 6379,
                         "Address": "rg-name-001.1abc4d.0001.usw2.cache.amazonaws.com"
                      },
                      "CacheClusterId": "rg-name-001"
                   },
                   {
                      "CurrentRole": "replica",
                      "PreferredAvailabilityZone": "us-west-2b",
                      "CacheNodeId": "0001",
                      "ReadEndpoint": {
                         "Port": 6379,
                         "Address": "rg-name-002.1abc4d.0001.usw2.cache.amazonaws.com"
                      },
                      "CacheClusterId": "rg-name-002"
                   },
                   {
                      "CurrentRole": "replica",
                      "PreferredAvailabilityZone": "us-west-2c",
                      "CacheNodeId": "0001",
                      "ReadEndpoint": {
                         "Port": 6379,
                         "Address": "rg-name-003.1abc4d.0001.usw2.cache.amazonaws.com"
                      },
                      "CacheClusterId": "rg-name-003"
                   }
                ],
                "NodeGroupId": "0001",
                "PrimaryEndpoint": {
                   "Port": 6379,
                   "Address": "rg-name.1abc4d.ng.0001.usw2.cache.amazonaws.com"
                }
```

```
            }
         ],
         "ReplicationGroupId": "rg-name",
         "AutomaticFailover": "enabled",
         "SnapshottingClusterId": "rg-name-002",
         "MemberClusters": [
            "rg-name-001",
            "rg-name-002",
            "rg-name-003"
         ],
         "PendingModifiedValues": {}
      },
      {
      ... some output omitted for brevity
      }
   ]
}
```

Untuk informasi selengkapnya, lihat ElastiCache topik AWS CLI untukdescribe-replication-groups.

Melihat detail grup replikasi (ElastiCache API)

Anda dapat melihat detail untuk replikasi menggunakan AWS CLI `DescribeReplicationGroups` operasi. Gunakan parameter opsional berikut untuk menyaring daftar. Menghilangkan parameter akan menampilkan detail hingga 100 grup replikasi.

Parameter Opsional

- `ReplicationGroupId` – Gunakan parameter ini untuk mencantumkan detail grup replikasi tertentu. Jika grup replikasi yang ditentukan memiliki lebih dari satu grup simpul, hasil yang ditampilkan akan dikelompokkan berdasarkan grup simpul.
- `MaxRecords` – Gunakan parameter ini untuk membatasi jumlah grup replikasi yang dicantumkan. Nilai dari `MaxRecords` tidak boleh kurang dari 20 atau lebih besar dari 100. Secara default, nilainya adalah 100.

Example

Kode berikut mencantumkan detail maksimum hingga 100 grup replikasi.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DescribeReplicationGroups
```

```
   &Version=2015-02-02
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &X-Amz-Credential=<credential>
```

Kode berikut menampilkan daftar detail untuk `myReplGroup`.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DescribeReplicationGroups
   &ReplicationGroupId=myReplGroup
   &Version=2015-02-02
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &X-Amz-Credential=<credential>
```

Kode berikut menampilkan daftar detail untuk maksimal 25 klaster.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DescribeReplicationGroups
   &MaxRecords=25
   &Version=2015-02-02
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya, lihat topik referensi ElastiCache API[DescribeReplicationGroups](#).

Menemukan titik akhir grup replikasi

Aplikasi dapat terhubung ke simpul apa pun dalam grup replikasi, asalkan memiliki titik akhir DNS dan nomor port untuk simpul tersebut. Bergantung pada apakah Anda menjalankan grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) atau Valkey atau Redis OSS (mode cluster diaktifkan), Anda akan tertarik pada titik akhir yang berbeda.

Valkey atau Redis OSS (mode cluster dinonaktifkan)

Cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan replika memiliki tiga jenis titik akhir; titik akhir primer, titik akhir pembaca, dan titik akhir node. Titik akhir primer adalah nama DNS yang selalu diresolusi ke simpul primer di klaster. Titik akhir primer tidak terpengaruh oleh perubahan klaster Anda, seperti promosi replika baca ke peran primer. Untuk aktivitas tulis, sebaiknya aplikasi Anda terhubung ke titik akhir primer.

Titik akhir pembaca akan membagi koneksi masuk secara merata ke titik akhir antara semua replika baca dalam sebuah cluster. ElastiCache Faktor lain seperti saat aplikasi membuat koneksi atau cara aplikasi menggunakan atau menggunakan ulang koneksi akan menentukan distribusi lalu lintas. Titik akhir pembaca tetap mengikuti perubahan klaster dalam waktu nyata saat replika ditambahkan atau dihapus. Anda dapat menempatkan beberapa replika baca klaster Redis OSS Anda ElastiCache di AWS Availability Zones (AZ) yang berbeda untuk memastikan ketersediaan titik akhir pembaca yang tinggi.

> **Note**
>
> Titik akhir pembaca bukan penyeimbang beban. Ini adalah catatan DNS yang akan diresolusi sebagai alamat IP dari salah satu simpul replika dengan metode round robin.

Untuk aktivitas baca, aplikasi juga dapat menghubungkan ke simpul mana pun di klaster. Tidak seperti titik akhir primer, titik akhir simpul diresolusi ke titik akhir tertentu. Jika Anda membuat perubahan dalam klaster Anda, seperti menambahkan atau menghapus replika, Anda harus memperbarui titik akhir simpul di aplikasi Anda.

Valkey atau Redis OSS (mode cluster diaktifkan)

Valkey atau Redis OSS (mode cluster enabled) cluster dengan replika, karena mereka memiliki beberapa pecahan (API/CLI: grup node), yang berarti mereka juga memiliki beberapa node primer, memiliki struktur endpoint yang berbeda dari Valkey atau Redis OSS (cluster mode disabled) cluster.

Valkey atau Redis OSS (mode cluster diaktifkan) memiliki titik akhir konfigurasi yang “mengetahui” semua titik akhir primer dan node di cluster. Aplikasi Anda terhubung ke titik akhir konfigurasi. Setiap kali aplikasi Anda menulis atau membaca dari titik akhir konfigurasi cluster, Valkey dan Redis OSS, di belakang layar, tentukan pecahan kunci mana dan titik akhir mana dalam pecahan itu untuk digunakan. Semua proses ini bersifat cukup transparan untuk aplikasi Anda.

Anda dapat menemukan titik akhir untuk klaster menggunakan ElastiCache konsol, the AWS CLI, atau ElastiCache API.

Menemukan Titik Akhir Grup Replikasi

Untuk menemukan titik akhir untuk grup replikasi Anda, lihat salah satu topik berikut:

- Menemukan Titik Akhir Cluster Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol)
- Menemukan Titik Akhir untuk Cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan) (Konsol)
- Menemukan Titik Akhir untuk Grup Replikasi Valkey atau Redis OSS ()AWS CLI
- Menemukan Titik Akhir untuk Grup Replikasi Valkey atau Redis OSS (API) ElastiCache

Mengubah grup replikasi

Batasan Penting

- Saat ini, ElastiCache mendukung modifikasi terbatas dari grup replikasi Valkey atau Redis OSS (mode cluster enabled), misalnya mengubah versi mesin, menggunakan operasi API (`ModifyReplicationGroup`CLI:). `modify-replication-group` Anda dapat memodifikasi jumlah pecahan (grup node) di cluster Valkey atau Redis OSS (mode cluster enabled) dengan operasi API (`ModifyReplicationGroupShardConfiguration`CLI:). `modify-replication-group-shard-configuration` Untuk informasi selengkapnya, lihat Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan).

  Modifikasi lain pada cluster Valkey atau Redis OSS (mode cluster enabled) mengharuskan Anda membuat cluster dengan cluster baru yang menggabungkan perubahan.
- Anda dapat memutakhirkan Valkey atau Redis OSS (mode cluster dinonaktifkan) dan Valkey atau Redis OSS (mode cluster diaktifkan) cluster dan grup replikasi ke versi mesin yang lebih baru. Namun, Anda tidak dapat menurunkan klaster ke versi mesin yang lebih lama kecuali dengan menghapus klaster atau grup replikasi yang sudah ada dan membuatnya lagi. Untuk informasi selengkapnya, lihat Manajemen Versi untuk ElastiCache.
- Anda dapat memutakhirkan cluster Valkey atau Redis OSS yang sudah ada ElastiCache yang menggunakan mode cluster dinonaktifkan untuk menggunakan mode cluster diaktifkan, menggunakan konsol, ModifyReplicationGroupAPI atau perintah modify-replication-groupCLI, seperti yang ditunjukkan pada contoh di bawah ini. Anda juga dapat mengikuti langkah-langkah di Mengubah mode klaster.

Anda dapat memodifikasi pengaturan cluster Valkey atau Redis OSS (mode cluster disabled) menggunakan ElastiCache konsol, API AWS CLI, atau API. ElastiCache Saat ini, ElastiCache mendukung sejumlah modifikasi terbatas pada grup replikasi Valkey atau Redis OSS (mode cluster enabled). Modifikasi lain mengharuskan Anda membuat cadangan grup replikasi saat ini kemudian menggunakan cadangan itu untuk menyemai grup replikasi Valkey atau Redis OSS (mode cluster enabled) baru.

Topik

- Menggunakan AWS Management Console

- Menggunakan AWS CLI
- Menggunakan ElastiCache API

Menggunakan AWS Management Console

Untuk memodifikasi cluster Valkey atau Redis OSS (mode cluster dinonaktifkan), lihat. Memodifikasi cluster ElastiCache

Menggunakan AWS CLI

Berikut ini adalah AWS CLI contoh `modify-replication-group` perintah. Anda dapat menggunakan perintah yang sama untuk membuat perubahan lain pada grup replikasi.

Aktifkan Multi-AZ pada grup replikasi Valkey atau Redis OSS yang ada:

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
   --replication-group-id myReplGroup \
   --multi-az-enabled = true
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
   --replication-group-id myReplGroup ^
   --multi-az-enabled
```

Ubah mode klaster dari dinonaktifkan menjadi diaktifkan:

Untuk mengubah mode klaster dari dinonaktifkan ke diaktifkan, Anda harus terlebih dahulu mengatur mode klaster ke kompatibel. Mode yang kompatibel memungkinkan klien Valkey atau Redis OSS Anda terhubung menggunakan mode cluster diaktifkan dan mode cluster dinonaktifkan. Setelah Anda memigrasikan semua klien Valkey atau Redis OSS untuk menggunakan mode cluster yang diaktifkan, Anda kemudian dapat menyelesaikan konfigurasi mode cluster dan mengatur mode cluster ke diaktifkan.

Untuk Linux, macOS, atau Unix:

Atur ke mode klaster kompatibel.

```
aws elasticache modify-replication-group \
   --replication-group-id myReplGroup \
   --cache-parameter-group-name myParameterGroupName \
   --cluster-mode compatible
```

Atur ke mode klaster diaktifkan.

```
aws elasticache modify-replication-group \
   --replication-group-id myReplGroup \
   --cluster-mode enabled
```

Untuk Windows:

Atur ke mode klaster kompatibel.

```
aws elasticache modify-replication-group ^
   --replication-group-id myReplGroup ^
   --cache-parameter-group-name myParameterGroupName ^
   --cluster-mode compatible
```

Atur ke mode klaster diaktifkan.

```
aws elasticache modify-replication-group ^
   --replication-group-id myReplGroup ^
   --cluster-mode enabled
```

Untuk informasi selengkapnya tentang AWS CLI `modify-replication-group` perintah, lihat modify-replication-group atau Memodifikasi mode cluster di Panduan ElastiCache Pengguna Redis OSS.

Menggunakan ElastiCache API

Operasi ElastiCache API berikut memungkinkan Multi-AZ pada grup replikasi Valkey atau Redis OSS yang ada. Anda dapat menggunakan operasi yang sama untuk membuat perubahan lain pada grup replikasi.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=ModifyReplicationGroup
   &AutomaticFailoverEnabled=true
   &Mutli-AZEnabled=true
```

```
    &ReplicationGroupId=myReplGroup
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &Version=2014-12-01
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Date=20141201T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20141201T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Untuk informasi selengkapnya tentang `ModifyReplicationGroup` operasi ElastiCache API, lihatModifyReplicationGroup.

Menghapus grup replikasi

Jika Anda tidak lagi memerlukan salah satu klaster dengan replika (disebut grup replikasi pada API/CLI), Anda dapat menghapusnya. Ketika Anda menghapus grup replikasi, ElastiCache menghapus semua node dalam grup itu.

Setelah Anda memulainya, operasi ini tidak dapat dihentikan atau dibatalkan.

> **Warning**
>
> - Saat Anda menghapus cluster ElastiCache untuk Redis OSS, snapshot manual Anda dipertahankan. Anda juga memiliki pilihan untuk membuat snapshot terakhir sebelum klaster dihapus. Snapshot cache otomatis tidak dipertahankan.
> - `CreateSnapshot`izin diperlukan untuk membuat snapshot akhir. Tanpa izin ini, panggilan API akan gagal dengan `Access Denied` pengecualian.

Menghapus Grup Replikasi (Konsol)

Untuk menghapus klaster yang memiliki replika, lihat Menghapus cluster di ElastiCache.

Menghapus Grup Replikasi (AWS CLI)

Gunakan perintah delete-replication-group untuk menghapus grup replikasi.

```
aws elasticache delete-replication-group --replication-group-id my-repgroup
```

Muncul prompt yang meminta konfirmasi keputusan Anda. Masukkan y (ya) untuk segera memulai operasi tersebut. Setelah dimulai, proses tidak dapat dibatalkan.

```
  After you begin deleting this replication group, all of its nodes will be deleted as
well.
  Are you sure you want to delete this replication group? [Ny]y

REPLICATIONGROUP  my-repgroup  My replication group  deleting
```

Menghapus grup replikasi (API) ElastiCache

Memanggil DeleteReplicationGroup dengan parameter `ReplicationGroup`.

Example

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DeleteReplicationGroup
   &ReplicationGroupId=my-repgroup
   &Version=2014-12-01
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20141201T220302Z
   &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
   &X-Amz-Date=20141201T220302Z
   &X-Amz-SignedHeaders=Host
   &X-Amz-Expires=20141201T220302Z
   &X-Amz-Credential=<credential>
   &X-Amz-Signature=<signature>
```

> **Note**
>
> Jika Anda menetapkan parameter `RetainPrimaryCluster` ke `true`, semua replika baca akan dihapus, tetapi klaster primer akan dipertahankan.

Mengubah jumlah replika

Anda dapat secara dinamis menambah atau mengurangi jumlah replika baca di grup replikasi Valkey atau Redis OSS Anda menggunakan AWS Management Console, the, atau API. AWS CLI ElastiCache Jika grup replikasi Anda adalah grup replikasi Valkey atau Redis OSS (mode cluster enabled), Anda dapat memilih pecahan (grup simpul) mana yang akan menambah atau mengurangi jumlah replika.

Untuk secara dinamis mengubah jumlah replika dalam grup replikasi Anda, pilih operasi dari tabel berikut yang sesuai dengan situasi Anda.

Untuk Melakukannya	Untuk Valkey atau Redis OSS (mode cluster diaktifkan)	Untuk Valkey atau Redis OSS (mode cluster dinonaktifkan)
Tambahkan replika	Menambah jumlah replika dalam serpihan	Menambah jumlah replika dalam serpihan Menambahkan replika baca untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)
Menghapus replika	Mengurangi jumlah replika dalam serpihan	Mengurangi jumlah replika dalam serpihan Menghapus replika baca untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)

Menambah jumlah replika dalam serpihan

Anda dapat meningkatkan jumlah replika dalam pecahan Valkey atau Redis OSS (mode cluster diaktifkan) atau Valkey atau Redis OSS (mode cluster dinonaktifkan) grup replikasi hingga maksimal lima. Anda dapat melakukannya dengan menggunakan AWS Management Console, the AWS CLI, atau ElastiCache API.

Topik

- Menggunakan AWS Management Console
- Menggunakan AWS CLI
- Menggunakan ElastiCache API

Menggunakan AWS Management Console

Prosedur berikut menggunakan konsol untuk meningkatkan jumlah replika dalam grup replikasi Valkey atau Redis OSS (mode cluster enabled).

Untuk meningkatkan jumlah replika dalam pecahan

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Valkey atau Redis OSS, lalu pilih nama grup replikasi yang ingin Anda tambahkan replika.
3. Pilih kotak untuk setiap serpihan yang replikanya akan ditambah.
4. Pilih Tambahkan replika.
5. Lengkapi halaman Tambahkan Replika ke Serpihan:

    - Untuk Jumlah replikasi/serpihan baru, masukkan jumlah replika yang Anda inginkan untuk semua serpihan yang dipilih. Nilai ini harus lebih besar atau sama dengan Jumlah replika per serpihan saat ini dan kurang dari atau sama dengan lima. Kami menyarankan setidaknya dua replika sebagai syarat minimum.
    - Untuk Availability Zone, pilih salah satu Tidak ada preferensi untuk ElastiCache memilih Availability Zone untuk setiap replika baru, atau Tentukan Availability Zone untuk memilih Availability Zone untuk setiap replika baru.

      Jika Anda memilih Tentukan Zona Ketersediaan, tentukan satu Zona Ketersediaan untuk setiap replika baru menggunakan daftar.

6. Pilih Tambahkan untuk menambahkan replika atau Batalkan untuk membatalkan operasi.

Menggunakan AWS CLI

Untuk meningkatkan jumlah replika dalam pecahan Valkey atau Redis OSS, gunakan `increase-replica-count` perintah dengan parameter berikut:

- `--replication-group-id` – Wajib. Identifikasi grup replikasi yang ingin ditambah jumlah replikanya.
- `--apply-immediately` atau `--no-apply-immediately` – Wajib. Menentukan apakah akan menambah jumlah replika segera (`--apply-immediately`) atau pada periode pemeliharaan berikutnya (`--no-apply-immediately`). Saat ini, `--no-apply-immediately` belum didukung.
- `--new-replica-count` – Opsional. Menentukan jumlah simpul replika yang Anda inginkan saat selesai, maksimum lima. Gunakan parameter ini untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) di mana hanya ada satu grup node atau grup Valkey atau Redis OSS (mode cluster diaktifkan), atau di mana Anda ingin semua grup node memiliki jumlah replika yang sama. Jika nilai ini tidak lebih besar dari jumlah replika saat ini dalam grup simpul, panggilan akan gagal dengan pengecualian.
- `--replica-configuration` – Opsional. Memungkinkan Anda menentukan jumlah replika dan Zona Ketersediaan untuk setiap grup simpul secara terpisah. Gunakan parameter ini untuk grup Valkey atau Redis OSS (mode cluster enabled) di mana Anda ingin mengkonfigurasi setiap grup node secara independen.

  `--replica-configuration` memiliki tiga anggota opsional:

  - `NodeGroupId` – ID empat digit untuk grup simpul yang sedang dikonfigurasi. Untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan), ID pecahan selalu ada. `0001` Untuk menemukan ID grup node (shard) Valkey atau Redis OSS (mode cluster enabled), lihat. Menemukan ID serpihan
  - `NewReplicaCount` – Jumlah replika yang Anda inginkan dalam grup simpul ini pada akhir operasi ini. Nilai ini harus lebih besar dari jumlah replika saat ini, maksimum lima. Jika nilai ini tidak lebih besar dari jumlah replika saat ini dalam grup simpul, panggilan akan gagal dengan pengecualian.
  - `PreferredAvailabilityZones` – Daftar string `PreferredAvailabilityZone` yang menentukan Zona Ketersediaan tempat simpul grup replikasi berada. Jumlah nilai `PreferredAvailabilityZone` harus sama dengan nilai `NewReplicaCount` ditambah

1 untuk memperhitungkan simpul primer. Jika anggota `--replica-configuration` ini dihilangkan, ElastiCache untuk Redis OSS memilih Availability Zone untuk masing-masing replika baru.

> **Important**
>
> Anda harus menyertakan parameter `--new-replica-count` atau `--replica-configuration`, tetapi tidak keduanya sekaligus, dalam panggilan Anda.

Example

Contoh berikut menambah jumlah replika dalam grup replikasi `sample-repl-group` menjadi tiga. Ketika contoh ini selesai, terdapat tiga replika di setiap grup simpul. Nomor ini berlaku apakah ini adalah grup Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan grup node tunggal atau grup Valkey atau Redis OSS (mode cluster diaktifkan) dengan beberapa grup node.

Untuk Linux, macOS, atau Unix:

```
aws elasticache increase-replica-count \
    --replication-group-id sample-repl-group \
    --new-replica-count 3 \
    --apply-immediately
```

Untuk Windows:

```
aws elasticache increase-replica-count ^
    --replication-group-id sample-repl-group ^
    --new-replica-count 3 ^
    --apply-immediately
```

Contoh berikut menambah jumlah replika dalam grup replikasi `sample-repl-group` ke nilai yang ditentukan untuk kedua grup simpul yang ditentukan tersebut. Mengingat bahwa ada beberapa grup node, ini adalah grup replikasi Valkey atau Redis OSS (mode cluster enabled). Saat menentukan `PreferredAvailabilityZones` opsional, jumlah Zona Ketersediaan yang tercantum harus sama dengan nilai dari `NewReplicaCount` ditambah 1. Pendekatan ini memperhitungkan simpul primer untuk grup yang diidentifikasi berdasarkan `NodeGroupId`.

Untuk Linux, macOS, atau Unix:

```
aws elasticache increase-replica-count \
    --replication-group-id sample-repl-group \
    --replica-configuration \
        NodeGroupId=0001,NewReplicaCount=2,PreferredAvailabilityZones=us-east-1a,us-east-1c,us-east-1b \
        NodeGroupId=0003,NewReplicaCount=3,PreferredAvailabilityZones=us-east-1a,us-east-1b,us-east-1c,us-east-1c \
    --apply-immediately
```

Untuk Windows:

```
aws elasticache increase-replica-count ^
    --replication-group-id sample-repl-group ^
    --replica-configuration ^
        NodeGroupId=0001,NewReplicaCount=2,PreferredAvailabilityZones=us-east-1a,us-east-1c,us-east-1b ^
        NodeGroupId=0003,NewReplicaCount=3,PreferredAvailabilityZones=us-east-1a,us-east-1b,us-east-1c,us-east-1c \
    --apply-immediately
```

Untuk informasi selengkapnya tentang meningkatkan jumlah replika menggunakan CLI, increase-replica-countlihat di Referensi Baris Perintah ElastiCache Amazon.

Menggunakan ElastiCache API

Untuk meningkatkan jumlah replika dalam pecahan Valkey atau Redis OSS, gunakan `IncreaseReplicaCount` tindakan dengan parameter berikut:

- `ReplicationGroupId` – Wajib. Identifikasi grup replikasi yang ingin ditambah jumlah replikanya.
- `ApplyImmediately` – Wajib. Menentukan apakah akan menambah jumlah replika segera (`ApplyImmediately=True`) atau pada periode pemeliharaan berikutnya (`ApplyImmediately=False`). Saat ini, `ApplyImmediately=False` belum didukung.
- `NewReplicaCount` – Opsional. Menentukan jumlah simpul replika yang Anda inginkan saat selesai, maksimum lima. Gunakan parameter ini untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) di mana hanya ada satu grup node, atau grup Valkey atau Redis OSS (mode cluster diaktifkan) di mana Anda ingin semua grup node memiliki jumlah replika yang sama. Jika nilai ini tidak lebih besar dari jumlah replika saat ini dalam grup simpul, panggilan akan gagal dengan pengecualian.

- `ReplicaConfiguration` – Opsional. Memungkinkan Anda menentukan jumlah replika dan Zona Ketersediaan untuk setiap grup simpul secara terpisah. Gunakan parameter ini untuk grup Valkey atau Redis OSS (mode cluster enabled) di mana Anda ingin mengkonfigurasi setiap grup node secara independen.

  `ReplicaConfiguraion` memiliki tiga anggota opsional:

  - `NodeGroupId` – ID empat digit untuk grup simpul yang sedang dikonfigurasi. Untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan), ID grup node (shard) selalu. `0001` Untuk menemukan ID grup node (shard) Valkey atau Redis OSS (mode cluster enabled), lihat. Menemukan ID serpihan
  - `NewReplicaCount` – Jumlah replika yang Anda inginkan dalam grup simpul ini pada akhir operasi ini. Nilai ini harus lebih dari jumlah replika saat ini dan maksimum lima. Jika nilai ini tidak lebih besar dari jumlah replika saat ini dalam grup simpul, panggilan akan gagal dengan pengecualian.
  - `PreferredAvailabilityZones` – Daftar string `PreferredAvailabilityZone` yang menentukan Zona Ketersediaan tempat simpul grup replikasi berada. Jumlah nilai `PreferredAvailabilityZone` harus sama dengan nilai `NewReplicaCount` ditambah 1 untuk memperhitungkan simpul primer. Jika anggota `ReplicaConfiguration` ini dihilangkan, ElastiCache untuk Redis OSS memilih Availability Zone untuk masing-masing replika baru.

> **Important**
>
> Anda harus menyertakan parameter `NewReplicaCount` atau `ReplicaConfiguration`, tetapi tidak keduanya sekaligus, dalam panggilan Anda.

Example

Contoh berikut menambah jumlah replika dalam grup replikasi `sample-repl-group` menjadi tiga. Ketika contoh ini selesai, terdapat tiga replika di setiap grup simpul. Nomor ini berlaku apakah ini adalah grup Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan grup node tunggal, atau grup Valkey atau Redis OSS (mode cluster diaktifkan) dengan beberapa grup node.

```
https://elasticache.us-west-2.amazonaws.com/
      ?Action=IncreaseReplicaCount
      &ApplyImmediately=True
      &NewReplicaCount=3
```

```
      &ReplicationGroupId=sample-repl-group
      &Version=2015-02-02
      &SignatureVersion=4
      &SignatureMethod=HmacSHA256
      &Timestamp=20150202T192317Z
      &X-Amz-Credential=<credential>
```

Contoh berikut menambah jumlah replika dalam grup replikasi `sample-repl-group` ke nilai yang ditentukan untuk kedua grup simpul yang ditentukan tersebut. Mengingat bahwa ada beberapa grup node, ini adalah grup replikasi Valkey atau Redis OSS (mode cluster enabled). Saat menentukan `PreferredAvailabilityZones` opsional, jumlah Zona Ketersediaan yang tercantum harus sama dengan nilai dari `NewReplicaCount` ditambah 1. Pendekatan ini memperhitungkan simpul primer, untuk grup yang diidentifikasi berdasarkan `NodeGroupId`.

```
https://elasticache.us-west-2.amazonaws.com/
      ?Action=IncreaseReplicaCount
      &ApplyImmediately=True
      &ReplicaConfiguration.ConfigureShard.1.NodeGroupId=0001
      &ReplicaConfiguration.ConfigureShard.1.NewReplicaCount=2

 &ReplicaConfiguration.ConfigureShard.1.PreferredAvailabilityZones.PreferredAvailabilityZone.1=
east-1a

 &ReplicaConfiguration.ConfigureShard.1.PreferredAvailabilityZones.PreferredAvailabilityZone.2=
east-1c

 &ReplicaConfiguration.ConfigureShard.1.PreferredAvailabilityZones.PreferredAvailabilityZone.3=
east-1b
      &ReplicaConfiguration.ConfigureShard.2.NodeGroupId=0003
      &ReplicaConfiguration.ConfigureShard.2.NewReplicaCount=3

 &ReplicaConfiguration.ConfigureShard.2.PreferredAvailabilityZones.PreferredAvailabilityZone.1=
east-1a

 &ReplicaConfiguration.ConfigureShard.2.PreferredAvailabilityZones.PreferredAvailabilityZone.2=
east-1b

 &ReplicaConfiguration.ConfigureShard.2.PreferredAvailabilityZones.PreferredAvailabilityZone.3=
east-1c

 &ReplicaConfiguration.ConfigureShard.2.PreferredAvailabilityZones.PreferredAvailabilityZone.4=
east-1c
```

```
    &ReplicationGroupId=sample-repl-group
    &Version=2015-02-02
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20150202T192317Z
    &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya tentang meningkatkan jumlah replika yang menggunakan API, lihat IncreaseReplicaCountdi Referensi Amazon ElastiCache API.

Mengurangi jumlah replika dalam serpihan

Anda dapat mengurangi jumlah replika dalam pecahan untuk Valkey atau Redis OSS (mode cluster diaktifkan), atau dalam grup replikasi untuk Valkey atau Redis OSS (mode cluster dinonaktifkan):

- Untuk Valkey atau Redis OSS (mode cluster dinonaktifkan), Anda dapat mengurangi jumlah replika menjadi satu jika Multi-AZ diaktifkan, dan menjadi nol jika tidak diaktifkan.
- Untuk Valkey atau Redis OSS (mode cluster diaktifkan), Anda dapat mengurangi jumlah replika menjadi nol. Namun, Anda tidak dapat melakukan failover ke replika jika simpul primer Anda gagal.

Anda dapat menggunakan AWS Management Console, the AWS CLI atau ElastiCache API untuk mengurangi jumlah replika dalam grup node (shard) atau grup replikasi.

Topik

- Menggunakan AWS Management Console
- Menggunakan AWS CLI
- Menggunakan ElastiCache API

Menggunakan AWS Management Console

Prosedur berikut menggunakan konsol untuk mengurangi jumlah replika dalam grup replikasi Valkey atau Redis OSS (mode cluster enabled).

Untuk mengurangi jumlah replika dalam pecahan Valkey atau Redis OSS

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Valkey atau Redis OSS, lalu pilih nama grup replikasi tempat Anda ingin menghapus replika.
3. Pilih kotak untuk setiap serpihan yang ingin dihapus simpul replika dalamnya.
4. Pilih Hapus replika.
5. Lengkapi halaman Hapus Replika dari Serpihan:

    a. Untuk Jumlah replikasi/serpihan baru, masukkan jumlah replika yang Anda inginkan untuk serpihan yang dipilih. Nilai ini harus lebih besar atau sama dengan 1. Kami menyarankan setidaknya dua replika per serpihan sebagai syarat minimum.

b. Pilih Hapus untuk menghapus replika atau Batalkan untuk membatalkan operasi.

> **Important**
>
> - Jika Anda tidak menentukan node replika yang akan dihapus, ElastiCache untuk Redis OSS secara otomatis memilih node replika untuk dihapus. Saat melakukannya, ElastiCache untuk Redis OSS mencoba mempertahankan arsitektur multi-AZ untuk grup replikasi Anda diikuti dengan mempertahankan replika dengan lag replikasi minimum dengan primer.
> - Anda tidak dapat menghapus primer atau simpul primer dalam grup replikasi. Jika Anda menentukan simpul primer untuk dihapus, operasi akan gagal dengan peristiwa kesalahan yang menunjukkan bahwa simpul primer telah dipilih untuk penghapusan.

Menggunakan AWS CLI

Untuk mengurangi jumlah replika dalam pecahan Valkey atau Redis OSS, gunakan `decrease-replica-count` perintah dengan parameter berikut:

- `--replication-group-id` – Wajib. Identifikasi grup replikasi yang ingin dikurangi jumlah replikanya.
- `--apply-immediately` atau `--no-apply-immediately` – Wajib. Menentukan apakah akan mengurangi jumlah replika dengan segera (`--apply-immediately`) atau pada periode pemeliharaan berikutnya (`--no-apply-immediately`). Saat ini, `--no-apply-immediately` belum didukung.
- `--new-replica-count` – Opsional. Menentukan jumlah simpul replika yang Anda inginkan. Nilai dari `--new-replica-count` harus nilai yang valid dan kurang dari jumlah replika saat ini dalam grup simpul. Untuk nilai minimum yang diizinkan, lihat Mengurangi jumlah replika dalam serpihan. Jika nilai `--new-replica-count` tidak memenuhi persyaratan ini, panggilan akan gagal.
- `--replicas-to-remove` – Opsional. Berisi daftar node yang IDs menentukan node replika untuk dihapus.
- `--replica-configuration` – Opsional. Memungkinkan Anda menentukan jumlah replika dan Zona Ketersediaan untuk setiap grup simpul secara terpisah. Gunakan parameter ini untuk grup Valkey atau Redis OSS (mode cluster enabled) di mana Anda ingin mengkonfigurasi setiap grup node secara independen.

`--replica-configuration` memiliki tiga anggota opsional:

- `NodeGroupId` – ID empat digit untuk grup simpul yang sedang dikonfigurasi. Untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan), ID pecahan selalu ada. `0001` Untuk menemukan ID grup node (shard) Valkey atau Redis OSS (mode cluster enabled), lihat. Menemukan ID serpihan
- `NewReplicaCount` – Parameter opsional yang menentukan jumlah simpul replika yang Anda inginkan. Nilai dari `NewReplicaCount` harus nilai yang valid dan kurang dari jumlah replika saat ini dalam grup simpul. Untuk nilai minimum yang diizinkan, lihat Mengurangi jumlah replika dalam serpihan. Jika nilai `NewReplicaCount` tidak memenuhi persyaratan ini, panggilan akan gagal.
- `PreferredAvailabilityZones` – Daftar string `PreferredAvailabilityZone` yang menentukan Zona Ketersediaan tempat simpul grup replikasi berada. Jumlah nilai `PreferredAvailabilityZone` harus sama dengan nilai `NewReplicaCount` ditambah 1 untuk memperhitungkan simpul primer. Jika anggota `--replica-configuration` ini dihilangkan, ElastiCache untuk Redis OSS memilih Availability Zone untuk masing-masing replika baru.

> **Important**
>
> Anda harus menyertakan satu dan hanya satu parameter `--new-replica-count`, `--replicas-to-remove`, atau `--replica-configuration`.

Example

Contoh berikut menggunakan `--new-replica-count` untuk mengurangi jumlah replika dalam grup replikasi `sample-repl-group` menjadi satu. Saat contoh ini selesai, terdapat satu replika di setiap grup simpul. Nomor ini berlaku apakah ini adalah grup Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan grup node tunggal atau grup Valkey atau Redis OSS (mode cluster diaktifkan) dengan beberapa grup node.

Untuk Linux, macOS, atau Unix:

```
aws elasticache decrease-replica-count
    --replication-group-id sample-repl-group \
    --new-replica-count 1 \
    --apply-immediately
```

Untuk Windows:

```
aws elasticache decrease-replica-count ^
    --replication-group-id sample-repl-group ^
    --new-replica-count 1 ^
    --apply-immediately
```

Contoh berikut mengurangi jumlah replika dalam grup replikasi `sample-repl-group` dengan menghapus dua replika yang ditentukan (`0001` dan `0003`) dari grup simpul.

Untuk Linux, macOS, atau Unix:

```
aws elasticache decrease-replica-count \
    --replication-group-id sample-repl-group \
    --replicas-to-remove 0001,0003 \
    --apply-immediately
```

Untuk Windows:

```
aws elasticache decrease-replica-count ^
    --replication-group-id sample-repl-group ^
    --replicas-to-remove 0001,0003 \
    --apply-immediately
```

Contoh berikut menggunakan `--replica-configuration` untuk mengurangi jumlah replika dalam grup replikasi `sample-repl-group` dengan nilai yang ditentukan untuk dua grup simpul yang ditentukan. Mengingat bahwa ada beberapa grup node, ini adalah grup replikasi Valkey atau Redis OSS (mode cluster enabled). Saat menentukan `PreferredAvailabilityZones` opsional, jumlah Zona Ketersediaan yang tercantum harus sama dengan nilai dari `NewReplicaCount` ditambah 1. Pendekatan ini memperhitungkan simpul primer untuk grup yang diidentifikasi berdasarkan `NodeGroupId`.

Untuk Linux, macOS, atau Unix:

```
aws elasticache decrease-replica-count \
    --replication-group-id sample-repl-group \
    --replica-configuration \
        NodeGroupId=0001,NewReplicaCount=1,PreferredAvailabilityZones=us-east-1a,us-
east-1c \
```

```
        NodeGroupId=0003,NewReplicaCount=2,PreferredAvailabilityZones=us-east-1a,us-
east-1b,us-east-1c \
    --apply-immediately
```

Untuk Windows:

```
aws elasticache decrease-replica-count ^
    --replication-group-id sample-repl-group ^
    --replica-configuration ^
        NodeGroupId=0001,NewReplicaCount=2,PreferredAvailabilityZones=us-east-1a,us-
east-1c ^
        NodeGroupId=0003,NewReplicaCount=3,PreferredAvailabilityZones=us-east-1a,us-
east-1b,us-east-1c \
    --apply-immediately
```

Untuk informasi selengkapnya tentang mengurangi jumlah replika menggunakan CLI, lihat di Referensi Baris decrease-replica-countPerintah Amazon. ElastiCache

Menggunakan ElastiCache API

Untuk mengurangi jumlah replika dalam pecahan Valkey atau Redis OSS, gunakan `DecreaseReplicaCount` tindakan dengan parameter berikut:

- `ReplicationGroupId` – Wajib. Identifikasi grup replikasi yang ingin dikurangi jumlah replikanya.
- `ApplyImmediately` – Wajib. Menentukan apakah akan mengurangi jumlah replika dengan segera (`ApplyImmediately=True`) atau pada periode pemeliharaan berikutnya (`ApplyImmediately=False`). Saat ini, `ApplyImmediately=False` belum didukung.
- `NewReplicaCount` – Opsional. Menentukan jumlah simpul replika yang Anda inginkan. Nilai dari `NewReplicaCount` harus nilai yang valid dan kurang dari jumlah replika saat ini dalam grup simpul. Untuk nilai minimum yang diizinkan, lihat Mengurangi jumlah replika dalam serpihan. Jika nilai `--new-replica-count` tidak memenuhi persyaratan ini, panggilan akan gagal.
- `ReplicasToRemove` – Opsional. Berisi daftar node yang IDs menentukan node replika untuk dihapus.
- `ReplicaConfiguration` – Opsional. Berisi daftar grup simpul yang memungkinkan Anda menentukan jumlah replika dan Zona Ketersediaan untuk setiap grup simpul secara terpisah. Gunakan parameter ini untuk grup Valkey atau Redis OSS (mode cluster enabled) di mana Anda ingin mengkonfigurasi setiap grup node secara independen.

  `ReplicaConfiguraion` memiliki tiga anggota opsional:

- `NodeGroupId` – ID empat digit untuk grup simpul yang sedang dikonfigurasi. Untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan), ID grup node selalu. `0001` Untuk menemukan ID grup node (shard) Valkey atau Redis OSS (mode cluster enabled), lihat. Menemukan ID serpihan
- `NewReplicaCount` – Jumlah replika yang Anda inginkan dalam grup simpul ini pada akhir operasi ini. Nilai ini harus kurang dari jumlah replika saat ini hingga minimum 1 jika Multi-AZ diaktifkan atau hingga 0 jika Multi-AZ dengan Failover Otomatis tidak diaktifkan. Jika nilai ini tidak lebih kecil dari jumlah replika saat ini dalam grup simpul, maka panggilan akan gagal dengan pengecualian.
- `PreferredAvailabilityZones` – Daftar string `PreferredAvailabilityZone` yang menentukan Zona Ketersediaan tempat simpul grup replikasi berada. Jumlah nilai `PreferredAvailabilityZone` harus sama dengan nilai `NewReplicaCount` ditambah 1 untuk memperhitungkan simpul primer. Jika anggota `ReplicaConfiguration` ini dihilangkan, ElastiCache untuk Redis OSS memilih Availability Zone untuk masing-masing replika baru.

> **Important**
>
> Anda harus menyertakan satu dan hanya satu parameter `NewReplicaCount`, `ReplicasToRemove`, atau `ReplicaConfiguration`.

Example

Contoh berikut menggunakan `NewReplicaCount` untuk mengurangi jumlah replika dalam grup replikasi `sample-repl-group` menjadi satu. Saat contoh ini selesai, terdapat satu replika di setiap grup simpul. Nomor ini berlaku apakah ini adalah grup Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan grup node tunggal atau grup Valkey atau Redis OSS (mode cluster diaktifkan) dengan beberapa grup node.

```
https://elasticache.us-west-2.amazonaws.com/
      ?Action=DecreaseReplicaCount
      &ApplyImmediately=True
      &NewReplicaCount=1
      &ReplicationGroupId=sample-repl-group
      &Version=2015-02-02
      &SignatureVersion=4
      &SignatureMethod=HmacSHA256
      &Timestamp=20150202T192317Z
```

```
      &X-Amz-Credential=<credential>
```

Contoh berikut mengurangi jumlah replika dalam grup replikasi `sample-repl-group` dengan menghapus dua replika yang ditentukan (`0001` dan `0003`) dari grup simpul.

```
https://elasticache.us-west-2.amazonaws.com/
      ?Action=DecreaseReplicaCount
      &ApplyImmediately=True
      &ReplicasToRemove.ReplicaToRemove.1=0001
      &ReplicasToRemove.ReplicaToRemove.2=0003
      &ReplicationGroupId=sample-repl-group
      &Version=2015-02-02
      &SignatureVersion=4
      &SignatureMethod=HmacSHA256
      &Timestamp=20150202T192317Z
      &X-Amz-Credential=<credential>
```

Contoh berikut menggunakan `ReplicaConfiguration` untuk mengurangi jumlah replika dalam grup replikasi `sample-repl-group` dengan nilai yang ditentukan untuk dua grup simpul yang ditentukan. Mengingat bahwa ada beberapa grup node, ini adalah grup replikasi Valkey atau Redis OSS (mode cluster enabled). Saat menentukan `PreferredAvailabilityZones` opsional, jumlah Zona Ketersediaan yang tercantum harus sama dengan nilai dari `NewReplicaCount` ditambah 1. Pendekatan ini memperhitungkan simpul primer untuk grup yang diidentifikasi berdasarkan `NodeGroupId`.

```
https://elasticache.us-west-2.amazonaws.com/
      ?Action=DecreaseReplicaCount
      &ApplyImmediately=True
      &ReplicaConfiguration.ConfigureShard.1.NodeGroupId=0001
      &ReplicaConfiguration.ConfigureShard.1.NewReplicaCount=1

 &ReplicaConfiguration.ConfigureShard.1.PreferredAvailabilityZones.PreferredAvailabilityZone.1=
east-1a

 &ReplicaConfiguration.ConfigureShard.1.PreferredAvailabilityZones.PreferredAvailabilityZone.2=
east-1c
      &ReplicaConfiguration.ConfigureShard.2.NodeGroupId=0003
      &ReplicaConfiguration.ConfigureShard.2.NewReplicaCount=2

 &ReplicaConfiguration.ConfigureShard.2.PreferredAvailabilityZones.PreferredAvailabilityZone.1=
east-1a
```

```
 &ReplicaConfiguration.ConfigureShard.2.PreferredAvailabilityZones.PreferredAvailabilityZone.2=
east-1b

 &ReplicaConfiguration.ConfigureShard.2.PreferredAvailabilityZones.PreferredAvailabilityZone.4=
east-1c
      &ReplicationGroupId=sample-repl-group
      &Version=2015-02-02
      &SignatureVersion=4
      &SignatureMethod=HmacSHA256
      &Timestamp=20150202T192317Z
      &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya tentang mengurangi jumlah replika yang menggunakan API, lihat DecreaseReplicaCountdi Referensi Amazon ElastiCache API.

Menambahkan replika baca untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)

Informasi dalam topik berikut hanya berlaku untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan).

Saat lalu lintas baca Anda meningkat, sebaiknya sebarkan operasi baca tersebut ke lebih banyak simpul dan kurangi kepadatan operasi baca pada salah satu simpul. Dalam topik ini, Anda dapat menemukan cara menambahkan replika baca ke cluster Valkey atau Redis OSS (mode cluster dinonaktifkan).

Grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) dapat memiliki maksimal lima replika baca. Jika Anda mencoba menambahkan replika baca ke grup replikasi yang sudah memiliki lima replika baca, maka operasi ini akan gagal.

Untuk informasi tentang menambahkan replika ke grup replikasi Valkey atau Redis OSS (mode cluster enabled), lihat berikut ini:

- Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan)
- Menambah jumlah replika dalam serpihan

Anda dapat menambahkan replika baca ke cluster Valkey atau Redis OSS (mode cluster disabled) menggunakan ElastiCache Console, the AWS CLI, atau API. ElastiCache

Topik terkait

- Menambahkan node ke ElastiCache cluster
- Menambahkan replika baca ke grup replikasi (AWS CLI)
- Menambahkan replika baca ke grup replikasi menggunakan API

Menambahkan replika baca ke grup replikasi (AWS CLI)

Untuk menambahkan replika baca ke grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan), gunakan AWS CLI `create-cache-cluster` perintah, dengan parameter `--replication-group-id` untuk menentukan grup replikasi mana yang akan ditambahkan cluster (node).

Contoh berikut membuat klaster `my-read replica` dan menambahkannya ke grup replikasi `my-replication-group`. Jenis simpul, grup parameter, grup keamanan, periode pemeliharaan, dan pengaturan lain untuk replika baca sama seperti yang digunakan untuk simpul lain, `my-replication-group`.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-cache-cluster \
      --cache-cluster-id my-read-replica \
      --replication-group-id my-replication-group
```

Untuk Windows:

```
aws elasticache create-cache-cluster ^
      --cache-cluster-id my-read-replica ^
      --replication-group-id my-replication-group
```

Untuk informasi selengkapnya tentang menambahkan replika baca menggunakan CLI, create-cache-cluster lihat di Referensi Baris Perintah ElastiCache Amazon.

Menambahkan replika baca ke grup replikasi menggunakan API

Untuk menambahkan replika baca ke grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan), gunakan ElastiCache `CreateCacheCluster` operasi, dengan parameter `ReplicationGroupId` untuk menentukan grup replikasi mana yang akan ditambahkan cluster (node).

Contoh berikut membuat klaster `myReadReplica` dan menambahkannya ke grup replikasi `myReplicationGroup`. Jenis simpul, grup parameter, grup keamanan, periode pemeliharaan, dan pengaturan lain untuk replika baca sama seperti yang digunakan untuk simpul lain, `myReplicationGroup`.

```
https://elasticache.us-west-2.amazonaws.com/
      ?Action=CreateCacheCluster
      &CacheClusterId=myReadReplica
      &ReplicationGroupId=myReplicationGroup
      &Version=2015-02-02
      &SignatureVersion=4
      &SignatureMethod=HmacSHA256
      &Timestamp=20150202T192317Z
      &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya tentang menambahkan replika baca menggunakan API, lihat CreateCacheCluster di Referensi Amazon ElastiCache API.

Menghapus replika baca untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)

Informasi dalam topik berikut hanya berlaku untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan).

Saat lalu lintas baca di grup replikasi Valkey atau Redis OSS Anda berubah, Anda mungkin ingin menambah atau menghapus replika baca. Menghapus simpul dari grup replikasi adalah sama seperti menghapus klaster saja, meskipun terdapat pembatasan:

- Anda tidak dapat menghapus primer dari grup replikasi. Jika Anda ingin menghapus replika primer, lakukan hal berikut:

  1. Promosikan replika baca menjadi primer. Untuk informasi tentang mempromosikan replika baca menjadi primer, lihat Mempromosikan replika baca ke primer, untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan).
  2. Hapus replika primer lama. Untuk batasan dari metode ini, lihat poin berikutnya.
- Jika Multi-AZ diaktifkan pada grup replikasi, Anda tidak dapat menghapus replika baca terakhir dari grup replikasi tersebut. Dalam kasus ini, lakukan hal berikut:

  1. Ubah grup replikasi dengan menonaktifkan Multi-AZ. Untuk informasi selengkapnya, lihat Mengubah grup replikasi.
  2. Hapus replika baca.

Anda dapat menghapus replika baca dari grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) menggunakan ElastiCache konsol, AWS CLI for ElastiCache, atau API. ElastiCache

Untuk petunjuk cara menghapus cluster dari grup replikasi Valkey atau Redis OSS, lihat berikut ini:

- Menggunakan AWS Management Console
- Menggunakan AWS CLI untuk menghapus ElastiCache cluster
- Menggunakan ElastiCache API
- Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan)
- Mengurangi jumlah replika dalam serpihan

Mempromosikan replika baca ke primer, untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan)

Informasi dalam topik berikut hanya berlaku untuk grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan).

Anda dapat mempromosikan replika baca Valkey atau Redis OSS (mode cluster dinonaktifkan) ke primer menggunakan AWS Management Console, the AWS CLI, atau API. ElastiCache Anda tidak dapat menaikkan replika baca ke primer jika Multi-AZ dengan Failover Otomatis diaktifkan pada grup replikasi . Untuk mempromosikan replika Valkey atau Redis OSS (mode cluster dinonaktifkan) ke primer pada grup replikasi berkemampuan multi-AZ, lakukan hal berikut:

1. Ubah grup replikasi untuk menonaktifkan Multi-AZ (melakukan hal ini tidak mensyaratkan semua klaster Anda berada dalam Zona Ketersediaan yang sama). Untuk informasi selengkapnya, lihat Mengubah grup replikasi.
2. Promosikan replika baca menjadi primer.
3. Ubah grup replikasi untuk mengaktifkan kembali Multi-AZ.

Multi-AZ tidak tersedia pada grup replikasi yang menjalankan Redis OSS 2.6.13 atau yang lebih lama.

Menggunakan AWS Management Console

Prosedur berikut menggunakan konsol untuk mempromosikan simpul replika menjadi primer.

Untuk mempromosikan replika baca menjadi primer (konsol)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Jika replika yang ingin Anda promosikan adalah anggota grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) tempat multi-AZ diaktifkan, modifikasi grup replikasi untuk menonaktifkan Multi-AZ sebelum Anda melanjutkan. Untuk informasi selengkapnya, lihat Mengubah grup replikasi.
3. Pilih Valkey atau Redis OSS, lalu dari daftar cluster, pilih grup replikasi yang ingin Anda modifikasi. Grup replikasi ini harus menjalankan mesin "Redis", bukan mesin "Redis Berklaster", dan harus memiliki dua atau lebih simpul.
4. Dari daftar simpul, pilih simpul replika yang ingin dipromosikan menjadi primer, kemudian untuk Tindakan, pilih Promosikan.

5. Pada kotak dialog Mempromosikan Replika Baca, lakukan hal berikut:

   a. Untuk Terapkan Segera, pilih Ya untuk mempromosikan replika baca segera, atau Tidak untuk mempromosikannya pada periode pemeliharaan berikutnya untuk klaster.

   b. Pilih Promosikan untuk mempromosikan replika baca atau Batalkan untuk membatalkan operasi.

6. Jika klaster Multi-AZ diaktifkan sebelum Anda memulai proses promosi, tunggu hingga status grup replikasi menjadi tersedia, lalu ubah klaster untuk mengaktifkan kembali Multi-AZ. Untuk informasi selengkapnya, lihat Mengubah grup replikasi.

Menggunakan AWS CLI

Anda tidak dapat mempromosikan replika baca menjadi primer jika grup replikasi mengaktifkan Multi-AZ. Dalam beberapa kasus, replika yang ingin dipromosikan mungkin adalah anggota grup replikasi yang mengaktifkan Multi-AZ. Dalam kasus ini, Anda harus mengubah grup replikasi untuk menonaktifkan Multi-AZ sebelum Anda melanjutkan. Untuk melakukan tindakan ini, semua klaster tidak harus berada dalam Zona Ketersediaan yang sama. Untuk informasi selengkapnya tentang mengubah grup replikasi, lihat Mengubah grup replikasi.

AWS CLI Perintah berikut memodifikasi grup replikasi`sample-repl-group`, membuat replika `my-replica-1` baca utama dalam grup replikasi.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
   --replication-group-id sample-repl-group \
   --primary-cluster-id my-replica-1
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
   --replication-group-id sample-repl-group ^
   --primary-cluster-id my-replica-1
```

Untuk informasi selengkapnya tentang memodifikasi grup replikasi, lihat modify-replication-group di Referensi Baris ElastiCache Perintah Amazon.

Menggunakan ElastiCache API

Anda tidak dapat mempromosikan replika baca menjadi primer jika grup replikasi mengaktifkan Multi-AZ. Dalam beberapa kasus, replika yang ingin dipromosikan mungkin adalah anggota grup replikasi yang mengaktifkan Multi-AZ. Dalam kasus ini, Anda harus mengubah grup replikasi untuk menonaktifkan Multi-AZ sebelum Anda melanjutkan. Untuk melakukan tindakan ini, semua klaster tidak harus berada dalam Zona Ketersediaan yang sama. Untuk informasi selengkapnya tentang mengubah grup replikasi, lihat Mengubah grup replikasi.

Tindakan ElastiCache API berikut memodifikasi grup replikasi`myReplGroup`, menjadikan replika baca sebagai yang `myReplica-1` utama dalam grup replikasi.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=ModifyReplicationGroup
   &ReplicationGroupId=myReplGroup
   &PrimaryClusterId=myReplica-1
   &Version=2014-12-01
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20141201T220302Z
   &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
   &X-Amz-Date=20141201T220302Z
   &X-Amz-SignedHeaders=Host
   &X-Amz-Expires=20141201T220302Z
   &X-Amz-Credential=<credential>
   &X-Amz-Signature=<signature>
```

Untuk informasi selengkapnya tentang memodifikasi grup replikasi, lihat ModifyReplicationGroup di Referensi Amazon ElastiCache API.

Mengelola pemeliharaan ElastiCache cluster

Setiap klaster memiliki periode pemeliharaan mingguan saat perubahan sistem akan diterapkan. Dengan Valkey dan Redis OSS, grup replikasi memiliki jendela pemeliharaan mingguan yang sama. Jika Anda tidak menentukan jendela pemeliharaan yang disukai saat membuat atau memodifikasi klaster atau grup replikasi, tetapkan ElastiCache jendela pemeliharaan 60 menit dalam jendela pemeliharaan wilayah Anda pada hari yang dipilih secara acak dalam seminggu.

Periode pemeliharaan 60 menit dipilih secara acak dari blok waktu 8 jam per wilayah. Tabel berikut menampilkan daftar blok waktu untuk setiap wilayah yang akan digunakan untuk menetapkan periode

pemeliharaan default. Anda dapat memilih periode pemeliharaan yang diinginkan di luar blok periode pemeliharaan wilayah.

Kode Wilayah	Nama wilayah	Periode Pemeliharaan Wilayah
ap-northeast-1	Wilayah Asia Pasifik (Tokyo)	13.00–21.00 UTC
ap-northeast-2	Wilayah Asia Pasifik (Seoul)	12.00–20.00 UTC
ap-northeast-3	Wilayah Asia Pasifik (Osaka)	12.00–20.00 UTC
ap-southeast-3	Wilayah Asia Pasifik (Jakarta)	14.00–22.00 UTC
ap-south-1	Wilayah Asia Pasifik (Mumbai)	17.30–01.30 UTC
ap-southeast-1	Wilayah Asia Pasifik (Singapura)	14.00–22.00 UTC
cn-north-1	Wilayah Tiongkok (Beijing)	14.00–22.00 UTC
cn-northwest-1	Wilayah Tiongkok (Ningxia)	14.00–22.00 UTC
ap-east-1	Wilayah Asia Pasifik (Hong Kong)	13.00–21.00 UTC
ap-southeast-2	Wilayah Asia Pasifik (Sydney)	12.00–20.00 UTC
eu-west-3	Wilayah Eropa (Paris)	23.59–07.29 UTC
af-south-1	Wilayah Afrika (Cape Town)	13.00–21.00 UTC
eu-central-1	Wilayah Eropa (Frankfurt)	23.00–07.00 UTC
eu-west-1	Wilayah Eropa (Irlandia)	22.00–06.00 UTC
eu-west-2	Wilayah Eropa (London)	23.00–07.00 UTC
me-south-1	Wilayah Timur Tengah (Bahrain)	13.00–21.00 UTC
me-central-1	Wilayah Timur Tengah (UEA)	13.00–21.00 UTC
eu-south-1	Wilayah Eropa (Milan)	21.00–05.00 UTC

Kode Wilayah	Nama wilayah	Periode Pemeliharaan Wilayah
sa-east-1	Wilayah South America (São Paulo)	01.00–09.00 UTC
us-east-1	Wilayah AS Timur (Virginia Utara)	03.00–11.00 UTC
us-east-2	Wilayah AS Timur (Ohio)	04.00–12.00 UTC
us-gov-west-1	AWS GovCloud (US) wilayah	06.00–14.00 UTC
us-west-1	Wilayah US West (N. California)	06.00–14.00 UTC
us-west-2	Wilayah US West (Oregon)	06.00–14.00 UTC

Mengubah jendela pemeliharaan klaster atau grup replikasi

Periode pemeliharaan harus berada dalam waktu penggunaan terendah, sehingga kemungkinan memerlukan perubahan dari waktu ke waktu. Anda dapat mengubah klaster atau grup replikasi Anda untuk menentukan rentang waktu hingga durasi 24 jam saat aktivitas pemeliharaan yang Anda minta akan dilakukan. Setiap perubahan klaster yang Anda minta yang ditangguhkan atau tertunda akan terjadi dalam kurun waktu ini.

> **Note**
>
> Jika Anda ingin menerapkan modifikasi tipe node upgrade and/or mesin segera menggunakan AWS Management Console pilih kotak Terapkan sekarang. Jika tidak, perubahan ini akan diterapkan selama periode pemeliharaan terjadwal berikutnya. Untuk menggunakan API, lihat modify-replication-groupatau modify-cache-cluster.

Informasi selengkapnya

Untuk informasi tentang periode pemeliharaan dan penggantian simpul, lihat yang berikut ini:

- ElastiCache Pemeliharaan —FAQ tentang pemeliharaan dan penggantian simpul
- Mengganti node (Memcached)—Mengelola penggantian node untuk Memcached
- Memodifikasi cluster ElastiCache —Mengubah periode pemeliharaan klaster

- Mengganti node (Valkey dan Redis OSS)—Mengelola penggantian simpul
- Mengubah grup replikasi—Mengubah periode pemeliharaan grup replikasi

Mengkonfigurasi parameter mesin menggunakan grup ElastiCache parameter

Amazon ElastiCache menggunakan parameter untuk mengontrol properti runtime node dan cluster Anda. Secara umum, versi mesin yang lebih baru mencakup parameter tambahan untuk mendukung fungsionalitas yang lebih baru. Untuk tabel parameter Memcached, lihat. Parameter spesifik Memcached Untuk tabel parameter Valkey dan Redis OSS, lihat. Parameter Valkey dan Redis OSS

Seperti yang Anda harapkan, beberapa nilai parameter, seperti `maxmemory`, ditentukan oleh mesin dan jenis simpul. Untuk tabel nilai parameter Memcached ini menurut jenis node, lihat. Parameter khusus jenis simpul Memcached Untuk tabel nilai parameter Valkey dan Redis OSS ini berdasarkan tipe node, lihat. Parameter spesifik tipe node Redis OSS

> **Note**
>
> Untuk daftar parameter kustom Memcached, lihat Parameter Khusus Memcached.

Topik

- Manajemen parameter di ElastiCache
- Tingkatan grup parameter cache di ElastiCache
- Membuat grup ElastiCache parameter
- Daftar grup ElastiCache parameter berdasarkan nama
- Daftar nilai kelompok ElastiCache parameter
- Memodifikasi grup ElastiCache parameter
- Menghapus grup ElastiCache parameter
- Parameter spesifik mesin

Manajemen parameter di ElastiCache

ElastiCache parameter dikelompokkan bersama ke dalam kelompok parameter bernama untuk manajemen parameter yang lebih mudah. Grup parameter merepresentasikan kombinasi nilai tertentu untuk parameter yang diteruskan ke perangkat lunak mesin selama pengaktifan. Nilai ini menentukan perilaku berbagai proses mesin di setiap simpul pada saat runtime. Nilai parameter pada grup parameter tertentu berlaku untuk semua simpul yang terkait dengan grup ini, terlepas dari klaster yang memiliki simpul tersebut.

Untuk menyesuaikan performa klaster, Anda dapat mengubah beberapa nilai parameter atau mengubah grup parameter klaster.

- Anda tidak dapat mengubah atau menghapus grup parameter default. Jika membutuhkan nilai parameter kustom, Anda harus membuat grup parameter kustom.
- Untuk Memcached, keluarga grup parameter dan cluster yang Anda tetapkan harus kompatibel. Misalnya, jika klaster menjalankan Memcached versi 1.4.8, Anda hanya dapat menggunakan grup parameter, default atau kustom, dari keluarga Memcached 1.4.

  Untuk Redis OSS, keluarga grup parameter dan cluster yang Anda tetapkan harus kompatibel. Misalnya, jika cluster Anda menjalankan Redis OSS versi 3.2.10, Anda hanya dapat menggunakan grup parameter, default atau kustom, dari keluarga Redis OSS 3.2.
- Jika Anda mengubah grup parameter klaster, nilai untuk setiap parameter yang dapat diubah secara bersyarat harus sama di kedua grup parameter baru dan saat ini.
- Untuk Memcached, saat Anda mengubah parameter cluster, perubahan diterapkan ke cluster segera. Hal ini berlaku terlepas dari apakah Anda mengubah grup parameter klaster itu sendiri atau nilai parameter dalam grup parameter klaster. Untuk menentukan waktu penerapan perubahan parameter tertentu, lihat kolom Perubahan Berlaku dalam tabel untuk Parameter spesifik Memcached. Untuk informasi tentang mem-boot ulang simpul klaster, lihat Mem-boot ulang klaster.
- Untuk Redis OSS, ketika Anda mengubah parameter cluster, perubahan diterapkan ke cluster baik segera atau, dengan pengecualian dicatat berikut, setelah node cluster di-boot ulang. Hal ini berlaku terlepas dari apakah Anda mengubah grup parameter klaster itu sendiri atau nilai parameter dalam grup parameter klaster. Untuk menentukan waktu penerapan perubahan parameter tertentu, lihat kolom Perubahan Berlaku dalam tabel untuk Parameter Valkey dan Redis OSS.

  Untuk informasi lebih lanjut tentang me-reboot node Valkey atau Redis OSS, lihat. Mem-boot ulang node

Perubahan parameter Valkey atau Redis OSS (Mode Cluster Diaktifkan)

Jika Anda membuat perubahan pada parameter berikut pada cluster Valkey atau Redis OSS (mode cluster enabled), ikuti langkah-langkah berikutnya.

- activerehashing
- databases

1. Buat cadangan manual klaster Anda. Lihat Membuat cadangan manual.
2. Hapus klaster . Lihat Menghapus klaster.
3. Pulihkan klaster menggunakan grup parameter dan cadangan yang sudah diubah untuk melakukan seeding klaster baru. Lihat Melakukan pemulihan dari cadangan ke dalam cache baru.

Perubahan pada parameter lain tidak memerlukan tindakan ini.

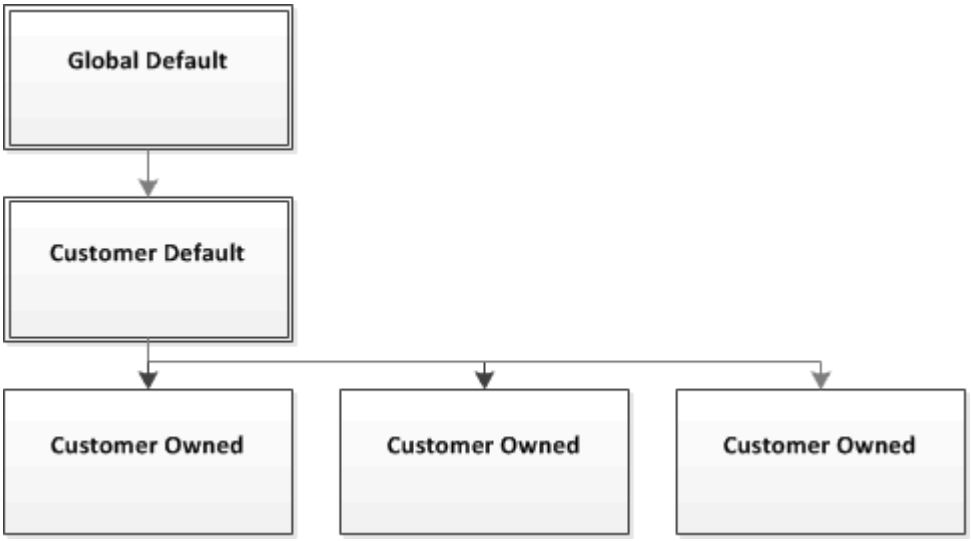
- Anda dapat mengaitkan grup parameter dengan datastores global Valkey dan Redis OSS. Datastores global adalah kumpulan dari satu atau lebih cluster yang menjangkau Wilayah. AWS Dalam hal ini, grup parameter digunakan bersama-sama oleh semua klaster yang membentuk penyimpanan data global. Perubahan apa pun ke grup parameter klaster primer akan direplikasi ke semua klaster lain di penyimpanan data global. Untuk informasi selengkapnya, lihat Replikasi lintas AWS Wilayah menggunakan datastores global.

  Anda dapat memeriksa apakah grup parameter merupakan bagian dari penyimpanan data global dengan melihatnya di lokasi berikut:

  - Pada ElastiCache konsol pada halaman Parameter Groups, atribut yes/no Global
  - yes/no `IsGlobal`Properti operasi CacheParameterGroupAPI

Tingkatan grup parameter cache di ElastiCache

Amazon ElastiCache memiliki tiga tingkatan grup parameter cache seperti yang ditunjukkan berikut.



Tingkatan grup ElastiCache parameter Amazon

Default Global

Grup parameter root tingkat atas untuk semua ElastiCache pelanggan Amazon di wilayah tersebut.

Grup parameter cache default global:

- Dicadangkan untuk ElastiCache dan tidak tersedia untuk pelanggan.

Default Pelanggan

Salinan grup parameter cache Default Global yang dibuat untuk penggunaan pelanggan.

Grup parameter cache Default Pelanggan:

- Dibuat dan dimiliki oleh ElastiCache.
- Tersedia bagi pelanggan untuk digunakan sebagai grup parameter cache untuk setiap klaster yang menjalankan versi mesin yang didukung oleh grup parameter cache ini.
- Tidak dapat diedit oleh pelanggan.

Milik Pelanggan

Salinan grup parameter cache Default Pelanggan. Grup parameter cache Milik Pelanggan dibuat setiap kali pelanggan membuat grup parameter cache.

Grup parameter cache Milik Pelanggan:

- Dibuat dan dimiliki oleh pelanggan.
- Dapat ditetapkan untuk salah satu klaster pelanggan yang kompatibel.
- Dapat diubah oleh pelanggan untuk membuat grup parameter cache kustom.

  Tidak semua nilai parameter dapat diubah. Untuk informasi selengkapnya tentang nilai Memcached, lihat. Parameter spesifik Memcached Untuk informasi lebih lanjut tentang nilai Valkey dan Redis OSS, lihat. Parameter Valkey dan Redis OSS

Membuat grup ElastiCache parameter

Anda perlu membuat grup parameter baru jika ada satu atau beberapa nilai parameter yang ingin Anda ubah dari nilai default. Anda dapat membuat grup parameter menggunakan ElastiCache konsol, the AWS CLI, atau ElastiCache API.

Membuat grup ElastiCache parameter (Konsol)

Prosedur berikut menunjukkan cara membuat grup parameter menggunakan konsol ElastiCache.

Untuk membuat grup parameter menggunakan ElastiCache konsol

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Untuk melihat daftar semua grup parameter yang tersedia, di panel navigasi sebelah kiri pilih Grup Parameter.
3. Untuk membuat grup parameter, pilih Buat Grup Parameter.

   Layar Buat Grup Parameter akan muncul.
4. Dari daftar Keluarga, pilih grup parameter yang akan menjadi templat untuk grup parameter Anda.

   Keluarga grup parameter, seperti memcached1.4 atau redis3.2 mendefinisikan parameter aktual dalam grup parameter Anda dan nilai awalnya. Keluarga grup parameter harus sesuai dengan mesin dan versi klaster.

5. Di kotak Nama, ketik nama unik untuk grup parameter ini.

   Saat membuat klaster atau mengubah grup parameter klaster, Anda akan memilih grup parameter berdasarkan namanya. Oleh karena itu, kami merekomendasikan agar namanya informatif dan dapat mengidentifikasi keluarga grup parameter.

   Batasan penamaan grup parameter adalah sebagai berikut:

   - Harus diawali dengan huruf ASCII.
   - Hanya dapat berisi huruf ASCII, angka, dan tanda hubung.
   - Harus memiliki panjang 1-255 karakter.
   - Tidak boleh berisi dua tanda hubung berurutan.
   - Tidak boleh diakhiri dengan sebuah tanda hubung.
6. Di kotak Deskripsi, ketik deskripsi untuk grup parameter.
7. Untuk membuat grup parameter, pilih Buat.

   Untuk mengakhiri proses tanpa membuat grup parameter, pilih Batalkan.
8. Ketika grup parameter dibuat, grup parameter ini akan memiliki nilai default keluarga. Untuk mengubah nilai default, Anda harus mengubah grup parameter. Untuk informasi selengkapnya, lihat Memodifikasi grup ElastiCache parameter.

Membuat grup ElastiCache parameter (AWS CLI)

Untuk membuat grup parameter menggunakan AWS CLI, gunakan perintah `create-cache-parameter-group` dengan parameter ini.

- `--cache-parameter-group-name` – Nama grup parameter.

  Batasan penamaan grup parameter adalah sebagai berikut:
  - Harus diawali dengan huruf ASCII.
  - Hanya dapat berisi huruf ASCII, angka, dan tanda hubung.
  - Harus memiliki panjang 1-255 karakter.
  - Tidak boleh berisi dua tanda hubung berurutan.
  - Tidak boleh diakhiri dengan tanda hubung.
- `--cache-parameter-group-family` – Keluarga mesin dan versi untuk grup parameter.
- `--description` – Deskripsi yang diberikan pengguna untuk grup parameter.

Example

Contoh berikut membuat grup parameter bernama myMem14 menggunakan keluarga memcached1.4 sebagai templat.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-cache-parameter-group \
    --cache-parameter-group-name myMem14  \
    --cache-parameter-group-family memcached1.4 \
    --description "My first parameter group"
```

Untuk Windows:

```
aws elasticache create-cache-parameter-group ^
    --cache-parameter-group-name myMem14  ^
    --cache-parameter-group-family memcached1.4 ^
    --description "My first parameter group"
```

Output dari perintah ini akan terlihat seperti ini.

```
{
    "CacheParameterGroup": {
        "CacheParameterGroupName": "myMem14",
        "CacheParameterGroupFamily": "memcached1.4",
        "Description": "My first  parameter group"
    }
}
```

Example

Contoh berikut membuat grup parameter bernama myRed28 menggunakan keluarga redis2.8 sebagai templat.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-cache-parameter-group \
    --cache-parameter-group-name myRed28  \
    --cache-parameter-group-family redis2.8 \
    --description "My first parameter group"
```

Untuk Windows:

```
aws elasticache create-cache-parameter-group ^
    --cache-parameter-group-name myRed28  ^
    --cache-parameter-group-family redis2.8 ^
    --description "My first parameter group"
```

Output dari perintah ini akan terlihat seperti ini.

```
{
    "CacheParameterGroup": {
        "CacheParameterGroupName": "myRed28",
        "CacheParameterGroupFamily": "redis2.8",
        "Description": "My first parameter group"
    }
}
```

Ketika grup parameter dibuat, grup parameter ini akan memiliki nilai default keluarga. Untuk mengubah nilai default, Anda harus mengubah grup parameter. Untuk informasi selengkapnya, lihat Memodifikasi grup ElastiCache parameter.

Untuk informasi selengkapnya, lihat `create-cache-parameter-group`.

Membuat grup ElastiCache parameter (ElastiCache API)

Untuk membuat grup parameter menggunakan ElastiCache API, gunakan `CreateCacheParameterGroup` tindakan dengan parameter ini.

- `ParameterGroupName` – Nama grup parameter.

  Batasan penamaan grup parameter adalah sebagai berikut:

  - Harus diawali dengan huruf ASCII.
  - Hanya dapat berisi huruf ASCII, angka, dan tanda hubung.
  - Harus memiliki panjang 1-255 karakter.
  - Tidak boleh berisi dua tanda hubung berurutan.
  - Tidak boleh diakhiri dengan tanda hubung.
- `CacheParameterGroupFamily` – Keluarga mesin dan versi untuk grup parameter. Misalnya, `memcached1.4`.
- `CacheParameterGroupFamily` – Keluarga mesin dan versi untuk grup parameter. Misalnya, `redis2.8`.

- `Description` – Deskripsi yang diberikan pengguna untuk grup parameter.

Example

Contoh berikut membuat grup parameter bernama myMem14 menggunakan keluarga memcached1.4 sebagai templat.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=CreateCacheParameterGroup
   &CacheParameterGroupFamily=memcached1.4
   &CacheParameterGroupName=myMem14
   &Description=My%20first%20parameter%20group
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &Version=2015-02-02
   &X-Amz-Credential=<credential>
```

Respons dari tindakan ini akan terlihat seperti ini.

```
<CreateCacheParameterGroupResponse xmlns="http://elasticache.amazonaws.com/
doc/2013-06-15/">
  <CreateCacheParameterGroupResult>
    <CacheParameterGroup>
      <CacheParameterGroupName>myMem14</CacheParameterGroupName>
      <CacheParameterGroupFamily>memcached1.4</CacheParameterGroupFamily>
      <Description>My first  parameter group</Description>
    </CacheParameterGroup>
  </CreateCacheParameterGroupResult>
  <ResponseMetadata>
    <RequestId>d8465952-af48-11e0-8d36-859edca6f4b8</RequestId>
  </ResponseMetadata>
</CreateCacheParameterGroupResponse>
```

Example

Contoh berikut membuat grup parameter bernama myRed28 menggunakan keluarga redis2.8 sebagai templat.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=CreateCacheParameterGroup
```

```
   &CacheParameterGroupFamily=redis2.8
   &CacheParameterGroupName=myRed28
   &Description=My%20first%20parameter%20group
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &Version=2015-02-02
   &X-Amz-Credential=<credential>
```

Respons dari tindakan ini akan terlihat seperti ini.

```
<CreateCacheParameterGroupResponse xmlns="http://elasticache.amazonaws.com/
doc/2013-06-15/">
  <CreateCacheParameterGroupResult>
    <CacheParameterGroup>
      <CacheParameterGroupName>myRed28</CacheParameterGroupName>
      <CacheParameterGroupFamily>redis2.8</CacheParameterGroupFamily>
      <Description>My first parameter group</Description>
    </CacheParameterGroup>
  </CreateCacheParameterGroupResult>
  <ResponseMetadata>
    <RequestId>d8465952-af48-11e0-8d36-859edca6f4b8</RequestId>
  </ResponseMetadata>
</CreateCacheParameterGroupResponse>
```

Ketika grup parameter dibuat, grup parameter ini akan memiliki nilai default keluarga. Untuk mengubah nilai default, Anda harus mengubah grup parameter. Untuk informasi selengkapnya, lihat Memodifikasi grup ElastiCache parameter.

Lihat informasi yang lebih lengkap di `CreateCacheParameterGroup`.

Daftar grup ElastiCache parameter berdasarkan nama

Anda dapat membuat daftar grup parameter menggunakan ElastiCache konsol, the AWS CLI, atau ElastiCache API.

Menampilkan daftar grup parameter berdasarkan nama (Konsol)

Prosedur berikut menunjukkan cara melihat daftar grup parameter menggunakan konsol ElastiCache .

Untuk membuat daftar grup parameter menggunakan ElastiCache konsol

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Untuk melihat daftar semua grup parameter yang tersedia, pilih Grup Parameter di panel navigasi sebelah kiri.

Daftar grup ElastiCache parameter berdasarkan nama (AWS CLI)

Untuk menghasilkan daftar kelompok parameter menggunakan AWS CLI, gunakan perintah`describe-cache-parameter-groups`. Jika Anda memberikan nama grup parameter, hanya grup parameter tersebut yang akan dicantumkan. Jika Anda memberikan nama grup parameter, hanya grup parameter hingga `--max-records` yang akan dicantumkan. Dalam kedua kasus, nama, keluarga, dan deskripsi grup parameter akan dicantumkan.

Example

Contoh kode berikut menampilkan daftar grup parameter myMem14.

Untuk Linux, macOS, atau Unix:

```
aws elasticache describe-cache-parameter-groups \
    --cache-parameter-group-name myMem14
```

Untuk Windows:

```
aws elasticache describe-cache-parameter-groups ^
    --cache-parameter-group-name myMem14
```

Output dari perintah ini akan terlihat seperti ini, mencantumkan nama, keluarga, dan deskripsi untuk grup parameter.

```
{
    "CacheParameterGroups": [
     {
         "CacheParameterGroupName": "myMem14",
         "CacheParameterGroupFamily": "memcached1.4",
         "Description": "My first parameter group"
     }
    ]
}
```

Example

Contoh kode berikut menampilkan daftar grup parameter myRed28.

Untuk Linux, macOS, atau Unix:

```
aws elasticache describe-cache-parameter-groups \
    --cache-parameter-group-name myRed28
```

Untuk Windows:

```
aws elasticache describe-cache-parameter-groups ^
    --cache-parameter-group-name myRed28
```

Output dari perintah ini akan terlihat seperti ini, mencantumkan nama, keluarga, dan deskripsi untuk grup parameter.

```
{
    "CacheParameterGroups": [
     {
         "CacheParameterGroupName": "myRed28",
         "CacheParameterGroupFamily": "redis2.8",
         "Description": "My first parameter group"
     }
    ]
}
```

Example

Kode contoh berikut mencantumkan grup parameter MyRed56 untuk grup parameter yang berjalan pada mesin Redis OSS versi 5.0.6 dan seterusnya. Jika grup parameter adalah bagian dari Replikasi

lintas AWS Wilayah menggunakan datastores global, nilai properti `IsGlobal` yang dikembalikan dalam output akan berupa `Yes`.

Untuk Linux, macOS, atau Unix:

```
aws elasticache describe-cache-parameter-groups \
    --cache-parameter-group-name myRed56
```

Untuk Windows:

```
aws elasticache describe-cache-parameter-groups ^
    --cache-parameter-group-name myRed56
```

Output dari perintah ini akan terlihat seperti ini, dengan menampilkan daftar nama, keluarga, isGlobal, dan deskripsi untuk grup parameter.

```
{
    "CacheParameterGroups": [
     {
         "CacheParameterGroupName": "myRed56",
         "CacheParameterGroupFamily": "redis5.0",
         "Description": "My first parameter group",
         "IsGlobal": "yes"
     }
    ]
}
```

Example

Contoh kode berikut menampilkan daftar hingga 10 grup parameter.

```
aws elasticache describe-cache-parameter-groups --max-records 10
```

Output JSON dari perintah ini akan terlihat seperti ini, mencantumkan nama, keluarga, deskripsi dan, pada redis5.6 menunjukkan apakah grup parameter bagian dari penyimpanan data global (isGlobal), untuk setiap grup parameter.

```
{
    "CacheParameterGroups": [
```

```
        {
            "CacheParameterGroupName": "custom-redis32",
            "CacheParameterGroupFamily": "redis3.2",
            "Description": "custom parameter group with reserved-memory > 0"
        },
        {
            "CacheParameterGroupName": "default.memcached1.4",
            "CacheParameterGroupFamily": "memcached1.4",
            "Description": "Default parameter group for memcached1.4"
        },
        {
            "CacheParameterGroupName": "default.redis2.6",
            "CacheParameterGroupFamily": "redis2.6",
            "Description": "Default parameter group for redis2.6"
        },
        {
            "CacheParameterGroupName": "default.redis2.8",
            "CacheParameterGroupFamily": "redis2.8",
            "Description": "Default parameter group for redis2.8"
        },
        {
            "CacheParameterGroupName": "default.redis3.2",
            "CacheParameterGroupFamily": "redis3.2",
            "Description": "Default parameter group for redis3.2"
        },
        {
            "CacheParameterGroupName": "default.redis3.2.cluster.on",
            "CacheParameterGroupFamily": "redis3.2",
            "Description": "Customized default parameter group for redis3.2 with
 cluster mode on"
        },
        {
            "CacheParameterGroupName": "default.redis5.6.cluster.on",
            "CacheParameterGroupFamily": "redis5.0",
            "Description": "Customized default parameter group for redis5.6 with
 cluster mode on",
            "isGlobal": "yes"
        },
    ]
}
```

Untuk informasi selengkapnya, lihat describe-cache-parameter-groups.

Daftar grup ElastiCache parameter berdasarkan nama (ElastiCache API)

Untuk membuat daftar grup parameter menggunakan ElastiCache API, gunakan `DescribeCacheParameterGroups` tindakan. Jika Anda memberikan nama grup parameter, hanya grup parameter tersebut yang akan dicantumkan. Jika Anda memberikan nama grup parameter, hanya grup parameter hingga `MaxRecords` yang akan dicantumkan. Dalam kedua kasus, nama, keluarga, dan deskripsi grup parameter akan dicantumkan.

Example

Contoh kode berikut menampilkan daftar grup parameter myMem14.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DescribeCacheParameterGroups
   &CacheParameterGroupName=myMem14
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &Version=2015-02-02
   &X-Amz-Credential=<credential>
```

Respons dari tindakan ini akan terlihat seperti ini, menampilkan daftar nama, keluarga, dan deskripsi untuk setiap grup parameter.

```
<DescribeCacheParameterGroupsResponse xmlns="http://elasticache.amazonaws.com/
doc/2013-06-15/">
  <DescribeCacheParameterGroupsResult>
    <CacheParameterGroups>
      <CacheParameterGroup>
        <CacheParameterGroupName>myMem14</CacheParameterGroupName>
        <CacheParameterGroupFamily>memcached1.4</CacheParameterGroupFamily>
        <Description>My custom Memcached 1.4 parameter group</Description>
      </CacheParameterGroup>
    </CacheParameterGroups>
  </DescribeCacheParameterGroupsResult>
  <ResponseMetadata>
    <RequestId>3540cc3d-af48-11e0-97f9-279771c4477e</RequestId>
  </ResponseMetadata>
</DescribeCacheParameterGroupsResponse>
```

Example

Contoh kode berikut menampilkan daftar hingga 10 grup parameter.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DescribeCacheParameterGroups
   &MaxRecords=10
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &Version=2015-02-02
   &X-Amz-Credential=<credential>
```

Respons dari tindakan ini akan terlihat seperti ini, menampilkan daftar nama, keluarga, deskripsi dan, dalam kasus redis5.6 jika grup parameter milik penyimpanan data global (isGlobal), untuk setiap grup parameter.

```
<DescribeCacheParameterGroupsResponse xmlns="http://elasticache.amazonaws.com/
doc/2013-06-15/">
  <DescribeCacheParameterGroupsResult>
    <CacheParameterGroups>
      <CacheParameterGroup>
        <CacheParameterGroupName>myRedis28</CacheParameterGroupName>
        <CacheParameterGroupFamily>redis2.8</CacheParameterGroupFamily>
        <Description>My custom Redis 2.8 parameter group</Description>
      </CacheParameterGroup>
      <CacheParameterGroup>
        <CacheParameterGroupName>myMem14</CacheParameterGroupName>
        <CacheParameterGroupFamily>memcached1.4</CacheParameterGroupFamily>
        <Description>My custom Memcached 1.4 parameter group</Description>
      </CacheParameterGroup>
       <CacheParameterGroup>
        <CacheParameterGroupName>myRedis56</CacheParameterGroupName>
        <CacheParameterGroupFamily>redis5.0</CacheParameterGroupFamily>
        <Description>My custom redis 5.6 parameter group</Description>
        <isGlobal>yes</isGlobal>
      </CacheParameterGroup>
    </CacheParameterGroups>
  </DescribeCacheParameterGroupsResult>
  <ResponseMetadata>
    <RequestId>3540cc3d-af48-11e0-97f9-279771c4477e</RequestId>
  </ResponseMetadata>
```

```
</DescribeCacheParameterGroupsResponse>
```

Example

Contoh kode berikut menampilkan daftar grup parameter myRed28.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DescribeCacheParameterGroups
   &CacheParameterGroupName=myRed28
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &Version=2015-02-02
   &X-Amz-Credential=<credential>
```

Respons dari tindakan ini akan terlihat seperti ini, menampilkan daftar nama, keluarga, dan deskripsi.

```
<DescribeCacheParameterGroupsResponse xmlns="http://elasticache.amazonaws.com/
doc/2013-06-15/">
  <DescribeCacheParameterGroupsResult>
    <CacheParameterGroups>
      <CacheParameterGroup>
        <CacheParameterGroupName>myRed28</CacheParameterGroupName>
        <CacheParameterGroupFamily>redis2.8</CacheParameterGroupFamily>
        <Description>My custom Redis 2.8 parameter group</Description>
      </CacheParameterGroup>
    </CacheParameterGroups>
  </DescribeCacheParameterGroupsResult>
  <ResponseMetadata>
    <RequestId>3540cc3d-af48-11e0-97f9-279771c4477e</RequestId>
  </ResponseMetadata>
</DescribeCacheParameterGroupsResponse>
```

Example

Contoh kode berikut menampilkan daftar grup parameter myRed56.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DescribeCacheParameterGroups
   &CacheParameterGroupName=myRed56
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
```

```
   &Timestamp=20150202T192317Z
   &Version=2015-02-02
   &X-Amz-Credential=<credential>
```

Respons dari tindakan ini akan terlihat seperti ini, dengan menampilkan daftar nama, keluarga, deskripsi, dan apakah grup parameter adalah bagian dari penyimpanan data global (isGlobal).

```
<DescribeCacheParameterGroupsResponse xmlns="http://elasticache.amazonaws.com/
doc/2013-06-15/">
  <DescribeCacheParameterGroupsResult>
    <CacheParameterGroups>
      <CacheParameterGroup>
        <CacheParameterGroupName>myRed56</CacheParameterGroupName>
        <CacheParameterGroupFamily>redis5.0</CacheParameterGroupFamily>
        <Description>My custom Redis 5.6 parameter group</Description>
        <isGlobal>yes</isGlobal>
      </CacheParameterGroup>
    </CacheParameterGroups>
  </DescribeCacheParameterGroupsResult>
  <ResponseMetadata>
    <RequestId>3540cc3d-af48-11e0-97f9-279771c4477e</RequestId>
  </ResponseMetadata>
</DescribeCacheParameterGroupsResponse>
```

Lihat informasi yang lebih lengkap di DescribeCacheParameterGroups.

Daftar nilai kelompok ElastiCache parameter

Anda dapat mencantumkan parameter dan nilainya untuk grup parameter menggunakan ElastiCache konsol, the AWS CLI, atau ElastiCache API.

Daftar nilai grup ElastiCache parameter (Konsol)

Prosedur berikut menunjukkan cara membuat daftar parameter dan nilainya untuk grup parameter menggunakan ElastiCache konsol.

Untuk membuat daftar parameter dan nilainya dari grup parameter menggunakan konsol ElastiCache

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Untuk melihat daftar semua grup parameter yang tersedia, di panel navigasi sebelah kiri pilih Grup Parameter.
3. Pilih grup parameter yang ingin Anda tampilkan daftar parameter dan nilainya dengan memilih kotak di sebelah kiri nama grup parameter.

   Parameter dan nilainya akan tercantum di bagian bawah layar. Karena jumlah parameter, Anda mungkin harus menggulir ke atas/bawah untuk menemukan parameter yang diinginkan.

Menampilkan daftar nilai grup parameter (AWS CLI)

Untuk membuat daftar parameter grup parameter dan nilainya menggunakan AWS CLI, gunakan perintah`describe-cache-parameters`.

Example

Kode contoh berikut mencantumkan semua parameter Memcached dan nilainya untuk grup parameter MyMEM14.

Untuk Linux, macOS, atau Unix:

```
aws elasticache describe-cache-parameters \
    --cache-parameter-group-name myMem14
```

Untuk Windows:

```
aws elasticache describe-cache-parameters ^
```

```
    --cache-parameter-group-name myMem14
```

Example

Contoh kode berikut menampilkan daftar semua parameter dan nilainya untuk grup parameter myRedis28.

Untuk Linux, macOS, atau Unix:

```
aws elasticache describe-cache-parameters \
    --cache-parameter-group-name myRedis28
```

Untuk Windows:

```
aws elasticache describe-cache-parameters ^
    --cache-parameter-group-name myRed28
```

Untuk informasi selengkapnya, lihat `describe-cache-parameters`.

Daftar nilai grup parameter (ElastiCache API)

Untuk membuat daftar parameter grup parameter dan nilainya menggunakan ElastiCache API, gunakan `DescribeCacheParameters` tindakan.

Example

Kode contoh berikut mencantumkan semua parameter Memcached untuk grup parameter MyMEM14.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DescribeCacheParameters
   &CacheParameterGroupName=myMem14
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &Version=2015-02-02
   &X-Amz-Credential=<credential>
```

Respons dari tindakan ini akan terlihat seperti ini. Respons ini telah terpotong.

```
<DescribeCacheParametersResponse xmlns="http://elasticache.amazonaws.com/
doc/2013-06-15/">
  <DescribeCacheParametersResult>
```

```
    <CacheClusterClassSpecificParameters>
      <CacheNodeTypeSpecificParameter>
        <DataType>integer</DataType>
        <Source>system</Source>
        <IsModifiable>false</IsModifiable>
        <Description>The maximum configurable amount of memory to use to store items,
 in megabytes.</Description>
        <CacheNodeTypeSpecificValues>
          <CacheNodeTypeSpecificValue>
            <Value>1000</Value>
            <CacheClusterClass>cache.c1.medium</CacheClusterClass>
          </CacheNodeTypeSpecificValue>
          <CacheNodeTypeSpecificValue>
            <Value>6000</Value>
            <CacheClusterClass>cache.c1.xlarge</CacheClusterClass>
          </CacheNodeTypeSpecificValue>
          <CacheNodeTypeSpecificValue>
            <Value>7100</Value>
            <CacheClusterClass>cache.m1.large</CacheClusterClass>
          </CacheNodeTypeSpecificValue>
          <CacheNodeTypeSpecificValue>
            <Value>1300</Value>
            <CacheClusterClass>cache.m1.small</CacheClusterClass>
          </CacheNodeTypeSpecificValue>

...output omitted...

    </CacheClusterClassSpecificParameters>
  </DescribeCacheParametersResult>
  <ResponseMetadata>
    <RequestId>6d355589-af49-11e0-97f9-279771c4477e</RequestId>
  </ResponseMetadata>
</DescribeCacheParametersResponse>
```

Example

Contoh kode berikut menampilkan daftar semua parameter untuk grup parameter myRed28.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DescribeCacheParameters
   &CacheParameterGroupName=myRed28
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
```

```
   &Version=2015-02-02
   &X-Amz-Credential=<credential>
```

Respons dari tindakan ini akan terlihat seperti ini. Respons ini telah terpotong.

```
<DescribeCacheParametersResponse xmlns="http://elasticache.amazonaws.com/
doc/2013-06-15/">
  <DescribeCacheParametersResult>
    <CacheClusterClassSpecificParameters>
      <CacheNodeTypeSpecificParameter>
        <DataType>integer</DataType>
        <Source>system</Source>
        <IsModifiable>false</IsModifiable>
        <Description>The maximum configurable amount of memory to use to store items,
 in megabytes.</Description>
        <CacheNodeTypeSpecificValues>
          <CacheNodeTypeSpecificValue>
            <Value>1000</Value>
            <CacheClusterClass>cache.c1.medium</CacheClusterClass>
          </CacheNodeTypeSpecificValue>
          <CacheNodeTypeSpecificValue>
            <Value>6000</Value>
            <CacheClusterClass>cache.c1.xlarge</CacheClusterClass>
          </CacheNodeTypeSpecificValue>
          <CacheNodeTypeSpecificValue>
            <Value>7100</Value>
            <CacheClusterClass>cache.m1.large</CacheClusterClass>
          </CacheNodeTypeSpecificValue>
          <CacheNodeTypeSpecificValue>
            <Value>1300</Value>
            <CacheClusterClass>cache.m1.small</CacheClusterClass>
          </CacheNodeTypeSpecificValue>

...output omitted...

    </CacheClusterClassSpecificParameters>
  </DescribeCacheParametersResult>
  <ResponseMetadata>
    <RequestId>6d355589-af49-11e0-97f9-279771c4477e</RequestId>
  </ResponseMetadata>
</DescribeCacheParametersResponse>
```

Lihat informasi yang lebih lengkap di DescribeCacheParameters.

Memodifikasi grup ElastiCache parameter

> **Important**
>
> Anda tidak dapat mengubah grup parameter default.

Anda dapat mengubah beberapa nilai parameter di grup parameter. Nilai parameter ini diterapkan ke klaster yang terkait dengan grup parameter. Untuk informasi selengkapnya tentang kapan perubahan nilai parameter diterapkan ke grup parameter, lihat Parameter Valkey dan Redis OSS danParameter spesifik Memcached.

Mengubah grup parameter (Konsol)

Prosedur berikut menunjukkan cara mengubah nilai `cluster-enabled` parameter menggunakan ElastiCache konsol. Anda akan menggunakan prosedur yang sama untuk mengubah nilai parameter apa pun.

Untuk mengubah nilai parameter menggunakan ElastiCache konsol

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Untuk melihat daftar semua grup parameter yang tersedia, pilih Grup Parameter di panel navigasi sebelah kiri.
3. Pilih grup parameter yang ingin Anda ubah dengan memilih kotak di sebelah kiri nama grup parameter.

   Parameter dari grup parameter akan tercantum di bagian bawah layar. Anda mungkin perlu menelusuri daftar untuk melihat semua parameter.
4. Untuk mengubah satu atau beberapa parameter, pilih Edit Parameter.
5. Di layar Edit Grup Parameter:, gulir menggunakan panah kiri dan kanan sampai Anda menemukan parameter `binding_protocol`, lalu ketik `ascii` di kolom Nilai.
6. Pilih Simpan Perubahan.
7. Untuk Memcached, untuk menemukan nama parameter yang Anda ubah, lihat. Parameter spesifik Memcached Jika perubahan parameter diterapkan Setelah pengaktifan ulang, boot ulang setiap klaster yang menggunakan grup parameter ini. Untuk informasi selengkapnya, lihat Mem-boot ulang klaster.

8. Dengan Valkey dan Redis OSS, untuk menemukan nama parameter yang Anda ubah, lihat. Parameter Valkey dan Redis OSS Jika Anda memiliki cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dan membuat perubahan pada parameter berikut, Anda harus me-reboot node di cluster:

   - activerehashing
   - databases

   Untuk informasi selengkapnya, lihat Mem-boot ulang simpul.

   > Perubahan parameter Valkey atau Redis OSS (Mode Cluster Diaktifkan)
   >
   > Jika Anda membuat perubahan pada parameter berikut pada cluster Valkey atau Redis OSS (mode cluster enabled), ikuti langkah-langkah berikutnya.
   >
   > - activerehashing
   > - databases
   >
   > 1. Dengan Redis OSS, Anda dapat membuat cadangan manual cluster Anda. Lihat Membuat cadangan manual.
   > 2. Hapus klaster . Lihat Menghapus klaster.
   > 3. Pulihkan klaster menggunakan grup parameter dan cadangan yang sudah diubah untuk melakukan seeding klaster baru. Lihat Melakukan pemulihan dari cadangan ke dalam cache baru.
   >
   > Perubahan pada parameter lain tidak memerlukan tindakan ini.

Mengubah grup parameter (AWS CLI)

Untuk mengubah nilai parameter menggunakan AWS CLI, gunakan perintah`modify-cache-parameter-group`.

Example

Dengan Memcached, untuk menemukan nama dan nilai yang diizinkan dari parameter yang ingin Anda ubah, lihat Parameter spesifik Memcached

Contoh kode berikut menetapkan nilai dua parameter, chunk_size dan chunk_size_growth_fact pada grup parameter `myMem14`.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-cache-parameter-group \
    --cache-parameter-group-name myMem14 \
    --parameter-name-values \
        ParameterName=chunk_size,ParameterValue=96 \
        ParameterName=chunk_size_growth_fact,ParameterValue=1.5
```

Untuk Windows:

```
aws elasticache modify-cache-parameter-group ^
    --cache-parameter-group-name myMem14 ^
    --parameter-name-values ^
        ParameterName=chunk_size,ParameterValue=96 ^
        ParameterName=chunk_size_growth_fact,ParameterValue=1.5
```

Output dari perintah ini akan terlihat seperti ini.

```
{
    "CacheParameterGroupName": "myMem14"
}
```

Example

Dengan Valkey dan Redis OSS, untuk menemukan nama dan nilai yang diizinkan dari parameter yang ingin Anda ubah, lihat Parameter Valkey dan Redis OSS

Kode sampel berikut menetapkan nilai dua parameter, reserved-memory-percentdan cluster diaktifkan pada kelompok parameter. `myredis32-on-30` Kami mengatur reserved-memory-percentke `30` (30 persen) dan mengaktifkan cluster `yes` sehingga grup parameter dapat digunakan dengan cluster Valkey atau Redis OSS (mode cluster diaktifkan) (grup replikasi).

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-cache-parameter-group \
    --cache-parameter-group-name myredis32-on-30 \
    --parameter-name-values \
```

```
        ParameterName=reserved-memory-percent,ParameterValue=30 \
        ParameterName=cluster-enabled,ParameterValue=yes
```

Untuk Windows:

```
aws elasticache modify-cache-parameter-group ^
    --cache-parameter-group-name myredis32-on-30 ^
    --parameter-name-values ^
        ParameterName=reserved-memory-percent,ParameterValue=30 ^
        ParameterName=cluster-enabled,ParameterValue=yes
```

Output dari perintah ini akan terlihat seperti ini.

```
{
    "CacheParameterGroupName": "my-redis32-on-30"
}
```

Untuk informasi selengkapnya, lihat `modify-cache-parameter-group`.

Untuk menemukan nama parameter yang Anda ubah, lihat Parameter Valkey dan Redis OSS.

Jika Anda memiliki cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dan membuat perubahan pada parameter berikut, Anda harus me-reboot node di cluster:

- activerehashing
- databases

Untuk informasi selengkapnya, lihat Mem-boot ulang simpul.

> **Perubahan parameter Valkey atau Redis OSS (Mode Cluster Diaktifkan)**
>
> Jika Anda membuat perubahan pada parameter berikut pada cluster Valkey atau Redis OSS (mode cluster enabled), ikuti langkah-langkah berikutnya.
>
> - activerehashing
> - databases
>
> 1. Buat cadangan manual klaster Anda. Lihat Membuat cadangan manual.
> 2. Hapus klaster . Lihat Menghapus klaster.

3. Pulihkan klaster menggunakan grup parameter dan cadangan yang sudah diubah untuk melakukan seeding klaster baru. Lihat Melakukan pemulihan dari cadangan ke dalam cache baru.

Perubahan pada parameter lain tidak memerlukan tindakan ini.

Memodifikasi grup parameter (ElastiCache API)

Untuk mengubah nilai parameter grup parameter menggunakan ElastiCache API, gunakan `ModifyCacheParameterGroup` tindakan.

Example

Dengan Memcached, untuk menemukan nama dan nilai yang diizinkan dari parameter yang ingin Anda ubah, lihat Parameter spesifik Memcached

Contoh kode berikut menetapkan nilai dua parameter, chunk_size dan chunk_size_growth_fact pada grup parameter `myMem14`.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=ModifyCacheParameterGroup
   &CacheParameterGroupName=myMem14
   &ParameterNameValues.member.1.ParameterName=chunk_size
   &ParameterNameValues.member.1.ParameterValue=96
   &ParameterNameValues.member.2.ParameterName=chunk_size_growth_fact
   &ParameterNameValues.member.2.ParameterValue=1.5
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &Version=2015-02-02
   &X-Amz-Credential=<credential>
```

Example

Dengan Valkey dan Redis OSS, untuk menemukan nama dan nilai yang diizinkan dari parameter yang ingin Anda ubah, lihat Parameter Valkey dan Redis OSS

Kode sampel berikut menetapkan nilai dua parameter, reserved-memory-percentdan cluster diaktifkan pada kelompok parameter. `myredis32-on-30` Kami mengatur reserved-memory-

percentke 30 (30 persen) dan mengaktifkan cluster yes sehingga grup parameter dapat digunakan dengan cluster Valkey atau Redis OSS (mode cluster diaktifkan) (grup replikasi).

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=ModifyCacheParameterGroup
   &CacheParameterGroupName=myredis32-on-30
   &ParameterNameValues.member.1.ParameterName=reserved-memory-percent
   &ParameterNameValues.member.1.ParameterValue=30
   &ParameterNameValues.member.2.ParameterName=cluster-enabled
   &ParameterNameValues.member.2.ParameterValue=yes
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &Version=2015-02-02
   &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya, lihat ModifyCacheParameterGroup.

Jika Anda memiliki cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dan membuat perubahan pada parameter berikut, Anda harus me-reboot node di cluster:

- activerehashing
- databases

Untuk informasi selengkapnya, lihat Mem-boot ulang node.

Perubahan parameter Valkey atau Redis OSS (Mode Cluster Diaktifkan)

Jika Anda membuat perubahan pada parameter berikut pada cluster Valkey atau Redis OSS (mode cluster enabled), ikuti langkah-langkah berikutnya.

- activerehashing
- databases

1. Buat cadangan manual klaster Anda. Lihat Membuat cadangan manual.
2. Hapus klaster . Lihat Menghapus cluster di ElastiCache.
3. Pulihkan klaster menggunakan grup parameter dan cadangan yang sudah diubah untuk melakukan seeding klaster baru. Lihat Melakukan pemulihan dari cadangan ke dalam cache baru.

Perubahan pada parameter lain tidak memerlukan tindakan ini.

Menghapus grup ElastiCache parameter

Anda dapat menghapus grup parameter kustom menggunakan ElastiCache konsol, file AWS CLI, atau ElastiCache API.

Anda tidak dapat menghapus grup parameter jika grup parameter ini dikaitkan dengan klaster. Anda juga tidak dapat menghapus grup parameter default.

Menghapus grup parameter (Konsol)

Prosedur berikut menunjukkan cara menghapus grup parameter menggunakan konsol ElastiCache.

Untuk menghapus grup parameter menggunakan ElastiCache konsol

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Untuk melihat daftar semua grup parameter yang tersedia, di panel navigasi sebelah kiri pilih Grup Parameter.
3. Pilih grup parameter yang ingin Anda hapus dengan memilih kotak di sebelah kiri nama grup parameter.

   Tombol Hapus akan menjadi aktif.
4. Pilih Hapus.

   Layar konfirmasi Hapus Grup Parameter akan muncul.
5. Untuk menghapus grup parameter, pada layar konfirmasi Hapus Grup Parameter, pilih Hapus.

   Untuk mempertahankan grup parameter, pilih Batalkan.

Menghapus grup parameter (AWS CLI)

Untuk menghapus grup parameter menggunakan AWS CLI, gunakan perintah`delete-cache-parameter-group`. Untuk grup parameter yang akan dihapus, grup parameter yang ditentukan berdasarkan `--cache-parameter-group-name` tidak boleh memiliki klaster yang terkait dengannya, dan juga tidak dapat berupa grup parameter default.

Contoh kode berikut menghapus grup parameter myMem14.

Example

Untuk Linux, macOS, atau Unix:

```
aws elasticache delete-cache-parameter-group \
    --cache-parameter-group-name myRed28
```

Untuk Windows:

```
aws elasticache delete-cache-parameter-group ^
    --cache-parameter-group-name myRed28
```

Untuk informasi selengkapnya, lihat delete-cache-parameter-group.

Menghapus grup parameter (ElastiCache API)

Untuk menghapus grup parameter menggunakan ElastiCache API, gunakan `DeleteCacheParameterGroup` tindakan. Untuk grup parameter yang akan dihapus, grup parameter yang ditentukan berdasarkan `CacheParameterGroupName` tidak boleh memiliki klaster yang terkait dengannya, dan juga tidak dapat berupa grup parameter default.

Example

Dengan Memcached, kode contoh berikut menghapus grup parameter MyMEM14.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DeleteCacheParameterGroup
   &CacheParameterGroupName=myMem14
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &Version=2015-02-02
   &X-Amz-Credential=<credential>
```

Example

Contoh kode berikut menghapus grup parameter myRed28.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DeleteCacheParameterGroup
   &CacheParameterGroupName=myRed28
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Timestamp=20150202T192317Z
   &Version=2015-02-02
```

```
    &X-Amz-Credential=<credential>
```

Lihat informasi yang lebih lengkap di DeleteCacheParameterGroup.

Parameter spesifik mesin

Valkey dan Redis OSS

Sebagian besar parameter Valkey 8 kompatibel dengan parameter Redis OSS 7.1. Parameter Valkey 7.2 sama dengan parameter Redis OSS 7.

Jika Anda tidak menentukan grup parameter untuk cluster Valkey atau Redis OSS Anda, maka grup parameter default yang sesuai dengan versi mesin Anda akan digunakan. Anda tidak dapat mengubah nilai parameter dalam grup parameter default. Namun, Anda dapat membuat grup parameter kustom dan menetapkannya ke klaster Anda setiap saat asalkan nilai parameter yang dapat diubah secara bersyarat di kedua grup parameter sama. Untuk informasi selengkapnya, lihat Membuat grup ElastiCache parameter.

Topik

Parameter Valkey dan Redis OSS

Topik

Perubahan parameter Valkey 8.1

Keluarga kelompok parameter: valkey8

> **Note**
>
> - Perubahan parameter Valkey 8.1 tidak berlaku untuk Valkey 8.0
> - Valkey 8.0 dan kelompok parameter di atas tidak kompatibel dengan Redis OSS 7.2.4.
> - di Valkey 8.1, perintah berikut tidak tersedia untuk cache tanpa server:`commandlog`,,,, dan `commandlog get` `commandlog help` `commandlog len` `commandlog reset`.

Grup parameter baru di Valkey 8.1

Nama	Detail	Deskripsi
commandlog-large-request-max-len (ditambahkan dalam 8.1)	Default: 1048576 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Ukuran maksimum, dalam byte, untuk permintaan yang akan dicatat oleh fitur Log Perintah Valkey.
commandlog-large-request-max-len (ditambahkan dalam 8.1)	Default: 128 Nilai yang diizinkan: 0-1024 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Panjang maksimum Log Perintah Valkey untuk permintaan.
commandlog-reply-larger-than (ditambahkan di 8.1)	Default: 1048576 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Ukuran maksimum, dalam byte, untuk respons yang akan dicatat oleh fitur Log Perintah Valkey.
commandlog-large-reply-max-len (ditambahkan dalam 8.1)	Default: 128 Nilai yang diizinkan: 0-1024 Jenis: integer Dapat diubah: Ya	Panjang maksimum Log Perintah Valkey untuk tanggapan.

Nama	Detail	Deskripsi
	Perubahan Berlaku: Segera	

Perubahan parameter Valkey 8.0

Keluarga kelompok parameter: valkey8

> **Note**
>
> Redis OSS 7.2.4 tidak kompatibel dengan Valkey 8 dan di atas kelompok parameter.

Perubahan parameter spesifik di Valkey 8.0

Nama	Detail	Deskripsi
repl-backlog-size	Standar: 10485760 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Ukuran, dalam byte, buffer backlog simpul primer. Backlog digunakan untuk mencatat pembaruan data pada simpul primer. Ketika replika baca terhubung ke primer, ia mencoba untuk melakukan sinkronisasi paral (psync), di mana ia menerapkan data dari backlog untuk mengejar ketinggalan dengan simpul utama. Jika psync gagal, maka sinkronis asi penuh diperlukan. Nilai minimum untuk parameter ini adalah 16384. Catatan: Dimulai dengan Redis OSS 2.8.22, parameter

Nama	Detail	Deskripsi
		ini berlaku untuk cluster primer serta replika baca.
maxmemory-sampel	Default: 3 Nilai yang diizinkan: 1 hingga 64 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Untuk perhitungan least-recently-used (LRU) dan time-to-live (TTL), parameter ini mewakili ukuran sampel kunci untuk diperiksa. Secara default, Redis OSS memilih 3 kunci dan menggunakan salah satu yang paling tidak digunakan baru-baru ini.

Grup parameter baru di Valkey 8.0

Nama	Detail	Deskripsi
extended-redis-compatibility	Nilai yang diizinkan: ya, tidak Default: yes Jenis: boolean Dapat diubah: Ya Perubahan berlaku: segera	Mode kompatibilitas Redis OSS yang diperluas membuat Valkey berpura-pura menjadi Redis OSS 7.2. Aktifkan ini hanya jika Anda memiliki masalah dengan alat atau klien. Dampak yang dihadapi pelanggan: • `LOADING`- Redis OSS memuat dataset dalam memori • `BUSY`- Redis OSS sibuk

Nama	Detail	Deskripsi
		• `MISCONF`- Redis OSS dikonfigurasi dengan salah satu cara ini: • `HELLO`Perintah mengembalikan “server” => “redis” dan “version” => “7.2.4" (versi kompatibilitas Redis OSS kami). • `INFO`Bidang untuk mode disebut “redis_mode”.

Dihapus kelompok parameter di Valkey 8.0

Nama	Detail	Deskripsi
lazyfree-lazy-eviction	Nilai yang diizinkan: ya, tidak Default: no Jenis: boolean Dapat diubah: Ya Perubahan akan diterapkan: segera	Melakukan penghapusan asinkron pada pengosongan.
lazyfree-lazy-expire	Nilai yang diizinkan: ya, tidak Default: no Jenis: boolean Dapat diubah: Ya	Melakukan penghapusan asinkron pada kunci kedaluwarsa.

Nama	Detail	Deskripsi
	Perubahan akan diterapkan: segera	
lazyfree-lazy-server-del	Nilai yang diizinkan: ya, tidak Default: no Jenis: boolean Dapat diubah: Ya Perubahan akan diterapkan: segera	Melakukan penghapusan asinkron untuk perintah yang memperbarui nilai.
lazyfree-lazy-user-del	Default: no Jenis: string Dapat diubah: Ya Perubahan berlaku: Segera di semua simpul dalam klaster	Ketika nilai diatur ke ya, perintah DEL bertindak sama seperti UNLINK.
replica-lazy-flush	Default: yes Jenis: boolean Dapat diubah: Tidak Nama sebelumnya: slave-lazy-flush	Melakukan flushDB asinkron selama sinkronisasi replika.

Perubahan parameter Valkey 7.2 dan Redis OSS 7

Keluarga kelompok parameter: valkey7

Grup parameter default Valkey 7.2 adalah sebagai berikut:

- `default.valkey7`— Gunakan grup parameter ini, atau yang diturunkan darinya, untuk cluster Valkey (mode cluster dinonaktifkan) dan grup replikasi.
- `default.valkey7.cluster.on`— Gunakan grup parameter ini, atau yang diturunkan darinya, untuk klaster Valkey (mode cluster diaktifkan) dan grup replikasi.

Keluarga grup parameter: redis7

Redis OSS 7 kelompok parameter default adalah sebagai berikut:

- `default.redis7`— Gunakan grup parameter ini, atau yang diturunkan darinya, untuk kelompok Redis OSS (mode cluster dinonaktifkan) dan grup replikasi.
- `default.redis7.cluster.on`— Gunakan grup parameter ini, atau yang diturunkan darinya, untuk kelompok Redis OSS (mode cluster diaktifkan) dan grup replikasi.

Perubahan parameter spesifik

Parameter yang ditambahkan dalam Redis OSS 7 adalah sebagai berikut. Valkey 7.2 juga mendukung parameter ini.

Nama	Detail	Deskripsi
`cluster-allow-pubsubshard-when-down`	Nilai yang diizinkan: `yes`, `no` Default: `yes` Jenis: string Dapat diubah: Ya Penerapan perubahan: Segera di semua simpul dalam klaster.	Ketika diatur ke nilai default ya, memungkinkan simpul melayani lalu lintas serpihan pubsub saat klaster dalam keadaan nonaktif, asalkan klaster ini mengetahui bahwa dirinya memiliki slot.
`cluster-preferred-endpoint-type`	Nilai yang diizinkan: `ip`, `tls-dynamic` Default: `tls-dynamic`	Nilai ini mengontrol titik akhir apa yang dikembalikan untuk permintaan MOVED/ASKING serta bidang titik akhir untuk `CLUSTER SLOTS` dan `CLUSTER SHARDS`. Ketika nilai diatur ke ip, simpul akan menyatakan alamat

Nama	Detail	Deskripsi
	Jenis: string Dapat diubah: Ya Penerapan perubahan: Segera di semua simpul dalam klaster.	ip-nya. Ketika nilai diatur ke tls-dynamic, node akan mengiklankan nama host saat encryption-in-transit diaktifkan dan alamat ip sebaliknya.
`latency-tracking`	Nilai yang diizinkan: `yes`, `no` Default: `no` Jenis: string Dapat diubah: Ya Penerapan perubahan: Segera di semua simpul dalam klaster.	Ketika diatur ke ya akan melacak latensi per perintah dan memungkinkan ekspor distribusi persentil melalui perintah statistik latensi `INFO`, dan distribusi latensi kumulatif (histogram) melalui perintah `LATENCY`.
`hash-max-listpack-entries`	Nilai yang diizinkan: `0+` Default: `512` Jenis: integer Dapat diubah: Ya Penerapan perubahan: Segera di semua simpul dalam klaster.	Jumlah maksimum entri hash agar set data dikompresi.

Nama	Detail	Deskripsi
`hash-max-listpack-value`	Nilai yang diizinkan: `0+` Default: `64` Jenis: integer Dapat diubah: Ya Penerapan perubahan: Segera di semua simpul dalam klaster.	Ambang entri hash terbesar agar set data dikompresi.
`zset-max-listpack-entries`	Nilai yang diizinkan: `0+` Default: `128` Jenis: integer Dapat diubah: Ya Penerapan perubahan: Segera di semua simpul dalam klaster.	Jumlah maksimum entri sorted set agar set data dikompresi.
`zset-max-listpack-value`	Nilai yang diizinkan: `0+` Default: `64` Jenis: integer Dapat diubah: Ya Penerapan perubahan: Segera di semua simpul dalam klaster.	Ambang batas entri sorted set terbesar agar set data dikompresi.

Parameter yang diubah dalam Redis OSS 7 adalah sebagai berikut.

Nama	Detail	Deskripsi
`activerehashing`	Dapat diubah: no. Di Redis OSS 7, parameter ini disembunyikan dan diaktifkan secara default. Untuk menonaktifkannya, Anda perlu membuat kasus dukungan.	Dapat diubah sebelumnya adalah ya.

Parameter dihapus dalam Redis OSS 7 adalah sebagai berikut.

Nama	Detail	Deskripsi
`hash-max-ziplist-entries`	Nilai yang diizinkan: `0+` Default: `512` Jenis: integer Dapat diubah: Ya Penerapan perubahan: Segera di semua simpul dalam klaster.	Gunakan `listpack` bukan `ziplist` untuk merepresentasikan pengenkodean hash kecil.
`hash-max-ziplist-value`	Nilai yang diizinkan: `0+` Default: `64` Jenis: integer Dapat diubah: Ya	Gunakan `listpack` bukan `ziplist` untuk merepresentasikan pengenkodean hash kecil.

Nama	Detail	Deskripsi
	Penerapan perubahan: Segera di semua simpul dalam klaster.	
`zset-max-ziplist-entries`	Nilai yang diizinkan: `0+` Default: `128` Jenis: integer Dapat diubah: Ya Penerapan perubahan: Segera di semua simpul dalam klaster.	Gunakan `listpack` bukan `ziplist` untuk merepresentasikan pengenkodean hash kecil.
`zset-max-ziplist-value`	Nilai yang diizinkan: `0+` Default: `64` Jenis: integer Dapat diubah: Ya Penerapan perubahan: Segera di semua simpul dalam klaster.	Gunakan `listpack` bukan `ziplist` untuk merepresentasikan pengenkodean hash kecil.

Nama	Detail	Deskripsi
`list-max-ziplist-size`	Nilai yang diizinkan: Default: `-2` Jenis: integer Dapat diubah: Ya Penerapan perubahan: Segera di semua simpul dalam klaster.	Jumlah entri yang diizinkan per simpul daftar internal.

Redis OSS 6.x perubahan parameter

Keluarga grup parameter: redis6.x

Redis OSS 6.x kelompok parameter default adalah sebagai berikut:

- `default.redis6.x`— Gunakan grup parameter ini, atau yang diturunkan darinya, untuk cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dan grup replikasi.
- `default.redis6.x.cluster.on`— Gunakan grup parameter ini, atau yang diturunkan darinya, untuk cluster Valkey atau Redis OSS (mode cluster diaktifkan) dan grup replikasi.

> **Note**
>
> Di mesin Redis OSS versi 6.2, ketika keluarga node r6gd diperkenalkan untuk digunakan dengan, hanya kebijakan noeviction Tingkatan data di ElastiCache, volatile-lru dan allkeys-lru max-memory yang didukung dengan tipe node r6gd.

Untuk informasi selengkapnya, lihat ElastiCache versi 6.2 untuk Redis OSS (ditingkatkan) dan ElastiCache versi 6.0 untuk Redis OSS (ditingkatkan).

Parameter yang ditambahkan dalam Redis OSS 6.x adalah sebagai berikut.

Detail	Deskripsi	
`acl-pubsub-default (added in 6.2)`	Nilai yang diizinkan : `resetchannels` , `allchannels` Default: `allchannels` Jenis: string Dapat diubah: Ya Perubahan berlaku: Pengguna Redis OSS yang ada yang terkait dengan cluster akan terus memiliki izin yang ada. Baik memperbarui pengguna atau reboot cluster untuk memperbarui pengguna Redis OSS yang ada.	Izin saluran pubsub default untuk pengguna ACL yang di-deploy ke klaster ini.
`cluster-allow-reads-when-down (added in 6.0)`	Default: no Jenis: string Dapat diubah: Ya Perubahan berlaku: Segera di semua simpul dalam klaster	Ketika disetel ke ya, grup replikasi Redis OSS (mode cluster enabled) terus memproses perintah baca bahkan ketika sebuah node tidak dapat mencapai kuorum primer. Jika diatur ke default tidak, grup replikasi menolak semua perintah. Kami merekomendasikan untuk mengatur nilai ini ke ya jika Anda menggunakan klaster dengan kurang dari tiga grup simpul atau aplikasi Anda dapat dengan aman menangani pembacaan yang usang dengan aman.
`tracking-table-max-keys`	Default: 1.000.000 Jenis: angka	Untuk membantu caching sisi klien, Redis OSS mendukung pelacakan klien mana yang telah mengakses kunci mana.

Detail	Deskripsi	
`(added in 6.0)`	Dapat diubah: Ya Perubahan berlaku: Segera di semua simpul dalam klaster	Ketika kunci yang dilacak diubah, pesan invalidasi dikirim ke semua klien untuk memberitahukan bahwa nilai cache-nya tidak valid lagi. Nilai ini memungkinkan Anda menentukan batas atas tabel ini. Setelah nilai parameter ini terlampaui, klien mendapatk an pesan invalidasi secara acak. Nilai ini harus diatur untuk membatasi penggunaan memori sambil masih melacak kunci. Kunci juga diinvalidasi dalam kondisi memori rendah.
`acllog-max-len (added in 6.0)`	Default: 128 Jenis: angka Dapat diubah: Ya Perubahan berlaku: Segera di semua simpul dalam klaster	Nilai ini sesuai dengan jumlah maksimum entri di log ACL.

Detail	Deskripsi	
`active-expire-effort (added in 6.0)`	Default: 1 Jenis: angka Dapat diubah: Ya Perubahan berlaku: Segera di semua simpul dalam klaster	Redis OSS menghapus kunci yang telah melampaui waktu mereka untuk hidup dengan dua mekanisme. Di satu sisi, kunci diakses dan ditemukan akan kedaluwarsa. Di sisi lain, pekerjaan berkala mengambil sampel kunci dan membuat kunci yang telah melebihi time-to-live (TTL)-nya menjadi kedaluwarsa. Parameter ini mendefinisikan jumlah upaya yang digunakan Redis OSS untuk mengakhiri item dalam pekerjaan periodik. Nilai default 1 akan mencoba mencegah adanya lebih dari 10 persen kunci kedaluwarsa yang masih berada dalam memori. Hal ini juga akan mencoba mencegah konsumsi lebih dari 25 persen dari total memori dan menambahk an latensi ke sistem. Anda dapat meningkat kan nilai ini hingga 10 untuk meningkatkan jumlah upaya yang digunakan untuk kunci kedaluwarsa. Kompromisnya adalah CPU lebih tinggi dan latensi berpotensi lebih tinggi. Kami merekomendasikan nilai 1 kecuali jika Anda melihat penggunaan memori tinggi dan dapat menoleransi peningkatan pemanfaatan CPU.
`lazyfree-lazy-user-del (added in 6.0)`	Default: no Jenis: string Dapat diubah: Ya Perubahan berlaku: Segera di semua simpul dalam klaster	Ketika nilai diatur ke ya, perintah `DEL` bertindak sama seperti `UNLINK`.

Parameter dihapus dalam Redis OSS 6.x adalah sebagai berikut.

Nama	Detail	Deskripsi
`lua-replicate-commands`	Nilai yang diizinkan: yes/no Default: yes Jenis: boolean Dapat diubah: Ya Perubahan berlaku: Segera.	Selalu mengaktifkan replikasi efek Lua atau tidak dalam skrip Lua

Redis OSS 5.0.3 perubahan parameter

Keluarga grup parameter: redis5.0

Redis OSS 5.0 grup parameter default

- `default.redis5.0`— Gunakan grup parameter ini, atau yang diturunkan darinya, untuk cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dan grup replikasi.
- `default.redis5.0.cluster.on`— Gunakan grup parameter ini, atau yang diturunkan darinya, untuk cluster Valkey atau Redis OSS (mode cluster diaktifkan) dan grup replikasi.

Parameter ditambahkan di Redis OSS 5.0.3

Nama	Detail	Deskripsi
`rename-commands`	Default: tidak ada Jenis: string Dapat diubah: Ya Perubahan berlaku: Segera di semua simpul dalam klaster	Daftar yang dipisahkan spasi dari perintah Redis OSS yang diganti namanya. Berikut adalah daftar terbatas perintah yang tersedia untuk diubah namanya: `APPEND AUTH BITCOUNT BITFIELD BITOP BITPOS BLPOP BRPOP BRPOPLPUSH BZPOPMIN BZPOPMAX CLIENT CLUSTER COMMAND DBSIZE DECR DECRBY DEL DISCARD DUMP ECHO EVAL EVALSHA EXEC EXISTS EXPIRE`

Nama	Detail	Deskripsi
		EXPIREAT FLUSHALL FLUSHDB GEOADD GEOHASH GEOPOS GEODIST GEORADIUS GEORADIUSBYMEMBER GET GETBIT GETRANGE GETSET HDEL HEXISTS HGET HGETALL HINCRBY HINCRBYFL OAT HKEYS HLEN HMGET HMSET HSET HSETNX HSTRLEN HVALS INCR INCRBY INCRBYFLOAT INFO KEYS LASTSAVE LINDEX LINSERT LLEN LPOP LPU SH LPUSHX LRANGE LREM LSET LTRIM MEMORY MGET MONITOR MOVE MSET MSETNX MULTI OBJECT PERSIST PEXPIRE PEXPIREAT PFADD PFCOUNT PFMERGE PING PSETEX PSUBSCRIBE PUBSUB PTTL PUBLISH PUNSUBSCRIBE RANDOMKEY READONLY READWRITE RENAME RENAMENX RESTORE ROLE RPOP RPOPLPUSH RPUSH RPUSHX SADD SCARD SCRIPT SDIFF SDIFFSTORE SELECT SET SETBIT SETEX SETNX SETRANGE SINTER SINTERSTORE SISMEMBER SLOWLOG SMEMBERS SMOVE SORT SPOP SRANDMEMBER SREM STRLEN SUBSCRIBE SUNION SUNIONSTORE SWAPDB TIME TOUCH TTL TYPE UNSUBSCRIBE UNLINK UNWATCH WAIT WATCH ZADD ZCARD ZCOUNT ZINCRBY ZINTERSTO RE ZLEXCOUNT ZPOPMAX ZPOPMIN ZRANGE ZRANGEBYLEX ZREVRANGE BYLEX ZRANGEBYSCORE ZRANK ZREM ZREMRANGEBYLEX ZREMRANGEBYRANK ZREMRANGEBYSCORE ZREVRANGE ZREVRANGEBYSCORE ZREVRANK ZSCORE ZUNIONSTORE SCAN SSCAN HSCAN

Nama	Detail	Deskripsi
		`ZSCAN XINFO XADD XTRIM XDEL XRANGE XREVRANGE XLEN XREAD XGROUP XREADGROUP XACK XCLAIM XPENDING GEORADIUS_RO GEORADIUSBYMEMBER_RO LOLWUT XSETID SUBSTR`

Untuk informasi selengkapnya, lihat ElastiCache versi 5.0.6 untuk Redis OSS (ditingkatkan).

Redis OSS 5.0.0 perubahan parameter

Keluarga grup parameter: redis5.0

Redis OSS 5.0 grup parameter default

- `default.redis5.0`— Gunakan grup parameter ini, atau yang diturunkan darinya, untuk cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dan grup replikasi.
- `default.redis5.0.cluster.on`— Gunakan grup parameter ini, atau yang diturunkan darinya, untuk cluster Valkey atau Redis OSS (mode cluster diaktifkan) dan grup replikasi.

Parameter ditambahkan di Redis OSS 5.0

Nama	Detail	Deskripsi
`stream-node-max-bytes`	Nilai yang diizinkan: 0+ Default: 4096 Jenis: integer Dapat diubah: Ya Perubahan berlaku: Segera.	Struktur aliran data adalah pohon radix simpul yang mengodekan beberapa item dalamnya. Gunakan konfigurasi ini untuk menentukan ukuran maksimum simpul tunggal dalam pohon radix dalam Byte. Jika diatur ke 0, ukuran simpul pohon adalah tidak terbatas.
`stream-node-max-entries`	Nilai yang diizinkan: 0+ Default: 100	Struktur data aliran adalah pohon radix simpul yang mengenkode beberapa item dalamnya. Gunakan konfigurasi ini untuk menentukan jumlah maksimum item yang dapat ditampung

Nama	Detail	Deskripsi
	Jenis: integer Dapat diubah: Ya Perubahan berlaku: Segera.	simpul tunggal sebelum beralih ke simpul baru saat menambahkan entri aliran baru. Jika diatur ke 0, jumlah item di simpul pohon adalah tidak terbatas
`active-defrag-max-scan-fields`	Nilai yang diizinkan: 1 hingga 1000000 Default: 1000 Jenis: integer Dapat diubah: Ya Perubahan berlaku: Segera.	Jumlah maksimum set/hash/zset/list bidang yang akan diproses dari pemindaian kamus utama
`lua-replicate-commands`	Nilai yang diizinkan: yes/no Default: yes Jenis: boolean Dapat diubah: Ya Perubahan berlaku: Segera.	Selalu mengaktifkan replikasi efek Lua atau tidak dalam skrip Lua
`replica-ignore-maxmemory`	Default: yes Jenis: boolean Dapat diubah: Tidak	Menentukan apakah replika mengabaikan pengaturan `maxmemory` dengan tidak mengosongkan item yang independen dari primer

Redis OSS telah mengganti nama beberapa parameter di engine versi 5.0 sebagai tanggapan atas umpan balik komunitas. Untuk informasi selengkapnya, lihat Apa yang Baru di Redis OSS 5? . Tabel berikut mencantumkan nama baru dan pemetaannya ke versi sebelumnya.

Parameter berganti nama dalam Redis OSS 5.0

Nama	Detail	Deskripsi
`replica-lazy-flush`	Default: yes Jenis: boolean Dapat diubah: Tidak Nama sebelumnya: slave-lazy-flush	Melakukan flushDB asinkron selama sinkronisasi replika.
`client-output-buffer-limit-replica-hard-limit`	Default: Untuk nilai, lihat Parameter spesifik tipe node Redis OSS Jenis: integer Dapat Diubah: Tidak Nama sebelumnya: client-output-buffer-limit - slave-hard-limit	Untuk Redis OSS baca replika: Jika buffer keluaran klien mencapai jumlah byte yang ditentukan, klien akan terputus.
`client-output-buffer-limit-replica-soft-limit`	Default: Untuk nilai, lihat Parameter spesifik tipe node Redis OSS Jenis: integer Dapat Diubah: Tidak Nama sebelumnya: client-output-buffer-limit - slave-soft-limit	Untuk Redis OSS baca replika: Jika buffer keluaran klien mencapai jumlah byte yang ditentukan, klien akan terputus, tetapi hanya jika kondisi ini berlanjut. `client-output-buffer-limit-replica-soft-seconds`
`client-output-buffer-limit-`	Default: 60 Jenis: integer	Untuk Redis OSS baca replika: Jika buffer keluaran klien tetap pada `client-output-buffer-limit-replica-soft-limit`

Nama	Detail	Deskripsi
`replica-soft-seconds`	Dapat Diubah: Tidak Nama sebelumnya: client-output-buffer-limit - slave-soft-seconds	byte lebih lama dari jumlah detik ini, klien akan terputus.
`replica-allow-chaining`	Default: no Jenis: string Dapat diubah: Tidak Nama sebelumnya: slave-allow-chaining	Menentukan apakah replika baca di Redis OSS dapat membaca replika sendiri.
`min-replicas-to-write`	Default: 0 Jenis: integer Dapat diubah: Ya Nama sebelumnya: min-slaves-to-write Perubahan berlaku: Segera	Jumlah minimum replika baca yang harus tersedia agar simpul primer dapat menerima penulisan dari klien. Jika jumlah replika yang tersedia di bawah jumlah ini, maka simpul primer tidak akan lagi menerima permintaan tulis. Jika parameter min-replicas-max-lag ini atau 0, maka node utama akan selalu menerima permintaan penulisan, bahkan jika tidak ada replika yang tersedia.

Nama	Detail	Deskripsi
`min-replicas-max-lag`	Default: 10 Jenis: integer Dapat diubah: Ya Nama sebelumnya: min-slaves-max-lag Perubahan Berlaku: Segera	Jumlah detik saat simpul primer harus menerima permintaan ping dari replika baca. Jika jumlah waktu ini berlalu dan primer tidak menerima ping, maka replika tidak lagi dianggap tersedia. Jika jumlah replika yang tersedia turun di bawah min-replicas-to-write, maka primer akan berhenti menerima penulisan pada saat itu. Jika salah satu parameter min-replicas-to-write ini atau 0, maka node utama akan selalu menerima permintaan tulis, bahkan jika tidak ada replika yang tersedia.
`close-on-replica-write`	Default: yes Jenis: boolean Dapat Diubah: Ya Nama sebelumnya: close-on-slave-write Perubahan Berlaku: Segera	Jika diaktifkan, klien yang mencoba menulis ke replika hanya baca akan terputus.

Parameter dihapus di Redis OSS 5.0

Nama	Detail	Deskripsi
`repl-timeout`	Default: 60 Dapat diubah: Tidak	Parameter tidak tersedia dalam versi ini.

Redis OSS 4.0.10 perubahan parameter

Keluarga grup parameter: redis4.0

Redis OSS 4.0.x grup parameter default

- `default.redis4.0`— Gunakan grup parameter ini, atau yang diturunkan darinya, untuk cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dan grup replikasi.
- `default.redis4.0.cluster.on`— Gunakan grup parameter ini, atau yang diturunkan darinya, untuk cluster Valkey atau Redis OSS (mode cluster diaktifkan) dan grup replikasi.

Parameter berubah di Redis OSS 4.0.10

Nama	Detail	Deskripsi
`maxmemory-policy`	Nilai yang diizinkan: `allkeys-lru` , `volatile-lru` , **`allkeys-lfu`** , **`volatile-lfu`** , `allkeys-random` , `volatile-random` , `volatile-ttl` , `noeviction` Default: volatile-lru Jenis: string Dapat diubah: Ya Perubahan berlaku: segera	`maxmemory-policy` telah ditambahkan dalam versi 2.6.13. Dalam versi 4.0.10 ditambahkan dua nilai baru yang diizinkan : `allkeys-lfu` , yang akan mengosongkan setiap kunci menggunakan LFU yang diperkirakan, dan `volatile-lfu` , yang akan mengosongkan kunci dengan set yang kedaluwarsa menggunakan LFU yang diperkirakan. Dalam versi 6.2, ketika keluarga simpul r6gd diperkenalkan untuk digunakan dengan tingkatan data, hanya kebijakan max-memory `noeviction` , `volatile-lru` dan `allkeys-lru` yang didukung dengan jenis simpul r6gd.

Parameter ditambahkan di Redis OSS 4.0.10

Nama	Detail	Deskripsi
Parameter penghapusan asinkron		
`lazyfree-lazy-eviction`	Nilai yang diizinkan: ya/tidak Default: no Jenis: boolean	Melakukan penghapusan asinkron pada pengosongan.

Nama	Detail	Deskripsi
	Dapat diubah: Ya Perubahan akan diterapkan: segera	
`lazyfree-lazy-expire`	Nilai yang diizinkan: ya/tidak Default: no Jenis: boolean Dapat diubah: Ya Perubahan akan diterapkan: segera	Melakukan penghapusan asinkron pada kunci kedaluwarsa.
`lazyfree-lazy-server-del`	Nilai yang diizinkan: ya/tidak Default: no Jenis: boolean Dapat diubah: Ya Perubahan akan diterapkan: segera	Melakukan penghapusan asinkron untuk perintah yang memperbarui nilai.
`slave-lazy-flush`	Nilai yang diizinkan: N/A Default: no Jenis: boolean Dapat Diubah: Tidak Perubahan akan diterapkan: N/A	Melakukan FlushDB asinkron selama sinkronisasi slave.

Nama	Detail	Deskripsi
Parameter LFU		
`lfu-log-factor`	Nilai yang diizinkan: semua integer > 0 Default: 10 Jenis: integer Dapat diubah: Ya Perubahan akan diterapkan: segera	Mengatur faktor log, yang menentukan jumlah temuan kunci untuk memenuhi penghitung kunci.
`lfu-decay-time`	Nilai yang diizinkan: integer apa pun Default: 1 Jenis: integer Dapat diubah: Ya Perubahan akan diterapkan: segera	Jumlah waktu dalam menit untuk mengurangi penghitung kunci.
Parameter defragmentasi aktif		

Nama	Detail	Deskripsi
activedefrag	Nilai yang diizinkan: ya/tidak Default: no Jenis: boolean Dapat diubah: Ya Perubahan akan diterapkan: segera	Defragmentasi aktif diaktifkan.
active-defrag-ignore-bytes	Nilai yang diizinkan: 10485760-104857600 Default: 104857600 Jenis: integer Dapat diubah: Ya Perubahan akan diterapkan: segera	Jumlah minimum sisa fragmentasi untuk memulai defrag aktif.
active-defrag-threshold-lower	Nilai yang diizinkan: 1-100 Default: 10 Jenis: integer Dapat diubah: Ya Perubahan akan diterapkan: segera	Persentase minimum fragmentasi untuk memulai defrag aktif.

Nama	Detail	Deskripsi
`active-defrag-threshold-upper`	Nilai yang diizinkan: 1-100 Default: 100 Jenis: integer Dapat diubah: Ya Perubahan akan diterapkan: segera	Persentase maksimum fragmentasi yang mana kita menggunakan upaya maksimal.
`active-defrag-cycle-min`	Nilai yang diizinkan: 1-75 Default: 25 Jenis: integer Dapat diubah: Ya Perubahan akan diterapkan: segera	Upaya minimal untuk defrag dalam persentase CPU.
`active-defrag-cycle-max`	Nilai yang diizinkan: 1-75 Default: 75 Jenis: integer Dapat diubah: Ya Perubahan akan diterapkan: segera	Upaya maksimal untuk defrag dalam persentase CPU.
Parameter buffer output klien		

Nama	Detail	Deskripsi
`client-query-buffer-limit`	Nilai yang diizinkan: 1048576-1073741824 Default: 1073741824 Jenis: integer Dapat diubah: Ya Perubahan akan diterapkan: segera	Ukuran maks buffer kueri klien tunggal.
`proto-max-bulk-len`	Nilai yang diizinkan: 1048576-536870912 Default: 536870912 Jenis: integer Dapat diubah: Ya Perubahan akan diterapkan: segera	Ukuran maks dari permintaan elemen tunggal.

Redis OSS 3.2.10 perubahan parameter

Keluarga grup parameter: redis3.2

ElastiCache untuk Redis OSS 3.2.10 tidak ada parameter tambahan yang didukung.

Redis OSS 3.2.6 perubahan parameter

Keluarga grup parameter: redis3.2

Untuk Redis OSS 3.2.6 tidak ada parameter tambahan yang didukung.

Redis OSS 3.2.4 perubahan parameter

Keluarga grup parameter: redis3.2

Dimulai dengan Redis OSS 3.2.4 ada dua kelompok parameter default.

- `default.redis3.2`— Saat menjalankan Redis OSS 3.2.4, tentukan grup parameter ini atau yang diturunkan darinya, jika Anda ingin membuat grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) dan masih menggunakan fitur tambahan Redis OSS 3.2.4.
- `default.redis3.2.cluster.on`— Tentukan grup parameter ini atau yang diturunkan darinya, saat Anda ingin membuat grup replikasi Valkey atau Redis OSS (mode cluster diaktifkan).

Topik

Parameter baru untuk Redis OSS 3.2.4

Keluarga grup parameter: redis3.2

Untuk Redis OSS 3.2.4 parameter tambahan berikut didukung.

Nama	Detail	Deskripsi
`list-max-ziplist-size`	Default: -2 Jenis: integer Dapat diubah: Tidak	Daftar dikodekan dengan cara khusus untuk menghemat ruang. Jumlah entri yang diizinkan per simpul daftar internal dapat ditentukan sebagai ukuran maksimum tetap atau jumlah maksimum elemen. Untuk ukuran maksimum tetap, gunakan -5 hingga -1, yang berarti: • -5: ukuran maks: 64 Kb - tidak direkomendasikan untuk beban kerja normal • -4: ukuran maks: 32 Kb - tidak direkomendasikan • -3: ukuran maks: 16 Kb - tidak direkomendasikan •

Nama	Detail	Deskripsi
		-2: ukuran maks: 8 Kb - direkomendasikan • -1: ukuran maks: 4 Kb - direkomendasikan • Angka positif berarti menyimpan hingga persis jumlah elemen per simpul daftar tersebut.
`list-compress-depth`	Default: 0 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Daftar juga dapat dikompresi. Kedalaman kompresi adalah jumlah simpul quicklist ziplist dari setiap sisi daftar yang akan dikecualikan dari kompresi. Kepala dan ekor dari daftar selalu tidak dikompresi untuk operasi "fast push and pop". Pengaturannya adalah: • 0: Menonaktifkan semua kompresi. • 1: Mulai mengompresi dengan simpul pertama masuk dari kepala dan ekor. [head]->node->node->...->node->[tail] Semua simpul kecuali jika [head] dan [tail] dikompresi. • 2: Mulai mengompresi dengan simpul kedua masuk dari kepala dan ekor. [head]->[next]->node->node->...->node->[prev]->[tail] [head], [next], [prev], [tail] tidak dikompresi. Semua simpul lainnya dikompresi. • Dll.

Nama	Detail	Deskripsi
`cluster-enabled`	Default: no/yes * Jenis: string Dapat diubah: Tidak	Menunjukkan apakah ini adalah grup replikasi Valkey atau Redis OSS (mode cluster diaktifkan) dalam mode cluster (ya) atau grup replikasi Valkey atau Redis OSS (mode cluster diaktifkan) dalam mode non-cluster (tidak). Grup replikasi Valkey atau Redis OSS (mode cluster enabled) dalam mode cluster dapat mempartisi data mereka hingga 500 grup node. * Redis OSS 3.2. x memiliki dua kelompok parameter default. • `default.redis3.2` – nilai default no. • `default.redis3.2.cluster.on` – nilai default `yes`. .

Nama	Detail	Deskripsi
`cluster-require-full-coverage`	Default: no Jenis: boolean Dapat diubah: ya Perubahan Berlaku: Segera	Ketika diatur `keyes`, node Valkey atau Redis OSS (mode cluster diaktifkan) dalam mode cluster berhenti menerima kueri jika mereka mendeteksi setidaknya ada satu slot hash yang ditemukan (tidak ada node yang tersedia yang menyajikannya). Dengan cara ini jika sebagian klaster berhenti, klaster menjadi tidak tersedia. Klaster secara otomatis menjadi tersedia lagi begitu semua slot tercakup lagi. Namun, terkadang Anda ingin subset klaster yang berfungsi terus menerima permintaan untuk bagian dari ruang kunci yang masih tercakup. Untuk melakukannya, cukup atur opsi `cluster-require-full-coverage` ke `no`.

Nama	Detail	Deskripsi
`hll-sparse-max-bytes`	Default: 3000 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	HyperLogLog batas byte representasi jarang. Batas termasuk header 16 byte. Ketika HyperLogLog menggunakan representasi jarang melintasi batas ini, itu diubah menjadi representasi padat. Nilai yang lebih besar dari 16000 tidak disarankan karena pada titik tersebut dense representation lebih hemat memori. Kami merekomendasikan nilai sekitar 3000 untuk mendapatkan manfaat dari pengenkodean hemat ruang tanpa terlalu memperlambat PFADD, yaitu O(N) dengan sparse encoding. Nilai dapat dinaikkan menjadi ~ 10000 ketika CPU tidak menjadi perhatian, tetapi ruang adalah, dan kumpulan data terdiri dari banyak HyperLogLogs dengan kardinalitas dalam kisaran 0 - 15000.

Nama	Detail	Deskripsi
`reserved-memory-percent`	Default: 25 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Persentase memori simpul yang dicadangkan untuk penggunaan nondata. Secara default, jejak data Redis OSS tumbuh hingga menghabiskan semua memori node. Jika ini terjadi, maka performa simpul kemungkinan akan terdampak negatif karena memory paging yang berlebihan. Dengan memesan memori, Anda dapat menyisihkan beberapa memori yang tersedia untuk tujuan OSS non-Redis untuk membantu mengurangi jumlah paging. Parameter ini khusus untuk ElastiCache, dan bukan bagian dari distribusi OSS Redis standar. Untuk informasi selengkapnya, lihat `reserved-memory` dan Mengelola memori cadangan untuk Valkey dan Redis OSS.

Parameter berubah di Redis OSS 3.2.4 (ditingkatkan)

Keluarga grup parameter: redis3.2

Untuk Redis OSS 3.2.4 parameter berikut diubah.

Nama	Detail	Perubahan
`activerehashing`	Dapat diubah: Ya jika grup parameter tidak terkait dengan klaster cache. Jika sebaliknya, tidak.	Dapat diubah adalah Tidak.
`databases`	Dapat diubah: Ya jika grup parameter tidak terkait dengan	Dapat diubah adalah Tidak.

Nama	Detail	Perubahan
	klaster cache. Jika sebaliknya, tidak.	
`appendonly`	Default: nonaktif Dapat diubah: Tidak	Jika Anda ingin memutakhirkan dari versi Redis OSS sebelumnya, Anda harus mematikan `appendonly` terlebih dahulu.
`appendfsync`	Default: nonaktif Dapat diubah: Tidak	Jika Anda ingin memutakhirkan dari versi Redis OSS sebelumnya, Anda harus mematikan `appendfsync` terlebih dahulu.
`repl-timeout`	Default: 60 Dapat diubah: Tidak	Sekarang tidak dapat diubah dengan default 60.
`tcp-keepalive`	Default: 300	Default adalah 0.
`list-max-ziplist-entries`		Parameter tidak lagi tersedia.
`list-max-ziplist-value`		Parameter tidak lagi tersedia.

Redis OSS 2.8.24 (ditingkatkan) menambahkan parameter

Keluarga grup parameter: redis2.8

Untuk Redis OSS 2.8.24 tidak ada parameter tambahan yang didukung.

Redis OSS 2.8.23 (ditingkatkan) menambahkan parameter

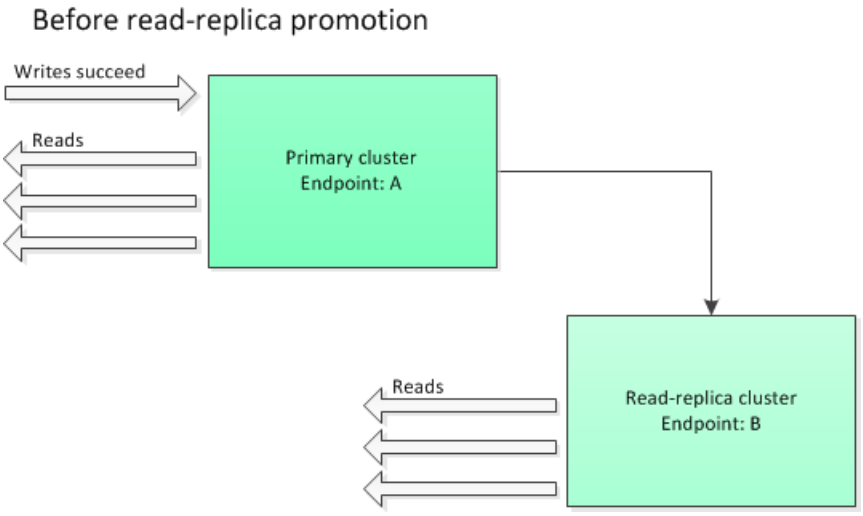
Keluarga grup parameter: redis2.8

Untuk Redis OSS 2.8.23 parameter tambahan berikut didukung.

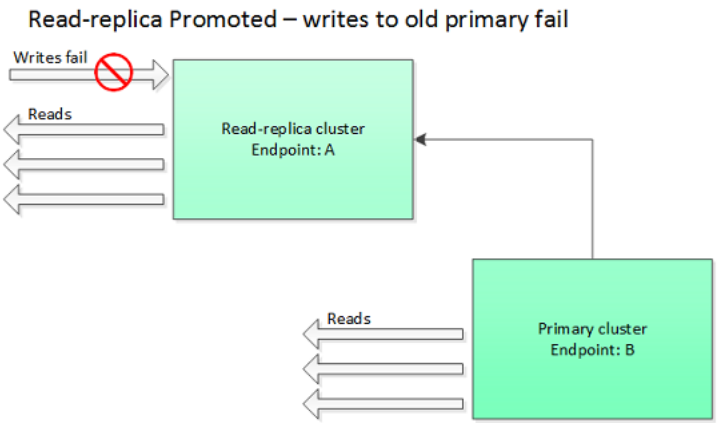
Nama	Detail	Deskripsi
`close-on-slave-write`	Default: yes Jenis: string (ya/tidak) Dapat diubah: Ya Perubahan Berlaku: Segera	Jika diaktifkan, klien yang mencoba menulis ke replika hanya baca akan terputus.

Cara kerja close-on-slave-write

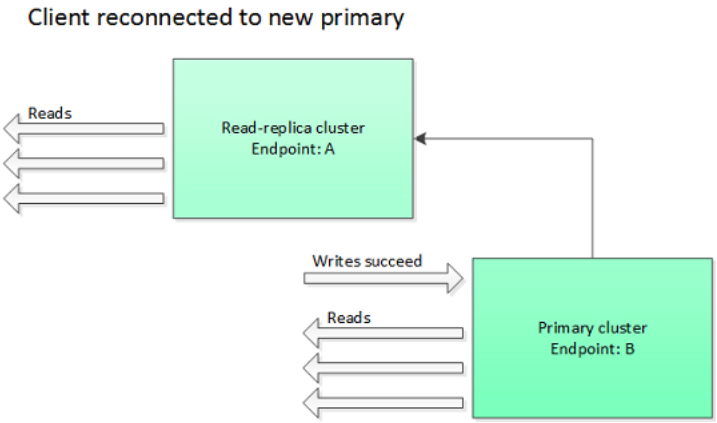
`close-on-slave-write`Parameter ini diperkenalkan oleh Amazon ElastiCache untuk memberi Anda kontrol lebih besar atas bagaimana cluster Anda merespons saat node utama dan node replika baca bertukar peran karena mempromosikan replika baca ke primer.



Jika klaster replika baca dipromosikan ke primer untuk alasan apa pun selain failover grup replikasi dengan Multi-AZ diaktifkan, klien akan terus mencoba menulis ke titik akhir A. Karena titik akhir A sekarang adalah titik akhir untuk replika baca, penulisan ini akan gagal. Ini adalah perilaku untuk Redis OSS sebelum ElastiCache memperkenalkan `close-on-replica-write` dan perilaku jika Anda menonaktifkan. `close-on-replica-write`



Dengan `close-on-replica-write` diaktifkan, setiap kali klien mencoba menulis ke replika baca, koneksi klien ke klaster ditutup. Logika aplikasi Anda harus mendeteksi pemutusan koneksi, memeriksa tabel DNS, dan menghubungkan kembali ke titik akhir primer, yang sekarang akan menjadi titik akhir B.



Kapan Anda mungkin menonaktifkan close-on-replica-write

Jika penonaktifan `close-on-replica-write` mengakibatkan gagalnya penulisan ke klaster, mengapa `close-on-replica-write` dinonaktifkan?

Seperti yang telah disebutkan, dengan mengaktifkan `close-on-replica-write`, setiap kali klien mencoba menulis ke replika baca, koneksi klien ke klaster akan ditutup. Pembuatan koneksi baru ke simpul membutuhkan waktu. Jadi, pemutusan koneksi dan pembuatan koneksi kembali sebagai akibat dari permintaan tulis ke replika juga memengaruhi latensi permintaan baca yang dilayani melalui koneksi yang sama. Efek ini tetap ada sampai dibuatnya koneksi baru. Jika aplikasi Anda sangat sarat dengan operasi baca atau sangat sensitif terhadap latensi, Anda dapat menjaga klien tetap terhubung untuk menghindari performa baca yang menurun.

Redis OSS 2.8.22 (ditingkatkan) menambahkan parameter

Keluarga grup parameter: redis2.8

Untuk Redis OSS 2.8.22 tidak ada parameter tambahan yang didukung.

> **Important**
>
> - Dimulai dengan Redis OSS versi 2.8.22, `repl-backlog-size` berlaku untuk cluster utama serta cluster replika.
> - Dimulai dengan Redis OSS versi 2.8.22, `repl-timeout` parameter tidak didukung. Jika diubah, ElastiCache akan menimpa dengan default (60an), seperti yang kita lakukan dengan`appendonly`.

Parameter berikut tidak lagi didukung.

- appendonly
- appendfsync
- repl-timeout

Redis OSS 2.8.21 menambahkan parameter

Keluarga grup parameter: redis2.8

Untuk Redis OSS 2.8.21, tidak ada parameter tambahan yang didukung.

Redis OSS 2.8.19 menambahkan parameter

Keluarga grup parameter: redis2.8

Untuk Redis OSS 2.8.19 tidak ada parameter tambahan yang didukung.

Redis OSS 2.8.6 menambahkan parameter

Keluarga grup parameter: redis2.8

Untuk Redis OSS 2.8.6 parameter tambahan berikut didukung.

Nama	Detail	Deskripsi
`min-slaves-max-lag`	Default: 10 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Jumlah detik saat simpul primer harus menerima permintaan ping dari replika baca. Jika jumlah waktu ini berlalu dan primer tidak menerima ping, maka replika tidak lagi dianggap tersedia. Jika jumlah replika yang tersedia turun di bawah min-slaves-to-write, maka primer akan berhenti menerima penulisan pada saat itu. Jika parameter min-slaves-to-writ e ini atau 0, maka node utama akan selalu menerima permintaa n penulisan, bahkan jika tidak ada replika yang tersedia.
`min-slaves-to-write`	Default: 0 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Jumlah minimum replika baca yang harus tersedia agar simpul primer dapat menerima penulisan dari klien. Jika jumlah replika yang tersedia di bawah jumlah ini, maka simpul primer tidak akan lagi menerima permintaan tulis. Jika parameter min-slaves-max-lag ini atau 0, maka node utama akan selalu menerima permintaa n penulisan, bahkan jika tidak ada replika yang tersedia.
`notify-keyspace-events`	Default: (string kosong) Jenis: string	Jenis-jenis acara keyspace yang Redis OSS dapat memberita hu klien. Setiap jenis peristiwa direpresentasikan oleh satu huruf:

Nama	Detail	Deskripsi
	Dapat diubah: Ya Perubahan Berlaku: Segera	• K – Peristiwa ruang kunci, dipublikasikan dengan awalan __keyspace@<db>__ • E – Peristiwa key-event, dipublikasikan dengan awalan __keyevent@<db>__ • g – Perintah generik non-spesifik seperti DEL, EXPIRE, RENAME, dll. • $ – Perintah String • l – Perintah Daftar • s – Perintah Set • h – Perintah Hash • z – Perintah Sorted set • x – Peristiwa kedaluwarsa (peristiwa yang dihasilkan setiap kali kunci kedaluwarsa) • e – Peristiwa pengosongan (peristiwa yang dihasilkan ketika kunci dikosongkan untuk maxmemory) • A – Alias untuk g$lshzxe

Nama	Detail	Deskripsi
		Anda dapat memiliki kombinasi semua jenis peristiwa ini. Misalnya, AKE berarti bahwa Redis OSS dapat mempublik asikan notifikasi dari semua jenis acara. Jangan gunakan karakter selain yang tercantum di atas; jika tidak, pesan kesalahan akan dihasilkan. Secara default, parameter ini diatur ke string kosong, yang berarti bahwa notifikasi peristiwa ruang kunci dinonaktifkan.

Nama	Detail	Deskripsi
`repl-backlog-size`	Default: 1048576 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Ukuran, dalam byte, buffer backlog simpul primer. Backlog digunakan untuk mencatat pembaruan data pada simpul primer. Ketika replika baca terhubung ke primer, replika ini mencoba melakukan sinkronisasi parsial (`psync`), yang menerapkan data dari backlog untuk mengejar simpul primer. Jika `psync` gagal, maka sinkronisasi penuh diperlukan. Nilai minimum untuk parameter ini adalah 16384. **Note** Dimulai dengan Redis OSS 2.8.22, parameter ini berlaku untuk cluster primer serta replika baca.

Nama	Detail	Deskripsi
`repl-backlog-ttl`	Default: 3600 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Jumlah detik saat simpul primer mempertahankan buffer backlog. Mulai dari waktu simpul replika terakhir terputus, data dalam backlog akan tetap utuh sampai `repl-backlog-ttl` kedaluwarsa. Jika replika tidak terhubung ke primer dalam waktu ini, maka primer akan melepaskan buffer backlog. Ketika replika akhirnya terhubung kembali, replikasi ini harus melakukan sinkronisasi penuh dengan primer. Jika parameter ini diatur ke 0, maka buffer backlog tidak akan pernah dilepas.
`repl-timeout`	Default: 60 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Merepresentasikan periode waktu habis, dalam detik, untuk: • Transfer data massal selama sinkronisasi, dari perspektif replika baca • Waktu habis simpul primer dari perspektif replika • Waktu habis replika dari perspektif simpul primer

Redis OSS 2.6.13 parameter

Keluarga grup parameter: redis2.6

Redis OSS 2.6.13 adalah versi pertama dari Redis OSS yang didukung oleh. ElastiCache Tabel berikut menunjukkan Redis OSS 2.6.13 parameter yang mendukung. ElastiCache

Nama	Detail	Deskripsi
`activerehashing`	Default: yes Jenis: string (ya/tidak) Dapat diubah: Ya Perubahan berlaku: Saat Pembuatan	Menentukan apakah akan mengaktifkan fitur rehashing aktif Redis. Tabel hash utama di-rehash sepuluh kali per detik; setiap operasi rehash mengonsumsi 1 milidetik waktu CPU. Nilai ini diatur saat Anda membuat grup parameter. Ketika menetapkan grup parameter baru untuk klaster, nilai ini harus sama dalam grup parameter lama dan baru.
`appendonly`	Default: no Jenis: string Dapat diubah: Ya Perubahan Berlaku: Segera	Mengaktifkan atau menonaktifkan fitur append only file (AOF) Redis. AOF menangkap setiap perintah Redis OSS yang mengubah data dalam cache, dan digunakan untuk memulihkan dari kegagalan node tertentu. Nilai default adalah tidak, yang berarti AOF dinonaktifkan. Atur parameter ini ke yes untuk mengaktifkan AOF. Untuk informasi selengkapnya, lihat Mitigasi Kegagalan. **Note** Append Only File (AOF) tidak didukung untuk simpul cache.t1.micro dan cache.t2.*. Untuk simpul jenis ini, nilai parameter `appendonly` akan diabaikan.

Nama	Detail	Deskripsi
		Note Untuk grup replikasi Multi-AZ, AOF tidak diizinkan.
`appendfsync`	Default: everysec Jenis: string Dapat diubah: Ya Perubahan Berlaku: Segera	Saat `appendonly` diatur ke yes, akan mengontrol seberapa sering buffer output AOF ditulis ke disk: • no – buffer di-flushing ke disk sesuai kebutuhan. • everysec – buffer di-flushing sekali per detik. Ini adalah opsi default. • always – buffer di-flushing setiap kali data dalam klaster diubah. • Appendfsync tidak didukung untuk versi 2.8.22 dan yang lebih baru.
`client-output-buffer-limit-normal-hard-limit`	Default: 0 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Jika buffer output klien mencapai jumlah byte tertentu, klien akan terputus. Default-nya adalah nol (tidak ada batas absolut).
`client-output-buffer-limit-normal-soft-limit`	Default: 0 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Jika buffer output klien mencapai jumlah byte tertentu, klien akan terputus, tetapi hanya jika kondisi ini bertahan selama `client-output-buffer-limit-normal-soft-seconds` . Default-nya adalah nol (tidak ada batas relatif).

Nama	Detail	Deskripsi
client-output-buffer-limit-normal-soft-seconds	Default: 0 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Jika buffer output klien tetap pada client-output-buffer-limit-normal-soft-limit byte lebih lama dari jumlah detik ini, klien akan terputus. Default-nya adalah nol (tidak ada batas waktu).
client-output-buffer-limit-pubsub-hard-limit	Default: 33554432 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Untuk publish/subscribe klien Redis OSS: Jika buffer keluaran klien mencapai jumlah byte yang ditentukan, klien akan terputus.
client-output-buffer-limit-pubsub-soft-limit	Default: 8388608 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Untuk publish/subscribe klien Redis OSS: Jika buffer keluaran klien mencapai jumlah byte yang ditentukan, klien akan terputus, tetapi hanya jika kondisi ini berlanjut. client-output-buffer-limit-pubsub-soft-seconds
client-output-buffer-limit-pubsub-soft-seconds	Default: 60 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Untuk publish/subscribe klien Redis OSS: Jika buffer keluaran klien tetap pada client-output-buffer-limit-pubsub-soft-limit byte lebih lama dari jumlah detik ini, klien akan terputus.
client-output-buffer-limit-slave-hard-limit	Default: Untuk nilai, lihat Parameter spesifik tipe node Redis OSS Jenis: integer Dapat Diubah: Tidak	Untuk Redis OSS baca replika: Jika buffer keluaran klien mencapai jumlah byte yang ditentukan, klien akan terputus.

Nama	Detail	Deskripsi
`client-output-buffer-limit-slave-soft-limit`	Default: Untuk nilai, lihat Parameter spesifik tipe node Redis OSS Jenis: integer Dapat Diubah: Tidak	Untuk Redis OSS baca replika: Jika buffer keluaran klien mencapai jumlah byte yang ditentukan, klien akan terputus, tetapi hanya jika kondisi ini berlanjut. `client-output-buffer-limit-slave-soft-seconds`
`client-output-buffer-limit-slave-soft-seconds`	Default: 60 Jenis: integer Dapat Diubah: Tidak	Untuk Redis OSS baca replika: Jika buffer keluaran klien tetap pada `client-output-buffer-limit-slave-soft-limit` byte lebih lama dari jumlah detik ini, klien akan terputus.
`databases`	Default: 16 Jenis: integer Dapat diubah: Tidak Perubahan Berlaku: Pada Pembuatan	Jumlah partisi logis yang membagi basis data. Kami merekomendasikan untuk menjaga nilai ini tetap rendah. Nilai ini diatur saat Anda membuat grup parameter. Ketika menetapkan grup parameter baru untuk klaster, nilai ini harus sama baik dalam grup parameter lama dan baru.
`hash-max-ziplist-entries`	Default: 512 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Menentukan jumlah memori yang digunakan untuk hash. Hash dengan jumlah entri kurang dari yang ditentukan akan disimpan menggunakan pengodean khusus yang menghemat ruang.

Nama	Detail	Deskripsi
`hash-max-ziplist-value`	Default: 64 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Menentukan jumlah memori yang digunakan untuk hash. Hash dengan entri yang lebih kecil dari jumlah byte yang ditentukan akan disimpan menggunakan pengodean khusus yang menghemat ruang.
`list-max-ziplist-entries`	Default: 512 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Menentukan jumlah memori yang digunakan untuk daftar. Daftar dengan jumlah entri kurang dari yang ditentukan akan disimpan menggunakan pengodean khusus yang menghemat ruang.
`list-max-ziplist-value`	Default: 64 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Menentukan jumlah memori yang digunakan untuk daftar. Daftar dengan entri yang lebih kecil dari jumlah byte yang ditentukan akan disimpan menggunakan pengodean khusus yang menghemat ruang.
`lua-time-limit`	Default: 5000 Jenis: integer Dapat Diubah: Tidak	Waktu eksekusi maksimum untuk skrip Lua, dalam milidetik, sebelum ElastiCache mengambil tindakan untuk menghentikan skrip. Jika *`lua-time-limit`* terlampaui, semua perintah Redis OSS akan mengembalikan kesalahan bentuk ____-BUSY. Karena keadaan ini dapat menyebabkan gangguan dengan banyak operasi Redis OSS penting, pertama-tama ElastiCache akan mengeluarkan perintah SCRIPT KILL. Jika ini tidak berhasil, ElastiCache akan secara paksa restart Redis OSS.

Nama	Detail	Deskripsi
`maxclients` Nilai ini berlaku untuk semua jenis instans kecuali yang ditentukan secara eksplisit	Default: 65000 Jenis: integer Dapat diubah: Tidak	Jumlah maksimum klien yang dapat dihubungkan pada satu waktu.
	t2.medium Default: 20000 Jenis: integer Dapat diubah: Tidak	
	t2.small Default: 20000 Jenis: integer Dapat diubah: Tidak	
	t2.micro Default: 20000 Jenis: integer Dapat diubah: Tidak	
	t4g.micro Default: 20000 Jenis: integer Dapat Diubah: Tidak	
	t3.medium Standar: 46000 Jenis: integer Dapat Diubah: Tidak	
	t3.small Default: 46000 Jenis: integer Dapat diubah: Tidak	

Nama	Detail	Deskripsi
	t3.micro Default: 20000 Jenis: integer Dapat diubah: Tidak	
`maxmemory-policy`	Default: volatile-lru Jenis: string Dapat diubah: Ya Perubahan Berlaku: Segera	Kebijakan pengosongan untuk kunci saat penggunaan memori maksimum tercapai. Nilai yang valid adalah: `volatile-lru | allkeys-lru | volatile-random | allkeys-random | volatile-ttl | noeviction` Untuk informasi selengkapnya, lihat Menggunakan Valkey atau Redis OSS sebagai cache LRU.
`maxmemory-samples`	Default: 3 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Untuk perhitungan least-recently-used (LRU) dan time-to-live (TTL), parameter ini mewakili ukuran sampel kunci untuk diperiksa. Secara default, Redis OSS memilih 3 kunci dan menggunakan salah satu yang paling tidak digunakan baru-baru ini.

Nama	Detail	Deskripsi
`reserved-memory`	Default: 0 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Total memori, dalam byte, yang dicadangkan untuk penggunaan non-data. Secara default, node Redis OSS akan tumbuh hingga mengkonsumsi node `maxmemory` (lihat). Parameter spesifik tipe node Redis OSS Jika ini terjadi, maka performa simpul kemungkinan akan terdampak negatif karena memory paging yang berlebihan. Dengan memesan memori Anda dapat menyisihkan beberapa memori yang tersedia untuk tujuan OSS non-Redis untuk membantu mengurangi jumlah paging. Parameter ini khusus untuk ElastiCache, dan bukan bagian dari distribusi OSS Redis standar. Untuk informasi selengkapnya, lihat `reserved-memory-percent` dan Mengelola memori cadangan untuk Valkey dan Redis OSS.
`set-max-intset-entries`	Default: 512 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Menentukan jumlah memori yang digunakan untuk jenis tertentu dari set (string yang berupa integer dalam radix 10 pada rentang integer bertanda 64 bit). Set seperti itu dengan jumlah entri kurang dari yang ditentukan akan disimpan menggunakan pengodean khusus yang menghemat ruang.
`slave-allow-chaining`	Default: no Jenis: string Dapat diubah: Tidak	Menentukan apakah replika baca di Redis OSS dapat membaca replika sendiri.

Nama	Detail	Deskripsi
`slowlog-log-slower-than`	Default: 10000 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Waktu eksekusi maksimum, dalam mikrodetik, untuk perintah yang akan dicatat oleh fitur Redis OSS Slow Log.
`slowlog-max-len`	Default: 128 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Panjang maksimum Redis OSS Slow Log.
`tcp-keepalive`	Default: 0 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Jika parameter ini diatur ke nilai bukan nol (N), simpul klien akan di-polling setiap N detik untuk memastikan bahwa simpul ini masih terhubung. Dengan pengaturan default 0, tidak ada polling yang terjadi. **Important** Beberapa aspek dari parameter ini berubah dalam Redis OSS versi 3.2.4. Lihat Parameter berubah di Redis OSS 3.2.4 (ditingkatkan).

Nama	Detail	Deskripsi
`timeout`	Default: 0 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Jumlah detik waktu tunggu simpul sebelum waktu habis Nilainya adalah: • `0` – tidak pernah memutus koneksi klien idle. • `1-19` – nilai tidak valid. • `>=20` – jumlah detik waktu tunggu simpul sebelum memutus koneksi klien yang idle.
`zset-max-ziplist-entries`	Default: 128 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Menentukan jumlah memori yang digunakan untuk sorted set. Sorted set dengan jumlah elemen kurang dari yang ditentukan akan disimpan menggunakan pengodean khusus yang menghemat ruang.
`zset-max-ziplist-value`	Default: 64 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Segera	Menentukan jumlah memori yang digunakan untuk sorted set. Sorted set dengan entri yang lebih kecil dari jumlah byte yang ditentukan disimpan menggunakan pengodean khusus yang menghemat ruang.

Note

Jika Anda tidak menentukan grup parameter untuk cluster Redis OSS 2.6.13 Anda, maka grup parameter default (`default.redis2.6`) akan digunakan. Anda tidak dapat mengubah nilai parameter dalam grup parameter default; namun, Anda selalu dapat membuat grup parameter kustom dan menetapkannya ke klaster Anda setiap saat.

Parameter spesifik tipe node Redis OSS

Meskipun sebagian besar parameter memiliki nilai tunggal, beberapa parameter memiliki nilai yang berbeda-beda bergantung pada jenis simpul yang digunakan. Tabel berikut menunjukkan nilai default untuk parameter `maxmemory`, `client-output-buffer-limit-slave-hard-limit`, dan `client-output-buffer-limit-slave-soft-limit` untuk setiap jenis simpul. Nilai `maxmemory` adalah jumlah maksimum byte yang tersedia untuk Anda gunakan, untuk data, dan untuk penggunaan lainnya, pada simpul. Untuk informasi selengkapnya, lihat Memori yang tersedia.

> **Note**
>
> Parameter `maxmemory` tidak dapat diubah.

Jenis simpul	Maxmemory	C lient-output-buffe r-limit - slave-hard-limit	C lient-output-buffe r-limit - slave-soft-limit
cache.t1.micro	142606336	14260633	14260633
cache.t2.micro	581959680	58195968	58195968
cache.t2.small	1665138688	166513868	166513868
cache.t2.medium	3461349376	346134937	346134937
cache.t3.micro	536870912	53687091	53687091
cache.t3.small	1471026299	147102629	147102629
cache.t3.medium	3317862236	331786223	331786223
cache.t4g.micro	536870912	53687091	53687091
cache.t4g.small	1471026299	147102629	147102629
cache.t4g.medium	3317862236	331786223	331786223
cache.m1.small	943718400	94371840	94371840

Jenis simpul	Maxmemory	C lient-output-buffe r-limit - slave-hard-limit	C lient-output-buffe r-limit - slave-soft-limit
cache.m1.medium	3093299200	309329920	309329920
cache.m1.large	7025459200	702545920	702545920
cache.m1.xlarge	14889779200	1488977920	1488977920
cache.m2.xlarge	17091788800	1709178880	1709178880
cache.m2.2xlarge	35022438400	3502243840	3502243840
cache.m2.4xlarge	70883737600	7088373760	7088373760
cache.m3.medium	2988441600	309329920	309329920
cache.m3.large	6501171200	650117120	650117120
cache.m3.xlarge	14260633600	1426063360	1426063360
cache.m3.2xlarge	29989273600	2998927360	2998927360
cache.m4.large	6892593152	689259315	689259315
cache.m4.xlarge	15328501760	1532850176	1532850176
cache.m4.2xlarge	31889126359	3188912636	3188912636
cache.m4.4xlarge	65257290629	6525729063	6525729063
cache.m4.10xlarge	166047614239	16604761424	16604761424
cache.m5.large	6854542746	685454275	685454275
cache.m5.xlarge	13891921715	1389192172	1389192172
cache.m5.2xlarge	27966669210	2796666921	2796666921
cache.m5.4xlarge	56116178125	5611617812	5611617812

Jenis simpul	Maxmemory	C lient-output-buffe r-limit - slave-hard-limit	C lient-output-buffe r-limit - slave-soft-limit
cache.m5.12xlarge	168715971994	16871597199	16871597199
cache.m5.24xlarge	337500562842	33750056284	33750056284
cache.m6g.large	6854542746	685454275	685454275
cache.m6g.xlarge	13891921715	1389192172	1389192172
cache.m6g.2xlarge	27966669210	2796666921	2796666921
cache.m6g.4xlarge	56116178125	5611617812	5611617812
cache.m6g.8xlarge	111325552312	11132555231	11132555231
cache.m6g.12xlarge	168715971994	16871597199	16871597199
cache.m6g.16xlarge	225000375228	22500037523	22500037523
cache.c1.xlarge	6501171200	650117120	650117120
cache.r3.large	14470348800	1468006400	1468006400
cache.r3.xlarge	30513561600	3040870400	3040870400
cache.r3.2xlarge	62495129600	6081740800	6081740800
cache.r3.4xlarge	126458265600	12268339200	12268339200
cache.r3.8xlarge	254384537600	24536678400	24536678400
cache.r4.large	13201781556	1320178155	1320178155
cache.r4.xlarge	26898228839	2689822883	2689822883
cache.r4.2xlarge	54197537997	5419753799	5419753799
cache.r4.4xlarge	108858546586	10885854658	10885854658

Jenis simpul	Maxmemory	C lient-output-buffe r-limit - slave-hard-limit	C lient-output-buffe r-limit - slave-soft-limit
cache.r4.8xlarge	218255432090	21825543209	21825543209
cache.r4.16xlarge	437021573120	43702157312	43702157312
cache.r5.large	14037181030	1403718103	1403718103
cache.r5.xlarge	28261849702	2826184970	2826184970
cache.r5.2xlarge	56711183565	5671118356	5671118356
cache.r5.4xlarge	113609865216	11360986522	11360986522
cache.r5.12xlarge	341206346547	34120634655	34120634655
cache.r5.24xlarge	682485973811	68248597381	68248597381
cache.r6g.large	14037181030	1403718103	1403718103
cache.r6g.xlarge	28261849702	2826184970	2826184970
cache.r6g.2xlarge	56711183565	5671118356	5671118356
cache.r6g.4xlarge	113609865216	11360986522	11360986522
cache.r6g.8xlarge	225000375228	22500037523	22500037523
cache.r6g.12xlarge	341206346547	34120634655	34120634655
cache.r6g.16xlarge	450000750456	45000075046	45000075046
cache.r6gd.xlarge	28261849702	2826184970	2826184970
cache.r6gd.2xlarge	56711183565	5671118356	5671118356
cache.r6gd.4xlarge	113609865216	11360986522	11360986522
cache.r6gd.8xlarge	225000375228	22500037523	22500037523

Jenis simpul	Maxmemory	C lient-output-buffe r-limit - slave-hard-limit	C lient-output-buffe r-limit - slave-soft-limit
cache.r6gd.12xlarge	341206346547	34120634655	34120634655
cache.r6gd.16xlarge	450000750456	45000075046	45000075046
cache.r7g.large	14037181030	1403718103	1403718103
cache.r7g.xlarge	28261849702	2826184970	2826184970
cache.r7g.2xlarge	56711183565	5671118356	5671118356
cache.r7g.4xlarge	113609865216	11360986522	11360986522
cache.r7g.8xlarge	225000375228	22500037523	22500037523
cache.r7g.12xlarge	341206346547	34120634655	34120634655
cache.r7g.16xlarge	450000750456	45000075046	45000075046
cache.m7g.large	6854542746	685454275	685454275
cache.m7g.xlarge	13891921715	1389192172	1389192172
cache.m7g.2xlarge	27966669210	2796666921	2796666921
cache.m7g.4xlarge	56116178125	5611617812	5611617812
cache.m7g.8xlarge	111325552312	11132555231	11132555231
cache.m7g.12xlarge	168715971994	16871597199	16871597199
cache.m7g.16xlarge	225000375228	22500037523	22500037523
cache.c7gn.large	3317862236	1403718103	1403718103
cache.c7gn.xlarge	6854542746	2826184970	2826184970
cache.c7gn.2xlarge	13891921715	5671118356	5671118356

Jenis simpul	Maxmemory	C lient-output-buffe r-limit - slave-hard-limit	C lient-output-buffe r-limit - slave-soft-limit
cache.c7gn.4xlarge	27966669210	11360986522	11360986522
cache.c7gn.8xlarge	56116178125	22500037523	22500037523
cache.c7gn.12xlarge	84357985997	34120634655	34120634655
cache.c7gn.16xlarge	113609865216	45000075046	45000075046

> **Note**
>
> Semua jenis instans generasi saat ini dibuat di Amazon Virtual Private Cloud (VPC) secara default.
> Instans T1 tidak mendukung Multi-AZ.
> Instans T1 dan T2 tidak mendukung Redis OSS AOF.
> Redis OSS variabel konfigurasi `appendonly` dan tidak `appendfsync` didukung pada Redis OSS versi 2.8.22 dan yang lebih baru.

Parameter spesifik Memcached

Memcache

Jika Anda tidak menentukan grup parameter untuk klaster Memcached, maka grup parameter default yang sesuai dengan versi mesin Anda akan digunakan. Anda tidak dapat mengubah nilai parameter dalam grup parameter default. Namun, Anda dapat membuat grup parameter kustom dan menetapkannya ke klaster Anda kapan saja. Untuk informasi selengkapnya, lihat Membuat grup ElastiCache parameter.

Topik

- Perubahan Memcached 1.6.17
- Parameter yang ditambahkan di Memcached 1.6.6
- Perubahan parameter Memcached 1.5.10
- Parameter yang ditambahkan di Memcached 1.4.34

- Parameter yang ditambahkan di Memcached 1.4.33
- Parameter yang ditambahkan di Memcached 1.4.24
- Parameter yang ditambahkan di Memcached 1.4.14
- Parameter yang didukung Memcached 1.4.5
- Overhead koneksi Memcached
- Parameter khusus jenis simpul Memcached

Perubahan Memcached 1.6.17

Mulai Memcached 1.6.17, kami tidak lagi mendukung perintah administratif ini: `lru_crawler`, `lru`, dan `slabs`. Dengan perubahan ini, Anda tidak akan dapat enable/disable `lru_crawler` saat runtime melalui perintah. Silakan enable/disable `lru_crawler` dengan memodifikasi grup parameter kustom Anda.

Parameter yang ditambahkan di Memcached 1.6.6

Untuk Memcached 1.6.6, tidak ada parameter tambahan yang didukung.

Keluarga grup parameter: memcached1.6

Perubahan parameter Memcached 1.5.10

Untuk Memcached 1.5.10, parameter tambahan berikut didukung.

Keluarga grup parameter: memcached1.5

Nama	Detail	Deskripsi
`no_modern`	Default: 1 Jenis: boolean Dapat diubah: Ya Nilai yang Diizinkan: 0,1 Perubahan Berlaku: Saat peluncuran	Alias untuk menonakti fkan`slab_reassign` ,, `lru_maintainer_thr ead lru_segmented` , dan perintah. `maxconns_fast` Saat menggunakan Memcached 1.5 dan yang lebih tinggi, atur

Nama	Detail	Deskripsi
		no_modern juga hash_algorithm ke. jenkins Selain itu, saat menggunakan Memcached 1.5.10, dikendalikan inline_ascii_reponse oleh parameter. parallelly Ini berarti bahwa jika no_modern dinonaktifkan maka inline_as cii_reponse dinonaktifkan. Dari mesin Memcached 1.5.16 dan seterusnya inline_as cii_response parameter tidak lagi berlaku, jadi no_modern dinonaktifkan atau dinonakti fkan tidak berpengaruh pada. inline_ascii_reponse Jika no_modern dinonakti fkan, maka slab_reas sign lru_maintainer_thr ead ,lru_segmented ,, dan maxconns_fast AKAN diaktifka n. Karena slab_automove dan hash_algorithm parameter bukan parameter SWITCH, pengaturannya didasarkan pada konfigurasi dalam grup parameter. Jika Anda ingin menonakti fkan no_modern dan kembali kemodern, Anda harus mengonfig urasi grup parameter khusus untuk menonaktifkan parameter ini dan

Nama	Detail	Deskripsi
		kemudian reboot agar perubahan ini diterapkan. **Note** Nilai konfigurasi default untuk parameter ini telah diubah dari 0 ke 1 mulai 20 Agustus 2021. Nilai default yang diperbarui akan diambil secara otomatis oleh ElastiCache pengguna baru untuk setiap wilayah setelah 20 Agustus 2021. ElastiCache Pengguna yang ada di wilayah sebelum 20 Agustus 2021 perlu memodifikasi grup parameter khusus mereka secara manual untuk mengambil perubahan baru ini.
`inline_ascii_resp`	Default: 0 Jenis: boolean Dapat diubah: Ya Nilai yang Diizinkan: 0,1 Perubahan Berlaku: Saat peluncuran	Menyimpan angka dari respons `VALUE`, dalam item, menggunakan hingga 24 byte. Perlambatan kecil untuk `get` ASCII, `faster` ditetapkan.

Untuk Memcached 1.5.10, parameter berikut dihapus.

Nama	Detail	Deskripsi
`expirezero_does_not_evict`	Default: 0 Jenis: boolean Dapat diubah: Ya Nilai yang Diizinkan: 0,1 Perubahan Berlaku: Saat peluncuran	Tidak lagi didukung di versi ini.
`modern`	Default: 1 Jenis: boolean Dapat Diubah: Ya (memerlukan peluncuran ulang jika diatur ke `no_modern`) Nilai yang Diizinkan: 0,1 Perubahan Berlaku: Saat peluncuran	Tidak lagi didukung di versi ini. Dimulai dari versi ini, `no-modern` diaktifkan secara default pada setiap peluncuran atau peluncuran ulang.

Parameter yang ditambahkan di Memcached 1.4.34

Untuk Memcached 1.4.34, tidak ada parameter tambahan yang didukung.

Keluarga grup parameter: memcached1.4

Parameter yang ditambahkan di Memcached 1.4.33

Untuk Memcached 1.4.33, parameter tambahan berikut didukung.

Keluarga grup parameter: memcached1.4

Nama	Detail	Deskripsi
`modern`	Default: diaktifkan Jenis: boolean Dapat diubah: Ya Perubahan Berlaku: Saat peluncuran	Alias untuk beberapa fitur. Pengaktifan `modern` setara dengan mengaktifkan perintah berikut dan menggunakan algoritma hash murmur3: `slab_reassign` , `slab_automove` , `lru_crawler` , `lru_maintainer` , `maxconns_fast` , dan `hash_algorithm=murmur3` .
`watch`	Default: diaktifkan Jenis: boolean Dapat diubah: Ya Perubahan Berlaku: Segera Log dapat dihapus jika pengguna mencapai batas `watcher_logbuf_size` dan `worker_logbuf_size` mereka.	Pengambilan, pengosongan, dan mutasi log. Ketika, misalnya, pengguna mengaktifkan `watch`, mereka dapat melihat log saat `get`, `set`, `delete`, atau `update` terjadi.
`idle_timeout`	Default: 0 (dinonaktifkan) Jenis: integer	Jumlah minimum detik saat klien akan diizinkan untuk idle sebelum diminta agar ditutup. Rentang nilai: 0 hingga 86400.

Nama	Detail	Deskripsi
	Dapat diubah: Ya Perubahan Berlaku: Saat Peluncuran	
`track_sizes`	Default: dinonaktifkan Jenis: boolean Dapat diubah: Ya Perubahan Berlaku: Saat Peluncuran	Menunjukkan ukuran setiap grup slab yang telah dikonsumsi. Pengaktifan `track_sizes` memungkinkan Anda menjalankan `stats sizes` tanpa perlu menjalankan `stats sizes_enable` .
`watcher_logbuf_size`	Default: 256 (KB) Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Saat Peluncuran	Perintah `watch` mengaktifkan pencatatan log stream untuk Memcached. Namun `watch` dapat menghapus log jika tingkat pengosongan, mutasi, atau pengambilan cukup tinggi untuk menyebabkan buffer pencatatan log menjadi penuh. Dalam situasi tersebut, pengguna dapat meningkatkan ukuran buffer untuk mengurangi kemungkinan kehilangan log.

Nama	Detail	Deskripsi
`worker_logbuf_size`	Default: 64 (KB) Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Saat Peluncuran	Perintah `watch` mengaktifkan pencatatan log stream untuk Memcached. Namun `watch` dapat menghapus log jika tingkat pengosongan, mutasi, atau pengambilan cukup tinggi untuk menyebabkan buffer pencatata n log menjadi penuh. Dalam situasi tersebut, pengguna dapat meningkatkan ukuran buffer untuk mengurangi kemungkinan kehilangan log.
`slab_chunk_max`	Default: 524288 (byte) Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Saat Peluncuran	Menentukan ukuran maksimum slab. Mengatur ukuran slab lebih kecil membuat penggunaan memori lebih efisien. Item yang lebih besar dari `slab_chun k_max` akan dibagi menjadi beberapa slab.
`lru_crawler metadump [all|1|2|3]`	Default: dinonaktifkan Jenis: boolean Dapat diubah: Ya Perubahan Berlaku: Segera	jika lru_crawler diaktifkan, perintah ini menghapus semua kunci. `all|1|2|3` - semua slab, atau tentukan nomor slab tertentu

Parameter yang ditambahkan di Memcached 1.4.24

Untuk Memcached 1.4.24, parameter tambahan berikut didukung.

Keluarga grup parameter: memcached1.4

Nama	Detail	Deskripsi
`disable_flush_all`	Default: 0 (dinonaktifkan) Jenis: boolean Dapat diubah: Ya Perubahan Berlaku: Saat peluncuran	Tambahkan parameter (`-F`) untuk menonaktifkan flush_all . Berguna jika Anda tidak ingin dapat menjalankan flush penuh pada instans produksi. Nilai: 0, 1 (pengguna dapat melakukan `flush_all` jika nilai adalah 0).
`hash_algorithm`	Default: jenkins Jenis: string Dapat diubah: Ya Perubahan Berlaku: Saat peluncuran	Algoritma hash yang akan digunakan. Nilai yang diizinkan: murmur3 dan jenkins.
`lru_crawler`	Default: 0 (dinonaktifkan) Jenis: boolean Dapat diubah: Ya Perubahan Berlaku: Setelah mulai ulang **Note** Anda dapat mengaktifkan `lru_crawler` untuk sementara pada saat	Membersihkan kelas slab item yang telah kedaluwarsa. Ini adalah proses berdampak rendah yang berjalan di latar belakang. Saat ini memerlukan inisiasi perayapan menggunakan perintah manual. Untuk mengaktifkan sementara, jalankan `lru_crawler enable` di baris perintah. `lru_crawler 1,3,5` merayapi kelas slab 1, 3, 5 dengan mencari item yang kedaluwarsa untuk ditambahkan ke daftar bebas. Nilai: 0,1

Nama	Detail	Deskripsi
	runtime dari baris perintah. Untuk informasi selengkapnya, lihat kolom Deskripsi.	**Note** Mengaktifkan `lru_crawler` pada baris perintah akan mengaktifkan perayap hingga dinonaktifkan pada baris perintah atau boot ulang berikutnya. Untuk mengaktifkan secara permanen, Anda harus mengubah nilai parameter. Untuk informasi selengkapnya, lihat Memodifikasi grup ElastiCache parameter.
`lru_maintainer`	Default: 0 (dinonaktifkan) Jenis: boolean Dapat diubah: Ya Perubahan Berlaku: Saat peluncuran	Benang latar belakang yang mengocokkan item di antara kapasitas LRUs as tercapai. Nilai: 0, 1.

Nama	Detail	Deskripsi
`expirezero_does_not_evict`	Default: 0 (dinonaktifkan) Jenis: boolean Dapat diubah: Ya Perubahan Berlaku: Saat peluncuran	Ketika digunakan dengan `lru_maintainer` , menjadikan item yang memiliki waktu kedaluwarsa 0 tidak dapat dikosongkan. **Warning** Hal ini dapat memenuhi memori yang tersedia untuk item lainnya yang dapat dikosongkan. Dapat diatur untuk mengabaikan `lru_maintainer` .

Parameter yang ditambahkan di Memcached 1.4.14

Untuk Memcached 1.4.14, parameter tambahan berikut didukung.

Keluarga grup parameter: memcached1.4

Parameter yang ditambahkan dalam Memcached 1.4.14

Nama	Deskripsi
`config_max`	Jumlah maksimum entri ElastiCache konfigurasi.

Nama	Deskripsi
`config_size_max`	Ukuran maksimum entri konfigurasi, dalam byte.
`hashpower_init`	Ukuran awal tabel ElastiCache hash, dinyatakan sebagai kekuatan dua. Default-nya adalah 16 (2^16), atau 65536 kunci.

Nama	Deskripsi
`maxconns_fast`	Mengubah cara permintaan koneksi baru ditangani ketika batas koneksi maksimum tercapai. Jika parameter ini diatur ke 0 (nol), koneksi baru ditambahkan ke antrean backlog dan akan menunggu sampai koneksi lain ditutup. Jika parameter diatur ke 1, ElastiCache mengirimkan kesalahan ke klien dan segera menutup koneksi.
`slab_automove`	Menyesuaikan algoritma automove slab: Jika parameter ini diatur ke 0 (nol), algoritma automove dinonaktifkan. Jika diatur ke 1, ElastiCache membutuhkan pendekatan konservatif yang lambat untuk memindahkan slab secara otomatis. Jika diatur ke 2, gerakkan lempengan ElastiCache secara agresif setiap kali ada penggusuran. (Mode ini tidak direkomen dasikan kecuali untuk tujuan pengujian.)

Nama	Deskripsi
`slab_reassign`	Mengaktifkan atau menonaktifkan penetapan ulang slab. Jika parameter ini diatur ke 1, Anda dapat menggunakan perintah "slab reassign" untuk secara manual menetapkan ulang memori.

Parameter yang didukung Memcached 1.4.5

Keluarga grup parameter: memcached1.4

Untuk Memcached 1.4.5, parameter tambahan berikut didukung.

Parameter yang ditambahkan di Memcached 1.4.5

Nama	Detail	Deskripsi
`backlog_queue_limit`	Default: 1024 Jenis: integer Dapat diubah: Tidak	Batas antrean backlog.
`binding_protocol`	Default: otomatis Jenis: string	Protokol pengikatan. Nilai yang diizinkan adalah `ascii` dan `auto`.

Nama	Detail	Deskripsi
	Dapat diubah: Ya Perubahan Berlaku: Setelah pengaktifan ulang	Untuk panduan dalam mengubah nilai `binding_protocol` , lihat Memodifikasi grup ElastiCache parameter.
`cas_disabled`	Default: 0 (salah) Jenis: Boolean Dapat diubah: Ya Perubahan Berlaku: Setelah pengaktifan ulang	Jika 1 (benar), operasi periksa dan atur (CAS) akan dinonaktifkan, dan item yang disimpan akan menggunakan 8 byte lebih sedikit dibandingkan dengan CAS diaktifkan.
`chunk_size`	Default: 48 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Setelah pengaktifan ulang	Jumlah minimum, dalam byte, ruang untuk mengalokasikan kunci, nilai, dan bendera item terkecil.
`chunk_size_growth_factor`	Default: 1,25 Jenis: float Dapat diubah: Ya Perubahan Berlaku: Setelah pengaktifan ulang	Faktor pertumbuhan yang mengontrol ukuran setiap potongan Memcached berturut-turut; setiap potongan akan `chunk_size_growth_factor` kali lebih besar dari potongan sebelumnya.
`error_on_memory_exhausted`	Default: 0 (salah) Jenis: Boolean Dapat diubah: Ya Perubahan Berlaku: Setelah pengaktifan ulang	Jika 1 (benar), ketika tidak ada lagi memori untuk menyimpan item, Memcached akan menampilkan kesalahan dan bukan mengosongkan item.

Nama	Detail	Deskripsi
large_memory_pages	Default: 0 (salah) Jenis: Boolean Dapat diubah: Tidak	Jika 1 (true), ElastiCache akan mencoba untuk menggunakan halaman memori besar.
lock_down_paged_memory	Default: 0 (salah) Jenis: Boolean Dapat diubah: Tidak	Jika 1 (true), ElastiCache akan mengunci semua memori halaman.
max_item_size	Default: 1048576 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Setelah pengaktifan ulang	Ukuran, dalam byte, item terbesar yang dapat disimpan dalam klaster.
max_simultaneous_connections	Default: 65000 Jenis: integer Dapat diubah: Tidak	Jumlah maksimum koneksi bersamaan.
maximize_core_file_limit	Default: 0 (salah) Jenis: Boolean Dapat diubah: Perubahan Berlaku: Setelah pengaktifan ulang	Jika 1 (true), ElastiCache akan memaksimalkan batas file inti.

Nama	Detail	Deskripsi
`memcached_connections_overhead`	Default: 100 Jenis: integer Dapat diubah: Ya Perubahan Berlaku: Setelah pengaktifan ulang	Jumlah memori yang akan dicadangkan untuk koneksi Memcached dan berbagai overhead lainnya. Untuk informasi tentang parameter ini, lihat Overhead koneksi Memcached.
`requests_per_event`	Default: 20 Jenis: integer Dapat diubah: Tidak	Jumlah maksimum permintaan per peristiwa untuk koneksi tertentu. Batas ini diperlukan untuk mencegah kekurangan sumber daya.

Overhead koneksi Memcached

Pada setiap simpul, memori yang tersedia untuk menyimpan item adalah total memori yang tersedia pada simpul tersebut (yang disimpan dalam parameter `max_cache_memory`) dikurangi memori yang digunakan untuk koneksi dan overhead lainnya (yang disimpan dalam parameter `memcached_connections_overhead`). Misalnya, sebuah simpul jenis `cache.m1.small` memiliki `max_cache_memory` sebesar 1.300 MB. Dengan nilai `memcached_connections_overhead` default 100 MB, proses Memcached akan memiliki 1.200 MB yang tersedia untuk menyimpan item.

Nilai default untuk parameter `memcached_connections_overhead` memenuhi sebagian besar kasus penggunaan; namun, jumlah alokasi yang diperlukan untuk overhead koneksi dapat bervariasi bergantung pada beberapa faktor, termasuk tingkat permintaan, ukuran muatan, dan jumlah koneksi.

Anda dapat mengubah nilai `memcached_connections_overhead` agar lebih sesuai dengan kebutuhan aplikasi Anda. Misalnya, peningkatan nilai parameter `memcached_connections_overhead` akan mengurangi jumlah memori yang tersedia untuk menyimpan item dan memberikan buffer yang lebih besar untuk overhead koneksi. Pengurangan nilai parameter `memcached_connections_overhead` akan memberi Anda lebih banyak memori untuk menyimpan item, tetapi dapat meningkatkan risiko penggunaan swap dan penurunan performa. Jika Anda melihat penggunaan swap dan penurunan performa, coba tingkatkan nilai parameter `memcached_connections_overhead`.

> **Important**
>
> Untuk jenis simpul `cache.t1.micro`, nilai untuk `memcached_connections_overhead` ditentukan sebagai berikut:
>
> - Jika cluster Anda menggunakan grup parameter default, ElastiCache akan menetapkan nilai `memcached_connections_overhead` untuk 13MB.
> - Jika klaster menggunakan grup parameter yang telah Anda buat sendiri, nilai `memcached_connections_overhead` dapat diatur ke nilai pilihan Anda.

Parameter khusus jenis simpul Memcached

Meskipun sebagian besar parameter memiliki nilai tunggal, beberapa parameter memiliki nilai yang berbeda-beda bergantung pada jenis simpul yang digunakan. Tabel berikut menunjukkan nilai default untuk parameter `max_cache_memory` dan `num_threads` untuk tiap jenis simpul. Nilai pada parameter ini tidak dapat diubah.

Jenis simpul	max_cache_memory (dalam megabyte)	num_threads
cache.t1.micro	213	1
cache.t2.micro	555	1
cache.t2.small	1588	1
cache.t2.medium	3301	2
cache.t3.micro	512	2
cache.t3.small	1402	2
cache.t3.medium	3364	2
cache.t4g.micro	512	2
cache.t4g.small	1402	2
cache.t4g.medium	3164	2

Jenis simpul	max_cache_memory (dalam megabyte)	num_threads
cache.m1.small	1301	1
cache.m1.medium	3350	1
cache.m1.large	7100	2
cache.m1.xlarge	14600	4
cache.m2.xlarge	33800	2
cache.m2.2xlarge	30412	4
cache.m2.4xlarge	68000	16
cache.m3.medium	2850	1
cache.m3.large	6200	2
cache.m3.xlarge	13600	4
cache.m3.2xlarge	28600	8
cache.m4.large	6573	2
cache.m4.xlarge	11496	4
cache.m4.2xlarge	30412	8
cache.m4.4xlarge	62234	16
cache.m4.10xlarge	158355	40
cache.m5.large	6537	2
cache.m5.xlarge	13248	4
cache.m5.2xlarge	26671	8
cache.m5.4xlarge	53516	16

Jenis simpul	max_cache_memory (dalam megabyte)	num_threads
cache.m5.12xlarge	160900	48
cache.m5.24xlarge	321865	96
cache.m6g.large	6537	2
cache.m6g.xlarge	13248	4
cache.m6g.2xlarge	26671	8
cache.m6g.4xlarge	53516	16
cache.m6g.8xlarge	107000	32
cache.m6g.12xlarge	160900	48
cache.m6g.16xlarge	214577	64
cache.c1.xlarge	6600	8
cache.r3.large	13800	2
cache.r3.xlarge	29100	4
cache.r3.2xlarge	59600	8
cache.r3.4xlarge	120600	16
cache.r3.8xlarge	120600	32
cache.r4.large	12590	2
cache.r4.xlarge	25652	4
cache.r4.2xlarge	51686	8
cache.r4.4xlarge	103815	16
cache.r4.8xlarge	208144	32

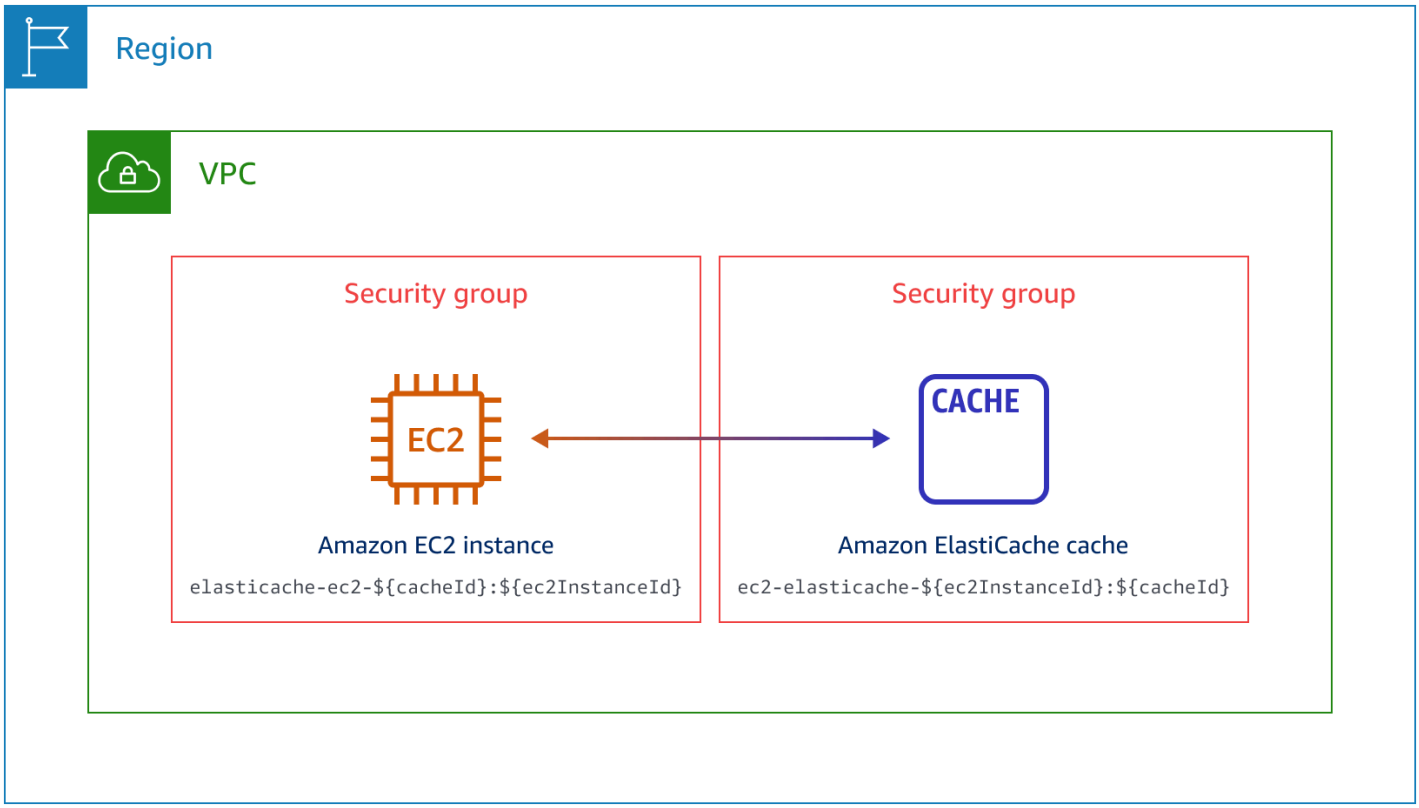
Jenis simpul	max_cache_memory (dalam megabyte)	num_threads
cache.r4.16xlarge	416776	64
cache.r5.large	13387	2
cache.r5.xlarge	26953	4
cache.r5.2xlarge	54084	8
cache.r5.4xlarge	108347	16
cache.r5.12xlarge	325400	48
cache.r5.24xlarge	650869	96
cache.r6g.large	13387	2
cache.r6g.xlarge	26953	4
cache.r6g.2xlarge	54084	8
cache.r6g.4xlarge	108347	16
cache.r6g.8xlarge	214577	32
cache.r6g.12xlarge	325400	48
cache.r6g.16xlarge	429154	64
cache.c7gn.large	3164	2
cache.c7gn.xlarge	6537	4
cache.c7gn.2xlarge	13248	8
cache.c7gn.4xlarge	26671	16
cache.c7gn.8xlarge	53516	32
cache.c7gn.12xlarge	325400	48

Jenis simpul	max_cache_memory (dalam megabyte)	num_threads
cache.c7gn.16xlarge	108347	64

> **Note**
>
> Semua instans T2 dibuat di Amazon Virtual Private Cloud (Amazon VPC).

Menghubungkan EC2 instance dan ElastiCache cache secara otomatis

Anda dapat menggunakan ElastiCache konsol untuk menyederhanakan pengaturan koneksi antara instans Amazon Elastic Compute Cloud EC2 (Amazon) dan cache. ElastiCache Seringkali, cache Anda berada di subnet pribadi dan EC2 instance Anda berada di subnet publik dalam VPC. Anda dapat menggunakan klien SQL pada EC2 instance Anda untuk terhubung ke ElastiCache cache Anda. EC2 Instance ini juga dapat menjalankan server web atau aplikasi yang mengakses ElastiCache cache pribadi Anda.



Topik

- Konektivitas otomatis dengan sebuah EC2 instance
- Melihat sumber daya komputasi terhubung

Konektivitas otomatis dengan sebuah EC2 instance

Ketika Anda mengatur koneksi antara EC2 instance dan ElastiCache cache, ElastiCache secara otomatis mengkonfigurasi grup keamanan VPC untuk instance EC2 Anda dan untuk ElastiCache cache Anda.

Berikut ini adalah persyaratan untuk menghubungkan EC2 instance dengan ElastiCache cache:

- EC2 Instance harus ada di VPC yang sama dengan cache. ElastiCache

  Jika tidak ada EC2 instance di VPC yang sama, maka konsol menyediakan tautan untuk membuatnya.
- Pengguna yang mengatur konektivitas harus memiliki izin untuk melakukan EC2 operasi Amazon berikut. Permisiosn ini umumnya ditambahkan ke EC2 akun saat dibuat. Untuk informasi

selengkapnya tentang EC2 izin, lihat Memberikan izin yang diperlukan untuk sumber daya Amazon. EC2

- `ec2:AuthorizeSecurityGroupEgress`
- `ec2:AuthorizeSecurityGroupIngress`
- `ec2:CreateSecurityGroup`
- `ec2:DescribeInstances`
- `ec2:DescribeNetworkInterfaces`
- `ec2:DescribeSecurityGroups`
- `ec2:ModifyNetworkInterfaceAttribute`
- `ec2:RevokeSecurityGroupEgress`

Saat Anda mengatur sambungan ke EC2 instance, ElastiCache bertindak sesuai dengan konfigurasi grup keamanan saat ini yang terkait dengan ElastiCache cache dan EC2 instance, seperti yang dijelaskan dalam tabel berikut.

Konfigurasi grup ElastiCache keamanan saat ini	Konfigurasi grup EC2 keamanan saat ini	ElastiCache aksi
Ada satu atau lebih grup keamanan yang terkait dengan ElastiCache cache dengan nama yang cocok dengan pola`elasticache-ec2-${cacheId}:${ec2InstanceId}` . Grup keamanan yang cocok dengan pola belum diubah. Grup keamanan ini hanya memiliki satu aturan masuk dengan grup keamanan VPC dari instance sebagai EC2 sumbernya.	Ada satu atau lebih grup keamanan yang terkait dengan EC2 instance dengan nama yang cocok dengan pola`elasticache-ec2-${cacheId}:${ec2InstanceId}` . Grup keamanan yang cocok dengan pola belum diubah. Grup keamanan ini hanya memiliki satu aturan keluar dengan grup keamanan VPC cache sebagai ElastiCache sumbernya.	ElastiCache tidak mengambil tindakan. Koneksi sudah dikonfigurasi secara otomatis antara EC2 instance dan ElastiCache cache. Karena koneksi sudah ada antara EC2 instance dan ElastiCache cache, grup keamanan tidak dimodifikasi.
Salah satu syarat berikut berlaku:	Salah satu syarat berikut dipenuhi:	ELC action: create new security groups

Konfigurasi grup ElastiCache keamanan saat ini	Konfigurasi grup EC2 keamanan saat ini	ElastiCache aksi
• Tidak ada grup keamanan yang terkait dengan ElastiCache cache dengan nama yang cocok dengan pola`elasticache-ec2-${cacheId}:${ec2InstanceId}` . • Ada satu atau lebih grup keamanan yang terkait dengan ElastiCache cache dengan nama yang cocok dengan pola`elasticache-ec2-${cacheId}:${ec2InstanceId}` . Namun, tidak ElastiCache dapat menggunakan salah satu grup keamanan ini untuk koneksi dengan EC2 instance. ElastiCache tidak dapat menggunakan grup keamanan yang tidak memiliki satu aturan masuk dengan grup keamanan VPC dari instance sebagai EC2 sumbernya. ElastiCache juga tidak dapat menggunakan grup keamanan yang telah dimodifikasi. Contoh-contoh perubahan meliputi penambahan aturan atau pengubahan port aturan yang ada.	• Tidak ada grup keamanan yang terkait dengan EC2 instance dengan nama yang cocok dengan pola`ec2-elasticache-${ec2InstanceId}:${cacheId}` . • Ada satu atau lebih grup keamanan yang terkait dengan EC2 instance dengan nama yang cocok dengan pola`ec2-elasticache-${ec2InstanceId}:${cacheId}` . Namun, tidak ElastiCache dapat menggunakan salah satu grup keamanan ini untuk koneksi dengan ElastiCache cache. ElastiCache tidak dapat menggunakan grup keamanan yang tidak memiliki satu aturan keluar dengan grup keamanan VPC cache sebagai ElastiCache sumbernya . ElastiCache juga tidak dapat menggunakan grup keamanan yang telah dimodifikasi.	

Konfigurasi grup ElastiCache keamanan saat ini	Konfigurasi grup EC2 keamanan saat ini	ElastiCache aksi
Ada satu atau lebih grup keamanan yang terkait dengan ElastiCache cache dengan nama yang cocok dengan pola`elasticache-ec2-${cacheId}:${ec2InstanceId}` . Grup keamanan yang cocok dengan pola belum diubah. Grup keamanan ini hanya memiliki satu aturan masuk dengan grup keamanan VPC dari instance sebagai EC2 sumbernya.	Ada satu atau lebih grup keamanan yang terkait dengan EC2 instance dengan nama yang cocok dengan pola`elasticache-ec2-${cacheId}:${ec2InstanceId}` . Namun, tidak ElastiCache dapat menggunakan salah satu grup keamanan ini untuk koneksi dengan ElastiCache cache. ElastiCache tidak dapat menggunakan grup keamanan yang tidak memiliki satu aturan keluar dengan grup keamanan VPC cache sebagai ElastiCache sumbernya. ElastiCache juga tidak dapat menggunakan grup keamanan yang telah dimodifikasi.	ELC action: create new security groups

Konfigurasi grup ElastiCache keamanan saat ini	Konfigurasi grup EC2 keamanan saat ini	ElastiCache aksi
Ada satu atau lebih grup keamanan yang terkait dengan ElastiCache cache dengan nama yang cocok dengan pola`elasticache-ec2-${cacheId}:${ec2InstanceId}` . Grup keamanan yang cocok dengan pola belum diubah. Grup keamanan ini hanya memiliki satu aturan masuk dengan grup keamanan VPC dari instance sebagai EC2 sumbernya.	Grup EC2 keamanan yang valid untuk koneksi ada, tetapi tidak terkait dengan EC2 instance. Grup keamanan ini memiliki nama yang cocok dengan pola `ec2-elasticache-${ec2InstanceId}:${cacheId}` . Grup tersebut belum diubah. Ini hanya memiliki satu aturan keluar dengan grup keamanan VPC cache sebagai theElasti Cache sumbernya.	ELC action: associate EC2 security group

Konfigurasi grup ElastiCache keamanan saat ini	Konfigurasi grup EC2 keamanan saat ini	ElastiCache aksi
Salah satu syarat berikut berlaku: • Tidak ada grup keamanan yang terkait dengan ElastiCache cache dengan nama yang cocok dengan pola`elasticache-ec2-${cacheId}:${ec2InstanceId}` . • Ada satu atau lebih grup keamanan yang terkait dengan ElastiCache cache dengan nama yang cocok dengan pola`elasticache-ec2-${cacheId}:${ec2InstanceId}` . Namun, tidak ElastiCache dapat menggunakan salah satu grup keamanan ini untuk koneksi dengan EC2 instance. ElastiCache tidak dapat menggunakan grup keamanan yang tidak memiliki satu aturan masuk dengan grup keamanan VPC dari instance sebagai EC2 sumbernya. ElastiCache juga tidak dapat menggunakan grup keamanan yang telah dimodifikasi.	Ada satu atau lebih grup keamanan yang terkait dengan EC2 instance dengan nama yang cocok dengan pola`ec2-elasticache-${ec2InstanceId}:${cacheId}` . Grup keamanan yang cocok dengan pola belum diubah. Grup keamanan ini hanya memiliki satu aturan keluar dengan grup keamanan VPC cache sebagai ElastiCache sumbernya.	ELC action: create new security groups

ElastiCachetindakan: buat grup keamanan baru

ElastiCache mengambil tindakan berikut:

- Membuat grup keamanan baru yang cocok dengan pola `elasticache-ec2-${cacheId}: ${ec2InstanceId}`. Grup keamanan ini memiliki aturan masuk dengan grup keamanan VPC dari instance sebagai EC2 sumbernya. Grup keamanan ini dikaitkan dengan ElastiCache cache dan memungkinkan EC2 instance untuk mengaksesnya.
- Membuat grup keamanan baru yang cocok dengan pola `elasticache-ec2-${cacheId}: ${ec2InstanceId}`. Grup keamanan ini memiliki aturan keluar dengan grup keamanan VPC cache sebagai target. ElastiCache Grup keamanan ini dikaitkan dengan EC2 instance dan memungkinkan EC2 instance untuk mengirim lalu lintas ke ElastiCache cache.

ElastiCache tindakan: kelompok EC2 keamanan asosiasi

ElastiCachemengaitkan grup EC2 keamanan yang valid dan sudah ada dengan EC2 instance. Grup keamanan ini memungkinkan EC2 instance untuk mengirim lalu lintas ke ElastiCache cache.

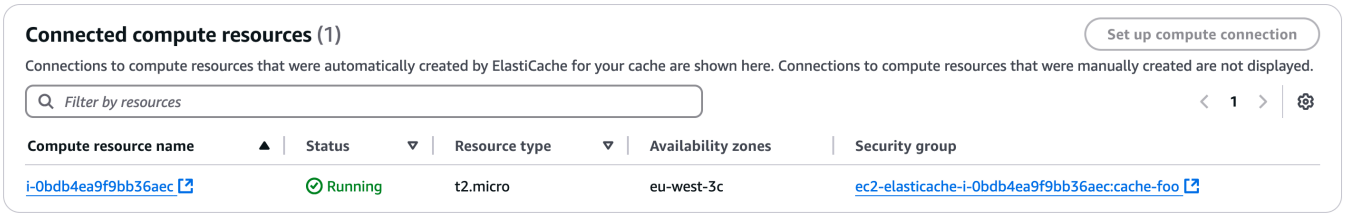
Melihat sumber daya komputasi terhubung

Anda dapat menggunakan AWS Management Console untuk melihat sumber daya komputasi yang terhubung ke ElastiCache cache. Sumber daya yang ditampilkan meliputi koneksi sumber daya komputasi yang disiapkan secara otomatis. Misalnya, Anda dapat mengizinkan sumber daya komputasi untuk mengakses cache secara manual dengan menambahkan aturan ke grup keamanan VPC yang terkait dengan cache. Sumber daya ini tidak akan muncul dalam daftar sumber daya komputasi yang terhubung.

Agar sumber daya komputasi dicantumkan, kondisi yang sama harus diterapkan seperti saat menghubungkan EC2 instance dan ElastiCache cache secara otomatis.

Untuk melihat sumber daya komputasi yang terhubung ke cache ElastiCache

1. Masuk ke AWS Management Console dan buka ElastiCache konsol
2. Di panel navigasi, pilih Cache, lalu pilih cache Valkey atau Redis OSS.
3. Pada tab Konektivitas & keamanan, lihat sumber daya komputasi di Koneksi Mengatur komputasi.



Penskalaan ElastiCache

Anda dapat menskalakan ElastiCache cache Anda sesuai dengan kebutuhan Anda. Cache tanpa server dan cluster yang dirancang sendiri menawarkan beberapa opsi penskalaan yang berbeda.

Penskalaan Tanpa Server ElastiCache

ElastiCache Tanpa server secara otomatis mengakomodasi lalu lintas beban kerja Anda saat naik atau turun. Untuk setiap cache ElastiCache Tanpa Server, ElastiCache terus melacak pemanfaatan sumber daya seperti CPU, memori, dan jaringan. Ketika salah satu sumber daya ini dibatasi, ElastiCache Serverless menskalakan dengan menambahkan pecahan baru dan mendistribusikan ulang data ke pecahan baru, tanpa downtime ke aplikasi Anda. Anda dapat memantau sumber daya yang dikonsumsi oleh cache Anda CloudWatch dengan memantau `BytesUsedForCache` metrik untuk penyimpanan data cache dan `ElastiCacheProcessingUnits` (ECPU) untuk penggunaan komputasi.

Menetapkan batas penskalaan untuk mengelola biaya

Anda dapat memilih untuk mengonfigurasi penggunaan maksimum pada penyimpanan data cache dan cache Anda ECPU/second untuk mengontrol biaya cache. Tindakan ini akan memastikan bahwa penggunaan cache Anda tidak pernah melebihi jumlah maksimum yang dikonfigurasi.

Jika Anda menetapkan penskalaan maksimum maka aplikasi Anda mungkin mengalami penurunan kinerja cache saat cache mencapai maksimum. Ketika Anda mengatur penyimpanan data cache maksimum dan penyimpanan data cache Anda mencapai maksimum, ElastiCache akan mulai mengusir data dalam cache Anda yang memiliki set Time-To-Live (TTL), menggunakan logika LRU. Jika tidak ada data yang dapat dikosongkan, maka permintaan untuk menulis data tambahan akan menerima pesan kesalahan Out Of Memory (OOM). Bila Anda menetapkan ECPU/second maksimum dan penggunaan komputasi beban kerja Anda melebihi nilai ini, ElastiCache akan mulai membatasi permintaan.

Jika Anda mengatur batas maksimum pada `BytesUsedForCache` atau`ElastiCacheProcessingUnits`, kami sangat menyarankan untuk menyiapkan CloudWatch alarm pada nilai yang lebih rendah dari batas maksimum sehingga Anda diberi tahu saat cache Anda beroperasi mendekati batas ini. Sebaiknya atur alarm pada 75% dari batas maksimum yang Anda tetapkan. Lihat dokumentasi tentang cara mengatur CloudWatch alarm.

Pra-penskalaan dengan Tanpa Server ElastiCache

ElastiCache Pra-penskalaan tanpa server

Dengan pra-penskalaan, juga disebut pra-pemanasan, Anda dapat menetapkan batas minimum yang didukung untuk cache Anda. ElastiCache Anda dapat mengatur minimum ini untuk Unit ElastiCache Pemrosesan (ECPUs) per detik atau penyimpanan data. Ini dapat berguna dalam persiapan untuk acara penskalaan yang diantisipasi. Misalnya, jika perusahaan game mengharapkan peningkatan 5x dalam login dalam menit pertama game baru mereka diluncurkan, mereka dapat menyiapkan cache mereka untuk lonjakan penggunaan yang signifikan ini.

Anda dapat melakukan pra-penskalaan menggunakan ElastiCache konsol, CLI, atau API. ElastiCache Tanpa server memperbarui yang tersedia ECPUs/second di cache dalam waktu 60 menit, dan mengirimkan pemberitahuan acara ketika pembaruan batas minimum selesai.

Cara kerja pra-penskalaan

Ketika batas minimum untuk ECPUs/second atau penyimpanan data diperbarui melalui konsol, CLI, atau API, batas baru itu tersedia dalam 1 jam. ElastiCache Tanpa server mendukung 30K ECPUs/second pada cache kosong, dan hingga 90K ECPUs/sec saat menggunakan fitur Baca dari Replika. ElastiCache Tanpa server untuk Valkey 8.0 dapat menggandakan permintaan per detik (RPS) yang didukung setiap 2-3 menit, mencapai 5M RPS per cache dari nol dalam waktu kurang dari 13 menit, dengan latensi baca p50 sub-milidetik yang konsisten. Jika Anda mengantisipasi bahwa acara penskalaan yang akan datang mungkin melebihi tingkat ini, maka sebaiknya atur minimum ECPUs/second ke puncak yang ECPUs/sec Anda harapkan setidaknya 60 menit sebelum acara puncak. Jika tidak, aplikasi mungkin mengalami peningkatan latensi dan pembatasan permintaan.

Setelah pembaruan batas minimum selesai, ElastiCache Tanpa Server akan mulai mengukur Anda untuk minimum baru ECPUs per detik atau penyimpanan minimum baru. Hal ini terjadi bahkan jika aplikasi Anda tidak mengeksekusi permintaan pada cache, atau jika penggunaan penyimpanan data Anda di bawah minimum. Saat Anda menurunkan batas minimum dari pengaturan saat ini, pembaruan segera dilakukan sehingga ElastiCache Tanpa Server akan segera mulai mengukur pada batas minimum yang baru.

> **Note**
>
> - Ketika Anda menetapkan batas penggunaan minimum, Anda dikenakan biaya untuk batas tersebut meskipun penggunaan aktual Anda lebih rendah dari batas penggunaan minimum. ECPU atau penggunaan penyimpanan data yang melebihi batas penggunaan minimum dikenakan tarif reguler. Misalnya, jika Anda menetapkan batas penggunaan minimum 100.000 ECPUs/second maka Anda akan dikenakan biaya setidaknya $1,224 per jam (menggunakan harga ECPU di us-east-1), bahkan jika penggunaan Anda lebih rendah dari minimum yang ditetapkan.
> - ElastiCache Tanpa server mendukung skala minimum yang diminta pada tingkat agregat pada cache. ElastiCache Serverless juga mendukung maksimum 30K ECPUs/second per slot (90K ECPUs/second saat menggunakan Read from Replica menggunakan koneksi READONLY). Sebagai praktik terbaik, aplikasi Anda harus memastikan bahwa distribusi kunci di seluruh slot Valkey atau Redis OSS dan lalu lintas lintas kunci seragam mungkin.

Mengatur batas penskalaan menggunakan konsol dan AWS CLI

Menyetel batas penskalaan menggunakan Konsol AWS

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih mesin yang berjalan pada cache yang ingin Anda modifikasi.
3. Daftar cache yang menjalankan mesin yang dipilih akan muncul.
4. Pilih cache yang akan diubah dengan memilih tombol radio di sebelah kiri nama cache.
5. Pilih Tindakan, lalu pilih Ubah.
6. Di bawah batas Penggunaan, tetapkan batas Memori atau Komputasi yang sesuai.
7. Klik Pratinjau perubahan lalu Simpan perubahan.

Menetapkan batas penskalaan menggunakan AWS CLI

Untuk mengubah batas penskalaan menggunakan CLI, gunakan modify-serverless-cache API.

Linux:

```
aws elasticache modify-serverless-cache --serverless-cache-name <cache name> \
```

```
--cache-usage-limits 'DataStorage={Minimum=10,Maximum=100,Unit=GB},
 ECPUPerSecond={Minimum=1000,Maximum=100000}'
```

Windows:

```
aws elasticache modify-serverless-cache --serverless-cache-name <cache name> ^
--cache-usage-limits 'DataStorage={Minimum=10,Maximum=100,Unit=GB},
 ECPUPerSecond={Minimum=1000,Maximum=100000}'
```

Menghapus batas penskalaan menggunakan CLI

Untuk menghapus batas penskalaan menggunakan CLI, atur parameter batas Minimum dan Maksimum ke 0.

Linux:

```
aws elasticache modify-serverless-cache --serverless-cache-name <cache name> \
--cache-usage-limits 'DataStorage={Minimum=0,Maximum=0,Unit=GB},
 ECPUPerSecond={Minimum=0,Maximum=0}'
```

Windows:

```
aws elasticache modify-serverless-cache --serverless-cache-name <cache name> ^
--cache-usage-limits 'DataStorage={Minimum=0,Maximum=0,Unit=GB},
 ECPUPerSecond={Minimum=0,Maximum=0}'
```

Penskalaan cluster yang dirancang sendiri

Jumlah data yang perlu diproses oleh aplikasi Anda jarang bersifat statis. Jumlahnya meningkat dan menurun sejalan dengan bisnis Anda yang bertumbuh atau mengalami fluktuasi permintaan yang normal. Jika cache Anda dikelola sendiri, Anda perlu menyediakan perangkat keras yang memadai untuk puncak permintaan. Hal ini dapat memakan banyak biaya. Dengan menggunakan Amazon, ElastiCache Anda dapat menskalakan untuk memenuhi permintaan saat ini, hanya membayar untuk apa yang Anda gunakan. ElastiCache memungkinkan Anda untuk menskalakan cache Anda agar sesuai dengan permintaan.

> **Note**
>
> Jika cluster Valkey atau Redis OSS direplikasi di satu atau lebih Wilayah, maka Wilayah tersebut diskalakan secara berurutan. Saat meningkatkan, Wilayah sekunder diskalakan

terlebih dahulu dan kemudian Wilayah primer. Saat menurunkan skala, Wilayah primer adalah yang pertama dan kemudian setiap Wilayah sekunder mengikuti.
Saat memperbarui versi mesin, urutannya adalah Wilayah sekunder dan kemudian Wilayah primer.

Topik

Penskalaan untuk cluster Memcached

ElastiCache untuk Memcached menawarkan layanan caching dalam memori yang dikelola sepenuhnya yang menyebarkan, mengoperasikan, dan menskalakan Memcached secara vertikal di cloud. AWS

Penskalaan vertikal sesuai permintaan

Dengan penskalaan vertikal, ElastiCache untuk Memcached menyediakan sistem caching memori terdistribusi berkinerja tinggi yang banyak digunakan untuk mempercepat aplikasi dinamis dengan mengurangi beban database. Ini menyimpan data dan objek dalam RAM, mengurangi kebutuhan untuk membaca dari sumber data eksternal.

Anda dapat menggunakan cluster yang dirancang sendiri ElastiCache untuk memcached untuk penskalaan vertikal. Anda dapat menerapkan penskalaan vertikal ke cluster yang ada serta yang baru. Ini dapat memberikan fleksibilitas dalam alokasi sumber daya, memungkinkan pengguna untuk secara efisien beradaptasi dengan perubahan beban kerja tanpa mengubah arsitektur cluster. Kemampuan untuk menskalakan ini meningkatkan kinerja dengan meningkatkan kapasitas cache selama periode permintaan tinggi, dan mengurangi untuk mengoptimalkan biaya selama periode permintaan rendah. Ini menyederhanakan operasi, menghilangkan kebutuhan untuk membuat cluster baru untuk menggeser kebutuhan sumber daya, dan memungkinkan respons cepat terhadap fluktuasi lalu lintas. Secara keseluruhan, penskalaan vertikal untuk cache yang dirancang sendiri Memcached dapat membantu meningkatkan efisiensi biaya, meningkatkan pemanfaatan sumber daya, dan bahkan memungkinkan pengguna mengubah jenis instance Memcached mereka. Semua

memudahkan pengguna untuk menyelaraskan infrastruktur caching mereka dengan kebutuhan aplikasi yang sebenarnya.

> **Note**
>
> - Cache yang dirancang sendiri dan modifikasi tipe node hanya tersedia untuk mesin Memcached versi 1.5 atau yang lebih baru.
> - Penemuan Otomatis harus diaktifkan untuk memanfaatkan penskalaan vertikal.

Menyiapkan penskalaan vertikal sesuai permintaan untuk cluster Memcached ElastiCache

Anda dapat mengonfigurasi penskalaan vertikal sesuai permintaan untuk Memcached with`scale-config`, yang berisi dua parameter:

1. ScaleIntervalMinutes: Waktu (dalam hitungan menit) antara penskalaan batch selama proses upgrade Memcached
2. ScalePercentage: Persentase node untuk diskalakan secara bersamaan selama proses pemutakhiran Memcached

Mengonversi tipe node Memcached yang ada ke cache yang dapat menskalakan secara vertikal melalui CLI

Untuk mengonversi cache yang dirancang sendiri Memcached yang ada ke cache yang dapat menskalakan secara vertikal, Anda dapat menggunakan melalui `elasticache modify-cache-cluster` CLI.

```
aws elasticache modify-cache-cluster \
    --cache-cluster-id <your-cluster-id> \
    --cache-node-type <new-node-type> \
    --scale-config <scale-config> \
    --apply-immediately
```

Menyiapkan penskalaan vertikal dengan CLI

Untuk mengatur penskalaan vertikal untuk cache yang dirancang sendiri Memcached Anda melalui CLI, gunakan `elasticache modify-cache-cluster` dengan dan parameternya dan. `scale-config` `ScalePercentage` `ScaleIntervalMinutes`

- scale-interval-minutes: Ini menentukan waktu (dalam menit) antara batch penskalaan. Pengaturan ini dapat berkisar dari 2-30 menit. Jika tidak ada nilai yang ditentukan, nilai default 5 menit diterapkan.
- scale-percentage: Ini menentukan persentase node untuk skala secara bersamaan di setiap batch. Pengaturan ini dapat berkisar dari 10-100. Pengaturan dibulatkan saat membagi, jadi misalnya jika hasilnya adalah 49,5 pengaturan 50 diterapkan. Jika tidak ada nilai yang ditentukan, nilai default 20 diterapkan.

Opsi konfigurasi ini akan memungkinkan Anda untuk menyempurnakan proses penskalaan sesuai dengan kebutuhan spesifik Anda, menyeimbangkan antara meminimalkan gangguan cluster dan mengoptimalkan kecepatan penskalaan. Parameter scale-config hanya akan berlaku untuk tipe engine Memcached dan akan diabaikan untuk mesin cache lainnya, memastikan kompatibilitas mundur dengan penggunaan API yang ada untuk cluster lain.

Panggilan API

```
aws elasticache modify-cache-cluster \
    --cache-cluster-id <your-cluster-id> \
    --cache-node-type <new-node-type> \
    --scale-config '{
            "ScalePercentage": 30,
            "ScaleIntervalMinutes": 2
          }'
    --apply-immediately
```

Hasil:

Mengembalikan ID cache cluster dan perubahan yang tertunda.

```
{
    "CacheCluster": {
        "CacheNodeType": "old_insance_type",
         ...
         ...
         "PendingModifiedValues": {
            "CacheNodeType": "new_instance_type"
         },
    }
}
```

Buat daftar pengaturan penskalaan vertikal cache Memcached Anda

Anda dapat mengambil opsi penskalaan untuk cache Memcached Anda, dan melihat opsi mereka saat ini untuk penskalaan vertikal.

Panggilan API

```
aws elasticache list-allowed-node-type-modifications --cache-cluster-id <your-cluster-id>
```

Hasil:

```
{
  "ScaleUpModifications": [
      "cache.x.xxxx",
      "cache.x.xxxx"
      ],
   "ScaleDownModifications": [
      "cache.x.xxxx",
      "cache.x.xxxx",
      "cache.x.xxxx"
      ]
}
```

Penskalaan vertikal untuk Memcached dengan AWS Management Console

Untuk mengonversi cache yang dirancang sendiri Memcached yang sudah ada menjadi cache yang dapat menskalakan secara vertikal https://console.aws.amazon.com/elasticache/, ikuti langkah-langkah ini.

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pilih cache Memcached untuk dikonversi.
3. Pilih tab Modify.
4. Buka bagian Pengaturan cache, dan pilih jenis Node yang diinginkan.
5. Pilih Pratinjau perubahan, dan tinjau perubahannya.
6. Pilih Ubah.

Penskalaan horizontal otomatis untuk Memcached

ElastiCache sekarang terintegrasi dengan layanan AWS Application Auto Scaling (AAS) untuk menyertakan penskalaan horizontal otomatis untuk cluster Memcached. Anda dapat menentukan kebijakan penskalaan melalui layanan AWS Application Auto Scaling, dan secara otomatis menyesuaikan jumlah node dalam klaster Memcached sesuai kebutuhan, berdasarkan metrik atau jadwal yang telah ditentukan sebelumnya.

> **Note**
>
> Penskalaan horizontal otomatis saat ini tidak tersedia di Wilayah Beijing dan Ningxia.

Ini adalah metode yang tersedia untuk secara otomatis menskalakan cache yang dirancang sendiri secara horizontal.

- Penskalaan Terjadwal: Penskalaan berdasarkan jadwal memungkinkan Anda mengatur jadwal penskalaan Anda sendiri untuk perubahan beban yang dapat diprediksi. Misalnya, setiap pekan lalu lintas ke aplikasi web Anda mulai meningkat pada hari Rabu, tetap tinggi pada hari Kamis, dan mulai berkurang pada hari Jumat. Anda dapat mengonfigurasi Auto Scaling untuk meningkatkan kapasitas pada hari Rabu dan mengurangi kapasitas pada hari Jumat.
- Pelacakan Target: Dengan kebijakan penskalaan pelacakan target, Anda memilih metrik penskalaan dan menetapkan nilai target. Application Auto Scaling membuat dan mengelola CloudWatch alarm yang memicu kebijakan penskalaan dan menghitung penyesuaian penskalaan berdasarkan metrik dan nilai target. Kebijakan penskalaan menambah atau menghapus kapasitas yang diperlukan untuk menjaga metrik berada pada, atau mendekati, nilai target yang ditentukan.

Cara mengatur penskalaan horizontal untuk cache yang dirancang sendiri ElastiCache untuk Memcached melalui CLI

Untuk penskalaan horizontal dengan ElastiCache Memcached, Anda dapat memiliki kebijakan pelacakan target, kebijakan terjadwal, atau keduanya.

1. Daftarkan sumber daya sebagai target yang dapat diskalakan

   Panggil `RegisterScalableTarget` API di AWS Application Auto Scaling untuk mendaftarkan target untuk dimensi yang dapat diskalakan. `elasticache:cache-cluster:Nodes`

   API: ApplicationAutoScaling. RegisterScalableTarget

Masukan:

```
{
 "ScalableDimension": "elasticache:cache-cluster:Nodes",
 "ResourceId": "cache-cluster/test-cluster-1",
 "ServiceNamespace": "elasticache",
 "MinCapacity": 20,
 "MaxCapacity": 50
}
```

2. Membuat kebijakan penskalaan pelacakan Target

   Selanjutnya, Anda dapat membuat kebijakan penskalaan pelacakan target untuk sumber daya dengan memanggil API kebijakan penskalaan put.

3. Metrik yang telah ditentukan

   Berikut ini adalah kebijakan yang menskalakan sepanjang dimensi Cache Node, menggunakan metrik yang telah ditentukan `ElastiCacheCPUUtilization`, ditetapkan pada 50 untuk cache cluster test-cluster-1. Saat menghapus node untuk scale-in, n node terakhir akan dihapus.

   API: ApplicationAutoScaling. PutScalingPolicy

   Masukan:

```
{
 "PolicyName": "cpu50-target-tracking-scaling-policy",
 "PolicyType": "TargetTrackingScaling",
 "TargetTrackingScalingPolicyConfiguration": {
  "TargetValue": 50,
  "PredefinedMetricSpecification": {
   "PredefinedMetricType": "ElastiCacheCPUUtilization"
   },
  "ScaleOutCooldown": 600,
  "ScaleInCooldown": 600
   },
 "ServiceNamespace": "elasticache",
 "ScalableDimension": "elasticache:cache-cluster:Nodes",
 "ResourceId": "cache-cluster/test-cluster-1"
}
```

   Output:

```
{
 "PolicyARN": "arn:aws:autoscaling:us-west-2:012345678910:scalingPolicy:6d8972f3-
efc8-437c-92d1-6270f29a66e7:resource/elasticache/cache-cluster/test-
cluster-1:policyName/cpu50-target-tracking-scaling-policy",
 "Alarms": [
  {
  "AlarmARN": "arn:aws:cloudwatch:us-west-2:012345678910:alarm:TargetTracking-
elasticache/cache-cluster/test-cluster-1-AlarmHigh-d4f0770c-b46e-434a-
a60f-3b36d653feca",
  "AlarmName": "TargetTracking-elasticache/cache-cluster/test-cluster-1-AlarmHigh-
d4f0770c-b46e-434a-a60f-3b36d653feca"
  },
  {
  "AlarmARN": "arn:aws:cloudwatch:us-west-2:012345678910:alarm:TargetTracking-
elasticache/cache-cluster/test-cluster-1-AlarmLow-1b437334-d19b-4a63-
a812-6c67aaf2910d",
  "AlarmName": "TargetTracking-elasticache/cache-cluster/test-cluster-1-
AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
  }
 ]
}
```

4. Metrik Kustom

   Anda juga dapat menyetel kebijakan penskalaan pada dimensi dengan menggunakan persentase kustom yang didasarkan pada metrik Cloudwatch.

   Masukan:

```
{
 "PolicyName": "cpu50-target-tracking-scaling-policy",
 "PolicyType": "TargetTrackingScaling",
 "TargetTrackingScalingPolicyConfiguration": {
  "CustomizedMetricSpecification": {
   "Dimensions": [
    {
    "Name": "MyMetricDimension",
    "Value": "DimensionValue"
    }
    ],
   "MetricName": "MyCustomMetric",
   "Namespace": "MyNamespace",
```

```
   "Statistic": "Average",
   "Unit": "Percent"
   },
  "TargetValue": 40,
  "ScaleOutCooldown": 600,
  "ScaleInCooldown": 600
  },
 "ServiceNamespace": "elasticache",
 "ScalableDimension": "elasticache:cache-cluster:Nodes",
 "ResourceId": "cache-cluster/test-cluster-1"
}
```

5. Tindakan Terjadwal

   Saat Anda perlu skala untuk acara tertentu dan kemudian menskalakan setelah acara, Anda dapat membuat dua tindakan terjadwal dengan memanggil `PutScheduledAction` API.

   Kebijakan 1: Penskalaan

   `at`Perintah dalam `--schedule` menjadwalkan tindakan yang akan dijalankan sekali pada tanggal dan waktu yang ditentukan di masa depan. Bidang jadwal juga mendukung laju (menit, jam, hari dll) dan cron (untuk ekspresi cron).

   Pada tanggal dan waktu yang ditentukan, Application Auto Scaling memperbarui nilai `MinCapacity` dan `MaxCapacity`. Application Auto Scaling MinCapacity menskalakan untuk menempatkan node cache ke 70.

   API: ApplicationAutoScaling. PutScheduledAction

   Masukan:

```
{
 "ResourceId": "elasticache:ache-cluster:test-cluster-1",
 "ScalableDimension": "elasticache:cache-cluster:Nodes",
  "ScalableTargetAction": {
   "MaxCapacity": 100,
   "MinCapacity": 70
   },
 "Schedule": "at(2020-05-20T17:05:00)",
 "ScheduledActionName": "ScalingOutScheduledAction",
 "ServiceNamespace": "elasticache",
}
```

Kebijakan 2: Penskalaan

Pada tanggal dan waktu yang ditentukan, Application Auto Scaling memperbarui tabel `MinCapacity` dan`MaxCapacity`, dan skala `MaxCapacity` untuk mengembalikan node cache ke 60.

API: ApplicationAutoScaling. PutScheduledAction

Masukan:

```
{
 "ResourceId": "elasticache:cache-cluster:test-cluster-1",
 "ScalableDimension": "elasticache:cache-cluster:Nodes",
 "ScalableTargetAction": {
  "MaxCapacity": 60,
  "MinCapacity": 40
  },
 "Schedule": "at(2020-05-21T17:05:00)",
 "ScheduledActionName": "ScalingInScheduledAction",
 "ServiceNamespace": "elasticache",
}
```

6. Lihat Aktivitas Penskalaan

Anda dapat melihat aktivitas penskalaan menggunakan `DescribeScalingActivities` API.

API: ApplicationAutoScaling. DescribeScalingActivities

Output:

```
{
 "ScalingActivities": [
  {
  "ScalableDimension": "elasticache:elasticache:DesiredCount",
  "Description": "Setting desired count to 30.",
  "ResourceId": "elasticache/cache-cluster/test-cluster-1",
  "ActivityId": "4d759079-a31f-4d0c-8468-504c56e2eecf",
  "StartTime": 1462574194.658,
  "elasticacheNamespace": "elasticache",
  "EndTime": 1462574276.686,
```

```
  "Cause": "monitor alarm TargetTracking-elasticache/cache-cluster/test-cluster-1-
AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca in state ALARM triggered policy
 cpu50-target-tracking-scaling-policy",
  "StatusMessage": "Failed to set desired count to 30",
  "StatusCode": "Failed"
  },
  {
  "ScalableDimension": "elasticache:elasticache:DesiredCount",
  "Description": "Setting desired count to 25.",
  "ResourceId": "elasticache/cache-cluster/test-cluster-1",
  "ActivityId": "90aff0eb-dd6a-443c-889b-b809e78061c1",
  "StartTime": 1462574254.223,
  "elasticacheNamespace": "elasticache",
  "EndTime": 1462574333.492,
  "Cause": "monitor alarm TargetTracking-elasticache/cache-cluster/test-cluster-1-
AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca in state ALARM triggered policy
 cpu50-target-tracking-scaling-policy",
  "StatusMessage": "Successfully set desired count to 25. Change successfully
 fulfilled by elasticache.",
  "StatusCode": "Successful"
  }
 ]
}
```

7. Edit/Hapus Kebijakan Penskalaan

   Anda dapat mengedit atau menghapus kebijakan dengan memanggil `PutScalingPolicy` API lagi, atau dengan memanggil `DeleteScalingPolicy` atau `DeleteScheduled` Tindakan.

8. De-register target yang dapat diskalakan

   Anda dapat membatalkan pendaftaran target yang dapat diskalakan melalui API. `DeregisterScalableTarget` Membatalkan pendaftaran target yang dapat diskalakan akan menghapus kebijakan penskalaan dan tindakan terjadwal yang terkait dengannya.

   API: ApplicationAutoScaling. DeregisterScalableTarget

   Masukan:

```
{
 "ResourceId": "elasticache/cache-cluster/test-cluster-1",
 "ServiceNamespace": "elasticache",
 "ScalableDimension": "elasticache:cache-cluster:Nodes"
```

```
}
```

9. Pembersihan Kebijakan Penskalaan
10. Beberapa Kebijakan Penskalaan

    Anda dapat membuat beberapa kebijakan penskalaan. Berikut ini adalah info kunci tentang perilaku dari pelacakan target penskalaan otomatis.

    - Anda dapat memiliki beberapa kebijakan penskalaan pelacakan target untuk target yang dapat diskalakan, asalkan setiapnya menggunakan metrik yang berbeda.
    - Tujuan Application Auto Scaling adalah untuk selalu memprioritaskan ketersediaan, sehingga perilakunya berbeda tergantung pada apakah kebijakan pelacakan target siap untuk diperkecil atau diperbesar. Hal ini akan memperkecil skala target yang dapat diskala jika salah satu kebijakan pelacakan target siap untuk diperkecil skalanya, tetapi hanya akan memperbesar skala jika semua kebijakan pelacakan target (dengan penskalaan dalam porsi aktif) siap untuk diperbesar skalanya.
    - Jika beberapa kebijakan menginstruksikan target yang dapat diskalakan untuk memperbesar atau memperkecil skalanya di saat yang sama, Application Auto Scaling akan menskalakan berdasarkan kebijakan yang menyediakan kapasitas terbesar untuk pembesaran dan pengecilan skala. Hal ini memberikan fleksibilitas yang lebih besar untuk mencakup beberapa skenario dan memastikan bahwa selalu ada kapasitas yang cukup untuk memproses beban kerja aplikasi Anda.

    > **Note**
    >
    > AWS Application Auto Scaling tidak mengantri kebijakan penskalaan. Application Auto Scaling akan menunggu penskalaan pertama selesai, lalu cooldown, dan kemudian ulangi algoritma di atas.

Secara otomatis menskalakan cache Memcached secara horizontal melalui AWS Management Console

Untuk mengonversi cache yang dirancang sendiri Memcached yang sudah ada menjadi cache yang dapat menskalakan secara horizontal https://console.aws.amazon.com/elasticache/, ikuti langkah-langkah ini.

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pilih cache Memcached untuk dikonversi.
3. Buka tab Autoscaling.
4. Pilih kebijakan penskalaan yang akan diterapkan, dengan memilih Tambahkan penskalaan dinamis atau Tambahkan penskalaan terjadwal.
5. Isi detail untuk kebijakan yang dipilih sesuai kebutuhan.
6. Klik Buat.

Penskalaan manual untuk Memcached

Menskalakan cluster Memcached masuk atau keluar secara horizontal secara manual semudah menambahkan atau menghapus node dari cluster. Cluster memcached terdiri dari 1 hingga 60 node.

Karena Anda dapat mempartisi data di semua simpul dalam klaster Memcached, menaikkan skala ke jenis simpul dengan memori yang lebih besar jarang diperlukan. Namun, karena mesin Memcached tidak menyimpan data, jika Anda melakukan skala ke jenis node yang berbeda maka cluster baru Anda mulai kosong kecuali aplikasi Anda mengisinya.

Untuk menskalakan klaster Memcached secara manual secara vertikal, Anda harus membuat klaster baru. Klaster Memcached selalu dimulai dalam kondisi kosong kecuali jika aplikasi Anda mengisinya.

Penskalaan Cluster Memcached secara manual

Tindakan	Topik
Menskalakan ke luar	Menambahkan simpul ke klaster
Menskalakan ke dalam	Menghapus simpul dari klaster
Mengubah jenis simpul	Penskalaan Memcached secara manual secara vertikal

Topik

- Penskalaan Memcached secara manual secara horizontal

- Penskalaan Memcached secara manual secara vertikal

Penskalaan Memcached secara manual secara horizontal

Mesin Memcached mendukung partisi data Anda di beberapa simpul. Oleh karena itu, klaster Memcached dapat diskalakan secara horizontal dengan mudah. Untuk menskalakan klaster Memcached Anda secara horizontal, cukup tambahkan atau hapus simpul.

Topik berikut memberikan detail cara menskalakan klaster Memcached Anda ke luar atau ke dalam dengan menambahkan atau menghapus simpul.

- Menambahkan simpul ke klaster
- Menghapus simpul dari klaster

Setiap kali Anda mengubah jumlah simpul dalam klaster Memcached, Anda harus memetakan ulang setidaknya beberapa ruang kunci agar dipetakan ke simpul yang benar. Untuk informasi lebih detail tentang penyeimbangan beban klaster Memcached Anda, lihat Mengkonfigurasi ElastiCache klien Anda untuk penyeimbangan beban yang efisien (Memcached).

Jika Anda menggunakan penemuan otomatis pada klaster Memcached, Anda tidak perlu mengubah titik akhir dalam aplikasi saat Anda menambahkan atau menghapus simpul. Untuk informasi selengkapnya tentang penemuan otomatis, lihat Secara otomatis mengidentifikasi node di cluster Anda (Memcached). Jika Anda tidak menggunakan penemuan otomatis, setiap kali Anda mengubah jumlah simpul dalam klaster Memcached, Anda harus memperbarui titik akhir dalam aplikasi.

Penskalaan Memcached secara manual secara vertikal

Saat Anda menskalakan cluster Memcached secara manual ke atas atau ke bawah, Anda harus membuat klaster baru. Klaster Memcached selalu dimulai dalam kondisi kosong kecuali jika aplikasi Anda mengisinya.

> **Important**
>
> Jika Anda menskalakan ke bawah ke tipe simpul yang lebih kecil, pastikan bahwa tipe simpul yang lebih kecil memadai untuk data dan overhead Anda. Untuk informasi selengkapnya, lihat Memilih ukuran simpul Anda.

Topik

- Menskalakan Memcached secara vertikal (Konsol)
- Menskalakan Memcached secara vertikal (AWS CLI)
- Penskalaan Memcached secara vertikal (API) ElastiCache

Menskalakan Memcached secara vertikal (Konsol)

Prosedur berikut memandu Anda melalui penskalaan cluster Anda secara vertikal menggunakan konsol. ElastiCache

Untuk menskalakan klaster Memcached secara vertikal (konsol)

1. Buat klaster baru dengan jenis simpul baru. Untuk informasi selengkapnya, lihat Membuat klaster Memcached (konsol).
2. Dalam aplikasi Anda, perbarui titik akhir ke titik akhir klaster baru. Untuk informasi selengkapnya, lihat Menemukan Titik Akhir Cluster (Konsol) (Memcached).
3. Hapus klaster lama. Untuk informasi selengkapnya, lihat Menghapus simpul baru di Memcached.

Menskalakan Memcached secara vertikal (AWS CLI)

Prosedur berikut memandu Anda dalam menskalakan klaster cache Memcached secara vertikal menggunakan AWS CLI.

Untuk menskalakan klaster cache Memcached secara vertikal (AWS CLI)

1. Buat klaster cache baru dengan tipe simpul baru. Untuk informasi selengkapnya, lihat Membuat klaster (AWS CLI).
2. Dalam aplikasi Anda, perbarui titik akhir ke titik akhir klaster baru. Untuk informasi selengkapnya, lihat Menemukan Titik Akhir (AWS CLI).
3. Hapus klaster cache lama. Untuk informasi selengkapnya, lihat Menggunakan AWS CLI untuk menghapus ElastiCache cluster.

Penskalaan Memcached secara vertikal (API) ElastiCache

Prosedur berikut memandu Anda melalui penskalaan cluster cache Memcached Anda secara vertikal menggunakan API. ElastiCache

Untuk menskalakan cluster cache Memcached secara vertikal (API) ElastiCache

1. Buat klaster cache baru dengan jenis simpul baru. Untuk informasi selengkapnya, lihat Membuat cluster untuk Memcached (API) ElastiCache
2. Dalam aplikasi Anda, perbarui titik akhir ke titik akhir klaster cache baru. Untuk informasi selengkapnya, lihat Menemukan Titik Akhir (ElastiCache API).
3. Hapus klaster cache lama. Lihat informasi yang lebih lengkap di Menggunakan ElastiCache API.

Cluster penskalaan untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)

Cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dapat berupa cluster simpul tunggal dengan 0 pecahan atau cluster multi-node dengan 1 pecahan. Klaster simpul tunggal menggunakan satu simpul untuk operasi baca dan tulis. Cluster multi-node selalu memiliki 1 node sebagai node read/write utama dengan 0 hingga 5 node replika read-only.

Topik

- Penskalaan cluster simpul tunggal untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)

Penskalaan cluster Valkey atau Redis OSS

Tindakan	Valkey atau Redis OSS (mode cluster dinonaktifkan)	Valkey atau Redis OSS (mode cluster diaktifkan)
Menskalakan ke dalam	Menghapus node dari ElastiCache cluster	Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan)
Menskalakan ke luar	Menambahkan simpul ke klaster	Resharding online untuk Valkey atau Redis OSS (mode cluster diaktifkan)
Mengubah jenis simpul	Ke jenis simpul yang lebih besar: • Meningkatkan kluster Valkey atau Redis OSS node tunggal • Meningkatkan klaster Valkey atau Redis OSS dengan replika Ke jenis simpul yang lebih kecil:	Penskalaan vertikal online dengan mengubah jenis simpul

Tindakan	Valkey atau Redis OSS (mode cluster dinonaktifkan)	Valkey atau Redis OSS (mode cluster diaktifkan)
	• Menurunkan kluster Valkey atau Redis OSS simpul tunggal • Menurunkan kluster Valkey atau Redis OSS dengan replika	
Mengubah jumlah grup simpul	Tidak didukung untuk klaster Valkey atau Redis OSS (mode cluster dinonaktifkan)	Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan)

Daftar Isi

Penskalaan cluster simpul tunggal untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)

Node Valkey atau Redis OSS (mode cluster dinonaktifkan) harus cukup besar untuk menampung semua data cache ditambah overhead Valkey atau Redis OSS. Untuk mengubah kapasitas data cluster Valkey atau Redis OSS (mode cluster disabled) Anda, Anda harus menskalakan secara vertikal; meningkatkan ke tipe node yang lebih besar untuk meningkatkan kapasitas data, atau memperkecil ke tipe node yang lebih kecil untuk mengurangi kapasitas data.

Proses ElastiCache penskalaan dirancang untuk melakukan upaya terbaik untuk mempertahankan data Anda yang ada dan membutuhkan replikasi Valkey atau Redis OSS yang sukses. Untuk cluster Valkey atau Redis OSS (mode cluster dinonaktifkan), kami merekomendasikan agar memori yang cukup tersedia untuk Valkey atau Redis OSS.

Anda tidak dapat mempartisi data Anda di beberapa cluster Valkey atau Redis OSS (mode cluster dinonaktifkan). Namun, jika Anda hanya perlu menambah atau mengurangi kapasitas baca cluster Anda, Anda dapat membuat cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan node replika dan menambah atau menghapus replika baca. Untuk membuat cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan node replika menggunakan cluster cache Valkey atau Redis OSS simpul tunggal Anda sebagai cluster utama, lihat. Membuat cluster Valkey (mode cluster dinonaktifkan) (Konsol)

Setelah membuat klaster dengan replika, Anda dapat meningkatkan kapasitas baca dengan menambahkan replika baca. Jika diperlukan, Anda dapat mengurangi kapasitas baca dengan menghapus replika baca di lain waktu. Untuk informasi selengkapnya, lihat Meningkatkan kapasitas baca atau Mengurangi kapasitas baca.

Selain dapat menskalakan kapasitas baca, klaster Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan replika memberikan keuntungan bisnis lainnya. Untuk informasi selengkapnya, lihat Ketersediaan tinggi menggunakan grup replikasi.

> **Important**
>
> Jika grup parameter Anda menggunakan `reserved-memory` untuk menyisihkan memori untuk overhead Valkey atau Redis OSS, sebelum Anda mulai menskalakan pastikan bahwa Anda memiliki grup parameter khusus yang menyimpan jumlah memori yang benar untuk jenis node baru Anda. Sebagai alternatif, Anda dapat mengubah grup parameter kustom agar menggunakan `reserved-memory-percent` dan menggunakan grup parameter tersebut untuk klaster baru Anda.
> Jika menggunakan `reserved-memory-percent`, Anda tidak perlu melakukan ini.
> Untuk informasi selengkapnya, lihat Mengelola memori cadangan untuk Valkey dan Redis OSS.

Topik

- Meningkatkan kluster Valkey atau Redis OSS node tunggal
- Menurunkan kluster Valkey atau Redis OSS simpul tunggal

Meningkatkan kluster Valkey atau Redis OSS node tunggal

Saat Anda meningkatkan kluster Valkey atau Redis OSS simpul tunggal, ElastiCache lakukan proses berikut, baik Anda menggunakan ElastiCache konsol,, atau API AWS CLI. ElastiCache

1. Klaster cache baru dengan jenis simpul baru diluncurkan di Zona Ketersediaan yang sama dengan klaster cache yang ada.
2. Data cache dalam klaster cache yang ada disalin ke klaster cache baru. Berapa lama proses ini berjalan bergantung pada jenis simpul dan berapa banyak data dalam klaster cache Anda.
3. Operasi baca dan tulis sekarang dilayani menggunakan klaster cache baru. Karena titik akhir klaster cache baru sama seperti klaster cache lama, Anda tidak perlu memperbarui titik akhir dalam aplikasi Anda. Anda akan merasakan gangguan singkat (beberapa detik) pada operasi baca dan tulis dari simpul primer saat entri DNS diperbarui.
4. ElastiCache menghapus cluster cache lama. Anda akan merasakan gangguan singkat (beberapa detik) pada operasi baca dan tulis dari simpul lama karena koneksi ke simpul lama akan terputus.

> **Note**
>
> Untuk klaster yang menjalankan simpul jenis r6gd, Anda hanya dapat menskalakan ke ukuran simpul dalam keluarga simpul r6gd.

Seperti yang ditunjukkan pada tabel berikut, operasi penskalaan Valkey atau Redis OSS Anda diblokir jika Anda memiliki pemutakhiran mesin yang dijadwalkan untuk jendela pemeliharaan berikutnya. Untuk informasi selengkapnya tentang Periode Pemeliharaan, lihat Mengelola pemeliharaan ElastiCache cluster.

Operasi Valkey atau Redis OSS yang diblokir

Operasi Tertunda	Operasi Diblokir
Penaikan skala	Peningkatan mesin segera
Peningkatan mesin	Penaikan skala segera
Penaikan skala dan peningkatan mesin	Penaikan skala segera

Operasi Tertunda	Operasi Diblokir
	Peningkatan mesin segera

Jika memiliki operasi tertunda yang memblokir, Anda dapat melakukan salah satu hal berikut.

- Jadwalkan operasi penskalaan Valkey atau Redis OSS Anda untuk jendela pemeliharaan berikutnya dengan membersihkan kotak centang Terapkan segera (penggunaan CLI:, Penggunaan API:). `--no-apply-immediately` `ApplyImmediately=false`
- Tunggu hingga jendela pemeliharaan berikutnya (atau setelahnya) untuk melakukan operasi penskalaan Valkey atau Redis OSS Anda.
- Tambahkan upgrade mesin Valkey atau Redis OSS ke modifikasi cluster cache ini dengan kotak centang Terapkan Segera dipilih (CLI use:`--apply-immediately`, API use:). `ApplyImmediately=true` Tindakan ini akan membuka blokir operasi penaikan skala dengan memicu agar peningkatan mesin segera dilakukan.

Anda dapat meningkatkan klaster Valkey atau Redis OSS (mode cluster dinonaktifkan) node tunggal menggunakan ElastiCache konsol, the AWS CLI, atau API. ElastiCache

> **Important**
>
> Jika grup parameter Anda menggunakan `reserved-memory` untuk menyisihkan memori untuk overhead Valkey atau Redis OSS, sebelum Anda mulai menskalakan pastikan bahwa Anda memiliki grup parameter khusus yang menyimpan jumlah memori yang benar untuk jenis node baru Anda. Sebagai alternatif, Anda dapat mengubah grup parameter kustom agar menggunakan `reserved-memory-percent` dan menggunakan grup parameter tersebut untuk klaster baru Anda.
> Jika menggunakan `reserved-memory-percent`, Anda tidak perlu melakukan ini.
> Untuk informasi selengkapnya, lihat Mengelola memori cadangan untuk Valkey dan Redis OSS.

Meningkatkan kluster simpul tunggal untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (Konsol)

Prosedur berikut menjelaskan cara meningkatkan cluster Valkey atau Redis OSS node tunggal menggunakan Management Console. ElastiCache Selama proses ini, klaster Valkey atau Redis OSS Anda akan terus melayani permintaan dengan waktu henti minimal.

Untuk meningkatkan cluster Valkey atau Redis OSS node tunggal (konsol)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cluster Valkey atau Redis OSS.
3. Dari daftar cluster, pilih cluster yang ingin Anda tingkatkan (harus menjalankan mesin Valkey atau Redis OSS, bukan mesin Valkey atau Redis OSS yang berkerumun).
4. Pilih Ubah.
5. Di wizard Ubah Klaster:
   a. Pilih jenis simpul sebagai tujuan penskalaan dari daftar Jenis simpul.
   b. Jika Anda menggunakan `reserved-memory` untuk mengelola memori, dari daftar Grup Parameter, pilih grup parameter kustom yang mencadangkan jumlah memori yang benar untuk jenis simpul baru Anda.
6. Jika Anda ingin segera menaikkan skala, pilih kotak Terapkan segera. Jika kotak Terapkan segera tidak dipilih, proses menaikkan skala dilakukan selama periode pemeliharaan berikutnya dari klaster ini.
7. Pilih Ubah.

   Jika Anda memilih Terapkan segera pada langkah sebelumnya, status klaster berubah ke mengubah. Ketika status berubah ke tersedia, pengubahan selesai dan Anda dapat mulai menggunakan klaster baru tersebut.

Meningkatkan cluster cache Valkey atau Redis OSS node tunggal ()AWS CLI

Prosedur berikut menjelaskan cara meningkatkan cluster cache Valkey atau Redis OSS node tunggal menggunakan. AWS CLI Selama proses ini, klaster Valkey atau Redis OSS Anda akan terus melayani permintaan dengan waktu henti minimal.

Untuk meningkatkan cluster cache Valkey atau Redis OSS simpul tunggal ()AWS CLI

1. Tentukan jenis node yang dapat Anda tingkatkan dengan menjalankan AWS CLI `list-allowed-node-type-modifications` perintah dengan parameter berikut.

   - `--cache-cluster-id`

   Untuk Linux, macOS, atau Unix:

   ```
   aws elasticache list-allowed-node-type-modifications \
        --cache-cluster-id my-cache-cluster-id
   ```

   Untuk Windows:

   ```
   aws elasticache list-allowed-node-type-modifications ^
        --cache-cluster-id my-cache-cluster-id
   ```

   Output dari perintah di atas terlihat seperti berikut (format JSON).

   ```
   {
        "ScaleUpModifications": [
            "cache.m3.2xlarge",
            "cache.m3.large",
            "cache.m3.xlarge",
            "cache.m4.10xlarge",
            "cache.m4.2xlarge",
            "cache.m4.4xlarge",
            "cache.m4.large",
            "cache.m4.xlarge",
            "cache.r3.2xlarge",
            "cache.r3.4xlarge",
            "cache.r3.8xlarge",
            "cache.r3.large",
            "cache.r3.xlarge"
        ]
           "ScaleDownModifications": [
            "cache.t2.micro",
            "cache.t2.small ",
            "cache.t2.medium ",
               "cache.t1.small ",
        ],
   ```

```
}
```

Untuk informasi selengkapnya, lihat list-allowed-node-type-modifications dalam Referensi AWS CLI .

2. Ubah cluster cache yang ada yang menentukan cluster cache untuk ditingkatkan dan tipe node baru yang lebih besar, menggunakan AWS CLI `modify-cache-cluster` perintah dan parameter berikut.

   - `--cache-cluster-id` – Nama klaster cache yang dinaikkan skalanya.
   - `--cache-node-type` – Jenis simpul baru yang Anda inginkan untuk menskalakan klaster cache. Nilai ini harus berupa salah satu dari jenis simpul yang dihasilkan oleh perintah `list-allowed-node-type-modifications` di langkah 1.
   - `--cache-parameter-group-name` – [Opsional] Gunakan parameter ini jika Anda menggunakan `reserved-memory` untuk mengelola memori cadangan klaster. Tentukan grup parameter cache kustom yang mencadangkan jumlah memori yang sesuai untuk jenis simpul yang baru. Jika menggunakan `reserved-memory-percent`, Anda dapat menghilangkan parameter ini.
   - `--apply-immediately` – Menyebabkan operasi penaikan skala segera diterapkan. Untuk menunda proses penaikan skala ke periode pemeliharaan berikutnya untuk klaster, gunakan parameter `--no-apply-immediately`.

   Untuk Linux, macOS, atau Unix:

   ```
   aws elasticache modify-cache-cluster \
        --cache-cluster-id my-redis-cache-cluster \
        --cache-node-type cache.m3.xlarge \
        --cache-parameter-group-name redis32-m2-xl \
        --apply-immediately
   ```

   Untuk Windows:

   ```
   aws elasticache modify-cache-cluster ^
        --cache-cluster-id my-redis-cache-cluster ^
        --cache-node-type cache.m3.xlarge ^
        --cache-parameter-group-name redis32-m2-xl ^
        --apply-immediately
   ```

Output dari perintah di atas terlihat seperti berikut (format JSON).

```
{
     "CacheCluster": {
         "Engine": "redis",
         "CacheParameterGroup": {
             "CacheNodeIdsToReboot": [],
             "CacheParameterGroupName": "default.redis6.x",
             "ParameterApplyStatus": "in-sync"
         },
         "SnapshotRetentionLimit": 1,
         "CacheClusterId": "my-redis-cache-cluster",
         "CacheSecurityGroups": [],
         "NumCacheNodes": 1,
         "SnapshotWindow": "00:00-01:00",
         "CacheClusterCreateTime": "2017-02-21T22:34:09.645Z",
         "AutoMinorVersionUpgrade": true,
         "CacheClusterStatus": "modifying",
         "PreferredAvailabilityZone": "us-west-2a",
         "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
         "CacheSubnetGroupName": "default",
         "EngineVersion": "6.0",
         "PendingModifiedValues": {
             "CacheNodeType": "cache.m3.2xlarge"
         },
         "PreferredMaintenanceWindow": "tue:11:30-tue:12:30",
         "CacheNodeType": "cache.m3.medium",
          "DataTiering": "disabled"
     }
 }
```

Untuk informasi selengkapnya, lihat modify-cache-cluster dalam Referensi AWS CLI .

3. Jika Anda menggunakan`--apply-immediately`, periksa status cluster cache baru menggunakan AWS CLI `describe-cache-clusters` perintah dengan parameter berikut. Ketika status berubah ke tersedia, Anda dapat mulai menggunakan klaster cache baru yang lebih besar.

   - `--cache-cache cluster-id`— Nama cluster cache Valkey atau Redis OSS simpul tunggal Anda. Gunakan parameter ini untuk mendeskripsikan klaster cache tertentu daripada semua klaster cache.

```
aws elasticache describe-cache-clusters --cache-cluster-id my-redis-cache-cluster
```

Untuk informasi selengkapnya, lihat describe-cache-clusters dalam Referensi AWS CLI .

Meningkatkan cluster cache Valkey atau Redis OSS node tunggal (API) ElastiCache

Prosedur berikut menjelaskan cara meningkatkan cluster cache Valkey atau Redis OSS node tunggal menggunakan API. ElastiCache Selama proses ini, klaster Valkey atau Redis OSS Anda akan terus melayani permintaan dengan waktu henti minimal.

Untuk meningkatkan cluster cache Valkey atau Redis OSS node tunggal (API) ElastiCache

1. Tentukan tipe node yang dapat Anda tingkatkan dengan menjalankan `ListAllowedNodeTypeModifications` aksi ElastiCache API dengan parameter berikut.

   - `CacheClusterId`— Nama cluster cache Valkey atau Redis OSS simpul tunggal yang ingin Anda tingkatkan.

   ```
   https://elasticache.us-west-2.amazonaws.com/
       ?Action=ListAllowedNodeTypeModifications
       &CacheClusterId=MyRedisCacheCluster
       &Version=2015-02-02
       &SignatureVersion=4
       &SignatureMethod=HmacSHA256
       &Timestamp=20150202T192317Z
       &X-Amz-Credential=<credential>
   ```

   Untuk informasi selengkapnya, lihat ListAllowedNodeTypeModifications di Referensi Amazon ElastiCache API.

2. Ubah cluster cache yang ada yang menentukan cluster cache untuk ditingkatkan dan tipe node baru yang lebih besar, menggunakan aksi `ModifyCacheCluster` ElastiCache API dan parameter berikut.

   - `CacheClusterId` – Nama klaster cache yang dinaikkan skalanya.

- `CacheNodeType` – Jenis simpul baru yang Anda inginkan untuk menaikkan skala klaster cache. Nilai ini harus menjadi salah satu jenis node yang dikembalikan oleh `ListAllowedNodeTypeModifications` tindakan pada langkah sebelumnya.
- `CacheParameterGroupName` – [Opsional] Gunakan parameter ini jika Anda menggunakan `reserved-memory` untuk mengelola memori cadangan klaster. Tentukan grup parameter cache kustom yang mencadangkan jumlah memori yang sesuai untuk jenis simpul yang baru. Jika menggunakan `reserved-memory-percent`, Anda dapat menghilangkan parameter ini.
- `ApplyImmediately` – Tetapkan ke `true` agar proses penaikan skala segera diterapkan. Untuk menunda proses menaikkan skala ke periode pemeliharaan berikutnya dari klaster, gunakan `ApplyImmediately=false`.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=ModifyCacheCluster
    &ApplyImmediately=true
    &CacheClusterId=MyRedisCacheCluster
    &CacheNodeType=cache.m3.xlarge
    &CacheParameterGroupName redis32-m2-xl
    &Version=2015-02-02
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20150202T192317Z
    &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya, lihat ModifyCacheCluster di Referensi Amazon ElastiCache API.

3. Jika Anda menggunakan `ApplyImmediately=true`, periksa status cluster cache baru menggunakan `DescribeCacheClusters` tindakan ElastiCache API dengan parameter berikut. Ketika status berubah ke tersedia, Anda dapat mulai menggunakan klaster cache baru yang lebih besar.

   - `CacheClusterId`— Nama cluster cache Valkey atau Redis OSS simpul tunggal Anda. Gunakan parameter ini untuk mendeskripsikan klaster cache tertentu bukannya semua klaster cache.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=DescribeCacheClusters
    &CacheClusterId=MyRedisCacheCluster
    &Version=2015-02-02
```

```
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20150202T192317Z
    &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya, lihat DescribeCacheClusters di Referensi Amazon ElastiCache API.

Menurunkan kluster Valkey atau Redis OSS simpul tunggal

Bagian berikut memandu Anda melalui cara menskalakan cluster Valkey atau Redis OSS node tunggal ke tipe node yang lebih kecil. Memastikan bahwa tipe node baru yang lebih kecil cukup besar untuk mengakomodasi semua data dan overhead Valkey atau Redis OSS penting untuk kesuksesan jangka panjang cluster Valkey atau Redis OSS baru Anda. Untuk informasi selengkapnya, lihat Memastikan Anda memiliki cukup memori untuk membuat snapshot Valkey atau Redis OSS.

> **Note**
>
> Untuk klaster yang menjalankan simpul jenis r6gd, Anda hanya dapat menskalakan ke ukuran simpul dalam keluarga simpul r6gd.

Topik

- Menurunkan cluster Valkey atau Redis OSS simpul tunggal (Konsol)
- Menurunkan cluster cache Valkey atau Redis OSS simpul tunggal ()AWS CLI
- Menurunkan cluster cache Valkey atau Redis OSS node tunggal (API) ElastiCache

Menurunkan cluster Valkey atau Redis OSS simpul tunggal (Konsol)

Prosedur berikut memandu Anda melalui penskalaan cluster Valkey atau Redis OSS node tunggal Anda ke tipe node yang lebih kecil menggunakan konsol. ElastiCache

> **Important**
>
> Jika grup parameter Anda menggunakan `reserved-memory` untuk menyisihkan memori untuk overhead Valkey atau Redis OSS, sebelum Anda mulai menskalakan pastikan bahwa Anda memiliki grup parameter khusus yang menyimpan jumlah memori yang benar untuk jenis node baru Anda. Sebagai alternatif, Anda dapat mengubah grup parameter kustom agar menggunakan `reserved-memory-percent` dan menggunakan grup parameter tersebut untuk klaster baru Anda.
> Jika menggunakan `reserved-memory-percent`, Anda tidak perlu melakukan ini.
> Untuk informasi selengkapnya, lihat Mengelola memori cadangan untuk Valkey dan Redis OSS.

Untuk mengurangi kluster Valkey atau Redis OSS node tunggal Anda (konsol)

1. Pastikan bahwa jenis simpul yang lebih kecil memadai untuk data dan kebutuhan overhead Anda.
2. Jika grup parameter Anda menggunakan `reserved-memory` untuk menyisihkan memori untuk overhead Valkey atau Redis OSS, pastikan Anda memiliki grup parameter khusus untuk menyisihkan jumlah memori yang benar untuk jenis node baru Anda.

   Sebagai alternatif, Anda dapat mengubah grup parameter kustom untuk menggunakan `reserved-memory-percent`. Untuk informasi selengkapnya, lihat Mengelola memori cadangan untuk Valkey dan Redis OSS.
3. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
4. Dari daftar klaster, pilih klaster yang ingin Anda turunkan skalanya. Cluster ini harus menjalankan mesin Valkey atau Redis OSS dan bukan mesin Valkey atau Redis OSS yang berkerumun.
5. Pilih Ubah.
6. Di wizard Ubah Klaster:
   a. Pilih jenis simpul yang Anda inginkan untuk penurunan skala dari daftar Jenis simpul.
   b. Jika Anda menggunakan `reserved-memory` untuk mengelola memori, dari daftar Grup Parameter, pilih grup parameter kustom yang mencadangkan jumlah memori yang benar untuk jenis simpul baru Anda.
7. Jika Anda ingin segera menurunkan skala, pilih kotak centang Terapkan segera. Jika kotak centang Terapkan segera tidak dipilih, proses penurunan skala akan dilakukan selama periode pemeliharaan berikutnya dari klaster ini.
8. Pilih Ubah.
9. Ketika status klaster berubah dari mengubah ke tersedia, artinya klaster Anda telah diskalakan ke jenis simpul baru. Tidak perlu memperbarui titik akhir dalam aplikasi Anda.

Menurunkan cluster cache Valkey atau Redis OSS simpul tunggal ()AWS CLI

Prosedur berikut menjelaskan cara menurunkan cluster cache Valkey atau Redis OSS node tunggal menggunakan. AWS CLI

Untuk mengurangi cluster cache Valkey atau Redis OSS simpul tunggal ()AWS CLI

1. Tentukan jenis node yang dapat Anda turunkan dengan menjalankan AWS CLI `list-allowed-node-type-modifications` perintah dengan parameter berikut.

   - `--cache-cluster-id`

   Untuk Linux, macOS, atau Unix:

   ```
   aws elasticache list-allowed-node-type-modifications \
        --cache-cluster-id my-cache-cluster-id
   ```

   Untuk Windows:

   ```
   aws elasticache list-allowed-node-type-modifications ^
        --cache-cluster-id my-cache-cluster-id
   ```

   Output dari perintah di atas terlihat seperti berikut (format JSON).

   ```
   {
        "ScaleUpModifications": [
            "cache.m3.2xlarge",
            "cache.m3.large",
            "cache.m3.xlarge",
            "cache.m4.10xlarge",
            "cache.m4.2xlarge",
            "cache.m4.4xlarge",
            "cache.m4.large",
            "cache.m4.xlarge",
            "cache.r3.2xlarge",
            "cache.r3.4xlarge",
            "cache.r3.8xlarge",
            "cache.r3.large",
            "cache.r3.xlarge"
        ]
           "ScaleDownModifications": [
            "cache.t2.micro",
            "cache.t2.small ",
            "cache.t2.medium ",
               "cache.t1.small ",
        ],
   ```

```
}
```

Untuk informasi selengkapnya, lihat list-allowed-node-type-modifications dalam Referensi AWS CLI .

2. Ubah cluster cache yang ada yang menentukan cluster cache untuk menurunkan skala dan tipe node baru yang lebih kecil, menggunakan AWS CLI `modify-cache-cluster` perintah dan parameter berikut.

   - `--cache-cluster-id` – Nama klaster cache yang diturunkan skalanya.
   - `--cache-node-type` – Jenis simpul baru yang Anda inginkan untuk menskalakan klaster cache. Nilai ini harus berupa salah satu dari jenis simpul yang dihasilkan oleh perintah `list-allowed-node-type-modifications` di langkah 1.
   - `--cache-parameter-group-name` – [Opsional] Gunakan parameter ini jika Anda menggunakan `reserved-memory` untuk mengelola memori cadangan klaster. Tentukan grup parameter cache kustom yang mencadangkan jumlah memori yang sesuai untuk jenis simpul yang baru. Jika menggunakan `reserved-memory-percent`, Anda dapat menghilangkan parameter ini.
   - `--apply-immediately` – Menyebabkan proses penurunan skala segera diterapkan. Untuk menunda proses penaikan skala ke periode pemeliharaan berikutnya untuk klaster, gunakan parameter `--no-apply-immediately`.

   Untuk Linux, macOS, atau Unix:

   ```
   aws elasticache modify-cache-cluster \
        --cache-cluster-id my-redis-cache-cluster \
        --cache-node-type cache.m3.xlarge \
        --cache-parameter-group-name redis32-m2-xl \
        --apply-immediately
   ```

   Untuk Windows:

   ```
   aws elasticache modify-cache-cluster ^
        --cache-cluster-id my-redis-cache-cluster ^
        --cache-node-type cache.m3.xlarge ^
        --cache-parameter-group-name redis32-m2-xl ^
        --apply-immediately
   ```

Output dari perintah di atas terlihat seperti berikut (format JSON).

```
{
     "CacheCluster": {
         "Engine": "redis",
         "CacheParameterGroup": {
             "CacheNodeIdsToReboot": [],
             "CacheParameterGroupName": "default.redis6,x",
             "ParameterApplyStatus": "in-sync"
         },
         "SnapshotRetentionLimit": 1,
         "CacheClusterId": "my-redis-cache-cluster",
         "CacheSecurityGroups": [],
         "NumCacheNodes": 1,
         "SnapshotWindow": "00:00-01:00",
         "CacheClusterCreateTime": "2017-02-21T22:34:09.645Z",
         "AutoMinorVersionUpgrade": true,
         "CacheClusterStatus": "modifying",
         "PreferredAvailabilityZone": "us-west-2a",
         "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
         "CacheSubnetGroupName": "default",
         "EngineVersion": "6.0",
         "PendingModifiedValues": {
             "CacheNodeType": "cache.m3.2xlarge"
         },
         "PreferredMaintenanceWindow": "tue:11:30-tue:12:30",
         "CacheNodeType": "cache.m3.medium",
          "DataTiering": "disabled"
     }
 }
```

Untuk informasi selengkapnya, lihat modify-cache-cluster dalam Referensi AWS CLI .

3. Jika Anda menggunakan`--apply-immediately`, periksa status cluster cache baru menggunakan AWS CLI `describe-cache-clusters` perintah dengan parameter berikut. Ketika status berubah ke tersedia, Anda dapat mulai menggunakan klaster cache baru yang lebih besar.

   - `--cache-cache cluster-id`— Nama cluster cache Valkey atau Redis OSS simpul tunggal Anda. Gunakan parameter ini untuk mendeskripsikan klaster cache tertentu daripada semua klaster cache.

```
aws elasticache describe-cache-clusters --cache-cluster-id my-redis-cache-cluster
```

Untuk informasi selengkapnya, lihat describe-cache-clusters dalam Referensi AWS CLI .

Menurunkan cluster cache Valkey atau Redis OSS node tunggal (API) ElastiCache

Prosedur berikut menjelaskan cara menskalakan updown cluster cache Valkey atau Redis OSS node tunggal menggunakan API. ElastiCache

Untuk mengurangi cluster cache Valkey atau Redis OSS node tunggal (API) ElastiCache

1. Tentukan tipe node yang dapat Anda turunkan dengan menjalankan `ListAllowedNodeTypeModifications` aksi ElastiCache API dengan parameter berikut.

   - `CacheClusterId`— Nama cluster cache Valkey atau Redis OSS simpul tunggal yang ingin Anda turunkan.

   ```
   https://elasticache.us-west-2.amazonaws.com/
       ?Action=ListAllowedNodeTypeModifications
       &CacheClusterId=MyRedisCacheCluster
       &Version=2015-02-02
       &SignatureVersion=4
       &SignatureMethod=HmacSHA256
       &Timestamp=20150202T192317Z
       &X-Amz-Credential=<credential>
   ```

   Untuk informasi selengkapnya, lihat ListAllowedNodeTypeModifications di Referensi Amazon ElastiCache API.

2. Ubah cluster cache yang ada yang menentukan cluster cache untuk ditingkatkan dan tipe node baru yang lebih besar, menggunakan aksi `ModifyCacheCluster` ElastiCache API dan parameter berikut.

   - `CacheClusterId` – Nama klaster cache yang diturunkan skalanya.
   - `CacheNodeType` – Jenis simpul baru yang Anda inginkan untuk menurunkan skala klaster cache. Nilai ini harus menjadi salah satu jenis node yang dikembalikan oleh `ListAllowedNodeTypeModifications` tindakan pada langkah sebelumnya.

- `CacheParameterGroupName` – [Opsional] Gunakan parameter ini jika Anda menggunakan `reserved-memory` untuk mengelola memori cadangan klaster. Tentukan grup parameter cache kustom yang mencadangkan jumlah memori yang sesuai untuk jenis simpul yang baru. Jika menggunakan `reserved-memory-percent`, Anda dapat menghilangkan parameter ini.
- `ApplyImmediately` – Tetapkan ke `true` agar proses penurunan skala segera diterapkan. Untuk menunda proses penaikan skala ke periode pemeliharaan berikutnya dari klaster, gunakan `ApplyImmediately=false`.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=ModifyCacheCluster
    &ApplyImmediately=true
    &CacheClusterId=MyRedisCacheCluster
    &CacheNodeType=cache.m3.xlarge
    &CacheParameterGroupName redis32-m2-xl
    &Version=2015-02-02
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20150202T192317Z
    &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya, lihat ModifyCacheCluster di Referensi Amazon ElastiCache API.

3. Jika Anda menggunakan `ApplyImmediately=true`, periksa status cluster cache baru menggunakan `DescribeCacheClusters` tindakan ElastiCache API dengan parameter berikut. Ketika status berubah ke tersedia, Anda dapat mulai menggunakan klaster cache baru yang lebih kecil.

   - `CacheClusterId`— Nama cluster cache Valkey atau Redis OSS simpul tunggal Anda. Gunakan parameter ini untuk mendeskripsikan klaster cache tertentu bukannya semua klaster cache.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=DescribeCacheClusters
    &CacheClusterId=MyRedisCacheCluster
    &Version=2015-02-02
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20150202T192317Z
```

```
    &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya, lihat DescribeCacheClusters di Referensi Amazon ElastiCache API.

Menskalakan node replika untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)

Cluster Valkey atau Redis OSS dengan node replika (disebut grup replikasi di API/CLI) menyediakan ketersediaan tinggi melalui replikasi yang memiliki multi-AZ dengan failover otomatis diaktifkan. Cluster dengan node replika adalah kumpulan logis hingga enam node Valkey atau Redis OSS di mana satu node, Primer, dapat melayani permintaan baca dan tulis. Semua simpul lain dalam klaster adalah replika hanya-baca dari Primer. Data yang ditulis ke Primer direplikasi secara asinkron ke semua replika baca di klaster. Karena Valkey atau Redis OSS (mode cluster dinonaktifkan) tidak mendukung partisi data Anda di beberapa cluster, setiap node dalam grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) berisi seluruh kumpulan data cache. Cluster Valkey atau Redis OSS (mode cluster enabled) mendukung partisi data Anda hingga 500 pecahan.

Untuk mengubah kapasitas data klaster, Anda harus menaikkan skalanya ke jenis simpul yang lebih besar, atau menurunkan skalanya ke jenis simpul yang lebih kecil.

Untuk mengubah kapasitas baca klaster Anda, tambahkan lebih banyak replika baca, hingga maksimum 5, atau hapus replika baca.

Proses ElastiCache penskalaan dirancang untuk melakukan upaya terbaik untuk mempertahankan data Anda yang ada dan membutuhkan replikasi Valkey atau Redis OSS yang sukses. Untuk cluster Valkey atau Redis OSS dengan replika, kami merekomendasikan agar memori yang cukup tersedia untuk Valkey atau Redis OSS.

Topik

- Meningkatkan klaster Valkey atau Redis OSS dengan replika
- Menurunkan kluster Valkey atau Redis OSS dengan replika
- Meningkatkan kapasitas baca
- Mengurangi kapasitas baca

Topik Terkait

- Ketersediaan tinggi menggunakan grup replikasi
- Replikasi: Mode Cluster Valkey dan Redis OSS Dinonaktifkan vs Diaktifkan
- Meminimalkan downtime ElastiCache dengan menggunakan Multi-AZ dengan Valkey dan Redis OSS
- Memastikan Anda memiliki cukup memori untuk membuat snapshot Valkey atau Redis OSS

Topik

- Meningkatkan klaster Valkey atau Redis OSS dengan replika
- Menurunkan kluster Valkey atau Redis OSS dengan replika
- Meningkatkan kapasitas baca
- Mengurangi kapasitas baca

Meningkatkan klaster Valkey atau Redis OSS dengan replika

Amazon ElastiCache menyediakan dukungan konsol, CLI, dan API untuk menskalakan grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) Anda.

Ketika proses peningkatan skala dimulai, ElastiCache lakukan hal berikut:

1. Meluncurkan grup replikasi menggunakan jenis simpul baru.
2. Menyalin semua data dari simpul primer saat ini ke simpul primer baru.
3. Menyinkronkan replika baca baru dengan simpul primer baru.
4. Memperbarui entri DNS sehingga mengarah ke simpul baru. Karena hal ini, Anda tidak perlu memperbarui titik akhir dalam aplikasi Anda. Untuk Valkey 7.2 dan yang lebih baru atau Redis OSS 5.0.5 ke atas, Anda dapat menskalakan cluster yang diaktifkan failover otomatis sementara cluster terus online dan melayani permintaan yang masuk. Pada Redis OSS versi 4.0.10 dan di bawahnya, Anda mungkin melihat gangguan singkat membaca dan menulis pada versi sebelumnya dari node utama saat entri DNS diperbarui.
5. Menghapus simpul lama (CLI/API: grup replikasi). Anda akan merasakan gangguan singkat (beberapa detik) pada operasi baca dan tulis dari simpul lama karena koneksi ke simpul lama akan terputus.

Seberapa lama proses ini berjalan bergantung pada jenis simpul dan jumlah data dalam klaster Anda.

Seperti yang ditunjukkan pada tabel berikut, operasi penskalaan Valkey atau Redis OSS Anda diblokir jika Anda memiliki pemutakhiran mesin yang dijadwalkan untuk jendela pemeliharaan cluster berikutnya.

Operasi Valkey atau Redis OSS yang diblokir

Operasi Tertunda	Operasi Diblokir
Penaikan skala	Peningkatan mesin segera
Peningkatan mesin	Penaikan skala segera
Penaikan skala dan peningkatan mesin	Penaikan skala segera
	Peningkatan mesin segera

Jika memiliki operasi tertunda yang memblokir, Anda dapat melakukan salah satu hal berikut.

- Jadwalkan operasi penskalaan Valkey atau Redis OSS Anda untuk jendela pemeliharaan berikutnya dengan membersihkan kotak centang Terapkan segera (penggunaan CLI:, Penggunaan API:). `--no-apply-immediately` `ApplyImmediately=false`
- Tunggu hingga jendela pemeliharaan berikutnya (atau setelah) untuk melakukan operasi penskalaan Valkey atau Redis OSS Anda.
- Tambahkan upgrade mesin Valkey atau Redis OSS ke modifikasi cluster cache ini dengan kotak centang Terapkan Segera dipilih (CLI use:`--apply-immediately`, API use:). `ApplyImmediately=true` Tindakan ini akan membuka blokir operasi penaikan skala dengan memicu peningkatan mesin agar segera dilakukan.

Bagian berikut menjelaskan cara menskalakan cluster Valkey atau Redis OSS Anda dengan replika menggunakan ElastiCache konsol, the AWS CLI, dan API. ElastiCache

> **Important**
>
> Jika grup parameter Anda menggunakan `reserved-memory` untuk menyisihkan memori untuk overhead Valkey atau Redis OSS, sebelum Anda mulai menskalakan pastikan bahwa Anda memiliki grup parameter khusus yang menyimpan jumlah memori yang benar untuk jenis node baru Anda. Sebagai alternatif, Anda dapat mengubah grup parameter kustom agar menggunakan `reserved-memory-percent` dan menggunakan grup parameter tersebut untuk klaster baru Anda.
> Jika menggunakan `reserved-memory-percent`, Anda tidak perlu melakukan ini.
> Untuk informasi selengkapnya, lihat Mengelola memori cadangan untuk Valkey dan Redis OSS.

Meningkatkan klaster Valkey atau Redis OSS dengan replika (Konsol)

Jumlah waktu yang dibutuhkan untuk menaikkan skala ke jenis simpul yang lebih besar bervariasi, bergantung pada jenis simpul dan jumlah data dalam klaster Anda saat ini.

Proses berikut menskalakan klaster Anda dengan replika dari tipe node saat ini ke tipe node baru yang lebih besar menggunakan ElastiCache konsol. Selama proses ini, mungkin ada gangguan singkat terhadap operasi baca dan tulis untuk versi lain dari simpul primer saat entri DNS diperbarui. Anda mungkin akan mengalami waktu henti kurang dari 1 detik untuk simpul yang menjalankan versi 5.0.6 ke atas dan beberapa detik untuk versi yang lebih lama.

Untuk meningkatkan klaster Valkey atau Redis OSS dengan replika (konsol)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cluster Valkey atau cluster Redis OSS
3. Dari daftar klaster, pilih klaster yang ingin Anda naikkan skalanya. Cluster ini harus menjalankan mesin Valkey atau Redis OSS dan bukan mesin Valkey atau Redis OSS yang berkerumun.
4. Pilih Ubah.
5. Di wizard Ubah Klaster:
   a. Pilih jenis simpul sebagai tujuan penskalaan dari daftar Jenis simpul. Perhatikan bahwa tidak semua jenis simpul tersedia sebagai pilihan penurunan skala.
   b. Jika Anda menggunakan `reserved-memory` untuk mengelola memori, dari daftar Grup Parameter, pilih grup parameter kustom yang mencadangkan jumlah memori yang benar untuk jenis simpul baru Anda.
6. Jika Anda ingin segera melakukan proses penaikan skala, centang kotak Terapkan segera. Jika kotak centang Terapkan segera tidak dipilih, proses penaikan skala akan dilakukan selama periode pemeliharaan berikutnya untuk klaster ini.
7. Pilih Ubah.
8. Ketika status klaster berubah dari mengubah ke tersedia, artinya klaster Anda telah diskalakan ke jenis simpul baru. Tidak perlu memperbarui titik akhir dalam aplikasi Anda.

Meningkatkan grup replikasi Valkey atau Redis OSS ()AWS CLI

Proses berikut menskalakan grup replikasi dari jenis simpul saat ini ke jenis simpul baru yang lebih besar menggunakan AWS CLI. Selama proses ini, ElastiCache update entri DNS sehingga mereka menunjuk ke node baru. Karena hal ini, Anda tidak perlu memperbarui titik akhir dalam aplikasi Anda. Untuk Valkey 7.2 dan yang lebih baru atau Redis OSS 5.0.5 ke atas, Anda dapat menskalakan cluster yang diaktifkan failover otomatis sementara cluster terus online dan melayani permintaan yang masuk. Pada versi 4.0.10 ke bawah, Anda mungkin merasakan gangguan singkat terhadap operasi baca dan tulis pada versi sebelumnya dari simpul primer saat entri DNS diperbarui.

Jumlah waktu yang dibutuhkan untuk menaikkan skala ke jenis simpul yang lebih besar bervariasi, bergantung pada jenis simpul dan jumlah data dalam klaster cache Anda saat ini.

Untuk meningkatkan Grup Replikasi Valkey atau Redis OSS ()AWS CLI

1. Tentukan jenis node mana yang dapat Anda tingkatkan dengan menjalankan AWS CLI `list-allowed-node-type-modifications` perintah dengan parameter berikut.

   - `--replication-group-id` – nama grup replikasi. Gunakan parameter ini untuk mendeskripsikan grup replikasi tertentu, bukan semua grup replikasi.

   Untuk Linux, macOS, atau Unix:

   ```
   aws elasticache list-allowed-node-type-modifications \
        --replication-group-id my-repl-group
   ```

   Untuk Windows:

   ```
   aws elasticache list-allowed-node-type-modifications ^
        --replication-group-id my-repl-group
   ```

   Output dari operasi ini terlihat seperti berikut (format JSON).

   ```
   {
        "ScaleUpModifications": [
            "cache.m3.2xlarge",
            "cache.m3.large",
            "cache.m3.xlarge",
            "cache.m4.10xlarge",
            "cache.m4.2xlarge",
            "cache.m4.4xlarge",
            "cache.m4.large",
            "cache.m4.xlarge",
            "cache.r3.2xlarge",
            "cache.r3.4xlarge",
            "cache.r3.8xlarge",
            "cache.r3.large",
            "cache.r3.xlarge"
        ]
    }
   ```

   Untuk informasi selengkapnya, lihat list-allowed-node-type-modifications dalam Referensi AWS CLI .

2. Skala grup replikasi Anda saat ini hingga tipe node baru menggunakan AWS CLI `modify-replication-group` perintah dengan parameter berikut.

   - `--replication-group-id` – nama grup replikasi.
   - `--cache-node-type` – jenis simpul baru yang lebih besar dari klaster cache dalam grup replikasi ini. Nilai ini harus menjadi salah satu jenis instance yang dikembalikan oleh `list-allowed-node-type-modifications` perintah pada langkah sebelumnya.
   - `--cache-parameter-group-name` – [Opsional] Gunakan parameter ini jika Anda menggunakan `reserved-memory` untuk mengelola memori cadangan klaster. Tentukan grup parameter cache kustom yang mencadangkan jumlah memori yang sesuai untuk jenis simpul yang baru. Jika menggunakan `reserved-memory-percent`, Anda dapat menghilangkan parameter ini.
   - `--apply-immediately` – Menyebabkan operasi penaikan skala segera diterapkan. Untuk menunda operasi penaikan skala ke periode pemeliharaan berikutnya, gunakan `--no-apply-immediately`.

   Untuk Linux, macOS, atau Unix:

   ```
   aws elasticache modify-replication-group \
        --replication-group-id my-repl-group \
        --cache-node-type cache.m3.xlarge \
        --cache-parameter-group-name redis32-m3-2xl \
        --apply-immediately
   ```

   Untuk Windows:

   ```
   aws elasticache modify-replication-group ^
        --replication-group-id my-repl-group ^
        --cache-node-type cache.m3.xlarge ^
        --cache-parameter-group-name redis32-m3-2xl \
        --apply-immediately
   ```

   Output dari perintah ini terlihat seperti berikut (format JSON).

   ```
   {
    "ReplicationGroup": {
     "Status": "available",
   ```

```
  "Description": "Some description",
  "NodeGroups": [{
   "Status": "available",
   "NodeGroupMembers": [{
     "CurrentRole": "primary",
     "PreferredAvailabilityZone": "us-west-2b",
     "CacheNodeId": "0001",
     "ReadEndpoint": {
      "Port": 6379,
      "Address": "my-repl-group-001.8fdx4s.0001.usw2.cache.amazonaws.com"
     },
     "CacheClusterId": "my-repl-group-001"
    },
    {
     "CurrentRole": "replica",
     "PreferredAvailabilityZone": "us-west-2c",
     "CacheNodeId": "0001",
     "ReadEndpoint": {
      "Port": 6379,
      "Address": "my-repl-group-002.8fdx4s.0001.usw2.cache.amazonaws.com"
     },
     "CacheClusterId": "my-repl-group-002"
    }
   ],
   "NodeGroupId": "0001",
   "PrimaryEndpoint": {
    "Port": 6379,
    "Address": "my-repl-group.8fdx4s.ng.0001.usw2.cache.amazonaws.com"
   }
  }],
  "ReplicationGroupId": "my-repl-group",
  "SnapshotRetentionLimit": 1,
  "AutomaticFailover": "disabled",
  "SnapshotWindow": "12:00-13:00",
  "SnapshottingClusterId": "my-repl-group-002",
  "MemberClusters": [
   "my-repl-group-001",
   "my-repl-group-002"
  ],
  "PendingModifiedValues": {}
 }
}
```

Untuk informasi selengkapnya, lihat modify-replication-group dalam Referensi AWS CLI .

3. Jika Anda menggunakan `--apply-immediately` parameter, pantau status grup replikasi menggunakan AWS CLI `describe-replication-group` perintah dengan parameter berikut. Saat status masih mengubah, Anda mungkin akan merasakan waktu henti kurang dari 1 detik untuk simpul yang berjalan di versi 5.0.6 ke atas dan gangguan singkat pada operasi baca dan tulis untuk simpul primer versi lebih lama saat entri DNS diperbarui.

   - `--replication-group-id` – nama grup replikasi. Gunakan parameter ini untuk mendeskripsikan grup replikasi tertentu, bukan semua grup replikasi.

   Untuk Linux, macOS, atau Unix:

   ```
   aws elasticache describe-replication-groups \
        --replication-group-id my-replication-group
   ```

   Untuk Windows:

   ```
   aws elasticache describe-replication-groups ^
        --replication-group-id my-replication-group
   ```

   Untuk informasi selengkapnya, lihat describe-replication-groups dalam Referensi AWS CLI .

Meningkatkan grup replikasi Valkey atau Redis OSS (API) ElastiCache

Proses berikut menskalakan grup replikasi Anda dari tipe node saat ini ke tipe node baru yang lebih besar menggunakan ElastiCache API. Untuk Valkey 7.2 dan yang lebih baru atau Redis OSS 5.0.5 ke atas, Anda dapat menskalakan cluster yang diaktifkan failover otomatis sementara cluster terus online dan melayani permintaan yang masuk. Pada versi Redis OSS 4.0.10 dan di bawahnya, Anda mungkin melihat gangguan singkat membaca dan menulis pada versi sebelumnya dari node utama saat entri DNS diperbarui.

Jumlah waktu yang dibutuhkan untuk menaikkan skala ke jenis simpul yang lebih besar bervariasi, bergantung pada jenis simpul dan jumlah data dalam klaster cache Anda saat ini.

Untuk meningkatkan Grup Replikasi Valkey atau Redis OSS (API) ElastiCache

1. Tentukan tipe node mana yang dapat Anda tingkatkan untuk menggunakan `ListAllowedNodeTypeModifications` aksi ElastiCache API dengan parameter berikut.

   - `ReplicationGroupId` – nama grup replikasi. Gunakan parameter ini untuk mendeskripsikan grup replikasi tertentu, bukan semua grup replikasi.

   ```
   https://elasticache.us-west-2.amazonaws.com/
       ?Action=ListAllowedNodeTypeModifications
       &ReplicationGroupId=MyReplGroup
       &Version=2015-02-02
       &SignatureVersion=4
       &SignatureMethod=HmacSHA256
       &Timestamp=20150202T192317Z
       &X-Amz-Credential=<credential>
   ```

   Untuk informasi selengkapnya, lihat ListAllowedNodeTypeModifications di Referensi Amazon ElastiCache API.

2. Skalakan grup replikasi Anda saat ini hingga tipe node baru menggunakan aksi `ModifyReplicationGroup` ElastiCache API dan dengan parameter berikut.

   - `ReplicationGroupId` – nama grup replikasi.
   - `CacheNodeType` – jenis simpul baru yang lebih besar dari klaster cache dalam grup replikasi ini. Nilai ini harus menjadi salah satu jenis instance yang dikembalikan oleh `ListAllowedNodeTypeModifications` tindakan pada langkah sebelumnya.
   - `CacheParameterGroupName` – [Opsional] Gunakan parameter ini jika Anda menggunakan `reserved-memory` untuk mengelola memori cadangan klaster. Tentukan grup parameter cache kustom yang mencadangkan jumlah memori yang sesuai untuk jenis simpul yang baru. Jika menggunakan `reserved-memory-percent`, Anda dapat menghilangkan parameter ini.
   - `ApplyImmediately` – Tetapkan ke `true` agar proses penaikan skala segera diterapkan. Untuk menunda proses penaikan skala ke periode pemeliharaan berikutnya, gunakan `ApplyImmediately=false`.

   ```
   https://elasticache.us-west-2.amazonaws.com/
       ?Action=ModifyReplicationGroup
       &ApplyImmediately=true
   ```

```
    &CacheNodeType=cache.m3.2xlarge
    &CacheParameterGroupName=redis32-m3-2xl
    &ReplicationGroupId=myReplGroup
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &Version=2014-12-01
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Date=20141201T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20141201T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Untuk informasi selengkapnya, lihat ModifyReplicationGroup di Referensi Amazon ElastiCache API.

3. Jika Anda menggunakan `ApplyImmediately=true`, pantau status grup replikasi menggunakan `DescribeReplicationGroups` aksi ElastiCache API dengan parameter berikut. Ketika status berubah dari mengubah ke tersedia, Anda dapat mulai menulis ke grup replikasi baru yang telah dinaikkan skalanya.

   - `ReplicationGroupId` – nama grup replikasi. Gunakan parameter ini untuk mendeskripsikan grup replikasi tertentu, bukan semua grup replikasi.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=DescribeReplicationGroups
    &ReplicationGroupId=MyReplGroup
    &Version=2015-02-02
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20150202T192317Z
    &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya, lihat DescribeReplicationGroups di Referensi Amazon ElastiCache API.

Menurunkan kluster Valkey atau Redis OSS dengan replika

Bagian berikut memandu Anda melalui cara menskalakan cluster cache Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan node replika ke tipe node yang lebih kecil. Untuk meraih keberhasilan, sangat penting untuk memastikan bahwa jenis simpul baru yang lebih kecil memiliki kapasitas cukup besar untuk mengakomodasi semua data dan overhead. Untuk informasi selengkapnya, lihat Memastikan Anda memiliki cukup memori untuk membuat snapshot Valkey atau Redis OSS.

> **Note**
>
> Untuk klaster yang menjalankan simpul jenis r6gd, Anda hanya dapat menskalakan ke ukuran simpul dalam keluarga simpul r6gd.

> **Important**
>
> Jika grup parameter Anda menggunakan `reserved-memory` untuk menyisihkan memori untuk overhead Valkey atau Redis OSS, sebelum Anda mulai menskalakan pastikan bahwa Anda memiliki grup parameter khusus yang menyimpan jumlah memori yang benar untuk jenis node baru Anda. Sebagai alternatif, Anda dapat mengubah grup parameter kustom agar menggunakan `reserved-memory-percent` dan menggunakan grup parameter tersebut untuk klaster baru Anda.
> Jika menggunakan `reserved-memory-percent`, Anda tidak perlu melakukan ini.
> Untuk informasi selengkapnya, lihat Mengelola memori cadangan untuk Valkey dan Redis OSS.

Menurunkan klaster Valkey atau Redis OSS dengan replika (Konsol)

Proses berikut menskalakan cluster Valkey atau Redis OSS Anda dengan node replika ke tipe node yang lebih kecil menggunakan konsol. ElastiCache

Untuk menurunkan kluster Valkey atau Redis OSS dengan node replika (konsol)

1. Pastikan bahwa jenis simpul yang lebih kecil memadai untuk data dan kebutuhan overhead Anda.

2. Jika grup parameter Anda menggunakan `reserved-memory` untuk menyisihkan memori untuk overhead Valkey atau Redis OSS, pastikan Anda memiliki grup parameter khusus untuk menyisihkan jumlah memori yang benar untuk jenis node baru Anda.

   Sebagai alternatif, Anda dapat mengubah grup parameter kustom untuk menggunakan `reserved-memory-percent`. Untuk informasi selengkapnya, lihat Mengelola memori cadangan untuk Valkey dan Redis OSS.
3. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
4. Dari daftar klaster, pilih klaster yang ingin Anda turunkan skalanya. Cluster ini harus menjalankan mesin Valkey atau Redis OSS dan bukan mesin Valkey atau Redis OSS yang berkerumun.
5. Pilih Ubah.
6. Di wizard Ubah Klaster:

   a. Pilih jenis simpul yang Anda inginkan untuk penurunan skala dari daftar Jenis simpul.

   b. Jika Anda menggunakan `reserved-memory` untuk mengelola memori, dari daftar Grup Parameter, pilih grup parameter kustom yang mencadangkan jumlah memori yang benar untuk jenis simpul baru Anda.
7. Jika Anda ingin segera menurunkan skala, pilih kotak centang Terapkan segera. Jika kotak centang Terapkan segera tidak dipilih, proses penurunan skala akan dilakukan selama periode pemeliharaan berikutnya dari klaster ini.
8. Pilih Ubah.
9. Ketika status klaster berubah dari mengubah ke tersedia, artinya klaster Anda telah diskalakan ke jenis simpul baru. Tidak perlu memperbarui titik akhir dalam aplikasi Anda.

Menurunkan grup replikasi Valkey atau Redis OSS ()AWS CLI

Proses berikut menskalakan grup replikasi Anda dari jenis simpul saat ini ke jenis simpul baru yang lebih kecil menggunakan AWS CLI. Selama proses ini, ElastiCache update entri DNS sehingga mereka menunjuk ke node baru. Karena hal ini, Anda tidak perlu memperbarui titik akhir dalam aplikasi Anda. Untuk Valkey 7.2 di atas atau Redis OSS 5.0.5 ke atas, Anda dapat menskalakan kluster yang diaktifkan failover otomatis sementara cluster terus online dan melayani permintaan yang masuk. Pada versi 4.0.10 ke bawah, Anda mungkin merasakan gangguan singkat terhadap operasi baca dan tulis pada versi sebelumnya dari simpul primer saat entri DNS diperbarui.

Namun, operasi baca dari klaster cache replika baca terus berfungsi tanpa gangguan.

Jumlah waktu yang dibutuhkan untuk menurunkan skala ke jenis simpul yang lebih kecil bervariasi, bergantung pada jenis simpul dan jumlah data dalam klaster cache Anda saat ini.

Untuk menurunkan Grup Replikasi Valkey atau Redis OSS ()AWS CLI

1. Tentukan jenis node mana yang dapat Anda turunkan dengan menjalankan AWS CLI `list-allowed-node-type-modifications` perintah dengan parameter berikut.

   - `--replication-group-id` – nama grup replikasi. Gunakan parameter ini untuk mendeskripsikan grup replikasi tertentu, bukan semua grup replikasi.

   Untuk Linux, macOS, atau Unix:

   ```
   aws elasticache list-allowed-node-type-modifications \
        --replication-group-id my-repl-group
   ```

   Untuk Windows:

   ```
   aws elasticache list-allowed-node-type-modifications ^
        --replication-group-id my-repl-group
   ```

   Output dari operasi ini terlihat seperti berikut (format JSON).

   ```
   {
        "ScaleDownModifications": [
            "cache.m3.2xlarge",
            "cache.m3.large",
            "cache.m3.xlarge",
            "cache.m4.10xlarge",
            "cache.m4.2xlarge",
            "cache.m4.4xlarge",
            "cache.m4.large",
            "cache.m4.xlarge",
            "cache.r3.2xlarge",
            "cache.r3.4xlarge",
            "cache.r3.8xlarge",
            "cache.r3.large",
            "cache.r3.xlarge"
        ]
    }
   ```

Untuk informasi selengkapnya, lihat list-allowed-node-type-modifications dalam Referensi AWS CLI .

2. Skala grup replikasi Anda saat ini hingga tipe node baru menggunakan AWS CLI `modify-replication-group` perintah dengan parameter berikut.

   - `--replication-group-id` – nama grup replikasi.
   - `--cache-node-type` – jenis simpul baru yang lebih kecil dari klaster cache dalam grup replikasi ini. Nilai ini harus menjadi salah satu jenis instance yang dikembalikan oleh `list-allowed-node-type-modifications` perintah pada langkah sebelumnya.
   - `--cache-parameter-group-name` – [Opsional] Gunakan parameter ini jika Anda menggunakan `reserved-memory` untuk mengelola memori cadangan klaster. Tentukan grup parameter cache kustom yang mencadangkan jumlah memori yang sesuai untuk jenis simpul yang baru. Jika menggunakan `reserved-memory-percent`, Anda dapat menghilangkan parameter ini.
   - `--apply-immediately` – Menyebabkan operasi penaikan skala segera diterapkan. Untuk menunda operasi penaikan skala ke periode pemeliharaan berikutnya, gunakan `--no-apply-immediately`.

   Untuk Linux, macOS, atau Unix:

   ```
   aws elasticache modify-replication-group \
        --replication-group-id my-repl-group \
        --cache-node-type cache.t2.small  \
        --cache-parameter-group-name redis32-m3-2xl \
        --apply-immediately
   ```

   Untuk Windows:

   ```
   aws elasticache modify-replication-group ^
        --replication-group-id my-repl-group ^
        --cache-node-type cache.t2.small  ^
        --cache-parameter-group-name redis32-m3-2xl \
        --apply-immediately
   ```

   Output dari perintah ini terlihat seperti berikut (format JSON).

```
{"ReplicationGroup": {
         "Status": "available",
         "Description": "Some description",
         "NodeGroups": [
             {
                 "Status": "available",
                 "NodeGroupMembers": [
                     {
                         "CurrentRole": "primary",
                         "PreferredAvailabilityZone": "us-west-2b",
                         "CacheNodeId": "0001",
                         "ReadEndpoint": {
                             "Port": 6379,
                             "Address": "my-repl-
group-001.8fdx4s.0001.usw2.cache.amazonaws.com"
                         },
                         "CacheClusterId": "my-repl-group-001"
                     },
                     {
                         "CurrentRole": "replica",
                         "PreferredAvailabilityZone": "us-west-2c",
                         "CacheNodeId": "0001",
                         "ReadEndpoint": {
                             "Port": 6379,
                             "Address": "my-repl-
group-002.8fdx4s.0001.usw2.cache.amazonaws.com"
                         },
                         "CacheClusterId": "my-repl-group-002"
                     }
                 ],
                 "NodeGroupId": "0001",
                 "PrimaryEndpoint": {
                     "Port": 6379,
                     "Address": "my-repl-
group.8fdx4s.ng.0001.usw2.cache.amazonaws.com"
                 }
             }
         ],
         "ReplicationGroupId": "my-repl-group",
         "SnapshotRetentionLimit": 1,
         "AutomaticFailover": "disabled",
         "SnapshotWindow": "12:00-13:00",
```

```
            "SnapshottingClusterId": "my-repl-group-002",
            "MemberClusters": [
                "my-repl-group-001",
                "my-repl-group-002",
            ],
            "PendingModifiedValues": {}
        }
    }
```

Untuk informasi selengkapnya, lihat modify-replication-group dalam Referensi AWS CLI .

3. Jika Anda menggunakan `--apply-immediately` parameter, pantau status grup replikasi menggunakan AWS CLI `describe-replication-group` perintah dengan parameter berikut. Ketika status berubah dari mengubah ke tersedia, Anda dapat mulai menulis ke grup replikasi baru yang telah diturunkan skalanya.

   - `--replication-group-id` – nama grup replikasi. Gunakan parameter ini untuk mendeskripsikan grup replikasi tertentu, bukan semua grup replikasi.

   Untuk Linux, macOS, atau Unix:

   ```
   aws elasticache describe-replication-group \
        --replication-group-id my-replication-group
   ```

   Untuk Windows:

   ```
   aws elasticache describe-replication-groups ^
        --replication-group-id my-replication-group
   ```

   Untuk informasi selengkapnya, lihat describe-replication-groups dalam Referensi AWS CLI .

Menurunkan grup replikasi Valkey atau Redis OSS (API) ElastiCache

Proses berikut menskalakan grup replikasi Anda dari tipe node saat ini ke tipe node baru yang lebih kecil menggunakan ElastiCache API. Selama proses ini, ElastiCache update entri DNS sehingga mereka menunjuk ke node baru. Karena hal ini, Anda tidak perlu memperbarui titik akhir dalam aplikasi Anda. Untuk Valkey 7.2 dan yang lebih baru atau Redis OSS 5.0.5 ke atas, Anda dapat menskalakan cluster yang diaktifkan failover otomatis sementara cluster terus online dan melayani permintaan yang masuk. Pada Redis OSS versi 4.0.10 dan di bawahnya, Anda mungkin melihat

gangguan singkat membaca dan menulis pada versi sebelumnya dari node utama saat entri DNS diperbarui.. Namun, operasi baca dari klaster cache replika baca terus berfungsi tanpa gangguan.

Jumlah waktu yang dibutuhkan untuk menurunkan skala ke jenis simpul yang lebih kecil bervariasi, bergantung pada jenis simpul dan jumlah data dalam klaster cache Anda saat ini.

Untuk mengurangi Grup Replikasi Valkey atau Redis OSS (API) ElastiCache

1. Tentukan tipe node mana yang dapat Anda turunkan untuk menggunakan `ListAllowedNodeTypeModifications` aksi ElastiCache API dengan parameter berikut.

   - `ReplicationGroupId` – nama grup replikasi. Gunakan parameter ini untuk mendeskripsikan grup replikasi tertentu, bukan semua grup replikasi.

   ```
   https://elasticache.us-west-2.amazonaws.com/
       ?Action=ListAllowedNodeTypeModifications
       &ReplicationGroupId=MyReplGroup
       &Version=2015-02-02
       &SignatureVersion=4
       &SignatureMethod=HmacSHA256
       &Timestamp=20150202T192317Z
       &X-Amz-Credential=<credential>
   ```

   Untuk informasi selengkapnya, lihat ListAllowedNodeTypeModifications di Referensi Amazon ElastiCache API.

2. Skalakan grup replikasi Anda saat ini hingga tipe node baru menggunakan aksi `ModifyReplicationGroup` ElastiCache API dan dengan parameter berikut.

   - `ReplicationGroupId` – nama grup replikasi.
   - `CacheNodeType` – jenis simpul baru yang lebih kecil dari klaster cache dalam grup replikasi ini. Nilai ini harus menjadi salah satu jenis instance yang dikembalikan oleh `ListAllowedNodeTypeModifications` tindakan pada langkah sebelumnya.
   - `CacheParameterGroupName` – [Opsional] Gunakan parameter ini jika Anda menggunakan `reserved-memory` untuk mengelola memori cadangan klaster. Tentukan grup parameter cache kustom yang mencadangkan jumlah memori yang sesuai untuk jenis simpul yang baru. Jika menggunakan `reserved-memory-percent`, Anda dapat menghilangkan parameter ini.

- `ApplyImmediately` – Tetapkan ke `true` agar proses penaikan skala segera diterapkan. Untuk menunda proses penurunan skala ke periode pemeliharaan berikutnya, gunakan `ApplyImmediately=false`.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=ModifyReplicationGroup
    &ApplyImmediately=true
    &CacheNodeType=cache.m3.2xlarge
    &CacheParameterGroupName=redis32-m3-2xl
    &ReplicationGroupId=myReplGroup
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &Version=2014-12-01
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Date=20141201T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20141201T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Untuk informasi selengkapnya, lihat ModifyReplicationGroup di Referensi Amazon ElastiCache API.

3. Jika Anda menggunakan `ApplyImmediately=true`, pantau status grup replikasi menggunakan `DescribeReplicationGroups` aksi ElastiCache API dengan parameter berikut. Ketika status berubah dari mengubah ke tersedia, Anda dapat mulai menulis ke grup replikasi baru yang telah diturunkan skalanya.

   - `ReplicationGroupId` – nama grup replikasi. Gunakan parameter ini untuk mendeskripsikan grup replikasi tertentu, bukan semua grup replikasi.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=DescribeReplicationGroups
    &ReplicationGroupId=MyReplGroup
    &Version=2015-02-02
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20150202T192317Z
```

```
    &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya, lihat DescribeReplicationGroups di Referensi Amazon ElastiCache API.

Meningkatkan kapasitas baca

Untuk meningkatkan kapasitas baca, tambahkan replika baca (hingga maksimal lima) ke grup replikasi Valkey atau Redis OSS Anda.

Anda dapat menskalakan kapasitas baca cluster Valkey atau Redis OSS Anda menggunakan ElastiCache konsol, the AWS CLI, atau API. ElastiCache Untuk informasi selengkapnya, lihat Menambahkan replika baca untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan).

Mengurangi kapasitas baca

Untuk mengurangi kapasitas baca, hapus satu atau lebih replika baca dari cluster Valkey atau Redis OSS Anda dengan replika (disebut grup replikasi di API/CLI). Jika klaster adalah Multi-AZ yang mengaktifkan failover otomatis, Anda tidak dapat menghapus replika baca terakhir tanpa menonaktifkan Multi-AZ terlebih dahulu. Untuk informasi selengkapnya, lihat Mengubah grup replikasi.

Lihat informasi yang lebih lengkap di Menghapus replika baca untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan).

Penskalaan cluster di Valkey atau Redis OSS (Mode Cluster Diaktifkan)

Saat permintaan pada klaster Anda berubah, Anda mungkin memutuskan untuk meningkatkan kinerja atau mengurangi biaya dengan mengubah jumlah pecahan di cluster Valkey atau Redis OSS (mode cluster enabled) Anda. Sebaiknya gunakan penskalaan horizontal online untuk melakukan tindakan tersebut. Dengan begitu, klaster Anda dapat terus melayani permintaan selama proses penskalaan.

Kondisi berikut dapat menjadi faktor yang membuat Anda memutuskan menskalakan ulang klaster:

- Tekanan memori:

  Jika simpul di klaster Anda mengalami tekanan memori, sebaiknya pertimbangkan untuk menskalakan ke luar agar memiliki lebih banyak sumber daya untuk menyimpan data dan melayani permintaan dengan lebih baik.

  Anda dapat menentukan apakah node Anda berada di bawah tekanan memori dengan memantau metrik berikut: FreeableMemory, SwapUsage, dan BytesUsedForCache.
- Hambatan CPU atau jaringan:

  Jika latency/throughput masalah mengganggu klaster Anda, Anda mungkin perlu meningkatkan skala untuk menyelesaikan masalah.

  Anda dapat memantau tingkat latensi dan throughput Anda dengan memantau metrik berikut: CPUUtilization,,, NetworkBytesInNetworkBytesOut, CurrConnectionsdan. NewConnections
- Klaster Anda diskalakan berlebih:

  Permintaan saat ini pada klaster Anda dalam kondisi yang membuat penskalaan ke dalam tidak mengganggu performa dan mengurangi biaya.

  Anda dapat memantau penggunaan klaster untuk menentukan apakah Anda dapat menskalakan dengan aman menggunakan metrik berikut: FreeableMemory,,, SwapUsage, BytesUsedForCache, CPUUtilization, NetworkBytesIn, NetworkBytesOutCurrConnections, dan NewConnections.

Dampak Performa dari Penskalaan

Ketika Anda menskalakan menggunakan proses offline, klaster akan offline selama sebagian besar proses dan dengan demikian tidak dapat melayani permintaan. Ketika Anda menskalakan menggunakan metode online, karena penskalaan adalah operasi sarat komputasi, ada beberapa

penurunan dalam performa, namun, klaster Anda terus melayani permintaan selama operasi penskalaan. Berapa banyak penurunan yang Anda alami bergantung pada pemanfaatan CPU normal dan data Anda.

Ada dua cara untuk menskalakan cluster Valkey atau Redis OSS (mode cluster enabled) Anda; penskalaan horizontal dan vertikal.

- Penskalaan horizontal mengizinkan Anda mengubah jumlah grup simpul (serpihan) dalam grup replikasi dengan menambahkan atau menghapus grup simpul (serpihan). Proses resharding online memungkinkan penskalaan in/out sementara klaster terus melayani permintaan yang masuk.

  Konfigurasikan slot di klaster baru Anda secara berbeda dari slot yang berada di klaster lama. Metode offline saja.
- Penskalaan Vertikal - Ubah jenis simpul untuk mengubah ukuran klaster. Penskalaan vertikal online memungkinkan penskalaan up/down sementara klaster terus melayani permintaan yang masuk.

Jika Anda mengurangi ukuran dan kapasitas memori cluster, baik dengan skala atau scaling down, pastikan bahwa konfigurasi baru memiliki memori yang cukup untuk data Anda dan Valkey atau Redis OSS overhead.

Untuk informasi selengkapnya, lihat Memilih ukuran simpul Anda.

Daftar Isi

- Penskalaan vertikal online dengan mengubah jenis simpul
  - Penskalaan ke atas secara online
    - Meningkatkan klaster cache Valkey atau Redis OSS (Konsol)
    - Meningkatkan klaster cache Valkey atau Redis OSS ()AWS CLI
    - Meningkatkan klaster cache Valkey atau Redis OSS (API) ElastiCache
  - Penskalaan ke bawah secara online
    - Menskalakan cluster cache Valkey atau Redis OSS (Konsol)
    - Menskalakan cluster cache Valkey atau Redis OSS ()AWS CLI
    - Menskalakan cluster cache Valkey atau Redis OSS (API) ElastiCache

Resharding offline untuk Valkey atau Redis OSS (mode cluster diaktifkan)

Keuntungan utama yang Anda dapatkan dari konfigurasi ulang serpihan secara offline adalah bahwa Anda dapat melakukan lebih dari sekadar menambahkan atau menghapus serpihan dari grup replikasi Anda. Saat Anda melakukan reshard dan menyeimbangkan ulang secara offline, selain mengubah jumlah pecahan di grup replikasi Anda, Anda dapat melakukan hal berikut:

> **Note**
>
> Resharding offline tidak didukung pada cluster Valkey atau Redis OSS dengan tiering data diaktifkan. Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.

- Ubah jenis simpul grup replikasi Anda.
- Tentukan Zona Ketersediaan untuk setiap simpul dalam grup replikasi.
- Tingkatkan ke versi mesin yang lebih baru.
- Tentukan jumlah simpul replika di setiap serpihan secara independen.
- Tentukan ruang kunci untuk setiap serpihan.

Kekurangan utama dari konfigurasi ulang serpihan secara offline adalah bahwa klaster Anda offline dimulai dari bagian pemulihan dari proses berlanjut sampai Anda memperbarui titik akhir dalam aplikasi Anda. Durasi offline klaster Anda akan bervariasi tergantung jumlah data dalam klaster Anda.

Untuk mengkonfigurasi ulang pecahan Anda Valkey atau Redis OSS (mode cluster diaktifkan) cluster offline

1. Buat cadangan manual dari cluster Valkey atau Redis OSS Anda yang ada. Untuk informasi selengkapnya, lihat Membuat cadangan manual.
2. Buat klaster baru dengan memulihkan dari cadangan. Untuk informasi selengkapnya, lihat Melakukan pemulihan dari cadangan ke dalam cache baru.
3. Perbarui titik akhir dalam aplikasi Anda untuk titik akhir klaster baru. Untuk informasi selengkapnya, lihat Menemukan titik akhir koneksi di ElastiCache.

Resharding online untuk Valkey atau Redis OSS (mode cluster diaktifkan)

Dengan menggunakan resharding online dan shard rebalancing dengan ElastiCache Valkey 7.2 atau yang lebih baru, atau Redis OSS versi 3.2.10 atau yang lebih baru, Anda dapat menskalakan cluster Valkey atau Redis OSS (mode cluster enabled) secara dinamis tanpa downtime. Pendekatan ini berarti bahwa klaster Anda dapat terus melayani permintaan bahkan saat penskalaan atau penyeimbangan ulang sedang dalam proses.

Anda dapat melakukan tindakan berikut:

- Tingkatkan kapasitas baca dan tulis dengan menambahkan pecahan (grup simpul) ke cluster Valkey atau Redis OSS (mode cluster diaktifkan) Anda (grup replikasi).

  Jika Anda menambahkan satu atau beberapa serpihan ke grup replikasi, simpul di setiap serpihan baru memiliki jumlah sama dengan jumlah simpul dalam serpihan terkecil yang ada.
- Skala - Kurangi kapasitas baca dan tulis, dan dengan demikian biayanya, dengan menghapus pecahan dari cluster Valkey atau Redis OSS (mode cluster enabled) Anda.
- Rebalance — Pindahkan ruang kunci di antara pecahan di cluster Valkey atau Redis OSS (mode cluster enabled) Anda sehingga mereka didistribusikan secara merata di antara pecahan mungkin.

Anda tidak dapat melakukan hal berikut:

- Mengonfigurasi serpihan secara independen:

  Anda tidak dapat menentukan ruang kunci untuk serpihan secara independen. Untuk melakukan tindakan ini, Anda harus menggunakan proses offline.

Saat ini, batasan berikut berlaku untuk resharding dan rebalancing ElastiCache online:

- Proses ini membutuhkan Valkey 7.2 dan yang lebih baru atau Redis OSS 3.2.10 atau yang lebih baru. Untuk informasi selengkapnya tentang cara meningkatkan versi mesin, lihat Manajemen Versi untuk ElastiCache.
- Terdapat batasan dengan slot atau ruang kunci dan item besar:

  Jika salah satu kunci dalam serpihan berisi item besar, kunci tersebut tidak dimigrasikan ke serpihan baru ketika penskalaan ke luar atau penyeimbangan ulang dilakukan. Fungsi ini dapat menghasilkan serpihan yang tidak seimbang.

  Jika salah satu kunci dalam serpihan berisi item besar (item yang lebih besar dari 256 MB setelah serialisasi), serpihan tersebut tidak dihapus ketika penskalaan ke dalam dilakukan. Fungsi ini dapat mengakibatkan beberapa serpihan tidak dihapus.
- Ketika menskalakan ke luar, jumlah simpul dalam serpihan baru sama dengan jumlah simpul dalam serpihan terkecil yang ada.
- Ketika menskalakan ke luar, setiap tag yang umum untuk semua serpihan yang ada akan disalin ke serpihan baru.
- Saat menskalakan cluster Global Data Store, tidak ElastiCache akan secara otomatis mereplikasi Fungsi dari salah satu node yang ada ke node baru. Sebaiknya muat Fungsi Anda di serpihan baru setelah menskalakan klaster ke luar sehingga setiap serpihan memiliki fungsi yang sama.

> **Note**
>
> ElastiCache Untuk Valkey 7.2 dan di atasnya, dan ElastiCache untuk Redis OSS versi 7 ke atas: Saat menskalakan cluster Anda, ElastiCache akan secara otomatis mereplikasi Fungsi yang dimuat di salah satu node yang ada (dipilih secara acak) ke node baru. Jika aplikasi Anda menggunakan Functions, kami sarankan memuat semua fungsi Anda ke semua pecahan sebelum melakukan scaling out sehingga cluster Anda tidak berakhir dengan definisi fungsi yang berbeda pada pecahan yang berbeda.

Untuk informasi selengkapnya, lihat Perubahan ukuran klaster online.

Anda dapat menskalakan atau menyeimbangkan kembali cluster Valkey atau Redis OSS (mode cluster enabled) secara horizontal menggunakan AWS Management Console, the, dan API. AWS CLI ElastiCache

Menambahkan serpihan dengan resharding online

Anda dapat menambahkan pecahan ke cluster Valkey atau Redis OSS (mode cluster enabled) menggunakan AWS Management Console,, AWS CLI atau API. ElastiCache Saat Anda menambahkan pecahan ke cluster Valkey atau Redis OSS (mode cluster enabled), tag apa pun pada pecahan yang ada disalin ke pecahan baru.

Menambahkan serpihan (Konsol)

Anda dapat menggunakan AWS Management Console untuk menambahkan satu atau lebih pecahan ke cluster Valkey atau Redis OSS (mode cluster enabled) Anda. Prosedur berikut menjelaskan prosesnya.

Untuk menambahkan pecahan ke klaster Valkey atau Redis OSS (mode cluster diaktifkan)

1. Buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cluster Valkey atau cluster Redis OSS.
3. Cari dan pilih nama, bukan kotak di sebelah kiri nama cluster, dari cluster Valkey atau Redis OSS (mode cluster enabled) yang ingin Anda tambahkan pecahan.

   > **Tip**
   >
   > Valkey atau Redis OSS (mode cluster diaktifkan) menunjukkan Clustered Valkey atau Clustered Redis OSS di kolom Mode

4. Pilih Tambahkan serpihan.
   a. Untuk Jumlah serpihan yang akan ditambahkan, pilih jumlah serpihan yang ingin Anda tambahkan ke klaster ini.
   b. Untuk Zona Ketersediaan, pilih Tidak ada preferensi atau Tentukan zona ketersediaan.
   c. Jika Anda memilih Tentukan zona ketersediaan, untuk setiap simpul dalam setiap serpihan, pilih Zona Ketersediaan simpul dari daftar Zona Ketersediaan.
   d. Pilih Tambahkan.

Menambahkan serpihan (AWS CLI)

Proses berikut menjelaskan cara mengkonfigurasi ulang pecahan di cluster Valkey atau Redis OSS (mode cluster enabled) Anda dengan menambahkan pecahan menggunakan file. AWS CLI

Gunakan parameter berikut dengan `modify-replication-group-shard-configuration`.

Parameter

- `--apply-immediately` – Wajib. Menentukan operasi konfigurasi ulang serpihan yang akan segera dimulai.
- `--replication-group-id` – Wajib. Menentukan grup replikasi (klaster) mana tempat operasi konfigurasi ulang serpihan akan dilakukan.
- `--node-group-count` – Wajib. Menentukan jumlah serpihan (grup simpul) yang ada ketika operasi selesai. Saat menambahkan serpihan, nilai `--node-group-count` harus lebih besar dari jumlah serpihan saat ini.

  Jika perlu, Anda dapat menentukan Zona Ketersediaan untuk setiap simpul dalam grup replikasi menggunakan `--resharding-configuration`.
- `--resharding-configuration` – Opsional. Daftar Zona Ketersediaan pilihan untuk setiap simpul di setiap serpihan dalam grup replikasi. Gunakan parameter ini hanya jika nilai `--node-group-count` lebih besar dari jumlah serpihan saat ini. Jika parameter ini dihilangkan saat menambahkan pecahan, Amazon ElastiCache memilih Availability Zones untuk node baru.

Contoh berikut mengkonfigurasi ulang ruang kunci di atas empat pecahan dalam klaster Valkey atau Redis OSS (mode cluster enabled) bernama. `my-cluster` Contoh ini juga menentukan Zona Ketersediaan untuk setiap simpul di setiap serpihan. Operasi segera dimulai.

Example - Menambahkan Serpihan

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group-shard-configuration \
    --replication-group-id my-cluster \
    --node-group-count 4 \
    --resharding-configuration \
        "PreferredAvailabilityZones=us-east-2a,us-east-2c" \
        "PreferredAvailabilityZones=us-east-2b,us-east-2a" \
        "PreferredAvailabilityZones=us-east-2c,us-east-2d" \
        "PreferredAvailabilityZones=us-east-2d,us-east-2c" \
    --apply-immediately
```

Untuk Windows:

```
aws elasticache modify-replication-group-shard-configuration ^
    --replication-group-id my-cluster ^
    --node-group-count 4 ^
    --resharding-configuration ^
        "PreferredAvailabilityZones=us-east-2a,us-east-2c" ^
        "PreferredAvailabilityZones=us-east-2b,us-east-2a" ^
        "PreferredAvailabilityZones=us-east-2c,us-east-2d" ^
        "PreferredAvailabilityZones=us-east-2d,us-east-2c" ^
    --apply-immediately
```

Untuk informasi selengkapnya, lihat modify-replication-group-shard-konfigurasi dalam AWS CLI dokumentasi.

Menambahkan pecahan (ElastiCache API)

Anda dapat menggunakan ElastiCache API untuk mengkonfigurasi ulang pecahan di klaster Valkey atau Redis OSS (mode cluster enabled) Anda secara online dengan menggunakan operasi. `ModifyReplicationGroupShardConfiguration`

Gunakan parameter berikut dengan `ModifyReplicationGroupShardConfiguration`.

Parameter

- `ApplyImmediately=true` – Wajib. Menentukan operasi konfigurasi ulang serpihan yang akan segera dimulai.
- `ReplicationGroupId` – Wajib. Menentukan grup replikasi (klaster) mana tempat operasi konfigurasi ulang serpihan akan dilakukan.
- `NodeGroupCount` – Wajib. Menentukan jumlah serpihan (grup simpul) yang ada ketika operasi selesai. Saat menambahkan serpihan, nilai `NodeGroupCount` harus lebih besar dari jumlah serpihan saat ini.

  Jika perlu, Anda dapat menentukan Zona Ketersediaan untuk setiap simpul dalam grup replikasi menggunakan `ReshardingConfiguration`.
- `ReshardingConfiguration` – Opsional. Daftar Zona Ketersediaan pilihan untuk setiap simpul di setiap serpihan dalam grup replikasi. Gunakan parameter ini hanya jika nilai `NodeGroupCount` lebih besar dari jumlah serpihan saat ini. Jika parameter ini dihilangkan saat menambahkan pecahan, Amazon ElastiCache memilih Availability Zones untuk node baru.

Proses berikut menjelaskan cara mengkonfigurasi ulang pecahan di klaster Valkey atau Redis OSS (mode cluster enabled) Anda dengan menambahkan pecahan menggunakan API. ElastiCache

Example - Menambahkan Serpihan

Contoh berikut menambahkan grup node ke cluster Valkey atau Redis OSS (mode cluster enabled)`my-cluster`, sehingga ada total empat kelompok node ketika operasi selesai. Contoh ini juga menentukan Zona Ketersediaan untuk setiap simpul di setiap serpihan. Operasi segera dimulai.

```
https://elasticache.us-east-2.amazonaws.com/
    ?Action=ModifyReplicationGroupShardConfiguration
    &ApplyImmediately=true
    &NodeGroupCount=4
    &ReplicationGroupId=my-cluster

 &ReshardingConfiguration.ReshardingConfiguration.1.PreferredAavailabilityZones.AvailabilityZone
east-2a

 &ReshardingConfiguration.ReshardingConfiguration.1.PreferredAvailabilityZones.AvailabilityZone
east-2c

 &ReshardingConfiguration.ReshardingConfiguration.2.PreferredAvailabilityZones.AvailabilityZone
east-2b

 &ReshardingConfiguration.ReshardingConfiguration.2.PreferredAvailabilityZones.AvailabilityZone
east-2a

 &ReshardingConfiguration.ReshardingConfiguration.3.PreferredAvailabilityZones.AvailabilityZone
east-2c

 &ReshardingConfiguration.ReshardingConfiguration.3.PreferredAvailabilityZones.AvailabilityZone
east-2d

 &ReshardingConfiguration.ReshardingConfiguration.4.PreferredAvailabilityZones.AvailabilityZone
east-2d

 &ReshardingConfiguration.ReshardingConfiguration.4.PreferredAvailabilityZones.AvailabilityZone
east-2c
    &Version=2015-02-02
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20171002T192317Z
    &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya, lihat ModifyReplicationGroupShardConfigurationdi Referensi ElastiCache API.

Menghapus serpihan dengan resharding online

Anda dapat menghapus pecahan dari cluster Valkey atau Redis OSS (mode cluster enabled) menggunakan AWS Management Console,, AWS CLI atau API. ElastiCache

Topik

- Menghapus serpihan (Konsol)
- Menghapus serpihan (AWS CLI)
- Menghapus serpihan (API ElastiCache)

Menghapus serpihan (Konsol)

Proses berikut menjelaskan cara mengkonfigurasi ulang pecahan di cluster Valkey atau Redis OSS (mode cluster enabled) Anda dengan menghapus pecahan menggunakan file. AWS Management Console

Sebelum menghapus grup node (pecahan) dari grup replikasi Anda, ElastiCache pastikan semua data Anda sesuai dengan pecahan yang tersisa. Jika data tercakup, serpihan yang ditentukan akan dihapus dari grup replikasi seperti yang diminta. Jika data tidak tercakup di grup simpul yang tersisa, proses akan dihentikan dan grup replikasi memiliki konfigurasi grup simpul yang sama seperti sebelum permintaan dibuat.

Anda dapat menggunakan AWS Management Console untuk menghapus satu atau lebih pecahan dari cluster Valkey atau Redis OSS (mode cluster enabled) Anda. Anda tidak dapat menghapus semua serpihan dalam grup replikasi. Sebaliknya, Anda harus menghapus grup replikasi. Untuk informasi selengkapnya, lihat Menghapus grup replikasi. Prosedur berikut menjelaskan proses untuk menghapus satu atau lebih serpihan.

Untuk menghapus pecahan dari cluster Valkey atau Redis OSS (mode cluster diaktifkan)

1. Buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cluster Valkey atau cluster Redis OSS.
3. Cari dan pilih nama, bukan kotak di sebelah kiri nama cluster, dari cluster Valkey atau Redis OSS (mode cluster diaktifkan) yang ingin Anda hapus dari pecahan.

> **Tip**
>
> Cluster Valkey atau Redis OSS (mode cluster enabled) memiliki nilai 1 atau lebih besar di kolom Shards.

4. Dari daftar serpihan, pilih kotak di sebelah kiri nama setiap serpihan yang ingin Anda hapus.
5. Pilih Hapus serpihan.

Menghapus serpihan (AWS CLI)

Proses berikut menjelaskan cara mengkonfigurasi ulang pecahan di cluster Valkey atau Redis OSS (mode cluster enabled) Anda dengan menghapus pecahan menggunakan file. AWS CLI

> **Important**
>
> Sebelum menghapus grup node (pecahan) dari grup replikasi Anda, ElastiCache pastikan semua data Anda sesuai dengan pecahan yang tersisa. Jika data tercakup, serpihan yang ditentukan (`--node-groups-to-remove`) akan dihapus dari grup replikasi seperti yang diminta dan ruang kunci terkait akan dipetakan ke serpihan yang tersisa. Jika data tidak tercakup di grup simpul yang tersisa, proses akan dihentikan dan grup replikasi yang tersisa memiliki konfigurasi grup simpul yang sama seperti sebelum permintaan dibuat.

Anda dapat menggunakan AWS CLI untuk menghapus satu atau lebih pecahan dari cluster Valkey atau Redis OSS (mode cluster enabled) Anda. Anda tidak dapat menghapus semua serpihan dalam grup replikasi. Sebaliknya, Anda harus menghapus grup replikasi. Untuk informasi selengkapnya, lihat Menghapus grup replikasi.

Gunakan parameter berikut dengan `modify-replication-group-shard-configuration`.

Parameter

- `--apply-immediately` – Wajib. Menentukan operasi konfigurasi ulang serpihan yang akan segera dimulai.
- `--replication-group-id` – Wajib. Menentukan grup replikasi (klaster) mana tempat operasi konfigurasi ulang serpihan akan dilakukan.

- `--node-group-count` – Wajib. Menentukan jumlah serpihan (grup simpul) yang ada ketika operasi selesai. Saat menghapus serpihan, nilai `--node-group-count` harus kurang dari jumlah serpihan saat ini.

- `--node-groups-to-remove` – Wajib jika `--node-group-count` kurang dari jumlah grup simpul (serpihan) saat ini. Daftar pecahan (grup simpul) IDs untuk dihapus dari grup replikasi.

Prosedur berikut menjelaskan proses untuk menghapus satu atau beberapa serpihan.

Example - Menghapus Serpihan

Contoh berikut menghapus dua kelompok node dari Valkey atau Redis OSS (cluster mode enabled) clustermy`-cluster`, sehingga ada total dua kelompok node ketika operasi selesai. Ruang Kunci dari serpihan yang dihapus didistribusikan secara merata ke serpihan yang tersisa.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group-shard-configuration \
    --replication-group-id my-cluster \
    --node-group-count 2 \
    --node-groups-to-remove "0002" "0003" \
    --apply-immediately
```

Untuk Windows:

```
aws elasticache modify-replication-group-shard-configuration ^
    --replication-group-id my-cluster ^
    --node-group-count 2 ^
    --node-groups-to-remove "0002" "0003" ^
    --apply-immediately
```

Menghapus serpihan (API ElastiCache)

Anda dapat menggunakan ElastiCache API untuk mengkonfigurasi ulang pecahan di klaster Valkey atau Redis OSS (mode cluster enabled) Anda secara online dengan menggunakan operasi. `ModifyReplicationGroupShardConfiguration`

Proses berikut menjelaskan cara mengkonfigurasi ulang pecahan di klaster Valkey atau Redis OSS (mode cluster enabled) Anda dengan menghapus pecahan menggunakan API. ElastiCache

> **Important**
>
> Sebelum menghapus grup node (pecahan) dari grup replikasi Anda, ElastiCache pastikan semua data Anda sesuai dengan pecahan yang tersisa. Jika data tercakup, serpihan yang ditentukan (`NodeGroupsToRemove`) akan dihapus dari grup replikasi seperti yang diminta dan ruang kunci terkait akan dipetakan ke serpihan yang tersisa. Jika data tidak tercakup di grup simpul yang tersisa, proses akan dihentikan dan grup replikasi yang tersisa memiliki konfigurasi grup simpul yang sama seperti sebelum permintaan dibuat.

Anda dapat menggunakan ElastiCache API untuk menghapus satu atau beberapa pecahan dari klaster Valkey atau Redis OSS (mode cluster enabled). Anda tidak dapat menghapus semua serpihan dalam grup replikasi. Sebaliknya, Anda harus menghapus grup replikasi. Untuk informasi selengkapnya, lihat Menghapus grup replikasi.

Gunakan parameter berikut dengan `ModifyReplicationGroupShardConfiguration`.

Parameter

- `ApplyImmediately=true` – Wajib. Menentukan operasi konfigurasi ulang serpihan yang akan segera dimulai.
- `ReplicationGroupId` – Wajib. Menentukan grup replikasi (klaster) mana tempat operasi konfigurasi ulang serpihan akan dilakukan.
- `NodeGroupCount` – Wajib. Menentukan jumlah serpihan (grup simpul) yang ada ketika operasi selesai. Saat menghapus serpihan, nilai `NodeGroupCount` harus kurang dari jumlah serpihan saat ini.
- `NodeGroupsToRemove` – Wajib jika `--node-group-count` kurang dari jumlah grup simpul (serpihan) saat ini. Daftar pecahan (grup simpul) IDs untuk dihapus dari grup replikasi.

Prosedur berikut menjelaskan proses untuk menghapus satu atau beberapa serpihan.

Example - Menghapus Serpihan

Contoh berikut menghapus dua kelompok node dari Valkey atau Redis OSS (cluster mode enabled) clustermy`-cluster`, sehingga ada total dua kelompok node ketika operasi selesai. Ruang Kunci dari serpihan yang dihapus didistribusikan secara merata ke serpihan yang tersisa.

```
https://elasticache.us-east-2.amazonaws.com/
```

```
    ?Action=ModifyReplicationGroupShardConfiguration
    &ApplyImmediately=true
    &NodeGroupCount=2
    &ReplicationGroupId=my-cluster
    &NodeGroupsToRemove.member.1=0002
    &NodeGroupsToRemove.member.2=0003
    &Version=2015-02-02
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20171002T192317Z
    &X-Amz-Credential=<credential>
```

Penyeimbangan ulang serpihan secara online

Anda dapat menyeimbangkan kembali pecahan di cluster Valkey atau Redis OSS (mode cluster enabled) menggunakan,, atau API. AWS Management Console AWS CLI ElastiCache

Topik

- Penyeimbangan Ulang Serpihan Secara Online (Konsol)
- Penyeimbangan ulang serpihan secara online (AWS CLI)
- Penyeimbangan kembali pecahan online (API) ElastiCache

Penyeimbangan Ulang Serpihan Secara Online (Konsol)

Proses berikut menjelaskan cara mengkonfigurasi ulang pecahan di cluster Valkey atau Redis OSS (mode cluster enabled) Anda dengan menyeimbangkan kembali pecahan menggunakan file. AWS Management Console

Untuk menyeimbangkan kembali ruang kunci di antara pecahan pada klaster Valkey atau Redis OSS (mode cluster diaktifkan)

1. Buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cluster Valkey atau cluster Redis OSS.
3. Pilih nama, bukan kotak di sebelah kiri nama, dari cluster Valkey atau Redis OSS (mode cluster enabled) yang ingin Anda seimbangkan kembali.

> **Tip**
>
> Cluster Valkey atau Redis OSS (mode cluster enabled) memiliki nilai 1 atau lebih besar di kolom Shards.

4. Pilih Seimbangkan ulang.
5. Saat diminta, pilih Seimbangkan ulang. Anda mungkin melihat pesan yang mirip dengan yang ini:*Slots in the replication group are uniformly distributed. Nothing to do. (Service: AmazonElastiCache; Status Code: 400; Error Code: InvalidReplicationGroupState; Request ID: 2246cebd-9721-11e7-8d5b-e1b0f086c8cf)*. Jika melihat pesan di atas, pilih Batalkan.

Penyeimbangan ulang serpihan secara online (AWS CLI)

Gunakan parameter berikut dengan `modify-replication-group-shard-configuration`.

Parameter

- `-apply-immediately` – Wajib. Menentukan operasi konfigurasi ulang serpihan yang akan segera dimulai.
- `--replication-group-id` – Wajib. Menentukan grup replikasi (klaster) mana tempat operasi konfigurasi ulang serpihan akan dilakukan.
- `--node-group-count` – Wajib. Untuk menyeimbangkan ulang ruang kunci di semua serpihan dalam klaster, nilai ini harus sama dengan jumlah serpihan saat ini.

Proses berikut menjelaskan cara mengkonfigurasi ulang pecahan di cluster Valkey atau Redis OSS (mode cluster enabled) Anda dengan menyeimbangkan kembali pecahan menggunakan file. AWS CLI

Example - Menyeimbangkan ulang serpihan dalam Klaster

Contoh berikut menyeimbangkan kembali slot di cluster Valkey atau Redis OSS (mode cluster enabled) `my-cluster` sehingga slot didistribusikan sama mungkin. Nilai dari `--node-group-count` (4) adalah jumlah serpihan saat ini dalam klaster.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group-shard-configuration \
    --replication-group-id my-cluster \
    --node-group-count 4 \
    --apply-immediately
```

Untuk Windows:

```
aws elasticache modify-replication-group-shard-configuration ^
    --replication-group-id my-cluster ^
    --node-group-count 4 ^
    --apply-immediately
```

Penyeimbangan kembali pecahan online (API) ElastiCache

Anda dapat menggunakan ElastiCache API untuk mengkonfigurasi ulang pecahan di klaster Valkey atau Redis OSS (mode cluster enabled) Anda secara online dengan menggunakan operasi. `ModifyReplicationGroupShardConfiguration`

Gunakan parameter berikut dengan `ModifyReplicationGroupShardConfiguration`.

Parameter

- `ApplyImmediately=true` – Wajib. Menentukan operasi konfigurasi ulang serpihan yang akan segera dimulai.
- `ReplicationGroupId` – Wajib. Menentukan grup replikasi (klaster) mana tempat operasi konfigurasi ulang serpihan akan dilakukan.
- `NodeGroupCount` – Wajib. Untuk menyeimbangkan ulang ruang kunci di semua serpihan dalam klaster, nilai ini harus sama dengan jumlah serpihan saat ini.

Proses berikut menjelaskan cara mengkonfigurasi ulang pecahan di klaster Valkey atau Redis OSS (mode cluster enabled) Anda dengan menyeimbangkan kembali pecahan menggunakan API. ElastiCache

Example - Menyeimbangkan Ulang Klaster

Contoh berikut menyeimbangkan kembali slot di cluster Valkey atau Redis OSS (mode cluster enabled) `my-cluster` sehingga slot didistribusikan sama mungkin. Nilai dari `NodeGroupCount` (`4`) adalah jumlah serpihan saat ini dalam klaster.

```
https://elasticache.us-east-2.amazonaws.com/
    ?Action=ModifyReplicationGroupShardConfiguration
    &ApplyImmediately=true
    &NodeGroupCount=4
    &ReplicationGroupId=my-cluster
    &Version=2015-02-02
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20171002T192317Z
    &X-Amz-Credential=<credential>
```

Penskalaan vertikal online dengan mengubah jenis simpul

Dengan menggunakan penskalaan vertikal online dengan Valkey versi 7.2 atau yang lebih baru, atau Redis OSS versi 3.2.10 atau yang lebih baru, Anda dapat menskalakan cluster Valkey atau Redis OSS Anda secara dinamis dengan waktu henti minimal. Ini memungkinkan cluster Valkey atau Redis OSS Anda untuk melayani permintaan bahkan saat penskalaan.

> **Note**
>
> Penskalaan tidak didukung antara klaster tingkatan data (misalnya, klaster yang menggunakan jenis simpul r6gd) dan klaster yang tidak menggunakan tingkatan data (misalnya, klaster yang menggunakan jenis simpul r6g). Untuk informasi selengkapnya, lihat Tingkatan data di ElastiCache.

Anda dapat melakukan tindakan berikut:

- Tingkatkan kapasitas baca dan tulis dengan menyesuaikan tipe node cluster Valkey atau Redis OSS Anda untuk menggunakan tipe node yang lebih besar.

  ElastiCache mengubah ukuran cluster Anda secara dinamis sambil tetap online dan melayani permintaan.
- Menurunkan skala – Mengurangi kapasitas baca dan tulis dengan menyesuaikan jenis simpul agar menggunakan simpul yang lebih kecil. Sekali lagi, mengubah ukuran cluster Anda ElastiCache secara dinamis sambil tetap online dan melayani permintaan. Dalam hal ini, Anda mengurangi biaya dengan menurunkan ukuran simpul.

> **Note**
>
> Proses penaikan dan penurunan skala bergantung pada pembuatan klaster dengan jenis simpul yang baru dipilih dan sinkronisasi simpul baru dengan yang sebelumnya. Untuk memastikan up/down aliran skala yang lancar, lakukan hal berikut:
>
> - Pastikan Anda memiliki kapasitas ENI (Elastic Network Interface) yang cukup. Jika menurunkan skala, pastikan simpul yang lebih kecil memiliki memori yang cukup untuk menyerap lalu lintas yang diperkirakan.
>
> Untuk praktik terbaik tentang manajemen memori, lihat Mengelola memori cadangan untuk Valkey dan Redis OSS.
> - Sementara proses penskalaan vertikal dirancang untuk tetap sepenuhnya online, hal ini tidak bergantung pada sinkronisasi data antara simpul lama dan simpul baru. Sebaiknya mulai penaikan/penurunan skala selama jam ketika lalu lintas data diperkirakan berada pada tingkat minimum.
> - Uji perilaku aplikasi Anda selama penskalaan ke dalam di lingkungan persiapan, jika memungkinkan.

Daftar Isi

Penskalaan ke atas secara online

Topik

- Meningkatkan klaster cache Valkey atau Redis OSS (API) ElastiCache

Meningkatkan klaster cache Valkey atau Redis OSS (Konsol)

Prosedur berikut menjelaskan cara meningkatkan klaster Valkey atau Redis OSS menggunakan Management Console. ElastiCache Selama proses ini, klaster Anda akan terus melayani permintaan dengan waktu henti minimal.

Untuk meningkatkan cluster Valkey atau Redis OSS (konsol)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cluster Valkey atau cluster Redis OSS.
3. Pada daftar klaster, pilih klaster tersebut.
4. Pilih Ubah.
5. Di wizard Ubah Klaster:
   - Pilih jenis simpul sebagai tujuan penskalaan dari daftar Jenis simpul. Untuk menaikkan skala, pilih jenis simpul yang lebih besar dari simpul saat ini.
6. Jika Anda ingin segera melakukan proses peningkatan skala, pilih kotak Terapkan segera. Jika kotak centang Terapkan segera tidak dipilih, proses peningkatan skala akan dilakukan selama periode pemeliharaan berikutnya dari klaster ini.
7. Pilih Ubah.

   Jika Anda memilih Terapkan segera pada langkah sebelumnya, status klaster berubah ke mengubah. Ketika status berubah ke tersedia, pengubahan selesai dan Anda dapat mulai menggunakan klaster baru tersebut.

Meningkatkan klaster cache Valkey atau Redis OSS ()AWS CLI

Prosedur berikut menjelaskan cara meningkatkan cluster cache Valkey atau Redis OSS menggunakan. AWS CLI Selama proses ini, klaster Anda akan terus melayani permintaan dengan waktu henti minimal.

Untuk meningkatkan cluster cache Valkey atau Redis OSS ()AWS CLI

1. Tentukan jenis node yang dapat Anda tingkatkan dengan menjalankan AWS CLI `list-allowed-node-type-modifications` perintah dengan parameter berikut.

Untuk Linux, macOS, atau Unix:

```
aws elasticache list-allowed-node-type-modifications \
     --replication-group-id my-replication-group-id
```

Untuk Windows:

```
aws elasticache list-allowed-node-type-modifications ^
     --replication-group-id my-replication-group-id
```

Output dari perintah di atas terlihat seperti berikut (format JSON).

```
{
     "ScaleUpModifications": [
         "cache.m3.2xlarge",
         "cache.m3.large",
         "cache.m3.xlarge",
         "cache.m4.10xlarge",
         "cache.m4.2xlarge",
         "cache.m4.4xlarge",
         "cache.m4.large",
         "cache.m4.xlarge",
         "cache.r3.2xlarge",
         "cache.r3.4xlarge",
         "cache.r3.8xlarge",
         "cache.r3.large",
         "cache.r3.xlarge"
     ]
        "ScaleDownModifications": [
         "cache.t2.micro",
         "cache.t2.small ",
         "cache.t2.medium",
         "cache.t1.small "
     ],
}
```

Untuk informasi selengkapnya, lihat list-allowed-node-type-modifications dalam Referensi AWS CLI .

2. Ubah grup replikasi Anda untuk meningkatkan ke tipe node baru yang lebih besar, menggunakan AWS CLI `modify-replication-group` perintah dan parameter berikut.

- `--replication-group-id` – Nama grup replikasi yang menjadi tujuan penaikan skala.
- `--cache-node-type` – Jenis simpul baru yang Anda inginkan untuk menskalakan klaster cache. Nilai ini harus berupa salah satu dari jenis simpul yang dihasilkan oleh perintah `list-allowed-node-type-modifications` di langkah 1.
- `--cache-parameter-group-name` – [Opsional] Gunakan parameter ini jika Anda menggunakan `reserved-memory` untuk mengelola memori cadangan klaster. Tentukan grup parameter cache kustom yang mencadangkan jumlah memori yang sesuai untuk jenis simpul yang baru. Jika menggunakan `reserved-memory-percent`, Anda dapat menghilangkan parameter ini.
- `--apply-immediately` – Menyebabkan operasi penaikan skala segera diterapkan. Untuk menunda proses penaikan skala ke periode pemeliharaan berikutnya untuk klaster, gunakan parameter `--no-apply-immediately`.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group  \
     --replication-group-id my-redis-cluster \
     --cache-node-type cache.m3.xlarge \
     --apply-immediately
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
     --replication-group-id my-redis-cluster ^
     --cache-node-type cache.m3.xlarge ^
     --apply-immediately
```

Output dari perintah di atas terlihat seperti berikut (format JSON).

```
{
  "ReplicationGroup": {
        "Status": "modifying",
        "Description": "my-redis-cluster",
        "NodeGroups": [
            {
                "Status": "modifying",
```

```
                "Slots": "0-16383",
                "NodeGroupId": "0001",
                "NodeGroupMembers": [
                    {
                        "PreferredAvailabilityZone": "us-east-1f",
                        "CacheNodeId": "0001",
                        "CacheClusterId": "my-redis-cluster-0001-001"
                    },
                    {
                        "PreferredAvailabilityZone": "us-east-1d",
                        "CacheNodeId": "0001",
                        "CacheClusterId": "my-redis-cluster-0001-002"
                    }
                ]
            }
        ],
        "ConfigurationEndpoint": {
            "Port": 6379,
            "Address": "my-redis-
cluster.r7gdfi.clustercfg.use1.cache.amazonaws.com"
        },
        "ClusterEnabled": true,
        "ReplicationGroupId": "my-redis-cluster",
        "SnapshotRetentionLimit": 1,
        "AutomaticFailover": "enabled",
        "SnapshotWindow": "07:30-08:30",
        "MemberClusters": [
            "my-redis-cluster-0001-001",
            "my-redis-cluster-0001-002"
        ],
        "CacheNodeType": "cache.m3.xlarge",
         "DataTiering": "disabled"
        "PendingModifiedValues": {}
    }
}
```

Untuk informasi selengkapnya, lihat modify-replication-group dalam Referensi AWS CLI .

3. Jika Anda menggunakan`--apply-immediately`, periksa status cluster cache menggunakan AWS CLI `describe-cache-clusters` perintah dengan parameter berikut. Ketika status berubah ke tersedia, Anda dapat mulai menggunakan simpul klaster cache baru yang lebih besar.

Meningkatkan klaster cache Valkey atau Redis OSS (API) ElastiCache

Proses berikut menskalakan cluster cache Anda dari tipe node saat ini ke tipe node baru yang lebih besar menggunakan ElastiCache API. Selama proses ini, ElastiCache update entri DNS sehingga mereka menunjuk ke node baru. Karena hal ini, Anda tidak perlu memperbarui titik akhir dalam aplikasi Anda. Untuk Valkey 7.2 dan di atasnya Redis OSS 5.0.5 dan yang lebih baru, Anda dapat menskalakan cluster yang diaktifkan failover otomatis sementara cluster terus online dan melayani permintaan yang masuk. Pada versi Redis OSS 4.0.10 dan di bawahnya, Anda mungkin melihat gangguan singkat membaca dan menulis pada versi sebelumnya dari node utama saat entri DNS diperbarui..

Jumlah waktu yang dibutuhkan untuk menaikkan skala ke jenis simpul yang lebih besar bervariasi, bergantung pada jenis simpul dan jumlah data dalam klaster cache Anda saat ini.

Untuk meningkatkan Valkey atau Redis OSS Cache Cluster (API) ElastiCache

1. Tentukan tipe node mana yang dapat Anda tingkatkan untuk menggunakan `ListAllowedNodeTypeModifications` aksi ElastiCache API dengan parameter berikut.

   - `ReplicationGroupId` – nama grup replikasi. Gunakan parameter ini untuk mendeskripsikan grup replikasi tertentu, bukan semua grup replikasi.

   ```
   https://elasticache.us-west-2.amazonaws.com/
       ?Action=ListAllowedNodeTypeModifications
       &ReplicationGroupId=MyReplGroup
       &Version=2015-02-02
       &SignatureVersion=4
       &SignatureMethod=HmacSHA256
       &Timestamp=20150202T192317Z
       &X-Amz-Credential=<credential>
   ```

   Untuk informasi selengkapnya, lihat ListAllowedNodeTypeModifications di Referensi Amazon ElastiCache API.

2. Skalakan grup replikasi Anda saat ini hingga tipe node baru menggunakan aksi `ModifyReplicationGroup` ElastiCache API dan dengan parameter berikut.

   - `ReplicationGroupId` – nama grup replikasi.

- `CacheNodeType` – jenis simpul baru yang lebih besar dari klaster cache dalam grup replikasi ini. Nilai ini harus menjadi salah satu jenis instance yang dikembalikan oleh `ListAllowedNodeTypeModifications` tindakan pada langkah sebelumnya.
- `CacheParameterGroupName` – [Opsional] Gunakan parameter ini jika Anda menggunakan `reserved-memory` untuk mengelola memori cadangan klaster. Tentukan grup parameter cache kustom yang mencadangkan jumlah memori yang sesuai untuk jenis simpul yang baru. Jika menggunakan `reserved-memory-percent`, Anda dapat menghilangkan parameter ini.
- `ApplyImmediately` – Tetapkan ke `true` agar proses penaikan skala segera diterapkan. Untuk menunda proses penaikan skala ke periode pemeliharaan berikutnya, gunakan `ApplyImmediately=false`.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=ModifyReplicationGroup
    &ApplyImmediately=true
    &CacheNodeType=cache.m3.2xlarge
    &CacheParameterGroupName=redis32-m3-2xl
    &ReplicationGroupId=myReplGroup
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &Version=2014-12-01
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Date=20141201T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20141201T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Untuk informasi selengkapnya, lihat ModifyReplicationGroup di Referensi Amazon ElastiCache API.

3. Jika Anda menggunakan `ApplyImmediately=true`, pantau status grup replikasi menggunakan `DescribeReplicationGroups` aksi ElastiCache API dengan parameter berikut. Ketika status berubah dari mengubah ke tersedia, Anda dapat mulai menulis ke grup replikasi baru yang telah dinaikkan skalanya.

   - `ReplicationGroupId` – nama grup replikasi. Gunakan parameter ini untuk mendeskripsikan grup replikasi tertentu, bukan semua grup replikasi.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=DescribeReplicationGroups
    &ReplicationGroupId=MyReplGroup
    &Version=2015-02-02
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20150202T192317Z
    &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya, lihat DescribeReplicationGroups di Referensi Amazon ElastiCache API.

Penskalaan ke bawah secara online

Topik

- Menskalakan cluster cache Valkey atau Redis OSS (Konsol)
- Menskalakan cluster cache Valkey atau Redis OSS ()AWS CLI
- Menskalakan cluster cache Valkey atau Redis OSS (API) ElastiCache

Menskalakan cluster cache Valkey atau Redis OSS (Konsol)

Prosedur berikut menjelaskan cara menurunkan klaster Valkey atau Redis OSS menggunakan Management Console. ElastiCache Selama proses ini, cluster Valkey atau Redis OSS Anda akan terus melayani permintaan dengan waktu henti minimal.

Untuk menurunkan kluster Valkey atau Redis OSS (konsol)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cluster Valkey atau cluster Redis OSS.
3. Dari daftar klaster, pilih klaster pilihan Anda.
4. Pilih Ubah.
5. Di wizard Ubah Klaster:

- Pilih jenis simpul sebagai tujuan penskalaan dari daftar Jenis simpul. Untuk menurunkan skala, pilih jenis simpul yang lebih kecil dari simpul saat ini. Perhatikan bahwa tidak semua jenis simpul tersedia sebagai pilihan penurunan skala.

6. Jika Anda ingin segera melakukan proses penurunan skala, pilih kotak Terapkan segera. Jika kotak Terapkan segera tidak dipilih, proses penurunan skala dilakukan selama periode pemeliharaan berikutnya dari klaster ini.
7. Pilih Ubah.

   Jika Anda memilih Terapkan segera pada langkah sebelumnya, status klaster berubah ke mengubah. Ketika status berubah ke tersedia, pengubahan selesai dan Anda dapat mulai menggunakan klaster baru tersebut.

Menskalakan cluster cache Valkey atau Redis OSS ()AWS CLI

Prosedur berikut menjelaskan cara menurunkan cluster cache Valkey atau Redis OSS menggunakan. AWS CLI Selama proses ini, klaster Anda akan terus melayani permintaan dengan waktu henti minimal.

Untuk mengurangi cluster cache Valkey atau Redis OSS ()AWS CLI

1. Tentukan jenis node yang dapat Anda turunkan dengan menjalankan AWS CLI `list-allowed-node-type-modifications` perintah dengan parameter berikut.

   Untuk Linux, macOS, atau Unix:

   ```
   aws elasticache list-allowed-node-type-modifications \
        --replication-group-id my-replication-group-id
   ```

   Untuk Windows:

   ```
   aws elasticache list-allowed-node-type-modifications ^
        --replication-group-id my-replication-group-id
   ```

   Output dari perintah di atas terlihat seperti berikut (format JSON).

   ```
   {
        "ScaleUpModifications": [
            "cache.m3.2xlarge",
   ```

```
          "cache.m3.large",
          "cache.m3.xlarge",
          "cache.m4.10xlarge",
          "cache.m4.2xlarge",
          "cache.m4.4xlarge",
          "cache.m4.large",
          "cache.m4.xlarge",
          "cache.r3.2xlarge",
          "cache.r3.4xlarge",
          "cache.r3.8xlarge",
          "cache.r3.large",
          "cache.r3.xlarge"
      ]

         "ScaleDownModifications": [
          "cache.t2.micro",
          "cache.t2.small ",
          "cache.t2.medium ",
          "cache.t1.small"
      ]
}
```

Untuk informasi selengkapnya, lihat list-allowed-node-type-modifications dalam Referensi AWS CLI .

2. Ubah grup replikasi Anda untuk menurunkan ke tipe node baru yang lebih kecil, menggunakan AWS CLI `modify-replication-group` perintah dan parameter berikut.

   - `--replication-group-id` – Nama grup replikasi yang menjadi tujuan penurunan skala.
   - `--cache-node-type` – Jenis simpul baru yang Anda inginkan untuk menskalakan klaster cache. Nilai ini harus berupa salah satu dari jenis simpul yang dihasilkan oleh perintah `list-allowed-node-type-modifications` di langkah 1.
   - `--cache-parameter-group-name` – [Opsional] Gunakan parameter ini jika Anda menggunakan `reserved-memory` untuk mengelola memori cadangan klaster. Tentukan grup parameter cache kustom yang mencadangkan jumlah memori yang sesuai untuk jenis simpul yang baru. Jika menggunakan `reserved-memory-percent`, Anda dapat menghilangkan parameter ini.
   - `--apply-immediately` – Menyebabkan operasi penaikan skala segera diterapkan. Guna menunda proses penurunan skala ke periode pemeliharaan berikutnya untuk klaster, gunakan parameter `--no-apply-immediately`.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group  \
     --replication-group-id my-redis-cluster \
     --cache-node-type cache.t2.micro \
     --apply-immediately
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
     --replication-group-id my-redis-cluster ^
     --cache-node-type cache.t2.micro ^
     --apply-immediately
```

Output dari perintah di atas terlihat seperti berikut (format JSON).

```
{
  "ReplicationGroup": {
        "Status": "modifying",
        "Description": "my-redis-cluster",
        "NodeGroups": [
            {
                "Status": "modifying",
                "Slots": "0-16383",
                "NodeGroupId": "0001",
                "NodeGroupMembers": [
                    {
                        "PreferredAvailabilityZone": "us-east-1f",
                        "CacheNodeId": "0001",
                        "CacheClusterId": "my-redis-cluster-0001-001"
                    },
                    {
                        "PreferredAvailabilityZone": "us-east-1d",
                        "CacheNodeId": "0001",
                        "CacheClusterId": "my-redis-cluster-0001-002"
                    }
                ]
            }
        ],
```

```
        "ConfigurationEndpoint": {
            "Port": 6379,
            "Address": "my-redis-
cluster.r7gdfi.clustercfg.use1.cache.amazonaws.com"
        },
        "ClusterEnabled": true,
        "ReplicationGroupId": "my-redis-cluster",
        "SnapshotRetentionLimit": 1,
        "AutomaticFailover": "enabled",
        "SnapshotWindow": "07:30-08:30",
        "MemberClusters": [
            "my-redis-cluster-0001-001",
            "my-redis-cluster-0001-002"
        ],
        "CacheNodeType": "cache.t2.micro",
         "DataTiering": "disabled"
        "PendingModifiedValues": {}
    }
}
```

Untuk informasi selengkapnya, lihat modify-replication-group dalam Referensi AWS CLI .

3. Jika Anda menggunakan`--apply-immediately`, periksa status cluster cache menggunakan AWS CLI `describe-cache-clusters` perintah dengan parameter berikut. Ketika status berubah ke tersedia, Anda dapat mulai menggunakan simpul klaster cache baru yang lebih kecil.

Menskalakan cluster cache Valkey atau Redis OSS (API) ElastiCache

Proses berikut menskalakan grup replikasi Anda dari tipe node saat ini ke tipe node baru yang lebih kecil menggunakan ElastiCache API. Selama proses ini, cluster Valkey atau Redis OSS Anda akan terus melayani permintaan dengan waktu henti minimal.

Jumlah waktu yang dibutuhkan untuk menurunkan skala ke jenis simpul yang lebih kecil bervariasi, bergantung pada jenis simpul dan jumlah data dalam klaster cache Anda saat ini.

Penskalaan ke bawah (ElastiCache API)

1. Tentukan tipe node mana yang dapat Anda turunkan untuk menggunakan `ListAllowedNodeTypeModifications` aksi ElastiCache API dengan parameter berikut.

   - `ReplicationGroupId` – nama grup replikasi. Gunakan parameter ini untuk mendeskripsikan grup replikasi tertentu, bukan semua grup replikasi.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=ListAllowedNodeTypeModifications
    &ReplicationGroupId=MyReplGroup
    &Version=2015-02-02
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20150202T192317Z
    &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya, lihat ListAllowedNodeTypeModifications di Referensi Amazon ElastiCache API.

2. Skalakan grup replikasi Anda saat ini ke tipe node baru menggunakan aksi `ModifyReplicationGroup` ElastiCache API dan dengan parameter berikut.

   - `ReplicationGroupId` – nama grup replikasi.
   - `CacheNodeType` – jenis simpul baru yang lebih kecil dari klaster cache dalam grup replikasi ini. Nilai ini harus menjadi salah satu jenis instance yang dikembalikan oleh `ListAllowedNodeTypeModifications` tindakan pada langkah sebelumnya.
   - `CacheParameterGroupName` – [Opsional] Gunakan parameter ini jika Anda menggunakan `reserved-memory` untuk mengelola memori cadangan klaster. Tentukan grup parameter cache kustom yang mencadangkan jumlah memori yang sesuai untuk jenis simpul yang baru. Jika menggunakan `reserved-memory-percent`, Anda dapat menghilangkan parameter ini.
   - `ApplyImmediately` – Tetapkan ke `true` agar proses penurunan skala segera diterapkan. Untuk menunda proses penurunan skala ke periode pemeliharaan berikutnya, gunakan `ApplyImmediately=false`.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=ModifyReplicationGroup
    &ApplyImmediately=true
    &CacheNodeType=cache.t2.micro
    &CacheParameterGroupName=redis32-m3-2xl
    &ReplicationGroupId=myReplGroup
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &Version=2014-12-01
```

```
&X-Amz-Algorithm=&AWS;4-HMAC-SHA256
&X-Amz-Date=20141201T220302Z
&X-Amz-SignedHeaders=Host
&X-Amz-Expires=20141201T220302Z
&X-Amz-Credential=<credential>
&X-Amz-Signature=<signature>
```

Untuk informasi selengkapnya, lihat ModifyReplicationGroup di Referensi Amazon ElastiCache API.

Memulai dengan filter Bloom

ElastiCache mendukung struktur data filter Bloom, yang menyediakan struktur data probabilistik ruang yang efisien untuk memeriksa apakah suatu elemen adalah anggota dari suatu set. Saat menggunakan filter Bloom, positif palsu dimungkinkan — filter dapat salah menunjukkan bahwa elemen ada, meskipun elemen itu tidak ditambahkan ke himpunan. Namun, menggunakan filter Bloom akan mencegah negatif palsu — indikasi salah bahwa elemen tidak ada, meskipun elemen itu ditambahkan ke set.

Anda dapat mengatur persentase potensi positif palsu ke tingkat yang diinginkan untuk beban kerja Anda, dengan menyesuaikan tingkat fp. Anda juga dapat mengonfigurasi kapasitas (jumlah item yang dapat disimpan oleh filter Bloom), properti penskalaan dan non-penskalaan, dan banyak lagi.

Setelah Anda membuat cluster dengan versi engine yang didukung, tipe data Bloom dan perintah terkait akan tersedia secara otomatis. Tipe `bloom` data kompatibel dengan API dengan sintaks perintah filter Bloom dari pustaka klien Valkey resmi termasuk`valkey-py`,, `valkey-java` dan. `valkey-go` Anda dapat dengan mudah memigrasikan aplikasi Valkey dan Redis OSS berbasis Bloom yang ada ke dalam. ElastiCache Untuk daftar lengkap perintah lihatPerintah filter Bloom.

Metrik terkait Bloom`BloomFilterBasedCmds`,`BloomFilterBasedCmdsLatency`, dan `BloomFilterBasedCmdsECPUs` dimasukkan ke dalam CloudWatch untuk memantau penggunaan tipe data ini. Untuk informasi selengkapnya, lihat Metrik untuk Valkey dan Redis OSS.

> **Note**
>
> - Untuk menggunakan filter Bloom, Anda harus menjalankan ElastiCache Valkey 8.1 dan yang lebih baru.

- Tipe data bloom tidak kompatibel dengan RDB dengan penawaran mekar berbasis non-Valkey lainnya.

Ikhtisar tipe data filter Bloom

Filter Bloom adalah struktur data probabilistik hemat ruang yang memungkinkan penambahan elemen dan memeriksa apakah ada elemen. Positif palsu dimungkinkan di mana filter salah menunjukkan bahwa suatu elemen ada, meskipun tidak ditambahkan. Namun, Filter Bloom menjamin bahwa negatif palsu (salah menunjukkan bahwa suatu elemen tidak ada, meskipun ditambahkan) tidak terjadi.

Sumber utama dokumentasi untuk filter mekar dapat ditemukan di halaman dokumentasi valkey.io. Ini berisi informasi berikut:

- Kasus penggunaan umum untuk filter mekar
  - Iklan/Deduplikasi acara
  - Deteksi penipuan
  - Memfilter konten berbahaya/spam
  - Deteksi pengguna yang unik
- Perbedaan antara filter mekar penskalaan dan non-penskalaan
  - Cara memutuskan antara filter mekar penskalaan dan non-penskalaan
- Properti mekar
  - Pelajari tentang properti filter Bloom yang dapat disetel. Ini termasuk tingkat positif palsu, kapasitas, penskalaan dan properti non-penskalaan, dan banyak lagi.
- Kinerja perintah mekar
- Memantau statistik filter mekar secara keseluruhan
- Menangani filter mekar besar
  - Rekomendasi dan detail tentang cara memeriksa apakah filter mekar mencapai batas penggunaan memorinya, dan apakah filter tersebut dapat diskalakan untuk mencapai kapasitas yang diinginkan.
  - Anda dapat secara khusus memeriksa jumlah memori yang dikonsumsi oleh dokumen filter mekar melalui penggunaan perintah BF.INFO.

Batas ukuran mekar

Konsumsi memori oleh satu objek filter Bloom dibatasi hingga 128 MB. Anda dapat memeriksa jumlah memori yang dikonsumsi oleh filter Bloom dengan menggunakan `BF.INFO <key> SIZE` perintah.

Mekar ACLs

Mirip dengan kategori per-tipe data yang ada (@string, @hash, dll.) Kategori baru @bloom ditambahkan untuk menyederhanakan pengelolaan akses ke perintah dan data Bloom. Tidak ada perintah Valkey atau Redis OSS lain yang ada sebagai anggota kategori @bloom.

Ada 3 kategori ACL yang ada yang diperbarui untuk menyertakan perintah Bloom baru: @read, @write dan @fast. Tabel berikut menunjukkan pemetaan perintah Bloom ke kategori yang sesuai.

Perintah Bloom	@read	@write	@fast	@bloom
BF.ADD		y	y	y
BF.KARTU	y		y	y
BF.EXISTS	y		y	y
BF.INFO	y		y	y
BF.INSERT		y	y	y
BF.MADD		y	y	y
BF.MEXISTS	y		y	y
BF.RESERVE	y		y	y

Metrik terkait filter Bloom

CloudWatch Metrik berikut yang terkait dengan struktur data bloom disediakan:

Metrik CW	Unit	Tanpa server/Dirancang sendiri	Deskripsi
BloomFilterBasedCmds	Hitungan	Keduanya	Jumlah total perintah filter Bloom, termasuk perintah baca dan tulis.
BloomFilterBasedCmdsLatency	Mikrodetik	swakelola	Latensi semua perintah filter Bloom, termasuk perintah baca dan tulis.
BloomFilterBasedCmdsECPUs	Hitungan	Nirserver	ECPUs dikonsumsi oleh semua perintah filter Bloom, termasuk perintah baca dan tulis.

Perintah filter Bloom

Perintah Bloom Filter didokumentasikan di situs web Valkey.io. Setiap halaman perintah memberikan ikhtisar komprehensif tentang perintah bloom, termasuk sintaks, perilaku, nilai pengembalian, dan kondisi kesalahan potensial.

Nama	Penjelasan
BF.ADD	Menambahkan satu item ke filter mekar.Jika filter belum ada, itu dibuat.
BF.CARD	Mengembalikan kardinalitas filter mekar.
BF.EXISTS	Menentukan apakah filter mekar berisi item yang ditentukan.
BF.INFO	Mengembalikan informasi penggunaan dan properti dari filter mekar tertentu.

Nama	Penjelasan
BF. SISIPKAN	Membuat filter mekar dengan 0 item atau lebih, atau menambahkan item ke filter mekar yang ada.
BF.MADD	Menambahkan satu atau lebih item ke filter mekar.
BF.MEKSIKO	Menentukan apakah filter mekar berisi 1 item atau lebih.
BF. CADANGAN	Membuat filter mekar kosong dengan properti yang ditentukan.

> **Note**
>
> BF.LOAD tidak didukung oleh. ElastiCache Ini hanya relevan untuk penggunaan AOF, yang ElastiCache tidak mendukung.

Memulai dengan JSON untuk Valkey dan Redis OSS

ElastiCache mendukung format asli JavaScript Object Notation (JSON), yang merupakan cara sederhana dan tanpa skema untuk menyandikan kumpulan data kompleks di dalam cluster Valkey dan Redis OSS. Anda dapat menyimpan dan mengakses data secara native menggunakan format JavaScript Object Notation (JSON) di dalam cluster, dan memperbarui data JSON yang disimpan dalam cluster tersebut—tanpa perlu mengelola kode kustom untuk membuat serial dan deserialisasinya.

Selain menggunakan operasi Valkey dan Redis OSS API untuk aplikasi yang beroperasi melalui JSON, Anda sekarang dapat secara efisien mengambil dan memperbarui bagian tertentu dari dokumen JSON tanpa perlu memanipulasi seluruh objek. Hal ini dapat meningkatkan performa dan mengurangi biaya. Anda juga dapat menelusuri konten dokumen JSON Anda menggunakan kueri `JSONPath` bergaya Goessner.

Setelah Anda membuat klaster dengan versi mesin yang didukung, jenis data JSON dan perintah terkait akan tersedia secara otomatis. Kompatibel dengan API dan RDB kompatibel dengan versi 2

modul JSON, sehingga Anda dapat dengan mudah memigrasikan aplikasi Valkey dan Redis OSS berbasis JSON yang ada ke dalam. ElastiCache Untuk informasi selengkapnya tentang perintah yang didukung, lihatPerintah Valkey dan Redis OSS yang didukung.

Metrik terkait JSON `JsonBasedCmds` dan `JsonBasedCmdsLatency` dimasukkan ke dalam CloudWatch untuk memantau penggunaan tipe data ini. Untuk informasi lebih lanjut, lihat Metrik untuk Valkey dan Redis OSS.

> **Note**
>
> Untuk menggunakan JSON, Anda harus menjalankan Valkey 7.2 dan yang lebih baru, atau Redis OSS 6.2.6 atau yang lebih baru.

Topik

Ikhtisar tipe data JSON

ElastiCache mendukung sejumlah perintah Valkey dan Redis OSS untuk bekerja dengan tipe data JSON. Berikut ini adalah ikhtisar tipe data JSON dan daftar rinci perintah yang didukung.

Terminologi

Istilah	Deskripsi
Dokumen JSON	Mengacu pada nilai kunci JSON.
Nilai JSON	Mengacu pada subset dari dokumen JSON, termasuk root yang merepresentasikan seluruh dokumen. Nilai bisa berupa kontainer atau entri dalam kontainer.
Elemen JSON	Setara dengan nilai JSON.

Standar JSON yang didukung

Format JSON sesuai dengan standar pertukaran data JSON RFC 7159 dan ECMA-404. Mendukung UTF-8 Unicode pada teks JSON.

Elemen root

Elemen root dapat berupa jenis data JSON apa pun. Perhatikan bahwa di RFC 4627 sebelumnya, hanya objek atau array yang diizinkan sebagai nilai root. Sejak pembaruan ke RFC 7159, root dokumen JSON dapat berupa jenis data JSON apa pun.

Batas ukuran dokumen

Dokumen JSON disimpan secara internal dalam format yang dioptimalkan untuk perubahan dan akses cepat. Format ini biasanya menimbulkan konsumsi cukup banyak memori daripada representasi terserialisasi yang setara dari dokumen yang sama.

Konsumsi memori oleh satu dokumen JSON dibatasi hingga 64 MB, yang merupakan ukuran struktur data dalam memori, bukan string JSON. Anda dapat memeriksa jumlah memori yang dikonsumsi oleh dokumen JSON menggunakan perintah `JSON.DEBUG MEMORY`.

JSON ACLs

- Serupa dengan kategori per jenis data yang ada (@string, @hash, dll.), kategori baru @json ditambahkan untuk menyederhanakan manajemen akses ke perintah dan data JSON. Tidak ada perintah Valkey atau Redis OSS lain yang ada sebagai anggota kategori @json. Semua perintah JSON memberlakukan batasan dan izin ruang kunci atau perintah apa pun.
- Ada lima kategori Valkey dan Redis OSS ACL yang diperbarui untuk menyertakan perintah JSON baru: @read, @write, @fast, @slow dan @admin. Tabel berikut menunjukkan pemetaan perintah JSON ke kategori yang sesuai.

ACL

Perintah JSON	@read	@write	@fast	@slow	@admin
JSON.ARRAPPEND		y	y		

Perintah JSON	@read	@write	@fast	@slow	@admin
JSON.ARRINDEX	y		y		
JSON.ARRINSERT		y	y		
JSON.ARRLEN	y		y		
JSON.ARRPOP		y	y		
JSON.ARRTRIM		y	y		
JSON.CLEAR		y	y		
JSON.DEBUG	y			y	y
JSON.DEL		y	y		
JSON.FORGET		y	y		
JSON.GET	y		y		
JSON.MGET	y		y		
JSON.NUMINCRBY		y	y		
JSON.NUMMULTBY		y	y		

Perintah JSON	@read	@write	@fast	@slow	@admin
JSON.OBJKEYS	y		y		
JSON.OBJLEN	y		y		
JSON.RESP	y		y		
JSON.SET		y		y	
JSON.STRAPPEND		y	y		
JSON.STRLEN	y		y		
JSON.STRLEN	y		y		
JSON.TOGGLE		y	y		
JSON.TYPE	y		y		
JSON.NUMINCRBY		y	y		

Batas kedalaman sarang

Ketika objek atau array JSON memiliki elemen yang merupakan objek atau array JSON lain, objek dalam atau array dianggap "bersarang" dalam objek luar atau array. Batas kedalaman sarang maksimum adalah 128. Setiap percobaan untuk membuat dokumen yang berisi kedalaman bersarang lebih dari 128 akan ditolak dengan kesalahan.

Sintaksis perintah

Kebanyakan perintah memerlukan nama kunci sebagai argumen pertama. Beberapa perintah juga memiliki argumen jalur. Argumen jalur ditetapkan secara default ke root jika bersifat opsional dan tidak disediakan.

Notasi:

- Argumen wajib diapit oleh tanda kurung sudut. Misalnya: <key>
- Argumen opsional diapit oleh tanda kurung siku. Misalnya: [path]
- Argumen opsional tambahan ditunjukkan oleh elipsis ("..."). Misalnya: [json...]

Sintaksis jalur

Redis JSON mendukung dua jenis sintaksis jalur:

- Sintaks yang disempurnakan - Mengikuti JSONPath sintaks yang dijelaskan oleh Goessner, seperti yang ditunjukkan pada tabel berikut. Kami telah menyusun ulang dan mengubah deskripsi dalam tabel agar jelas.
- Sintaksis terbatas – Memiliki kemampuan kueri terbatas.

> **Note**
>
> Hasil dari beberapa perintah bersifat sensitif terhadap jenis sintaksis jalur yang digunakan.

Jika jalur kueri diawali dengan '$', kueri tersebut menunjukkan penggunaan sintaksis yang ditingkatkan. Jika tidak, maka kueri tersebut menggunakan sintaksis terbatas.

Sintaksis yang ditingkatkan

Simbol/Ekspresi	Deskripsi
$	Elemen root.
. atau []	Operator turunan.
..	Penurunan rekursif.

Simbol/Ekspresi	Deskripsi
*	Wildcard. Semua elemen dalam sebuah objek atau array.
[]	Operator subskrip array. Indeks berbasis 0.
[,]	Operator Union.
[start:end:step]	Operator irisan array.
?()	Menerapkan ekspresi filter (skrip) ke array atau objek saat ini.
()	Ekspresi filter.
@	Digunakan dalam ekspresi filter yang merujuk ke simpul saat ini yang sedang diproses.
==	Sama dengan, digunakan dalam ekspresi filter.
!=	Tidak sama dengan, digunakan dalam ekspresi filter.
>	Lebih dari, digunakan dalam ekspresi filter.
>=	Lebih dari atau sama dengan, digunakan dalam ekspresi filter.
<	Kurang dari, digunakan dalam ekspresi filter.
<=	Kurang dari atau sama dengan, digunakan dalam ekspresi filter.
&&	Logika AND, digunakan untuk menggabungkan beberapa ekspresi filter.
||	Logika OR, digunakan untuk menggabungkan beberapa ekspresi filter.

Contoh

Contoh-contoh berikut dibuat berdasarkan contoh data XML Goessner, yang telah kami perubahan dengan menambahkan bidang tambahan.

```
{ "store": {
    "book": [
      { "category": "reference",
        "author": "Nigel Rees",
        "title": "Sayings of the Century",
        "price": 8.95,
        "in-stock": true,
        "sold": true
      },
      { "category": "fiction",
        "author": "Evelyn Waugh",
        "title": "Sword of Honour",
        "price": 12.99,
        "in-stock": false,
        "sold": true
      },
      { "category": "fiction",
        "author": "Herman Melville",
        "title": "Moby Dick",
        "isbn": "0-553-21311-3",
        "price": 8.99,
        "in-stock": true,
        "sold": false
      },
      { "category": "fiction",
        "author": "J. R. R. Tolkien",
        "title": "The Lord of the Rings",
        "isbn": "0-395-19395-8",
        "price": 22.99,
        "in-stock": false,
        "sold": false
      }
    ],
    "bicycle": {
      "color": "red",
      "price": 19.95,
      "in-stock": true,
      "sold": false
    }
```

```
  }
}
```

Jalur	Deskripsi
$.store.book[*].author	Penulis semua buku di toko.
$..author	Semua penulis.
$.store.*	Semua anggota toko.
$["store"].*	Semua anggota toko.
$.store..price	Harga semua yang ada di toko.
$..*	Semua anggota rekursif dari struktur JSON.
$..book[*]	Semua buku.
$..book[0]	Buku pertama.
$..book[-1]	Buku terakhir.
$..book[0:2]	Dua buku pertama.
$..book[0,1]	Dua buku pertama.
$..book[0:4]	Buku dari indeks 0 hingga 3 (indeks akhir tidak inklusif).
$..book[0:4:2]	Buku pada indeks 0, 2.
$..book[?(@.isbn)]	Semua buku dengan nomor ISBN.
$..book[?(@.price<10)]	Semua buku yang lebih murah dari $10.
‘$..book[?(@.price < 10)]’	Semua buku yang lebih murah dari $10. (Jalur harus diberi tanda kutip jika berisi spasi.)
‘$..book[?(@.["price"] < 10)]’	Semua buku yang lebih murah dari $10.

Jalur	Deskripsi
'$..book[?(@.["price"] < 10)]'	Semua buku yang lebih murah dari $10.
$..book[?(@.price>=10&&@.price<=100)]	Semua buku dalam kisaran harga $10 hingga $100, inklusif.
'$..book[?(@.price>=10 && @.price<=100)]'	Semua buku dalam kisaran harga $10 hingga $100, inklusif. (Jalur harus diberi tanda kutip jika berisi spasi.)
$..book[?(@.sold==true||@.in-stock==false)]	Semua buku yang terjual atau habis.
'$..book[?(@.sold == true || @.in-stock == false)]'	Semua buku yang terjual atau habis. (Jalur harus diberi tanda kutip jika berisi spasi.)
'$.store.book[?(@.["category"] == "fiction")]'	Semua buku dalam kategori fiksi.
'$.store.book[?(@.["category"] != "fiction")]'	Semua buku dalam kategori nonfiksi.

Contoh ekspresi filter tambahan:

```
127.0.0.1:6379> JSON.SET k1 . '{"books": [{"price":5,"sold":true,"in-
stock":true,"title":"foo"}, {"price":15,"sold":false,"title":"abc"}]}'
OK
127.0.0.1:6379> JSON.GET k1 $.books[?(@.price>1&&@.price<20&&@.in-stock)]
"[{\"price\":5,\"sold\":true,\"in-stock\":true,\"title\":\"foo\"}]"
127.0.0.1:6379> JSON.GET k1 '$.books[?(@.price>1 && @.price<20 && @.in-stock)]'
"[{\"price\":5,\"sold\":true,\"in-stock\":true,\"title\":\"foo\"}]"
127.0.0.1:6379> JSON.GET k1 '$.books[?((@.price>1 && @.price<20) && (@.sold==false))]'
"[{\"price\":15,\"sold\":false,\"title\":\"abc\"}]"
127.0.0.1:6379> JSON.GET k1 '$.books[?(@.title == "abc")]'
[{"price":15,"sold":false,"title":"abc"}]

127.0.0.1:6379> JSON.SET k2 . '[1,2,3,4,5]'
127.0.0.1:6379> JSON.GET k2 $.*.[?(@>2)]
"[3,4,5]"
127.0.0.1:6379> JSON.GET k2 '$.*.[?(@ > 2)]'
"[3,4,5]"

127.0.0.1:6379> JSON.SET k3 . '[true,false,true,false,null,1,2,3,4]'
```

```
OK
127.0.0.1:6379> JSON.GET k3 $.*.[?(@==true)]
"[true,true]"
127.0.0.1:6379> JSON.GET k3 '$.*.[?(@ == true)]'
"[true,true]"
127.0.0.1:6379> JSON.GET k3 $.*.[?(@>1)]
"[2,3,4]"
127.0.0.1:6379> JSON.GET k3 '$.*.[?(@ > 1)]'
"[2,3,4]"
```

Sintaksis terbatas

Simbol/Ekspresi	Deskripsi
. atau []	Operator turunan.
[]	Operator subskrip array. Indeks berbasis 0.

Contoh

Jalur	Deskripsi
.store.book[0].author	Penulis dari buku pertama.
.store.book[-1].author	Penulis dari buku terakhir.
.address.city	Nama kota.
["store"]["book"][0]["title"]	Judul buku pertama.
["store"]["book"][-1]["title"]	Judul buku terakhir.

> **Note**
>
> Semua konten Goessner yang dikutip dalam dokumentasi ini diatur dengan Lisensi Creative Commons.

Awalan kesalahan umum

Setiap pesan kesalahan memiliki awalan. Berikut ini adalah daftar awalan kesalahan umum.

Awalan	Deskripsi
ERR	Kesalahan umum.
LIMIT	Kesalahan yang terjadi ketika batas ukuran terlampaui. Misalnya, batas ukuran dokumen atau batas kedalaman sarang telah terlampaui.
NONEXISTENT	Kunci atau jalur tidak ada.
OUTOFBOUNDARIES	Indeks array di luar batas.
SYNTAXERR	Kesalahan sintaksis.
WRONGTYPE	Jenis nilai yang salah.

Metrik terkait JSON

Tersedia metrik info JSON berikut:

Info	Deskripsi
json_total_memory_bytes	Total memori yang dialokasikan untuk objek JSON.
json_num_documents	Jumlah total dokumen di Valkey atau Redis OSS.

Untuk menanyakan metrik inti, jalankan perintah berikut:

```
info json_core_metrics
```

Bagaimana ElastiCache Valkey dan Redis OSS berinteraksi dengan JSON

Bagian berikut menjelaskan bagaimana ElastiCache Valkey dan Redis OSS berinteraksi dengan tipe data JSON.

Prasyarat operator

Saat mengevaluasi ekspresi bersyarat untuk pemfilteran, &&s diutamakan terlebih dahulu, lalu ||s dievaluasi, seperti yang umum di sebagian besar bahasa. Operasi dalam tanda kurung dijalankan terlebih dahulu.

Perilaku batas bersarang jalur maksimum

Batas bersarang jalur maksimum ElastiCache untuk Redis OSS adalah 128. Jadi nilai seperti `$.a.b.c.d...` hanya bisa mencapai 128 tingkat.

Menangani nilai numerik

JSON tidak memiliki jenis data terpisah untuk angka integer dan floating point. Semuanya disebut angka.

Representasi numerik:

Ketika angka JSON diterima pada input, angka tersebut diubah menjadi salah satu dari dua representasi biner internal: integer bertanda 64-bit atau floating point presisi ganda IEEE 64-bit. String asli dan semua formatnya tidak dipertahankan. Jadi, ketika angka dihasilkan sebagai bagian dari respons JSON, angka tersebut dikonversi dari representasi biner internal ke string yang dapat dicetak yang menggunakan aturan pemformatan generik. Aturan-aturan ini dapat menghasilkan string yang berbeda dari string yang diterima.

Perintah aritmetika `NUMINCRBY` dan `NUMMULTBY`:

- Jika kedua angka ini adalah integer dan hasilnya berada di luar rentang `int64`, maka angka tersebut secara otomatis akan menjadi angka floating point presisi ganda IEEE 64-bit.
- Jika minimal salah satu angkanya adalah floating point, hasilnya adalah angka floating point presisi ganda IEEE 64-bit.
- Jika hasilnya melebihi rentang IEEE 64-bit ganda, perintah ini menampilkan kesalahan `OVERFLOW`.

Untuk daftar lengkap perintah yang tersedia, lihat Perintah Valkey dan Redis OSS yang didukung.

Pemfilteran array langsung

ElastiCache untuk Valkey atau Redis OSS menyaring objek array secara langsung.

Untuk data seperti `[0,1,2,3,4,5,6]` dan kueri jalur seperti`$[?(@<4)]`, atau data seperti `{"my_key":[0,1,2,3,4,5,6]}` dan kueri jalur seperti`$.my_key[?(@<4)]`, ElastiCache akan mengembalikan [1,2,3] dalam kedua keadaan.

Perilaku pengindeksan array

ElastiCache untuk Valkey atau Redis OSS memungkinkan indeks positif dan negatif untuk array. Untuk array dengan panjang lima, 0 akan mengueri elemen pertama, 1 yang kedua, dan seterusnya. Angka negatif dimulai pada akhir array, jadi -1 akan mengueri elemen kelima, -2 akan mengueri elemen keempat, dan seterusnya.

Untuk memastikan perilaku yang dapat diprediksi bagi pelanggan, ElastiCache tidak membulatkan indeks array ke bawah atau ke atas, jadi jika Anda memiliki array dengan panjang 5, memanggil indeks 5 atau lebih tinggi, atau -6 atau lebih rendah, tidak akan menghasilkan hasil.

Evaluasi sintaksis yang ketat

MemoryDB tidak mengizinkan jalur JSON dengan sintaksis yang tidak valid, bahkan jika subset dari jalur berisi jalur yang valid. Hal ini dimaksudkan untuk menjaga perilaku yang benar bagi pelanggan kami.

Perintah Valkey dan Redis OSS yang didukung

ElastiCache mendukung perintah Valkey dan Redis OSS JSON berikut:

Topik

- JSON.ARRAPPEND
- JSON.ARRINDEX
- JSON.ARRINSERT
- JSON.ARRLEN
- JSON.ARRPOP
- JSON.ARRTRIM
- JSON.CLEAR
- JSON.DEBUG
- JSON.DEL

- JSON.FORGET
- JSON.GET
- JSON.MGET
- JSON.MSET
- JSON.NUMINCRBY
- JSON.NUMMULTBY
- JSON.OBJLEN
- JSON.OBJKEYS
- JSON.RESP
- JSON.SET
- JSON.STRAPPEND
- JSON.STRLEN
- JSON.TOGGLE
- JSON.TYPE

JSON.ARRAPPEND

Menambahkan satu atau beberapa nilai ke nilai array di jalur.

Sintaksis

```
JSON.ARRAPPEND <key> <path> <json> [json ...]
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (wajib) – Sebuah jalur JSON.
- json (wajib) - Nilai JSON yang akan ditambahkan ke array.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array integer yang merepresentasikan panjang baru array di setiap jalur.
- Jika nilai bukan array, nilai yang akan dikembalikan adalah kosong.
- Kesalahan `NONEXISTENT` jika jalur tidak ada.

Jika jalur adalah sintaksis terbatas:

- Integer, panjang baru array.
- Jika beberapa nilai array dipilih, perintah mengembalikan panjang baru dari array yang diperbarui pertama.
- Kesalahan `WRONGTYPE` jika nilai di jalur bukan array.
- Kesalahan `SYNTAXERR` jika salah satu argumen input json bukan string JSON yang valid.
- Kesalahan `NONEXISTENT` jika jalur tidak ada.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"]]'
OK
127.0.0.1:6379> JSON.ARRAPPEND  k1 $[*] '"c"'
1) (integer) 1
2) (integer) 2
3) (integer) 3
127.0.0.1:6379> JSON.GET k1
"[[\"c\"],[\"a\",\"c\"],[\"a\",\"b\",\"c\"]]"
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"]]'
OK
127.0.0.1:6379> JSON.ARRAPPEND  k1 [-1] '"c"'
(integer) 3
127.0.0.1:6379> JSON.GET k1
"[[],[\"a\"],[\"a\",\"b\",\"c\"]]"
```

JSON.ARRINDEX

Mencari kemunculan pertama dari nilai JSON skalar dalam array di jalur.

- Kesalahan di luar jangkauan diatasi dengan membulatkan indeks ke awal dan akhir array.
- Jika awal > akhir, mengembalikan -1 (tidak ditemukan).

Sintaksis

```
JSON.ARRINDEX <key> <path> <json-scalar> [start [end]]
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (wajib) – Sebuah jalur JSON.
- json-skalar (wajib) - Nilai skalar yang dicari. Skalar JSON mengacu pada nilai yang bukan merupakan objek atau array. Artinya, string, angka, Boolean, dan kosong (null) adalah nilai skalar.
- awal (opsional) – Indeks awal, inklusif. Default ke 0 jika tidak disediakan.
- akhir (opsional) - Indeks akhir, eksklusif. Default ke 0 jika tidak disediakan, yang berarti bahwa elemen terakhir disertakan. 0 atau -1 berarti elemen terakhir disertakan.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array integer. Setiap nilai adalah indeks dari elemen yang cocok dalam array di jalur. Nilainya -1 jika tidak ditemukan.
- Jika nilai bukan array, nilai yang akan dikembalikan adalah kosong.

Jika jalur adalah sintaksis terbatas:

- Integer, indeks elemen yang cocok, atau -1 jika tidak ditemukan.
- Kesalahan `WRONGTYPE` jika nilai di jalur bukan array.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"], ["a", "b", "c"]]'
OK
127.0.0.1:6379> JSON.ARRINDEX k1 $[*] '"b"'
1) (integer) -1
2) (integer) -1
3) (integer) 1
4) (integer) 1
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 . '{"children": ["John", "Jack", "Tom", "Bob", "Mike"]}'
OK
127.0.0.1:6379> JSON.ARRINDEX k1 .children '"Tom"'
(integer) 2
```

JSON.ARRINSERT

Menyisipkan satu atau beberapa nilai ke dalam nilai array di jalur sebelum indeks.

Sintaksis

```
JSON.ARRINSERT <key> <path> <index> <json> [json ...]
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (wajib) – Sebuah jalur JSON.
- index (wajib) – Sebuah indeks array yang dimasukkan setelah nilai.
- json (wajib) - Nilai JSON yang akan ditambahkan ke array.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array integer yang merepresentasikan panjang baru array di setiap jalur.
- Jika nilai adalah array kosong, nilai yang akan dikembalikan adalah kosong.
- Jika nilai bukan array, nilai yang akan dikembalikan adalah kosong.
- Kesalahan `OUTOFBOUNDARIES` jika argumen indeks di luar batas.

Jika jalur adalah sintaksis terbatas:

- Integer, panjang baru array.
- Kesalahan `WRONGTYPE` jika nilai di jalur bukan array.
- Kesalahan `OUTOFBOUNDARIES` jika argumen indeks di luar batas.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"]]'
OK
127.0.0.1:6379> JSON.ARRINSERT k1 $[*] 0 '"c"'
1) (integer) 1
2) (integer) 2
3) (integer) 3
127.0.0.1:6379> JSON.GET k1
"[[\"c\"],[\"c\",\"a\"],[\"c\",\"a\",\"b\"]]"
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"]]'
OK
127.0.0.1:6379> JSON.ARRINSERT k1 . 0 '"c"'
(integer) 4
127.0.0.1:6379> JSON.GET k1
"[\"c\",[],[\"a\"],[\"a\",\"b\"]]"
```

JSON.ARRLEN

Mendapatkan panjang nilai array di jalur.

Sintaksis

```
JSON.ARRLEN <key> [path]
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (opsional) – Sebuah jalur JSON. Diatur secara default ke root jika tidak disediakan.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array integer yang merepresentasikan panjang array di setiap jalur.
- Jika nilai bukan array, nilai yang akan dikembalikan adalah kosong.
- Kosong jika kunci dokumen tidak ada.

Jika jalur adalah sintaksis terbatas:

- Integer, panjang array.
- Jika memilih beberapa objek, perintah ini mengembalikan panjang array pertama.
- Kesalahan `WRONGTYPE` jika nilai di jalur bukan array.
- Kesalahan `NONEXISTENT JSON` jika jalur tidak ada.
- Kosong jika kunci dokumen tidak ada.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"], ["a", "b", "c"]]'
OK
127.0.0.1:6379> JSON.ARRLEN k1 $[*]
1) (integer) 0
2) (integer) 1
3) (integer) 2
4) (integer) 3

127.0.0.1:6379> JSON.SET k2 . '[[], "a", ["a", "b"], ["a", "b", "c"], 4]'
OK
127.0.0.1:6379> JSON.ARRLEN k2 $[*]
1) (integer) 0
2) (nil)
3) (integer) 2
4) (integer) 3
5) (nil)
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"], ["a", "b", "c"]]'
OK
127.0.0.1:6379> JSON.ARRLEN k1 [*]
(integer) 0
127.0.0.1:6379> JSON.ARRLEN k1 [1]
(integer) 1
127.0.0.1:6379> JSON.ARRLEN k1 [2]
(integer) 2
```

```
127.0.0.1:6379> JSON.SET k2 . '[[], "a", ["a", "b"], ["a", "b", "c"], 4]'
OK
127.0.0.1:6379> JSON.ARRLEN k2 [1]
(error) WRONGTYPE JSON element is not an array
127.0.0.1:6379> JSON.ARRLEN k2 [0]
(integer) 0
127.0.0.1:6379> JSON.ARRLEN k2 [6]
(error) OUTOFBOUNDARIES Array index is out of bounds
127.0.0.1:6379> JSON.ARRLEN k2 a.b
(error) NONEXISTENT JSON path does not exist
```

JSON.ARRPOP

Menghapus dan mengembalikan elemen pada indeks dari array. Popping array kosong akan menampilkan null.

Sintaksis

```
JSON.ARRPOP <key> [path [index]]
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (opsional) – Sebuah jalur JSON. Default ke root jika tidak disediakan.
- indeks (opsional) – Posisi dalam array tempat popping dimulai.
  - Default ke -1 jika tidak disediakan, yang berarti elemen terakhir.
  - Nilai negatif berarti posisi dari elemen terakhir.
  - Indeks di luar batas akan dibulatkan ke batas array masing-masing.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array string massal yang merepresentasikan nilai yang di-popping di setiap jalur.
- Jika nilai adalah array kosong, nilai yang akan dikembalikan adalah kosong.
- Jika nilai bukan array, nilai yang akan dikembalikan adalah kosong.

Jika jalur adalah sintaksis terbatas:

- String massal, yang merepresentasikan nilai JSON yang di-popping.
- Null jika array kosong.
- Kesalahan `WRONGTYPE` jika nilai di jalur bukan array.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"]]'
OK
127.0.0.1:6379> JSON.ARRPOP k1 $[*]
1) (nil)
2) "\"a\""
3) "\"b\""
127.0.0.1:6379> JSON.GET k1
"[[],[],[\"a\"]]"
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"]]'
OK
127.0.0.1:6379> JSON.ARRPOP k1
"[\"a\",\"b\"]"
127.0.0.1:6379> JSON.GET k1
"[[],[\"a\"]]"

127.0.0.1:6379> JSON.SET k2 . '[[], ["a"], ["a", "b"]]'
OK
127.0.0.1:6379> JSON.ARRPOP k2 . 0
"[]"
127.0.0.1:6379> JSON.GET k2
"[[\"a\"],[\"a\",\"b\"]]"
```

JSON.ARRTRIM

Memangkas array di jalur sehingga menjadi subarray [awal, akhir], keduanya inklusif.

- Jika array kosong, tidak melakukan apa pun, mengembalikan 0.

- Jika awal <0, perlakukan itu sebagai 0.
- Jika akhir >= ukuran (ukuran array), perlakukan itu sebagai ukuran-1.
- Jika awal >= ukuran atau awal > akhir, mengosongkan array dan menampilkan 0.

Sintaksis

```
JSON.ARRTRIM <key> <path> <start> <end>
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (wajib) – Sebuah jalur JSON.
- Awal (wajib) – Indeks awal, inklusif.
- akhir (wajib) – Indeks akhir, inklusif.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array integer yang merepresentasikan panjang baru array di setiap jalur.
- Jika nilai adalah array kosong, nilai yang akan dikembalikan adalah kosong.
- Jika nilai bukan array, nilai yang akan dikembalikan adalah kosong.
- Kesalahan `OUTOFBOUNDARIES` jika argumen indeks di luar batas.

Jika jalur adalah sintaksis terbatas:

- Integer, panjang baru array.
- Null jika array kosong.
- Kesalahan `WRONGTYPE` jika nilai di jalur bukan array.
- Kesalahan `OUTOFBOUNDARIES` jika argumen indeks di luar batas.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 . '[[], ["a"], ["a", "b"], ["a", "b", "c"]]'
```

```
OK
127.0.0.1:6379> JSON.ARRTRIM k1 $[*] 0 1
1) (integer) 0
2) (integer) 1
3) (integer) 2
4) (integer) 2
127.0.0.1:6379> JSON.GET k1
"[[],[\"a\"],[\"a\",\"b\"],[\"a\",\"b\"]]"
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 . '{"children": ["John", "Jack", "Tom", "Bob", "Mike"]}'
OK
127.0.0.1:6379> JSON.ARRTRIM k1 .children 0 1
(integer) 2
127.0.0.1:6379> JSON.GET k1 .children
"[\"John\",\"Jack\"]"
```

JSON.CLEAR

Membersihkan array atau objek di jalur.

Sintaksis

```
JSON.CLEAR <key> [path]
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (opsional) – Sebuah jalur JSON. Diatur secara default ke root jika tidak disediakan.

Nilai yang ditampilkan

- Integer, jumlah kontainer dihapus.
- Menghapus array kosong atau akun objek untuk 1 kontainer yang dihapus.
- Menghapus nilai non-kontainer mengembalikan 0.

Contoh

```
127.0.0.1:6379> JSON.SET k1 . '[[], [0], [0,1], [0,1,2], 1, true, null, "d"]'
OK
127.0.0.1:6379>  JSON.CLEAR k1  $[*]
(integer) 7
127.0.0.1:6379> JSON.CLEAR k1  $[*]
(integer) 4
127.0.0.1:6379> JSON.SET k2 . '{"children": ["John", "Jack", "Tom", "Bob", "Mike"]}'
OK
127.0.0.1:6379> JSON.CLEAR k2 .children
(integer) 1
127.0.0.1:6379> JSON.GET k2 .children
"[]"
```

JSON.DEBUG

Melaporkan informasi. Subperintah yang didukung adalah:

- MEMORY <key> [path] – Melaporkan penggunaan memori dalam byte dari nilai JSON. Jalur akan diatur secara default ke root jika tidak disediakan.
- FIELDS <key> [path] – Melaporkan jumlah bidang di jalur dokumen yang ditentukan. Jalur akan diatur secara default ke root jika tidak disediakan. Setiap nilai JSON non-kontainer dihitung sebagai satu bidang. Objek dan array secara rekursif menghitung satu bidang untuk masing-masing nilai JSON. Setiap nilai kontainer, kecuali kontainer root, dihitung sebagai satu bidang tambahan.
- HELP – Mencetak pesan bantuan dari perintah.

Sintaksis

```
JSON.DEBUG <subcommand & arguments>
```

Tergantung pada subperintah:

MEMORY

- Jika jalur adalah sintaksis yang ditingkatkan:
  - Menampilkan array integer yang merepresentasikan ukuran memori (dalam byte) dari nilai JSON di setiap jalur.
  - Mengembalikan array kosong jika kunci Valkey atau Redis OSS tidak ada.

- Jika jalur adalah sintaksis terbatas:
  - Menampilkan integer, ukuran memori, dan nilai JSON dalam byte.
  - Mengembalikan null jika kunci Valkey atau Redis OSS tidak ada.

FIELD

- Jika jalur adalah sintaksis yang ditingkatkan:
  - Menampilkan array integer yang merepresentasikan jumlah bidang nilai JSON di setiap jalur.
  - Mengembalikan array kosong jika kunci Valkey atau Redis OSS tidak ada.
- Jika jalur adalah sintaksis terbatas:
  - Menampilkan integer, jumlah bidang nilai JSON.
  - Mengembalikan null jika kunci Valkey atau Redis OSS tidak ada.

HELP – Menampilkan array pesan bantuan.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 . '[1, 2.3, "foo", true, null, {}, [], {"a":1, "b":2},
 [1,2,3]]'
OK
127.0.0.1:6379> JSON.DEBUG MEMORY k1 $[*]
1) (integer) 16
2) (integer) 16
3) (integer) 19
4) (integer) 16
5) (integer) 16
6) (integer) 16
7) (integer) 16
8) (integer) 50
9) (integer) 64
127.0.0.1:6379> JSON.DEBUG FIELDS k1 $[*]
1) (integer) 1
2) (integer) 1
3) (integer) 1
4) (integer) 1
5) (integer) 1
6) (integer) 0
```

```
7) (integer) 0
8) (integer) 2
9) (integer) 3
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 .
 '{"firstName":"John","lastName":"Smith","age":27,"weight":135.25,"isAlive":true,"address":
{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021-3100"},"phoneNumbers":
[{"type":"home","number":"212 555-1234"},{"type":"office","number":"646
 555-4567"}],"children":[],"spouse":null}'
OK
127.0.0.1:6379> JSON.DEBUG MEMORY k1
(integer) 632
127.0.0.1:6379> JSON.DEBUG MEMORY k1 .phoneNumbers
(integer) 166

127.0.0.1:6379> JSON.DEBUG FIELDS k1
(integer) 19
127.0.0.1:6379> JSON.DEBUG FIELDS k1 .address
(integer) 4

127.0.0.1:6379> JSON.DEBUG HELP
1) JSON.DEBUG MEMORY <key> [path] - report memory size (bytes) of the JSON element.
 Path defaults to root if not provided.
2) JSON.DEBUG FIELDS <key> [path] - report number of fields in the JSON element. Path
 defaults to root if not provided.
3) JSON.DEBUG HELP - print help message.
```

JSON.DEL

Menghapus nilai JSON pada jalur dalam kunci dokumen. Jika jalurnya adalah root, itu setara dengan menghapus kunci dari Valkey atau Redis OSS.

Sintaksis

```
JSON.DEL <key> [path]
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (opsional) – Sebuah jalur JSON. Diatur secara default ke root jika tidak disediakan.

Nilai yang ditampilkan

- Jumlah elemen yang dihapus.
- 0 jika kunci Valkey atau Redis OSS tidak ada.
- 0 jika jalur JSON tidak valid atau tidak ada.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":{}, "b":{"a":1}, "c":{"a":1, "b":2}, "d":{"a":1,
 "b":2, "c":3}, "e": [1,2,3,4,5]}'
OK
127.0.0.1:6379> JSON.DEL k1 $.d.*
(integer) 3
127.0.0.1:6379> JSOn.GET k1
"{\"a\":{},\"b\":{\"a\":1},\"c\":{\"a\":1,\"b\":2},\"d\":{},\"e\":[1,2,3,4,5]}"
127.0.0.1:6379> JSON.DEL k1 $.e[*]
(integer) 5
127.0.0.1:6379> JSOn.GET k1
"{\"a\":{},\"b\":{\"a\":1},\"c\":{\"a\":1,\"b\":2},\"d\":{},\"e\":[]}"
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":{}, "b":{"a":1}, "c":{"a":1, "b":2}, "d":{"a":1,
 "b":2, "c":3}, "e": [1,2,3,4,5]}'
OK
127.0.0.1:6379> JSON.DEL k1 .d.*
(integer) 3
127.0.0.1:6379> JSON.GET k1
"{\"a\":{},\"b\":{\"a\":1},\"c\":{\"a\":1,\"b\":2},\"d\":{},\"e\":[1,2,3,4,5]}"
127.0.0.1:6379> JSON.DEL k1 .e[*]
(integer) 5
127.0.0.1:6379> JSON.GET k1
"{\"a\":{},\"b\":{\"a\":1},\"c\":{\"a\":1,\"b\":2},\"d\":{},\"e\":[]}"
```

JSON.FORGET

Sebuah alias dari JSON.DEL.

JSON.GET

Menampilkan JSON terserialisasi pada satu atau beberapa jalur.

Sintaksis

```
JSON.GET <key>
[INDENT indentation-string]
[NEWLINE newline-string]
[SPACE space-string]
[NOESCAPE]
[path ...]
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- INDENT/NEWLINE/SPACE(opsional) - Mengontrol format string JSON yang dikembalikan, yaitu, “cetak cantik”. Nilai default masing-masing adalah string kosong. Nilai dapat diganti dalam kombinasi apa pun. Nilai dapat ditentukan dalam urutan apa pun.
- NOESCAPE – Opsional, boleh ada untuk kompatibilitas lama dan tidak memiliki efek lain.
- jalur (opsional) - Nol atau lebih jalur JSON, default ke root jika tidak ada yang diberikan. Argumen jalur harus ditempatkan di akhir.

Nilai yang ditampilkan

Sintaksis jalur yang ditingkatkan:

Jika disediakan satu jalur:

- Mengembalikan string terserialisasi dari array nilai.
- Jika tidak ada nilai yang dipilih, perintah ini mengembalikan array kosong.

Jika beberapa jalur diberikan:

- Menampilkan objek JSON yang di-stringify, di mana setiap jalur adalah kunci.
- Jika terdapat campuran sintaksis jalur yang ditingkatkan dan dibatasi, hasilnya sesuai dengan sintaksis yang ditingkatkan.

- Jika jalur tidak ada, nilai yang sesuai adalah array kosong.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 .
 '{"firstName":"John","lastName":"Smith","age":27,"weight":135.25,"isAlive":true,"address":
{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021-3100"},"phoneNumbers":
[{"type":"home","number":"212 555-1234"},{"type":"office","number":"646
 555-4567"}],"children":[],"spouse":null}'
OK
127.0.0.1:6379> JSON.GET k1 $.address.*
"[\"21 2nd Street\",\"New York\",\"NY\",\"10021-3100\"]"
127.0.0.1:6379> JSON.GET k1 indent "\t" space " " NEWLINE "\n" $.address.*
"[\n\t\"21 2nd Street\",\n\t\"New York\",\n\t\"NY\",\n\t\"10021-3100\"\n]"
127.0.0.1:6379> JSON.GET k1 $.firstName $.lastName $.age
"{\"$.firstName\":[\"John\"],\"$.lastName\":[\"Smith\"],\"$.age\":[27]}"
127.0.0.1:6379> JSON.SET k2 . '{"a":{}, "b":{"a":1}, "c":{"a":1, "b":2}}'
OK
127.0.0.1:6379> json.get k2 $..*
"[{},{\"a\":1},{\"a\":1,\"b\":2},1,1,2]"
```

Sintaksis jalur terbatas:

```
 127.0.0.1:6379> JSON.SET k1 .
 '{"firstName":"John","lastName":"Smith","age":27,"weight":135.25,"isAlive":true,"address":
{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021-3100"},"phoneNumbers":
[{"type":"home","number":"212 555-1234"},{"type":"office","number":"646
 555-4567"}],"children":[],"spouse":null}'
OK
127.0.0.1:6379> JSON.GET k1 .address
"{\"street\":\"21 2nd Street\",\"city\":\"New York\",\"state\":\"NY\",\"zipcode\":
\"10021-3100\"}"
127.0.0.1:6379> JSON.GET k1 indent "\t" space " " NEWLINE "\n" .address
"{\n\t\"street\": \"21 2nd Street\",\n\t\"city\": \"New York\",\n\t\"state\": \"NY\",\n
\t\"zipcode\": \"10021-3100\"\n}"
127.0.0.1:6379> JSON.GET k1 .firstName .lastName .age
"{\".firstName\":\"John\",\".lastName\":\"Smith\",\".age\":27}"
```

JSON.MGET

Diserialisasi JSONs di jalur dari beberapa kunci dokumen. Perintah ini menampilkan kosong untuk kunci atau jalur JSON yang tidak ada.

Sintaksis

```
JSON.MGET <key> [key ...] <path>
```

- kunci (wajib) - Satu atau lebih kunci Valkey atau Redis OSS dari jenis dokumen.
- jalur (wajib) – Sebuah jalur JSON.

Nilai yang ditampilkan

- Array string massal. Ukuran array sama dengan jumlah kunci dalam perintah. Setiap elemen array diisi dengan (a) JSON terserialisasi seperti yang dapat ditemukan melalui jalur atau (b) null jika kunci tidak ada, jalur tidak ada dalam dokumen, atau jalur tidak valid (kesalahan sintaksis).
- Jika salah satu kunci yang ditentukan ada dan bukan kunci JSON, perintah ini mengembalikan kesalahan `WRONGTYPE`.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 . '{"address":{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021"}}'
OK
127.0.0.1:6379> JSON.SET k2 . '{"address":{"street":"5 main
 Street","city":"Boston","state":"MA","zipcode":"02101"}}'
OK
127.0.0.1:6379> JSON.SET k3 . '{"address":{"street":"100 Park
 Ave","city":"Seattle","state":"WA","zipcode":"98102"}}'
OK
127.0.0.1:6379> JSON.MGET k1 k2 k3 $.address.city
1) "[\"New York\"]"
2) "[\"Boston\"]"
3) "[\"Seattle\"]"
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 . '{"address":{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021"}}'
OK
127.0.0.1:6379> JSON.SET k2 . '{"address":{"street":"5 main
 Street","city":"Boston","state":"MA","zipcode":"02101"}}'
OK
127.0.0.1:6379> JSON.SET k3 . '{"address":{"street":"100 Park
 Ave","city":"Seattle","state":"WA","zipcode":"98102"}}'
OK

127.0.0.1:6379> JSON.MGET k1 k2 k3 .address.city
1) "\"New York\""
2) "\"Seattle\""
3) "\"Seattle\""
```

JSON.MSET

Didukung untuk Valkey versi 8.1 dan di atas.

Tetapkan nilai JSON untuk beberapa kunci. Operasi adalah atom. Entah semua nilai ditetapkan atau tidak ada yang disetel.

Sintaksis

```
JSON.MSET key path json [ key path json ... ]
```

- Jika jalur memanggil anggota objek:
  - Jika elemen induk tidak ada, perintah akan mengembalikan kesalahan NONEXISTENT.
  - Jika elemen induk ada tetapi bukan objek, perintah akan mengembalikan ERROR.
  - Jika elemen induk ada dan merupakan objek:
    - Jika anggota tidak ada, anggota baru akan ditambahkan ke objek induk jika dan hanya jika objek induk adalah turunan terakhir di jalur. Jika tidak, perintah akan mengembalikan kesalahan NONEXISTENT.
    - Jika anggota ada, nilainya akan diganti dengan nilai JSON.

- Jika jalur memanggil indeks array:
  - Jika elemen induk tidak ada, perintah akan mengembalikan kesalahan NONEXISTENT.
  - Jika elemen induk ada tetapi bukan array, perintah akan mengembalikan ERROR.
  - Jika elemen induk ada tetapi indeks berada di luar batas, perintah akan mengembalikan kesalahan OUTOFBORDIES.
  - Jika elemen induk ada dan indeks valid, elemen akan diganti dengan nilai JSON baru.
- Jika jalur memanggil objek atau array, nilai (objek atau array) akan digantikan oleh nilai JSON baru.

Nilai yang ditampilkan

- Balasan string sederhana: 'OK' jika operasi berhasil.
- Balasan kesalahan sederhana: Jika operasi gagal.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.MSET k1 . '[1,2,3,4,5]' k2 . '{"a":{"a":1, "b":2, "c":3}}' k3 .
 '{"a": [1,2,3,4,5]}'
OK
127.0.0.1:6379> JSON.GET k1
"[1,2,3,4,5]"
127.0.0.1:6379> JSON.GET k2
"{\"a\":{\"a\":1,\"b\":2,\"c\":3}}"
127.0.0.1:6379> JSON.MSET k2 $.a.* '0' k3 $.a[*] '0'
OK
127.0.0.1:6379> JSON.GET k2
"{\"a\":{\"a\":0,\"b\":0,\"c\":0}}"
127.0.0.1:6379> JSON.GET k3
"{\"a\":[0,0,0,0,0]}"
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.MSET k1 . '{"name": "John","address": {"street": "123 Main
 St","city": "Springfield"},"phones": ["555-1234","555-5678"]}'
OK
127.0.0.1:6379> JSON.MSET k1 .address.street '"21 2nd Street"' k1 .address.city '"New
 York"'
OK
```

```
127.0.0.1:6379> JSON.GET k1 .address.street
"\"21 2nd Street\""
127.0.0.1:6379> JSON.GET k1 .address.city
"\"New York\""
```

JSON.NUMINCRBY

Menambah nilai angka di jalur sebanyak sebuah angka tertentu.

Sintaksis

```
JSON.NUMINCRBY <key> <path> <number>
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (wajib) – Sebuah jalur JSON.
- angka (wajib) – Sebuah angka.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array string massal yang merepresentasikan nilai yang dihasilkan di setiap jalur.
- Jika nilai bukan angka, nilai yang akan ditampilkan adalah kosong.
- Kesalahan `WRONGTYPE` jika angka tidak dapat diuraikan.
- Kesalahan `OVERFLOW` jika hasilnya di luar rentang double 64-bit IEEE.
- `NONEXISTENT` jika kunci dokumen tidak ada.

Jika jalur adalah sintaksis terbatas:

- String massal yang merepresentasikan nilai yang dikembalikan.
- Jika memilih beberapa nilai, perintah ini mengembalikan hasil nilai yang terakhir diperbarui.
- Kesalahan `WRONGTYPE` jika nilai di jalur bukan angka.
- Kesalahan `WRONGTYPE` jika angka tidak dapat diuraikan.
- Kesalahan `OVERFLOW` jika hasilnya di luar rentang double 64-bit IEEE.
- `NONEXISTENT` jika kunci dokumen tidak ada.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k1 $.d[*] 10
"[11,12,13]"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[1],\"c\":[1,2],\"d\":[11,12,13]}"

127.0.0.1:6379> JSON.SET k1 $ '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k1 $.a[*] 1
"[]"
127.0.0.1:6379> JSON.NUMINCRBY k1 $.b[*] 1
"[2]"
127.0.0.1:6379> JSON.NUMINCRBY k1 $.c[*] 1
"[2,3]"
127.0.0.1:6379> JSON.NUMINCRBY k1 $.d[*] 1
"[2,3,4]"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[2],\"c\":[2,3],\"d\":[2,3,4]}"

127.0.0.1:6379> JSON.SET k2 $ '{"a":{}, "b":{"a":1}, "c":{"a":1, "b":2}, "d":{"a":1,
 "b":2, "c":3}}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k2 $.a.* 1
"[]"
127.0.0.1:6379> JSON.NUMINCRBY k2 $.b.* 1
"[2]"
127.0.0.1:6379> JSON.NUMINCRBY k2 $.c.* 1
"[2,3]"
127.0.0.1:6379> JSON.NUMINCRBY k2 $.d.* 1
"[2,3,4]"
127.0.0.1:6379> JSON.GET k2
"{\"a\":{},\"b\":{\"a\":2},\"c\":{\"a\":2,\"b\":3},\"d\":{\"a\":2,\"b\":3,\"c\":4}}"

127.0.0.1:6379> JSON.SET k3 $ '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"b"}, "d":{"a":1, "b":"b", "c":3}}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k3 $.a.* 1
"[null]"
127.0.0.1:6379> JSON.NUMINCRBY k3 $.b.* 1
```

```
"[null,2]"
127.0.0.1:6379> JSON.NUMINCRBY k3 $.c.* 1
"[null,null]"
127.0.0.1:6379> JSON.NUMINCRBY k3 $.d.* 1
"[2,null,4]"
127.0.0.1:6379> JSON.GET k3
"{\"a\":{\"a\":\"a\"},\"b\":{\"a\":\"a\",\"b\":2},\"c\":{\"a\":\"a\",\"b\":\"b\"},\"d
\":{\"a\":2,\"b\":\"b\",\"c\":4}}"
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k1 .d[1] 10
"12"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[1],\"c\":[1,2],\"d\":[1,12,3]}"

127.0.0.1:6379> JSON.SET k1 . '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k1 .a[*] 1
(error) NONEXISTENT JSON path does not exist
127.0.0.1:6379> JSON.NUMINCRBY k1 .b[*] 1
"2"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[2],\"c\":[1,2],\"d\":[1,2,3]}"
127.0.0.1:6379> JSON.NUMINCRBY k1 .c[*] 1
"3"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[2],\"c\":[2,3],\"d\":[1,2,3]}"
127.0.0.1:6379> JSON.NUMINCRBY k1 .d[*] 1
"4"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[2],\"c\":[2,3],\"d\":[2,3,4]}"

127.0.0.1:6379> JSON.SET k2 . '{"a":{}, "b":{"a":1}, "c":{"a":1, "b":2}, "d":{"a":1,
 "b":2, "c":3}}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k2 .a.* 1
(error) NONEXISTENT JSON path does not exist
127.0.0.1:6379> JSON.NUMINCRBY k2 .b.* 1
"2"
```

```
127.0.0.1:6379> JSON.GET k2
"{\"a\":{},\"b\":{\"a\":2},\"c\":{\"a\":1,\"b\":2},\"d\":{\"a\":1,\"b\":2,\"c\":3}}"
127.0.0.1:6379> JSON.NUMINCRBY k2 .c.* 1
"3"
127.0.0.1:6379> JSON.GET k2
"{\"a\":{},\"b\":{\"a\":2},\"c\":{\"a\":2,\"b\":3},\"d\":{\"a\":1,\"b\":2,\"c\":3}}"
127.0.0.1:6379> JSON.NUMINCRBY k2 .d.* 1
"4"
127.0.0.1:6379> JSON.GET k2
"{\"a\":{},\"b\":{\"a\":2},\"c\":{\"a\":2,\"b\":3},\"d\":{\"a\":2,\"b\":3,\"c\":4}}"

127.0.0.1:6379> JSON.SET k3 . '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"b"}, "d":{"a":1, "b":"b", "c":3}}'
OK
127.0.0.1:6379> JSON.NUMINCRBY k3 .a.* 1
(error) WRONGTYPE JSON element is not a number
127.0.0.1:6379> JSON.NUMINCRBY k3 .b.* 1
"2"
127.0.0.1:6379> JSON.NUMINCRBY k3 .c.* 1
(error) WRONGTYPE JSON element is not a number
127.0.0.1:6379> JSON.NUMINCRBY k3 .d.* 1
"4"
```

JSON.NUMMULTBY

Mengalikan nilai angka di jalur dengan angka tertentu.

Sintaksis

```
JSON.NUMMULTBY <key> <path> <number>
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (wajib) – Sebuah jalur JSON.
- angka (wajib) – Sebuah angka.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array string massal yang merepresentasikan nilai yang dihasilkan di setiap jalur.

- Jika nilai bukan angka, nilai yang akan ditampilkan adalah kosong.
- Kesalahan `WRONGTYPE` jika angka tidak dapat diuraikan.
- Kesalahan `OVERFLOW` jika hasilnya di luar rentang angka floating point presisi ganda 64-bit IEEE.
- `NONEXISTENT` jika kunci dokumen tidak ada.

Jika jalur adalah sintaksis terbatas:

- String massal yang merepresentasikan nilai yang dikembalikan.
- Jika memilih beberapa nilai, perintah ini mengembalikan hasil nilai yang terakhir diperbarui.
- Kesalahan `WRONGTYPE` jika nilai di jalur bukan angka.
- Kesalahan `WRONGTYPE` jika angka tidak dapat diuraikan.
- Kesalahan `OVERFLOW` jika hasilnya di luar rentang double 64-bit IEEE.
- `NONEXISTENT` jika kunci dokumen tidak ada.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k1 $.d[*] 2
"[2,4,6]"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[1],\"c\":[1,2],\"d\":[2,4,6]}"

127.0.0.1:6379> JSON.SET k1 $ '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k1 $.a[*] 2
"[]"
127.0.0.1:6379> JSON.NUMMULTBY k1 $.b[*] 2
"[2]"
127.0.0.1:6379> JSON.NUMMULTBY k1 $.c[*] 2
"[2,4]"
127.0.0.1:6379> JSON.NUMMULTBY k1 $.d[*] 2
"[2,4,6]"

127.0.0.1:6379> JSON.SET k2 $ '{"a":{}, "b":{"a":1}, "c":{"a":1, "b":2}, "d":{"a":1,
 "b":2, "c":3}}'
```

```
OK
127.0.0.1:6379> JSON.NUMMULTBY k2 $.a.* 2
"[]"
127.0.0.1:6379> JSON.NUMMULTBY k2 $.b.* 2
"[2]"
127.0.0.1:6379> JSON.NUMMULTBY k2 $.c.* 2
"[2,4]"
127.0.0.1:6379> JSON.NUMMULTBY k2 $.d.* 2
"[2,4,6]"

127.0.0.1:6379> JSON.SET k3 $ '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"b"}, "d":{"a":1, "b":"b", "c":3}}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k3 $.a.* 2
"[null]"
127.0.0.1:6379> JSON.NUMMULTBY k3 $.b.* 2
"[null,2]"
127.0.0.1:6379> JSON.NUMMULTBY k3 $.c.* 2
"[null,null]"
127.0.0.1:6379> JSON.NUMMULTBY k3 $.d.* 2
"[2,null,6]"
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k1 .d[1] 2
"4"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[1],\"c\":[1,2],\"d\":[1,4,3]}"

127.0.0.1:6379> JSON.SET k1 . '{"a":[], "b":[1], "c":[1,2], "d":[1,2,3]}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k1 .a[*] 2
(error) NONEXISTENT JSON path does not exist
127.0.0.1:6379> JSON.NUMMULTBY k1 .b[*] 2
"2"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[2],\"c\":[1,2],\"d\":[1,2,3]}"
127.0.0.1:6379> JSON.NUMMULTBY k1 .c[*] 2
"4"
127.0.0.1:6379> JSON.GET k1
```

```
"{\"a\":[],\"b\":[2],\"c\":[2,4],\"d\":[1,2,3]}"
127.0.0.1:6379> JSON.NUMMULTBY k1 .d[*] 2
"6"
127.0.0.1:6379> JSON.GET k1
"{\"a\":[],\"b\":[2],\"c\":[2,4],\"d\":[2,4,6]}"

127.0.0.1:6379> JSON.SET k2 . '{"a":{}, "b":{"a":1}, "c":{"a":1, "b":2}, "d":{"a":1,
 "b":2, "c":3}}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k2 .a.* 2
(error) NONEXISTENT JSON path does not exist
127.0.0.1:6379> JSON.NUMMULTBY k2 .b.* 2
"2"
127.0.0.1:6379> JSON.GET k2
"{\"a\":{},\"b\":{\"a\":2},\"c\":{\"a\":1,\"b\":2},\"d\":{\"a\":1,\"b\":2,\"c\":3}}"
127.0.0.1:6379> JSON.NUMMULTBY k2 .c.* 2
"4"
127.0.0.1:6379> JSON.GET k2
"{\"a\":{},\"b\":{\"a\":2},\"c\":{\"a\":2,\"b\":4},\"d\":{\"a\":1,\"b\":2,\"c\":3}}"
127.0.0.1:6379> JSON.NUMMULTBY k2 .d.* 2
"6"
127.0.0.1:6379> JSON.GET k2
"{\"a\":{},\"b\":{\"a\":2},\"c\":{\"a\":2,\"b\":4},\"d\":{\"a\":2,\"b\":4,\"c\":6}}"

127.0.0.1:6379> JSON.SET k3 . '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"b"}, "d":{"a":1, "b":"b", "c":3}}'
OK
127.0.0.1:6379> JSON.NUMMULTBY k3 .a.* 2
(error) WRONGTYPE JSON element is not a number
127.0.0.1:6379> JSON.NUMMULTBY k3 .b.* 2
"2"
127.0.0.1:6379> JSON.GET k3
"{\"a\":{\"a\":\"a\"},\"b\":{\"a\":\"a\",\"b\":2},\"c\":{\"a\":\"a\",\"b\":\"b\"},\"d
\":{\"a\":1,\"b\":\"b\",\"c\":3}}"
127.0.0.1:6379> JSON.NUMMULTBY k3 .c.* 2
(error) WRONGTYPE JSON element is not a number
127.0.0.1:6379> JSON.NUMMULTBY k3 .d.* 2
"6"
127.0.0.1:6379> JSON.GET k3
"{\"a\":{\"a\":\"a\"},\"b\":{\"a\":\"a\",\"b\":2},\"c\":{\"a\":\"a\",\"b\":\"b\"},\"d
\":{\"a\":2,\"b\":\"b\",\"c\":6}}"
```

JSON.OBJLEN

Mendapatkan jumlah kunci dalam nilai objek di jalur.

Sintaksis

```
JSON.OBJLEN <key> [path]
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (opsional) – Sebuah jalur JSON. Diatur secara default ke root jika tidak disediakan.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array integer yang merepresentasikan panjang array di setiap jalur.
- Jika nilai bukan objek, nilai yang akan dikembalikan adalah nilai yang kosong.
- Kosong jika kunci dokumen tidak ada.

Jika jalur adalah sintaksis terbatas:

- Integer, jumlah kunci dalam objek.
- Jika memilih beberapa objek, perintah ini mengembalikan panjang objek pertama.
- Kesalahan `WRONGTYPE` jika nilai di jalur bukan objek.
- Kesalahan `NONEXISTENT JSON` jika jalur tidak ada.
- Kosong jika kunci dokumen tidak ada.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 $ '{"a":{}, "b":{"a":"a"}, "c":{"a":"a", "b":"bb"}, "d":
{"a":1, "b":"b", "c":{"a":3,"b":4}}, "e":1}'
OK
127.0.0.1:6379> JSON.OBJLEN k1 $.a
1) (integer) 0
127.0.0.1:6379> JSON.OBJLEN k1 $.a.*
```

```
(empty array)
127.0.0.1:6379> JSON.OBJLEN k1 $.b
1) (integer) 1
127.0.0.1:6379> JSON.OBJLEN k1 $.b.*
1) (nil)
127.0.0.1:6379> JSON.OBJLEN k1 $.c
1) (integer) 2
127.0.0.1:6379> JSON.OBJLEN k1 $.c.*
1) (nil)
2) (nil)
127.0.0.1:6379> JSON.OBJLEN k1 $.d
1) (integer) 3
127.0.0.1:6379> JSON.OBJLEN k1 $.d.*
1) (nil)
2) (nil)
3) (integer) 2
127.0.0.1:6379> JSON.OBJLEN k1 $.*
1) (integer) 0
2) (integer) 1
3) (integer) 2
4) (integer) 3
5) (nil)
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":{}, "b":{"a":"a"}, "c":{"a":"a", "b":"bb"}, "d":
{"a":1, "b":"b", "c":{"a":3,"b":4}}, "e":1}'
OK
127.0.0.1:6379> JSON.OBJLEN k1 .a
(integer) 0
127.0.0.1:6379> JSON.OBJLEN k1 .a.*
(error) NONEXISTENT JSON path does not exist
127.0.0.1:6379> JSON.OBJLEN k1 .b
(integer) 1
127.0.0.1:6379> JSON.OBJLEN k1 .b.*
(error) WRONGTYPE JSON element is not an object
127.0.0.1:6379> JSON.OBJLEN k1 .c
(integer) 2
127.0.0.1:6379> JSON.OBJLEN k1 .c.*
(error) WRONGTYPE JSON element is not an object
127.0.0.1:6379> JSON.OBJLEN k1 .d
(integer) 3
```

```
127.0.0.1:6379> JSON.OBJLEN k1 .d.*
(integer) 2
127.0.0.1:6379> JSON.OBJLEN k1 .*
(integer) 0
```

JSON.OBJKEYS

Mendapatkan nama kunci pada nilai objek di jalur.

Sintaksis

```
JSON.OBJKEYS <key> [path]
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (opsional) – Sebuah jalur JSON. Diatur secara default ke root jika tidak disediakan.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array string massal. Setiap elemen adalah array kunci dalam objek yang cocok.
- Jika nilai bukan objek, nilai yang akan dikembalikan adalah nilai yang kosong.
- Kosong jika kunci dokumen tidak ada.

Jika jalur adalah sintaksis terbatas:

- Array string massal. Setiap elemen adalah nama kunci dalam objek.
- Jika memilih beberapa objek, perintah ini mengembalikan kunci objek pertama.
- Kesalahan `WRONGTYPE` jika nilai di jalur bukan objek.
- Kosong jika kunci dokumen tidak ada.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 $ '{"a":{}, "b":{"a":"a"}, "c":{"a":"a", "b":"bb"}, "d":
{"a":1, "b":"b", "c":{"a":3,"b":4}}, "e":1}'
```

```
OK
127.0.0.1:6379> JSON.OBJKEYS k1 $.*
1) (empty array)
2) 1) "a"
3) 1) "a"
   2) "b"
4) 1) "a"
   2) "b"
   3) "c"
5) (empty array)
127.0.0.1:6379> JSON.OBJKEYS k1 $.d
1) 1) "a"
   2) "b"
   3) "c"
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 $ '{"a":{}, "b":{"a":"a"}, "c":{"a":"a", "b":"bb"}, "d":
{"a":1, "b":"b", "c":{"a":3,"b":4}}, "e":1}'
OK
127.0.0.1:6379> JSON.OBJKEYS k1 .*
1) "a"
127.0.0.1:6379> JSON.OBJKEYS k1 .d
1) "a"
2) "b"
3) "c"
```

JSON.RESP

Mengembalikan nilai JSON pada jalur yang diberikan di Valkey atau Redis OSS Serialization Protocol (RESP). Jika nilainya adalah kontainer, responsnya adalah array RESP atau array bersarang.

- JSON kosong dipetakan ke String Massal Kosong RESP.
- Nilai JSON Boolean dipetakan ke masing-masing String Sederhana RESP.
- Angka integer dipetakan ke Integer RESP.
- Nomor floating point ganda IEEE 64-bit dipetakan ke String Massal RESP.
- String JSON dipetakan ke String Massal RESP.

- Array JSON direpresentasikan sebagai Array RESP, dengan elemen pertama adalah string sederhana [, diikuti oleh elemen array.
- Objek JSON direpresentasikan sebagai Array RESP, dengan elemen pertama adalah string sederhana {, diikuti oleh pasangan kunci-nilai, yang masing-masing adalah string massal RESP.

Sintaksis

```
JSON.RESP <key> [path]
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (opsional) – Sebuah jalur JSON. Diatur secara default ke root jika tidak disediakan.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array dari array. Setiap elemen array merepresentasikan bentuk RESP dari nilai pada satu jalur.
- Array kosong jika kunci dokumen tidak ada.

Jika jalur adalah sintaksis terbatas:

- Array yang merepresentasikan format RESP dari nilai pada jalur.
- Kosong jika kunci dokumen tidak ada.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 .
 '{"firstName":"John","lastName":"Smith","age":27,"weight":135.25,"isAlive":true,"address":
{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021-3100"},"phoneNumbers":
[{"type":"home","number":"212 555-1234"},{"type":"office","number":"646
 555-4567"}],"children":[],"spouse":null}'
OK

127.0.0.1:6379> JSON.RESP k1 $.address
```

```
1) 1) {
   2) 1) "street"
      2) "21 2nd Street"
   3) 1) "city"
      2) "New York"
   4) 1) "state"
      2) "NY"
   5) 1) "zipcode"
      2) "10021-3100"

127.0.0.1:6379> JSON.RESP k1 $.address.*
1) "21 2nd Street"
2) "New York"
3) "NY"
4) "10021-3100"

127.0.0.1:6379> JSON.RESP k1 $.phoneNumbers
1) 1) [
   2) 1) {
      2) 1) "type"
         2) "home"
      3) 1) "number"
         2) "555 555-1234"
   3) 1) {
      2) 1) "type"
         2) "office"
      3) 1) "number"
         2) "555 555-4567"

127.0.0.1:6379> JSON.RESP k1 $.phoneNumbers[*]
1) 1) {
   2) 1) "type"
      2) "home"
   3) 1) "number"
      2) "212 555-1234"
2) 1) {
   2) 1) "type"
      2) "office"
   3) 1) "number"
      2) "555 555-4567"
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 .
 '{"firstName":"John","lastName":"Smith","age":27,"weight":135.25,"isAlive":true,"address":
{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021-3100"},"phoneNumbers":
[{"type":"home","number":"212 555-1234"},{"type":"office","number":"646
 555-4567"}],"children":[],"spouse":null}'
OK

127.0.0.1:6379> JSON.RESP k1 .address
1) {
2) 1) "street"
   2) "21 2nd Street"
3) 1) "city"
   2) "New York"
4) 1) "state"
   2) "NY"
5) 1) "zipcode"
   2) "10021-3100"

127.0.0.1:6379> JSON.RESP k1
 1) {
 2) 1) "firstName"
    2) "John"
 3) 1) "lastName"
    2) "Smith"
 4) 1) "age"
    2) (integer) 27
 5) 1) "weight"
    2) "135.25"
 6) 1) "isAlive"
    2) true
 7) 1) "address"
    2) 1) {
       2) 1) "street"
          2) "21 2nd Street"
       3) 1) "city"
          2) "New York"
       4) 1) "state"
          2) "NY"
       5) 1) "zipcode"
          2) "10021-3100"
 8) 1) "phoneNumbers"
    2) 1) [
```

```
       2) 1) {
          2) 1) "type"
             2) "home"
          3) 1) "number"
             2) "212 555-1234"
       3) 1) {
          2) 1) "type"
             2) "office"
          3) 1) "number"
             2) "555 555-4567"
 9) 1) "children"
    2) 1) [
10) 1) "spouse"
    2) (nil)
```

JSON.SET

Menetapkan nilai JSON di jalur.

Jika jalur memanggil anggota objek:

- Jika elemen induk tidak ada, perintah ini mengembalikan kesalahan NONEXISTENT.
- Jika elemen induk ada, tetapi bukan objek, perintah ini menampilkan ERROR.
- Jika elemen induk ada dan merupakan objek:
  - Jika anggota tidak ada, anggota baru akan ditambahkan ke objek induk jika dan hanya jika objek induk adalah turunan terakhir di jalur. Jika tidak, perintah ini mengembalikan kesalahan NONEXISTENT.
  - Jika anggota ada, nilainya akan diganti dengan nilai JSON.

Jika jalur memanggil indeks array:

- Jika elemen induk tidak ada, perintah ini mengembalikan kesalahan NONEXISTENT.
- Jika elemen induk ada tetapi bukan array, perintah ini mengembalikan ERROR.
- Jika elemen induk ada tetapi indeks di luar batas, perintah ini mengembalikan kesalahan OUTOFBOUNDARIES.
- Jika elemen induk ada dan indeks valid, elemen akan diganti dengan nilai JSON baru.

Jika jalur memanggil objek atau array, nilai (objek atau array) akan digantikan oleh nilai JSON baru.

Sintaksis

```
JSON.SET <key> <path> <json> [NX | XX]
```

[NX | XX] Di mana Anda dapat memiliki 0 atau 1 pengidentifikasi [NX | XX].

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (wajib) – Sebuah jalur JSON. Untuk kunci baru, jalur JSON harus menjadi root “.”.
- NX (opsional) - Jika jalurnya adalah root, tetapkan nilainya hanya jika kuncinya tidak ada. Artinya, masukkan dokumen baru. Jika jalur bukan root, atur nilainya hanya jika jalur tidak ada. Artinya, masukkan nilai ke dalam dokumen.
- XX (opsional) - Jika jalurnya adalah root, tetapkan nilainya hanya jika kuncinya ada. Artinya, ganti dokumen yang ada. Jika jalur bukan root, atur nilainya hanya jika jalur ada. Artinya, perbarui nilai yang ada.

Nilai yang ditampilkan

- String sederhana 'OK' jika berhasil.
- Kosong jika kondisi NX atau XX tidak terpenuhi.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":{"a":1, "b":2, "c":3}}'
OK
127.0.0.1:6379> JSON.SET k1 $.a.* '0'
OK
127.0.0.1:6379> JSON.GET k1
"{\"a\":{\"a\":0,\"b\":0,\"c\":0}}"

127.0.0.1:6379> JSON.SET k2 . '{"a": [1,2,3,4,5]}'
OK
127.0.0.1:6379> JSON.SET k2 $.a[*] '0'
OK
127.0.0.1:6379> JSON.GET k2
```

```
"{\"a\":[0,0,0,0,0]}"
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 . '{"c":{"a":1, "b":2}, "e": [1,2,3,4,5]}'
OK
127.0.0.1:6379> JSON.SET k1 .c.a '0'
OK
127.0.0.1:6379> JSON.GET k1
"{\"c\":{\"a\":0,\"b\":2},\"e\":[1,2,3,4,5]}"
127.0.0.1:6379> JSON.SET k1 .e[-1] '0'
OK
127.0.0.1:6379> JSON.GET k1
"{\"c\":{\"a\":0,\"b\":2},\"e\":[1,2,3,4,0]}"
127.0.0.1:6379> JSON.SET k1 .e[5] '0'
(error) OUTOFBOUNDARIES Array index is out of bounds
```

JSON.STRAPPEND

Menambahkan string ke string JSON di jalur.

Sintaksis

```
JSON.STRAPPEND <key> [path] <json_string>
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (opsional) – Sebuah jalur JSON. Default ke root jika tidak disediakan.
- json_string (wajib) - Representasi JSON dari string. Perhatikan bahwa string JSON harus diberi tanda kutip. Misalnya: '"contoh string"'.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array integer yang merepresentasikan panjang baru string di setiap jalur.
- Jika nilai di jalur bukan string, nilai yang akan dikembalikan adalah kosong.
- Kesalahan SYNTAXERR jika argumen input json bukan string JSON yang valid.

- Kesalahan NONEXISTENT jika jalur tidak ada.

Jika jalur adalah sintaksis terbatas:

- Integer, panjang baru string.
- Jika memilih beberapa nilai string, perintah ini mengembalikan panjang baru dari string yang terakhir diperbarui.
- Kesalahan WRONGTYPE jika nilai di jalur bukan string.
- Kesalahan WRONGTYPE jika argumen input json bukan string JSON yang valid.
- Kesalahan NONEXISTENT jika jalur tidak ada.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 $ '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"bb"}, "d":{"a":1, "b":"b", "c":3}}'
OK
127.0.0.1:6379> JSON.STRAPPEND k1 $.a.a '"a"'
1) (integer) 2
127.0.0.1:6379> JSON.STRAPPEND k1 $.a.* '"a"'
1) (integer) 3
127.0.0.1:6379> JSON.STRAPPEND k1 $.b.* '"a"'
1) (integer) 2
2) (nil)
127.0.0.1:6379> JSON.STRAPPEND k1 $.c.* '"a"'
1) (integer) 2
2) (integer) 3
127.0.0.1:6379> JSON.STRAPPEND k1 $.c.b '"a"'
1) (integer) 4
127.0.0.1:6379> JSON.STRAPPEND k1 $.d.* '"a"'
1) (nil)
2) (integer) 2
3) (nil)
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"bb"}, "d":{"a":1, "b":"b", "c":3}}'
```

```
OK
127.0.0.1:6379> JSON.STRAPPEND k1 .a.a '"a"'
(integer) 2
127.0.0.1:6379> JSON.STRAPPEND k1 .a.* '"a"'
(integer) 3
127.0.0.1:6379> JSON.STRAPPEND k1 .b.* '"a"'
(integer) 2
127.0.0.1:6379> JSON.STRAPPEND k1 .c.* '"a"'
(integer) 3
127.0.0.1:6379> JSON.STRAPPEND k1 .c.b '"a"'
(integer) 4
127.0.0.1:6379> JSON.STRAPPEND k1 .d.* '"a"'
(integer) 2
```

JSON.STRLEN

Mendapatkan panjang nilai string JSON di jalur.

Sintaksis

```
JSON.STRLEN <key> [path]
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (opsional) – Sebuah jalur JSON. Diatur secara default ke root jika tidak disediakan.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array integer yang merepresentasikan panjang nilai array di setiap jalur.
- Jika nilai bukan string, nilai yang akan dikembalikan adalah kosong.
- Kosong jika kunci dokumen tidak ada.

Jika jalur adalah sintaksis terbatas:

- Integer, panjang string.
- Jika memilih beberapa nilai string, perintah ini mengembalikan panjang string pertama.
- Kesalahan `WRONGTYPE` jika nilai di jalur bukan string.

- Kesalahan `NONEXISTENT` jika jalur tidak ada.
- Kosong jika kunci dokumen tidak ada.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 $ '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"bb"}, "d":{"a":1, "b":"b", "c":3}}'
OK
127.0.0.1:6379> JSON.STRLEN k1 $.a.a
1) (integer) 1
127.0.0.1:6379> JSON.STRLEN k1 $.a.*
1) (integer) 1
127.0.0.1:6379> JSON.STRLEN k1 $.c.*
1) (integer) 1
2) (integer) 2
127.0.0.1:6379> JSON.STRLEN k1 $.c.b
1) (integer) 2
127.0.0.1:6379> JSON.STRLEN k1 $.d.*
1) (nil)
2) (integer) 1
3) (nil)
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 $ '{"a":{"a":"a"}, "b":{"a":"a", "b":1}, "c":{"a":"a",
 "b":"bb"}, "d":{"a":1, "b":"b", "c":3}}'
OK
127.0.0.1:6379> JSON.STRLEN k1 .a.a
(integer) 1
127.0.0.1:6379> JSON.STRLEN k1 .a.*
(integer) 1
127.0.0.1:6379> JSON.STRLEN k1 .c.*
(integer) 1
127.0.0.1:6379> JSON.STRLEN k1 .c.b
(integer) 2
127.0.0.1:6379> JSON.STRLEN k1 .d.*
(integer) 1
```

JSON.TOGGLE

Mengalihkan nilai Boolean antara true dan false di jalur.

Sintaksis

```
JSON.TOGGLE <key> [path]
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (opsional) – Sebuah jalur JSON. Diatur secara default ke root jika tidak disediakan.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array integer (0 – false, 1 – true) yang merepresentasikan nilai Boolean yang dihasilkan di setiap jalur.
- Jika nilai bukan nilai Boolean, nilai yang akan dikembalikan adalah kosong.
- `NONEXISTENT` jika kunci dokumen tidak ada.

Jika jalur adalah sintaksis terbatas:

- String ("true" /"false") yang merepresentasikan nilai Boolean yang dihasilkan.
- `NONEXISTENT` jika kunci dokumen tidak ada.
- Kesalahan `WRONGTYPE` jika nilai di jalur bukan nilai Boolean.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 . '{"a":true, "b":false, "c":1, "d":null, "e":"foo", "f":
[], "g":{}}'
OK
127.0.0.1:6379> JSON.TOGGLE k1 $.*
1) (integer) 0
2) (integer) 1
3) (nil)
```

```
4) (nil)
5) (nil)
6) (nil)
7) (nil)
127.0.0.1:6379> JSON.TOGGLE k1 $.*
1) (integer) 1
2) (integer) 0
3) (nil)
4) (nil)
5) (nil)
6) (nil)
7) (nil)
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 . true
OK
127.0.0.1:6379> JSON.TOGGLE k1
"false"
127.0.0.1:6379> JSON.TOGGLE k1
"true"

127.0.0.1:6379> JSON.SET k2 . '{"isAvailable": false}'
OK
127.0.0.1:6379> JSON.TOGGLE k2 .isAvailable
"true"
127.0.0.1:6379> JSON.TOGGLE k2 .isAvailable
"false"
```

JSON.TYPE

Melaporkan jenis nilai di jalur yang diberikan.

Sintaksis

```
JSON.TYPE <key> [path]
```

- kunci (wajib) - Kunci Valkey atau Redis OSS dari jenis dokumen JSON.
- jalur (opsional) – Sebuah jalur JSON. Diatur secara default ke root jika tidak disediakan.

Nilai yang ditampilkan

Jika jalur adalah sintaksis yang ditingkatkan:

- Array string yang merepresentasikan jenis nilai di setiap jalur. Jenisnya adalah salah satu dari {"null", "boolean", "string", "number", "integer", "object" dan "array"}.
- Jika jalur tidak ada, nilai yang ditampilkan adalah kosong.
- Array kosong jika kunci dokumen tidak ada.

Jika jalur adalah sintaksis terbatas:

- String, jenis nilai
- Kosong jika kunci dokumen tidak ada.
- Kosong jika jalur JSON tidak valid atau tidak ada.

Contoh

Sintaksis jalur yang ditingkatkan:

```
127.0.0.1:6379> JSON.SET k1 . '[1, 2.3, "foo", true, null, {}, []]'
OK
127.0.0.1:6379> JSON.TYPE k1 $[*]
1) integer
2) number
3) string
4) boolean
5) null
6) object
7) array
```

Sintaksis jalur terbatas:

```
127.0.0.1:6379> JSON.SET k1 .
 '{"firstName":"John","lastName":"Smith","age":27,"weight":135.25,"isAlive":true,"address":
{"street":"21 2nd Street","city":"New
 York","state":"NY","zipcode":"10021-3100"},"phoneNumbers":
[{"type":"home","number":"212 555-1234"},{"type":"office","number":"646
 555-4567"}],"children":[],"spouse":null}'
OK
```

```
127.0.0.1:6379> JSON.TYPE k1
object
127.0.0.1:6379> JSON.TYPE k1 .children
array
127.0.0.1:6379> JSON.TYPE k1 .firstName
string
127.0.0.1:6379> JSON.TYPE k1 .age
integer
127.0.0.1:6379> JSON.TYPE k1 .weight
number
127.0.0.1:6379> JSON.TYPE k1 .isAlive
boolean
127.0.0.1:6379> JSON.TYPE k1 .spouse
null
```

Menandai sumber daya Anda ElastiCache

Untuk membantu mengelola cluster dan ElastiCache sumber daya lainnya, Anda dapat menetapkan metadata Anda sendiri ke setiap sumber daya dalam bentuk tag. Tag memungkinkan Anda untuk mengkategorikan AWS sumber daya Anda dengan cara yang berbeda, misalnya, berdasarkan tujuan, pemilik, atau lingkungan. Hal ini berguna ketika Anda memiliki banyak sumber daya dengan jenis yang sama—Anda dapat dengan cepat mengidentifikasi sumber daya tertentu berdasarkan tag yang telah Anda tetapkan. Topik ini menjelaskan tag dan menunjukkan cara membuatnya.

> **Warning**
>
> Sebagai praktik terbaik, sebaiknya Anda tidak menyertakan data sensitif ke dalam tag.

Dasar-dasar tag

Tag adalah label yang Anda tetapkan ke AWS sumber daya. Setiap tag terdiri dari kunci dan nilai opsional, yang keduanya Anda tentukan. Tag memungkinkan Anda untuk mengkategorikan AWS sumber daya Anda dengan cara yang berbeda, misalnya, berdasarkan tujuan atau pemilik. Misalnya, Anda dapat menentukan satu set tag untuk ElastiCache klaster akun Anda yang membantu Anda melacak pemilik dan grup pengguna setiap instans.

Sebaiknya rancang serangkaian kunci tag yang memenuhi kebutuhan setiap jenis sumber daya. Penggunaan set kunci tag yang konsisten akan memudahkan manajemen sumber daya Anda. Anda

dapat mencari dan memfilter sumber daya berdasarkan tag yang Anda tambahkan. Untuk informasi selengkapnya tentang cara mengimplementasikan strategi pemberian tag sumber daya yang efektif, lihat Laporan resmi AWS Praktik Terbaik Pemberian Tag.

Tag tidak memiliki arti semantik ElastiCache dan ditafsirkan secara ketat sebagai serangkaian karakter. Selain itu, tag tidak secara otomatis ditetapkan ke sumber daya Anda. Anda dapat mengedit kunci dan nilai tag, serta menghapus tag dari sumber daya kapan saja. Anda dapat menetapkan nilai tag ke `null`. Jika Anda menambahkan tag yang memiliki kunci yang sama dengan tag yang sudah ada di sumber daya tersebut, nilai yang baru akan menimpa nilai yang lama. Jika Anda menghapus sumber daya, semua tag untuk sumber daya tersebut juga akan dihapus. Selain itu, jika Anda menambahkan atau menghapus tag di grup replikasi, semua simpul dalam grup replikasi itu juga akan mendapatkan penambahan atau penghapusan tag yang sama.

Anda dapat bekerja dengan tag menggunakan AWS Management Console, the AWS CLI, dan ElastiCache API.

Jika Anda menggunakan IAM, Anda dapat mengontrol pengguna mana di AWS akun Anda yang memiliki izin untuk membuat, mengedit, atau menghapus tag. Untuk informasi selengkapnya, lihat Izin tingkat sumber daya.

Sumber daya yang dapat Anda beri tag

Anda dapat menandai sebagian besar ElastiCache sumber daya yang sudah ada di akun Anda. Tabel di bawah ini mencantumkan sumber daya yang mendukung pemberian tag. Jika Anda menggunakan AWS Management Console, Anda dapat menerapkan tag ke sumber daya dengan menggunakan Editor Tag. Beberapa layar sumber daya memungkinkan Anda menentukan tag untuk sebuah sumber daya saat sumber daya tersebut dibuat; misalnya, tag dengan kunci Nama dan nilai yang Anda tentukan. Dalam kebanyakan kasus, konsol menerapkan tag segera setelah sumber daya dibuat (bukan selama pembuatan sumber daya). Konsol dapat mengatur sumber daya sesuai dengan tag Nama, tetapi tag ini tidak memiliki arti semantik untuk layanan. ElastiCache

Selain itu, beberapa tindakan pembuatan sumber daya memungkinkan Anda menentukan tag untuk sumber daya saat sumber daya tersebut dibuat. Jika tag tidak dapat diterapkan selama pembuatan sumber daya, kami akan mengembalikan proses pembuatan sumber daya. Hal ini untuk memastikan bahwa sumber daya dibuat dengan tag atau tidak akan dibuat sama sekali, dan tidak akan ada sumber daya yang dibiarkan tidak bertanda. Dengan memberikan tag pada sumber daya saat pembuatan, Anda tidak perlu menjalankan skrip pemberian tag kustom setelah pembuatan sumber daya.

Jika Anda menggunakan Amazon ElastiCache API, AWS CLI, atau AWS SDK, Anda dapat menggunakan `Tags` parameter pada tindakan ElastiCache API yang relevan untuk menerapkan tag. File tersebut adalah:

- `CreateServerlessCache`
- `CreateCacheCluster`
- `CreateReplicationGroup`
- `CopyServerlessCacheSnapshot`
- `CopySnapshot`
- `CreateCacheParameterGroup`
- `CreateCacheSecurityGroup`
- `CreateCacheSubnetGroup`
- `CreateServerlessCacheSnapshot`
- `CreateSnapshot`
- `CreateUserGroup`
- `CreateUser`
- `PurchaseReservedCacheNodesOffering`

Tabel berikut menjelaskan ElastiCache sumber daya yang dapat diberi tag, dan sumber daya yang dapat diberi tag saat pembuatan menggunakan ElastiCache API, AWS CLI, atau SDK. AWS

Menandai dukungan untuk sumber daya ElastiCache

Mendukung tag	Mendukung pemberian tag saat pembuatan
Ya	Ya
Ya	Ya
Ya	Ya

Mendukung tag	Mendukung pemberian tag saat pembuatan
Ya	Ya
Ya	Ya
Ya	Ya
Ya	Ya
Ya	Ya
Ya	Ya
Ya	Ya
Ya	Ya

> **Note**
>
> Anda tidak dapat memberikan tag pada Penyimpanan Data Global.

Anda dapat menerapkan izin tingkat sumber daya berbasis tag dalam kebijakan IAM Anda ke tindakan ElastiCache API yang mendukung penandaan pada pembuatan untuk menerapkan kontrol terperinci atas pengguna dan grup yang dapat menandai sumber daya saat pembuatan. Sumber daya Anda diamankan secara tepat sejak pembuatan—tag yang diterapkan segera ke sumber daya Anda. Oleh karena itu, izin tingkat sumber daya berbasis tag apa pun yang mengontrol penggunaan sumber daya akan langsung diterapkan. Sumber daya Anda dapat dilacak dan dilaporkan dengan lebih akurat. Anda dapat menerapkan penggunaan pemberian tag pada sumber daya baru serta mengontrol kunci dan nilai tag mana yang ditetapkan pada sumber daya Anda.

Untuk informasi selengkapnya, lihat Contoh pemberian tag sumber daya.

Untuk informasi selanjutnya tentang pemberian tag sumber daya Anda untuk penagihan, lihat Memantau biaya dengan tag alokasi biaya.

Memberi tag pada cache dan snapshot

Aturan berikut berlaku untuk pemberian tag sebagai bagian dari operasi permintaan:

- CreateReplicationGroup:
  - Jika parameter `--primary-cluster-id` dan `--tags` termasuk dalam permintaan, tag permintaan akan ditambahkan ke grup replikasi dan disebarkan ke semua klaster cache di grup replikasi. Jika klaster cache primer sebelumnya sudah memiliki tag, tag ini akan ditimpa dengan tag permintaan agar tag menjadi konsisten di semua simpul.

    Jika tidak ada tag permintaan, tag klaster cache primer akan ditambahkan ke grup replikasi dan disebarkan ke semua klaster cache.
  - Jika `--snapshot-name` atau `--serverless-cache-snapshot-name` disediakan:

    Jika tag disertakan dalam permintaan, grup replikasi hanya akan ditandai dengan tag tersebut. Jika tidak ada tag yang disertakan dalam permintaan, tag snapshot akan ditambahkan ke grup replikasi.
  - Jika `--global-replication-group-id` disediakan:

    Jika tag disertakan dalam permintaan, tag permintaan akan ditambahkan ke grup replikasi dan disebarkan ke semua klaster cache.
- CreateCacheCluster :
  - Jika `--replication-group-id` disediakan:

    Jika tag disertakan dalam permintaan, klaster cache akan diberi tag hanya dengan tag tersebut. Jika tidak ada tag yang disertakan dalam permintaan, klaster cache akan mewarisi tag grup replikasi, bukan tag dari klaster cache primer.
  - Jika `--snapshot-name` disediakan:

    Jika tag disertakan dalam permintaan, klaster cache akan diberi tag hanya dengan tag tersebut. Jika tidak ada tag yang disertakan dalam permintaan, tag snapshot akan ditambahkan ke klaster cache.
- CreateServerlessCache :

  - Jika tag disertakan dalam permintaan, hanya tag permintaan yang akan ditambahkan ke cache nirserver.
- CreateSnapshot :
  - Jika `--replication-group-id` disediakan:

    Jika tag disertakan dalam permintaan, hanya tag permintaan yang akan ditambahkan ke snapshot. Jika tidak ada tag yang disertakan dalam permintaan, tag grup replikasi akan ditambahkan ke snapshot.
  - Jika `--cache-cluster-id` disediakan:

    Jika tag disertakan dalam permintaan, hanya tag permintaan yang akan ditambahkan ke snapshot. Jika tidak ada tag yang disertakan dalam permintaan, tag klaster cache akan ditambahkan ke snapshot.
  - Untuk snapshot otomatis:

    Tag akan disebarkan dari tag grup replikasi.
- CreateServerlessCacheSnapshot :
  - Jika tag disertakan dalam permintaan, hanya tag permintaan yang akan ditambahkan ke snapshot cache nirserver.
- CopySnapshot :
  - Jika tag disertakan dalam permintaan, hanya tag permintaan yang akan ditambahkan ke snapshot. Jika tidak ada tag yang disertakan dalam permintaan, tag snapshot sumber akan ditambahkan ke snapshot salinan.
- CopyServerlessCacheSnapshot :
  - Jika tag disertakan dalam permintaan, hanya tag permintaan yang akan ditambahkan ke snapshot cache nirserver.
- AddTagsToResourcedan RemoveTagsFromResource:
  - Tag akan added/removed berasal dari grup replikasi dan tindakan akan disebarkan ke semua cluster di grup replikasi.

> **Note**
>
> AddTagsToResourcedan RemoveTagsFromResourcetidak dapat digunakan untuk parameter default dan grup keamanan.

- IncreaseReplicaCount dan ModifyReplicationGroupShardConfiguration:
  - Semua klaster baru yang ditambahkan ke grup replikasi akan memiliki tag yang sama dengan grup replikasi.

Batasan tag

Batasan dasar berikut berlaku untuk tag:

- Jumlah maksimum tag per sumber daya – 50
- Untuk setiap sumber daya, setiap kunci tag harus unik, dan setiap kunci tag hanya dapat memiliki satu nilai.
- Panjang kunci maksimum – 128 karakter Unicode dalam UTF-8.
- Panjang nilai maksimum – 256 karakter Unicode dalam UTF-8.
- Meskipun ElastiCache memungkinkan karakter apa pun dalam tagnya, layanan lain dapat membatasi. Karakter yang diizinkan di semua layanan adalah huruf, angka, dan spasi yang dapat direpresentasikan dalam UTF-8, serta karakter berikut: + - = . _ : / @
- Kunci dan nilai tag peka huruf besar dan kecil.
- `aws:`Awalan dicadangkan untuk AWS digunakan. Jika tag memiliki kunci tag dengan awalan ini, Anda tidak dapat mengedit atau menghapus kunci atau nilai tag tersebut. Tag dengan awalan `aws:` tidak dihitung terhadap tag per batas sumber daya.

Anda tidak dapat mengakhiri, menghentikan, atau menghapus sumber daya berdasarkan tandanya saja; Anda harus menentukan pengidentifikasi sumber daya tersebut. Misalnya, untuk menghapus snapshot yang Anda beri tag dengan tag kunci yang disebut `DeleteMe`, Anda harus menggunakan tindakan `DeleteSnapshot` dengan pengidentifikasi sumber daya snapshot tersebut, seperti `snap-1234567890abcdef0`.

Untuk informasi selengkapnya tentang ElastiCache sumber daya yang dapat Anda beri tag, lihatSumber daya yang dapat Anda beri tag.

Contoh pemberian tag sumber daya

- Membuat cache tanpa server menggunakan tag. Contoh ini menggunakan Memcached sebagai mesin.

```
aws elasticache create-serverless-cache \
```

```
    --serverless-cache-name CacheName \
    --engine memcached
    --tags Key="Cost Center", Value="1110001" Key="project",Value="XYZ"
```

- Menambahkan tag ke cache nirserver

```
aws elasticache add-tags-to-resource \
--resource-name arn:aws:elasticache:us-east-1:111111222233:serverlesscache:my-cache \
--tags Key="project",Value="XYZ" Key="Elasticache",Value="Service"
```

- Menambahkan tanda ke Grup Replikasi.

```
aws elasticache add-tags-to-resource \
--resource-name arn:aws:elasticache:us-east-1:111111222233:replicationgroup:my-rg \
--tags Key="project",Value="XYZ" Key="Elasticache",Value="Service"
```

- Membuat Klaster Cache menggunakan tanda.

```
aws elasticache create-cache-cluster \
--cluster-id testing-tags \
--cluster-description cluster-test \
--cache-subnet-group-name test \
--cache-node-type cache.t2.micro \
--engine valkey \
--tags Key="project",Value="XYZ" Key="Elasticache",Value="Service"
```

- Membuat Klaster Cache menggunakan tag. Contoh ini menggunakan Redis sebagai mesin.

```
aws elasticache create-cache-cluster \
--cluster-id testing-tags \
--cluster-description cluster-test \
--cache-subnet-group-name test \
--cache-node-type cache.t2.micro \
--engine valkey \
--tags Key="project",Value="XYZ" Key="Elasticache",Value="Service"
```

- Membuat snapshot nirserver dengan tag. Contoh ini menggunakan Memcached sebagai mesin.

```
aws elasticache create-serverless-cache-snapshot \
--serverless-cache-name testing-tags \
--serverless-cache-snapshot-name bkp-testing-tags-scs \
```

```
--tags Key="work",Value="foo"
```

- Membuat Snapshot dengan tag.

  Snapshot saat ini hanya tersedia untuk Redis. Untuk kasus ini, jika Anda menambahkan tag pada permintaan, bahkan jika grup replikasi berisi tag, snapshot hanya akan menerima tag permintaan.

```
aws elasticache create-snapshot \
--replication-group-id testing-tags \
--snapshot-name bkp-testing-tags-rg \
--tags Key="work",Value="foo"
```

Contoh kebijakan kontrol akses Berbasis Tag

1. Mengizinkan tindakan `AddTagsToResource` untuk klaster hanya jika klaster memiliki tag Project=XYZ.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "elasticache:AddTagsToResource",
            "Resource": [
                "arn:aws:elasticache:*:*:cluster:*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Project": "XYZ"
                }
            }
        }
    ]
}
```

2. Mengizinkan tindakan `RemoveTagsFromResource` dari grup replikasi jika grup berisi tag Project dan Service serta kunci yang berbeda dari Project dan Service.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "elasticache:RemoveTagsFromResource",
            "Resource": [
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Service": "Elasticache",
                    "aws:ResourceTag/Project": "XYZ"
                },
                "ForAnyValue:StringNotEqualsIgnoreCase": {
                    "aws:TagKeys": [
                        "Project",
                        "Service"
                    ]
                }
            }
        }
    ]
}
```

3. Mengizinkan `AddTagsToResource` untuk sumber daya apa pun hanya jika tag berbeda dari Project dan Service.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "elasticache:AddTagsToResource",
            "Resource": [
                "arn:aws:elasticache:*:*:*:*"
            ],
            "Condition": {
                "ForAnyValue:StringNotEqualsIgnoreCase": {
```

```
                    "aws:TagKeys": [
                        "Service",
                        "Project"
                    ]
                }
            }
        }
    ]
}
```

4. Menolak tindakan `CreateReplicationGroup` jika permintaan memiliki `Tag Project=Foo`.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "elasticache:CreateReplicationGroup",
            "Resource": [
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/Project": "Foo"
                }
            }
        }
    ]
}
```

5. Menolak tindakan `CopySnapshot` jika sumber snapshot memiliki tag Project=XYZ dan tag permintaan adalah Service=Elasticache.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "elasticache:CopySnapshot",
            "Resource": [
```

```
                "arn:aws:elasticache:*:*:snapshot:*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Project": "XYZ",
                    "aws:RequestTag/Service": "Elasticache"
                }
            }
        }
    ]
}
```

6. Menolak `CreateCacheCluster` tindakan jika tag permintaan `Project` hilang atau tidak sama dengan `Dev`, `QA` atau `Prod`.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
          {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*",
                "arn:aws:elasticache:*:*:securitygroup:*",
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ]
        },
        {
            "Effect": "Deny",
            "Action": [
                "elasticache:CreateCacheCluster"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:cluster:*"
            ],
            "Condition": {
                "Null": {
                    "aws:RequestTag/Project": "true"
                }
```

```
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster",
                "elasticache:AddTagsToResource"
            ],
            "Resource": "arn:aws:elasticache:*:*:cluster:*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/Project": [
                        "Dev",
                        "Prod",
                        "QA"
                    ]
                }
            }
        }
    ]
}
```

Untuk informasi terkait kunci kondisi ini, lihat Menggunakan kunci kondisi.

Memantau biaya dengan tag alokasi biaya

Saat menambahkan tag alokasi biaya ke sumber daya di Amazon ElastiCache, Anda dapat melacak biaya dengan mengelompokkan pengeluaran pada faktur berdasarkan nilai tag sumber daya.

Tag alokasi ElastiCache biaya adalah pasangan kunci-nilai yang Anda tentukan dan kaitkan dengan sumber daya. ElastiCache Kunci dan nilai peka terhadap huruf besar dan kecil. Anda dapat menggunakan kunci tag untuk menentukan kategori, dan nilai tag dapat berupa item dalam kategori tersebut. Misalnya, Anda dapat menentukan kunci tag `CostCenter` dan nilai tag `10010`, yang menunjukkan bahwa sumber daya tersebut ditetapkan ke pusat pembiayaan 10010. Anda juga dapat menggunakan tag untuk menunjukkan bahwa sumber daya sedang digunakan untuk pengujian atau produksi menggunakan kunci seperti `Environment` dan nilai seperti `test` atau `production`. Sebaiknya gunakan kumpulan kunci tag yang konsisten untuk mempermudah pelacakan biaya yang terkait dengan sumber daya Anda.

Gunakan tag alokasi biaya untuk mengatur AWS tagihan Anda untuk mencerminkan struktur biaya Anda sendiri. Untuk melakukan ini, daftar untuk mendapatkan tagihan AWS akun Anda dengan nilai

kunci tag yang disertakan. Kemudian, untuk melihat biaya sumber daya gabungan, atur informasi penagihan Anda sesuai dengan sumber daya Anda dengan nilai kunci tag yang sama. Misalnya, Anda dapat memberikan tag pada beberapa sumber daya dengan nama aplikasi tertentu, kemudian mengatur informasi penagihan untuk melihat biaya total aplikasi tersebut pada beberapa layanan.

Anda juga dapat menggabungkan tag untuk melacak biaya dengan tingkat detail yang lebih besar. Misalnya, untuk melacak biaya layanan Anda menurut wilayah, Anda dapat menggunakan kunci tag `Service` dan `Region`. Di salah satu sumber daya Anda mungkin memiliki nilai `ElastiCache` dan `Asia Pacific (Singapore)`, serta di sumber daya lain Anda mempunyai nilai `ElastiCache` dan `Europe (Frankfurt)`. Anda kemudian dapat melihat total ElastiCache biaya Anda dipecah berdasarkan wilayah. Untuk informasi selengkapnya, lihat Menggunakan Tag Alokasi Biaya dalam Panduan Pengguna AWS Billing .

Anda dapat menambahkan tag alokasi ElastiCache biaya ke cluster yang ElastiCache dirancang sendiri. Saat Anda menambahkan, menampilkan daftar, mengubah, menyalin, atau menghapus tag, operasi tersebut hanya akan diterapkan ke klaster yang ditentukan.

Karakteristik ElastiCache tag alokasi biaya

- Tag alokasi biaya diterapkan ke ElastiCache sumber daya yang ditentukan dalam operasi CLI dan API sebagai ARN. Jenis sumber daya akan berupa "klaster".

  Contoh ARN: `arn:aws:elasticache:<region>:<customer-id>:<resource-type>:<resource-name>`

  Contoh arn: `arn:aws:elasticache:us-west-2:1234567890:cluster:my-cluster`
- Kunci tag adalah nama tag yang wajib diisi. Nilai string kunci dapat terdiri dari 1 hingga 128 karakter Unicode dan tidak boleh diawali dengan `aws:`. String dapat berisi hanya kumpulan huruf Unicode, angka, spasi kosong, garis bawah (_), titik (.), titik dua (:), garis miring terbalik (\), tanda sama dengan (=), tanda plus (+), tanda hubung (-), atau tanda at (@).

- Nilai tag adalah nilai tag opsional. Nilai string dari nilai dapat terdiri dari 1 hingga 256 karakter Unicode dan tidak boleh diawali dengan `aws:`. String dapat berisi hanya kumpulan huruf Unicode, angka, spasi kosong, garis bawah (_), titik (.), titik dua (:), garis miring terbalik (\), tanda sama dengan (=), tanda plus (+), tanda hubung (-), atau tanda at (@).

- ElastiCache Sumber daya dapat memiliki maksimal 50 tag.

- Nilai tidak harus unik dalam kumpulan tag. Misalnya, Anda dapat memiliki kumpulan tag dengan kunci `Service` dan `Application` yang memiliki nilai `ElastiCache`.

AWS tidak menerapkan makna semantik apa pun pada tag Anda. Tag ditafsirkan secara ketat sebagai string karakter. AWS tidak secara otomatis mengatur tag apa pun pada ElastiCache sumber daya apa pun.

Mengelola tag alokasi biaya Anda menggunakan AWS CLI

Anda dapat menggunakan AWS CLI untuk menambah, memodifikasi, atau menghapus tag alokasi biaya.

Tag alokasi biaya diterapkan ke ElastiCache cluster. Klaster yang akan diberi tag ditentukan menggunakan ARN (Amazon Resource Name).

arn Sampel: `arn:aws:elasticache:us-west-2:1234567890:cluster:my-cluster`

Topik

Listing tag menggunakan AWS CLI

Anda dapat menggunakan tag AWS CLI untuk daftar pada ElastiCache sumber daya yang ada dengan menggunakan list-tags-for-resource operasi.

Kode berikut menggunakan daftar tag pada cluster Memcached di wilayah `my-cluster` us-west-2. AWS CLI

Untuk Linux, macOS, atau Unix:

```
aws elasticache list-tags-for-resource \
  --resource-name arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster
```

Untuk Windows:

```
aws elasticache list-tags-for-resource ^
  --resource-name arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster
```

Kode berikut menggunakan AWS CLI untuk daftar tag pada node Valkey atau Redis OSS di `my-cluster` cluster `my-cluster-001` di wilayah us-west-2.

Untuk Linux, macOS, atau Unix:

```
aws elasticache list-tags-for-resource \
  --resource-name arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster-001
```

Untuk Windows:

```
aws elasticache list-tags-for-resource ^
  --resource-name arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster-001
```

Output dari operasi ini akan terlihat seperti berikut ini, daftar semua tag pada sumber daya.

```
{
   "TagList": [
      {
         "Value": "10110",
         "Key": "CostCenter"
      },
      {
         "Value": "EC2",
         "Key": "Service"
      }
   ]
}
```

Jika tidak ada tag pada sumber daya, output akan kosong TagList.

```
{
   "TagList": []
}
```

Untuk informasi lebih lanjut, lihat AWS CLI untuk ElastiCache list-tags-for-resource.

Menambahkan tag menggunakan AWS CLI

Anda dapat menggunakan AWS CLI untuk menambahkan tag ke ElastiCache sumber daya yang ada dengan menggunakan operasi add-tags-to-resource CLI. Jika kunci tag tidak ada di sumber daya, kunci dan nilai akan ditambahkan ke sumber daya. Jika kunci sudah ada di sumber daya, nilai yang terkait dengan kunci tersebut akan diperbarui ke nilai yang baru.

Kode berikut menggunakan AWS CLI untuk menambahkan kunci `Service` dan `Region` dengan nilai-nilai `elasticache` dan `us-west-2` masing-masing ke node di cluster `my-cluster-001` `my-cluster` di wilayah us-west-2.

Memcache

Untuk Linux, macOS, atau Unix:

```
aws elasticache add-tags-to-resource \
 --resource-name arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster \
 --tags Key=Service,Value=elasticache \
        Key=Region,Value=us-west-2
```

Untuk Windows:

```
aws elasticache add-tags-to-resource ^
 --resource-name arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster ^
 --tags Key=Service,Value=elasticache ^
        Key=Region,Value=us-west-2
```

Redis

Untuk Linux, macOS, atau Unix:

```
aws elasticache add-tags-to-resource \
 --resource-name arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster-001 \
 --tags Key=Service,Value=elasticache \
        Key=Region,Value=us-west-2
```

Untuk Windows:

```
aws elasticache add-tags-to-resource ^
```

```
  --resource-name arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster-001 ^
  --tags Key=Service,Value=elasticache ^
         Key=Region,Value=us-west-2
```

Output dari operasi ini akan terlihat seperti berikut ini, daftar semua tag pada sumber daya mengikuti operasi tersebut.

```
{
   "TagList": [
      {
         "Value": "elasticache",
         "Key": "Service"
      },
      {
         "Value": "us-west-2",
         "Key": "Region"
      }
   ]
}
```

Untuk informasi lebih lanjut, lihat AWS CLI untuk ElastiCache add-tags-to-resource.

Anda juga dapat menggunakan AWS CLI untuk menambahkan tag ke cluster saat Anda membuat cluster baru dengan menggunakan operasicreate-cache-cluster. Anda tidak dapat menambahkan tag saat membuat klaster menggunakan konsol ElastiCache manajemen. Setelah klaster dibuat, Anda kemudian dapat menggunakan konsol untuk menambahkan tag pada klaster tersebut.

Memodifikasi tag menggunakan AWS CLI

Anda dapat menggunakan AWS CLI untuk memodifikasi tag pada ElastiCache cluster.

Untuk mengubah tag:

- Gunakan add-tags-to-resource untuk menambahkan tag dan nilai baru atau untuk mengubah nilai yang terkait dengan tag yang ada.
- Gunakan remove-tags-from-resource untuk menghapus tag tertentu dari sumber daya.

Output dari kedua operasi tersebut akan berupa daftar tag dan nilai-nilainya di klaster yang ditentukan.

Menghapus tag menggunakan AWS CLI

Anda dapat menggunakan AWS CLI untuk menghapus tag dari cluster yang ada ElastiCache untuk Memcached dengan menggunakan operasi. remove-tags-from-resource

Untuk Memcached, kode berikut menggunakan AWS CLI untuk menghapus tag dengan kunci `Service` dan `Region` dari node `my-cluster-001` di cluster `my-cluster` di wilayah us-west-2.

Untuk Linux, macOS, atau Unix:

```
aws elasticache remove-tags-from-resource \
 --resource-name arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster \
 --tag-keys PM Service
```

Untuk Windows:

```
aws elasticache remove-tags-from-resource ^
 --resource-name arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster ^
 --tag-keys PM Service
```

Untuk Redis OSS, kode berikut menggunakan AWS CLI untuk menghapus tag dengan kunci `Service` dan `Region` dari node `my-cluster-001` di cluster `my-cluster` di wilayah us-west-2.

Untuk Linux, macOS, atau Unix:

```
aws elasticache remove-tags-from-resource \
 --resource-name arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster-001 \
 --tag-keys PM Service
```

Untuk Windows:

```
aws elasticache remove-tags-from-resource ^
 --resource-name arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster-001 ^
 --tag-keys PM Service
```

Output dari operasi ini akan terlihat seperti berikut ini, daftar semua tag pada sumber daya mengikuti operasi tersebut.

```
{
   "TagList": []
}
```

Untuk informasi lebih lanjut, lihat AWS CLI untuk ElastiCache remove-tags-from-resource.

Mengelola tag alokasi biaya Anda menggunakan API ElastiCache

Anda dapat menggunakan ElastiCache API untuk menambahkan, memodifikasi, atau menghapus tag alokasi biaya.

Tag alokasi biaya diterapkan ElastiCache untuk cluster Memcached. Klaster yang akan diberi tag ditentukan menggunakan ARN (Amazon Resource Name).

arn Sampel: `arn:aws:elasticache:us-west-2:1234567890:cluster:my-cluster`

Topik

- Listing tag menggunakan ElastiCache API
- Menambahkan tag menggunakan ElastiCache API
- Memodifikasi tag menggunakan API ElastiCache
- Menghapus tag menggunakan ElastiCache API

Listing tag menggunakan ElastiCache API

Anda dapat menggunakan ElastiCache API untuk mencantumkan tag pada sumber daya yang ada dengan menggunakan ListTagsForResource operasi.

Untuk Memcached, kode berikut menggunakan ElastiCache API untuk mencantumkan tag pada sumber daya `my-cluster` di wilayah us-west-2.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=ListTagsForResource
   &ResourceName=arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Version=2015-02-02
   &Timestamp=20150202T192317Z
   &X-Amz-Credential=<credential>
```

Untuk Redis OSS, kode berikut menggunakan ElastiCache API untuk mencantumkan tag pada sumber daya `my-cluster-001` di wilayah us-west-2.

```
https://elasticache.us-west-2.amazonaws.com/
```

```
   ?Action=ListTagsForResource
   &ResourceName=arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster-001
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Version=2015-02-02
   &Timestamp=20150202T192317Z
   &X-Amz-Credential=<credential>
```

Menambahkan tag menggunakan ElastiCache API

Anda dapat menggunakan ElastiCache API untuk menambahkan tag ke ElastiCache cluster yang ada dengan menggunakan AddTagsToResource operasi. Jika kunci tag tidak ada di sumber daya, kunci dan nilai akan ditambahkan ke sumber daya. Jika kunci sudah ada di sumber daya, nilai yang terkait dengan kunci tersebut akan diperbarui ke nilai yang baru.

Kode berikut menggunakan ElastiCache API untuk menambahkan kunci `Service` dan `Region` dengan nilai `elasticache` dan `us-west-2` masing-masing. Untuk Memcached, ini diterapkan ke sumber daya. `my-cluster` Untuk Redis OSS, ini diterapkan pada sumber daya `my-cluster-001` di wilayah us-west-2.

Memcache

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=AddTagsToResource
   &ResourceName=arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &Tags.member.1.Key=Service
   &Tags.member.1.Value=elasticache
   &Tags.member.2.Key=Region
   &Tags.member.2.Value=us-west-2
   &Version=2015-02-02
   &Timestamp=20150202T192317Z
   &X-Amz-Credential=<credential>
```

Redis

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=AddTagsToResource
   &ResourceName=arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster-001
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
```

```
   &Tags.member.1.Key=Service
   &Tags.member.1.Value=elasticache
   &Tags.member.2.Key=Region
   &Tags.member.2.Value=us-west-2
   &Version=2015-02-02
   &Timestamp=20150202T192317Z
   &X-Amz-Credential=<credential>
```

Untuk informasi selengkapnya, lihat AddTagsToResource di Referensi Amazon ElastiCache API.

Memodifikasi tag menggunakan API ElastiCache

Anda dapat menggunakan ElastiCache API untuk memodifikasi tag pada sebuah ElastiCache cluster.

Untuk mengubah nilai tag:

- Gunakan operasi AddTagsToResource untuk menambahkan tag dan nilai baru atau untuk mengubah nilai tag yang ada.
- Gunakan RemoveTagsFromResource untuk menghapus tag dari sumber daya.

Hasil dari kedua operasi tersebut akan berupa daftar tag dan nilai-nilainya di sumber daya yang ditentukan.

Gunakan RemoveTagsFromResource untuk menghapus tag dari sumber daya.

Menghapus tag menggunakan ElastiCache API

Anda dapat menggunakan ElastiCache API untuk menghapus tag dari cluster yang sudah ada ElastiCache untuk Memcached dengan menggunakan operasi. RemoveTagsFromResource

Kode berikut menggunakan ElastiCache API untuk menghapus tag dengan kunci `Service` dan `Region` dari node di cluster `my-cluster-001` `my-cluster` di wilayah us-west-2.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=RemoveTagsFromResource
   &ResourceName=arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster-001
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &TagKeys.member.1=Service
   &TagKeys.member.2=Region
   &Version=2015-02-02
   &Timestamp=20150202T192317Z
```

```
    &X-Amz-Credential=<credential>
```

Menggunakan Lensa Amazon ElastiCache Well-Architected

Bagian ini menjelaskan Amazon ElastiCache Well-Architected Lens, kumpulan prinsip desain dan panduan untuk merancang beban kerja yang dirancang dengan baik. ElastiCache

- ElastiCache Lensa bersifat aditif untuk AWS Well-Architected Framework.
- Setiap Pilar memiliki serangkaian pertanyaan untuk membantu memulai diskusi seputar ElastiCache Arsitektur.
  - Setiap pertanyaan memiliki sejumlah praktik terkemuka bersama dengan skornya untuk pelaporan.
    - Wajib - Diperlukan sebelum mulai ke prod (tidak memenuhinya menimbulkan risiko tinggi)
    - Terbaik - Kemungkinan kondisi terbaik yang dapat dimiliki pelanggan
    - Bagus - Apa yang kami rekomendasikan untuk dimiliki pelanggan (tidak memenuhinya menimbulkan risiko sedang)
- Terminology Well-Architected
  - Komponen — Kode, konfigurasi, dan AWS Sumber Daya yang bersama-sama memenuhi persyaratan. Komponen berinteraksi dengan komponen lain, dan sering disamakan dengan layanan dalam arsitektur microservice.
  - Beban Kerja - Satu set komponen yang bersama-sama memberikan nilai bisnis. Contoh beban kerja adalah situs web pemasaran, situs web e-commerce, back-end untuk aplikasi seluler, platform analitik, dll.

> **Note**
>
> Panduan ini belum diperbarui untuk menyertakan informasi tentang caching ElastiCache tanpa server dan mesin Valkey baru.

Topik

- Pilar ElastiCache Efisiensi Kinerja Lensa Well-Architected Amazon
- Pilar ElastiCache Pengoptimalan Biaya Lensa Well-Architected Amazon

Pilar ElastiCache Keunggulan Operasional Lensa Well-Architected Amazon

Pilar keunggulan operasional berfokus untuk menjalankan dan memantau sistem guna memberikan nilai bisnis, dan terus meningkatkan proses dan prosedur. Topik utamanya meliputi mengotomatisasi perubahan, menanggapi peristiwa, dan mendefinisikan standar untuk mengelola operasi harian.

Topik

- OE 1: Bagaimana Anda memahami dan menanggapi peringatan dan peristiwa yang dipicu oleh cluster Anda? ElastiCache
- OE 2: Kapan dan bagaimana Anda menskalakan ElastiCache cluster yang ada?
- OE 3: Bagaimana Anda mengelola sumber daya ElastiCache cluster dan memelihara cluster Anda? up-to-date
- OE 4: Bagaimana Anda mengelola koneksi klien ke cluster Anda? ElastiCache
- OE 5: Bagaimana Anda menerapkan ElastiCache Komponen untuk Beban Kerja?
- OE 6: Bagaimana cara merencanakan dan mengurangi kegagalan?
- OE 7: Bagaimana Anda memecahkan masalah peristiwa mesin Valkey atau Redis OSS?

OE 1: Bagaimana Anda memahami dan menanggapi peringatan dan peristiwa yang dipicu oleh cluster Anda? ElastiCache

Pengenalan tingkat pertanyaan: Saat Anda mengoperasikan ElastiCache kluster, Anda dapat menerima pemberitahuan dan peringatan secara opsional saat peristiwa tertentu terjadi. ElastiCache, secara default, mencatat peristiwa yang berhubungan dengan sumber daya Anda, seperti failover, penggantian node, operasi penskalaan, pemeliharaan terjadwal, dan banyak lagi. Setiap peristiwa mencakup tanggal dan waktu, nama sumber dan jenis sumber, dan deskripsi.

Manfaat tingkat pertanyaan: Mampu memahami dan mengelola alasan mendasar di balik peristiwa yang memicu peringatan yang dihasilkan oleh klaster Anda memungkinkan Anda beroperasi lebih efektif dan merespons peristiwa dengan tepat.

- [Diperlukan] Tinjau peristiwa yang dihasilkan oleh ElastiCache di ElastiCache konsol (setelah memilih wilayah Anda) atau menggunakan perintah mendeskripsikan peristiwa Amazon Command Line Interface(AWS CLI) dan API. ElastiCache Konfigurasikan ElastiCache untuk mengirim

notifikasi untuk peristiwa kluster penting menggunakan Amazon Simple Notification Service (Amazon SNS). Menggunakan Amazon SNS dengan cluster Anda memungkinkan Anda untuk secara terprogram mengambil tindakan atas peristiwa. ElastiCache

- Ada dua kategori besar peristiwa: peristiwa terkini dan terjadwal. Daftar peristiwa terkini meliputi: pembuatan dan penghapusan sumber daya, operasi penskalaan, failover, boot ulang simpul, snapshot dibuat, perubahan parameter klaster, pembaruan sertifikat CA, peristiwa kegagalan (kegagalan penyediaan klaster - VPC atau ENI-, kegagalan penskalaan - ENI-, dan kegagalan snapshot). Daftar peristiwa terjadwal meliputi: simpul yang dijadwalkan akan diganti selama periode pemeliharaan dan penggantian simpul yang dijadwalkan ulang.
- Meskipun Anda mungkin tidak perlu segera bereaksi terhadap beberapa peristiwa ini, penting untuk terlebih dahulu melihat semua peristiwa kegagalan:
  - ElastiCache:AddCacheNodeFailed
  - ElastiCache:CacheClusterProvisioningFailed
  - ElastiCache:CacheClusterScalingFailed
  - ElastiCache:CacheNodesRebooted
  - ElastiCache: SnapshotFailed (Valkey atau Redis OSS saja)
- [Sumber Daya]:
  - Mengelola ElastiCache notifikasi Amazon SNS
  - Notifikasi Peristiwa dan Amazon SNS

- [Terbaik] Untuk mengotomatiskan respons terhadap peristiwa, manfaatkan kemampuan AWS produk dan layanan seperti SNS dan Fungsi Lambda. Ikuti praktik terbaik dengan membuat perubahan yang kecil, sering, dan dapat dikembalikan, sebagai kode untuk mengembangkan operasi Anda dari waktu ke waktu. Anda harus menggunakan CloudWatch metrik Amazon untuk memantau cluster Anda.

  [Sumber Daya]: Monitor ElastiCache (mode cluster dinonaktifkan) baca titik akhir replika menggunakan AWS Lambda, Amazon Route 53, dan Amazon SNS untuk kasus penggunaan yang menggunakan Lambda dan SNS.

OE 2: Kapan dan bagaimana Anda menskalakan ElastiCache cluster yang ada?

Pengenalan tingkat pertanyaan: Ukuran kanan ElastiCache klaster Anda adalah tindakan penyeimbangan yang perlu dievaluasi setiap kali ada perubahan pada jenis beban kerja yang mendasarinya. Tujuan Anda adalah beroperasi dengan lingkungan yang telah di-rightsizing untuk beban kerja Anda.

Manfaat tingkat pertanyaan: Pemanfaatan sumber daya Anda yang berlebihan dapat mengakibatkan peningkatan latensi dan penurunan performa secara keseluruhan. Kurangnya pemanfaatan, di sisi lain, dapat mengakibatkan sumber daya yang disediakan secara berlebihan dengan optimisasi biaya yang tidak optimal. Dengan melakukan rightsizing lingkungan, Anda dapat mencapai keseimbangan antara efisiensi performa dan optimisasi biaya. Untuk memulihkan di atas atau di bawah pemanfaatan sumber daya Anda, ElastiCache dapat skala dalam dua dimensi. Anda dapat menskalakan secara vertikal dengan menambah atau mengurangi kapasitas simpul. Anda juga dapat menskalakan secara horizontal dengan menambahkan dan menghapus simpul.

- [Wajib] Pemanfaatan CPU dan jaringan yang berlebihan pada simpul primer harus diatasi dengan memindahkan dan mengarahkan ulang operasi baca ke simpul replika. Gunakan simpul replika untuk operasi baca guna mengurangi pemanfaatan simpul primer. Ini dapat dikonfigurasi di pustaka klien Valkey atau Redis OSS Anda dengan menghubungkan ke titik akhir ElastiCache pembaca untuk mode cluster dinonaktifkan, atau dengan menggunakan perintah READONLY untuk mode cluster diaktifkan.

  [Sumber Daya]:

  - Menemukan titik akhir koneksi di ElastiCache
  - Rightsizing Klaster
  - Perintah READONLY
- [Wajib] Pantau pemanfaatan sumber daya klaster penting seperti CPU, memori, dan jaringan. Pemanfaatan sumber daya klaster khusus ini perlu dilacak untuk mematangkan keputusan Anda dalam penskalaan, dan jenis operasi penskalaan. Untuk mode ElastiCache cluster dinonaktifkan, node primer dan replika dapat menskalakan secara vertikal. Simpul replika juga dapat diskalakan secara horizontal dari 0 ke 5 simpul. Untuk mode klaster diaktifkan, hal yang sama berlaku dalam setiap serpihan klaster. Selain itu, Anda dapat menambah atau mengurangi jumlah serpihan.

  [Sumber Daya]:

  - Memantau praktik terbaik dengan ElastiCache menggunakan Amazon CloudWatch
  - Penskalaan ElastiCache Cluster untuk Valkey dan Redis OSS
  - Penskalaan ElastiCache Cluster untuk Memcached
- [Terbaik] Memantau tren dari waktu ke waktu dapat membantu Anda mendeteksi perubahan beban kerja yang akan luput dari perhatian jika dipantau pada titik waktu tertentu. Untuk mendeteksi tren jangka panjang, gunakan CloudWatch metrik untuk memindai rentang waktu yang lebih lama. Pembelajaran dari mengamati CloudWatch metrik periode yang diperpanjang harus

menginformasikan perkiraan Anda seputar pemanfaatan sumber daya cluster. CloudWatch Titik data dan metrik tersedia hingga 455 hari.

[Sumber Daya]:

- Pemantauan ElastiCache dengan CloudWatch Metrik
- Memantau Memcached dengan Metrik CloudWatch
- Memantau praktik terbaik dengan ElastiCache menggunakan Amazon CloudWatch

- [Terbaik] Jika ElastiCache sumber daya Anda dibuat dengan CloudFormation itu adalah praktik terbaik untuk melakukan perubahan menggunakan CloudFormation template untuk menjaga konsistensi operasional dan menghindari perubahan konfigurasi yang tidak dikelola dan penyimpangan tumpukan.

  [Sumber Daya]:

  - ElastiCache referensi tipe sumber daya untuk CloudFormation

- [Terbaik] Otomatiskan operasi penskalaan Anda menggunakan data operasional klaster dan tentukan ambang batas untuk menyiapkan alarm. CloudWatch Gunakan CloudWatch Events and Simple Notification Service (SNS) untuk memicu fungsi Lambda dan menjalankan ElastiCache API untuk menskalakan cluster Anda secara otomatis. Contohnya adalah menambahkan serpihan ke klaster Anda ketika metrik `EngineCPUUtilization` mencapai 80% untuk jangka waktu yang lama. Pilihan lain adalah menggunakan `DatabaseMemoryUsedPercentages` untuk ambang batas berbasis memori.

  [Sumber Daya]:

  - Menggunakan CloudWatch Alarm Amazon
  - Apa itu CloudWatch acara Amazon?
  - Menggunakan AWS Lambda dengan Amazon Simple Notification Service
  - Referensi ElastiCache API

OE 3: Bagaimana Anda mengelola sumber daya ElastiCache cluster dan memelihara cluster Anda? up-to-date

Pengenalan tingkat pertanyaan: Saat beroperasi dalam skala besar, penting bagi Anda untuk dapat menentukan dan mengidentifikasi semua sumber daya Anda. ElastiCache Saat meluncurkan fitur aplikasi baru, Anda perlu membuat simetri versi cluster di semua jenis ElastiCache lingkungan Anda: dev, testing, dan production. Atribut sumber daya memungkinkan Anda memisahkan lingkungan

untuk tujuan operasional yang berbeda-beda, seperti saat meluncurkan fitur baru dan mengaktifkan mekanisme keamanan baru.

Manfaat tingkat pertanyaan: Memisahkan lingkungan pengembangan, pengujian, dan produksi Anda adalah praktik operasional terbaik. Praktik terbaik lainnya adalah klaster dan simpul Anda di seluruh lingkungan memiliki patch perangkat lunak terbaru yang diterapkan menggunakan proses yang dipahami dan terdokumentasi dengan baik. Mengambil keuntungan dari ElastiCache fitur asli memungkinkan tim teknik Anda untuk fokus pada memenuhi tujuan bisnis dan bukan pada ElastiCache pemeliharaan.

- [Terbaik] Jalankan pada versi mesin terbaru yang tersedia dan terapkan Pembaruan Layanan Mandiri secepat tersedia. ElastiCache secara otomatis memperbarui infrastruktur dasarnya selama jendela pemeliharaan cluster yang Anda tentukan. Namun, simpul yang berjalan di klaster Anda diperbarui melalui Pembaruan Layanan Mandiri. Pembaruan ini dapat terdiri dari dua jenis: patch keamanan atau pembaruan perangkat lunak minor. Pastikan Anda memahami perbedaan berbagai jenis patch dan waktu penerapannya.

  [Sumber Daya]:

  - Pembaruan Layanan Mandiri di Amazon ElastiCache
  - Halaman Bantuan Pemeliharaan dan Pembaruan Layanan ElastiCache Terkelola Amazon
- [Terbaik] Atur ElastiCache sumber daya Anda menggunakan tag. Gunakan tag pada grup replikasi dan bukan pada simpul individual. Anda dapat mengonfigurasi tag agar ditampilkan saat Anda mengueri sumber daya dan Anda dapat menggunakan tag untuk melakukan pencarian dan menerapkan filter. Anda harus menggunakan Grup Sumber Daya agar dapat dengan mudah membuat dan memelihara kumpulan sumber daya yang memiliki set tag yang sama.

  [Sumber Daya]:

  - Praktik Terbaik Pemberian Tag
  - ElastiCache referensi tipe sumber daya untuk CloudFormation
  - Grup Parameter

OE 4: Bagaimana Anda mengelola koneksi klien ke cluster Anda? ElastiCache

Pengenalan tingkat pertanyaan: Saat beroperasi dalam skala besar, Anda perlu memahami bagaimana klien Anda terhubung dengan ElastiCache klaster untuk mengelola aspek operasional aplikasi Anda (seperti waktu respons).

Manfaat tingkat pertanyaan: Memilih mekanisme koneksi yang paling tepat memastikan bahwa aplikasi Anda tidak terputus karena kesalahan konektivitas, seperti waktu habis.

- [Wajib] Pisahkan operasi baca dari tulis dan hubungkan ke simpul replika untuk menjalankan operasi baca. Namun, ketahuilah ketika Anda memisahkan tulisan dari bacaan, Anda akan kehilangan kemampuan untuk membaca kunci segera setelah menulisnya karena sifat asinkron dari replikasi Valkey dan Redis OSS. Perintah WAIT dapat dimanfaatkan untuk meningkatkan keamanan data dunia nyata dan memaksa replika mengonfirmasi penulisan sebelum merespons klien, dengan mengorbankan performa secara keseluruhan. Menggunakan node replika untuk operasi baca dapat dikonfigurasi di pustaka ElastiCache klien Anda menggunakan endpoint ElastiCache pembaca untuk mode cluster dinonaktifkan. Untuk mode cluster diaktifkan, gunakan perintah READONLY. Untuk banyak pustaka ElastiCache klien, READONLY diimplementasikan secara default atau melalui pengaturan konfigurasi.

  [Sumber Daya]:

  - Menemukan titik akhir koneksi di ElastiCache
  - READONLY
- [Wajib] Gunakan pooling koneksi. Pembuatan koneksi TCP akan menghabiskan waktu CPU di sisi klien dan server dan pooling memungkinkan Anda menggunakan kembali koneksi TCP.

  Untuk mengurangi overhead koneksi, Anda harus menggunakan pooling koneksi. Dengan pool koneksi, aplikasi Anda dapat menggunakan kembali dan melepaskan koneksi 'sesuka hati', tanpa perlu membuat koneksi. Anda dapat menerapkan penyatuan koneksi melalui pustaka ElastiCache klien Anda (jika didukung), dengan Framework yang tersedia untuk lingkungan aplikasi Anda, atau membangunnya dari bawah ke atas.
- [Terbaik] Pastikan waktu habis soket klien diatur ke setidaknya satu detik (vs. default "tidak ada" di beberapa klien).
  - Mengatur nilai waktu habis terlalu rendah dapat menyebabkan kemungkinan waktu habis ketika beban server tinggi. Pengaturan yang terlalu tinggi dapat mengakibatkan aplikasi Anda membutuhkan waktu lama untuk mendeteksi masalah koneksi.
  - Kendalikan volume koneksi baru dengan menerapkan pooling koneksi di aplikasi klien Anda. Ini mengurangi latensi dan pemanfaatan CPU yang diperlukan untuk membuka dan menutup koneksi, dan melakukan handshake TLS jika TLS diaktifkan di klaster.

  [Sumber Daya]: Konfigurasikan ElastiCache untuk ketersediaan yang lebih tinggi

- [Baik] Menggunakan pipelining (jika kasus penggunaan Anda memungkinkannya) dapat meningkatkan performa secara signifikan.
  - Dengan pipelining, Anda akan mengurangi Waktu Pulang Pergi (RTT) antara klien aplikasi dan klaster serta permintaan baru dapat diproses bahkan jika klien belum membaca respons sebelumnya.
  - Dengan pipelining Anda dapat mengirim beberapa perintah ke server tanpa menunggu replies/ack. Kelemahan dari pipelining adalah ketika Anda akhirnya mengambil semua respons secara massal, mungkin ada kesalahan yang tidak akan Anda temukan sampai akhir.
  - Terapkan metode untuk mencoba kembali permintaan ketika ditampilkan kesalahan yang menghilangkan permintaan buruk.

  [Sumber Daya]: Pipelining

OE 5: Bagaimana Anda menerapkan ElastiCache Komponen untuk Beban Kerja?

Pengenalan tingkat pertanyaan: ElastiCache lingkungan dapat digunakan secara manual melalui AWS Konsol, atau secara terprogram melalui, CLI, toolkitAPIs, dll. Praktik terbaik Keunggulan Operasional menyarankan untuk mengotomatiskan deployment melalui kode jika memungkinkan. Selain itu, ElastiCache cluster dapat diisolasi oleh beban kerja atau digabungkan untuk tujuan pengoptimalan biaya.

Manfaat tingkat pertanyaan: Memilih mekanisme penyebaran yang paling tepat untuk ElastiCache lingkungan Anda dapat meningkatkan Keunggulan Operasi dari waktu ke waktu. Sebaiknya lakukan operasi sebagai kode jika memungkinkan untuk meminimalkan kesalahan manusia dan meningkatkan pengulangan, fleksibilitas, dan waktu respons terhadap peristiwa.

Dengan memahami persyaratan isolasi beban kerja, Anda dapat memilih untuk memiliki ElastiCache lingkungan khusus per beban kerja atau menggabungkan beberapa beban kerja menjadi satu cluster, atau kombinasinya. Memahami kompromi dapat membantu mencapai keseimbangan antara Keunggulan Operasional dan Optimalisasi Biaya

- [Wajib] Pahami opsi penerapan yang tersedia ElastiCache, dan otomatiskan prosedur ini bila memungkinkan. Kemungkinan jalan otomatisasi termasuk CloudFormation, AWS CLI/SDK, dan. APIs

  [Sumber Daya]:

  - Referensi jenis ElastiCache sumber daya Amazon

  - elasticache
  - Amazon ElastiCache API Referensi
- [Wajib] Untuk semua beban kerja, tentukan tingkat isolasi klaster yang diperlukan.
  - [Terbaik]: Isolasi Tinggi – pemetaan beban kerja ke klaster 1:1. Memungkinkan kontrol berbutir terbaik atas akses, ukuran, penskalaan, dan pengelolaan ElastiCache sumber daya berdasarkan per beban kerja.
  - [Lebih Baik]: Isolasi Sedang – M:1 diisolasi berdasarkan tujuan tetapi mungkin dibagi di beberapa beban kerja (misalnya klaster yang dikhususkan untuk caching beban kerja, dan yang lain dikhususkan untuk pesan).
  - [Baik]: Isolasi Rendah - M:1 semua tujuan, dibagikan sepenuhnya. Direkomendasikan untuk beban kerja di mana akses bersama dapat diterima.

OE 6: Bagaimana cara merencanakan dan mengurangi kegagalan?

Pengenalan tingkat pertanyaan: Keunggulan Operasional mencakup mengantisipasi kegagalan dengan melakukan latihan “pra-mortem” reguler untuk mengidentifikasi sumber kegagalan potensial sehingga dapat dihilangkan atau dikurangi. ElastiCache menawarkan API Failover yang memungkinkan simulasi kejadian kegagalan node, untuk tujuan pengujian.

Manfaat tingkat pertanyaan: Dengan menguji skenario kegagalan lebih dahulu, Anda dapat mempelajari bagaimana pengaruhnya terhadap beban kerja Anda. Ini memungkinkan pengujian prosedur respons yang aman dan efektivitasnya, serta membuat tim Anda terbiasa dengan eksekusinya.

[Diperlukan] Secara teratur melakukan pengujian failover di akun dev/test. TestFailover

OE 7: Bagaimana Anda memecahkan masalah peristiwa mesin Valkey atau Redis OSS?

Pengenalan tingkat pertanyaan: Keunggulan Operasional membutuhkan kemampuan untuk menyelidiki informasi tingkat layanan dan tingkat mesin untuk menganalisis kesehatan dan status cluster Anda. ElastiCache dapat memancarkan log mesin Valkey atau Redis OSS ke Amazon CloudWatch dan Amazon Kinesis Data Firehose.

Manfaat tingkat pertanyaan: Mengaktifkan log mesin Valkey atau Redis OSS pada ElastiCache cluster memberikan wawasan tentang peristiwa yang berdampak pada kesehatan dan kinerja cluster. Log mesin Valkey atau Redis OSS menyediakan data langsung dari mesin yang tidak

tersedia melalui mekanisme kejadian. ElastiCache Melalui pengamatan yang cermat terhadap kedua ElastiCache peristiwa (lihat OE-1 sebelumnya) dan log mesin, dimungkinkan untuk menentukan urutan peristiwa saat pemecahan masalah dari perspektif layanan dan perspektif mesin. ElastiCache

- [Wajib] Pastikan fungsionalitas pencatatan mesin Redis OSS diaktifkan, yang tersedia pada ElastiCache versi 6.2 untuk Redis OSS dan yang lebih baru. Ini dapat dilakukan selama pembuatan klaster atau dengan mengubah klaster setelah pembuatan.
  - Tentukan apakah Amazon CloudWatch Log atau Amazon Kinesis Data Firehose adalah target yang tepat untuk log mesin Redis OSS.
  - Pilih log target yang sesuai dalam salah satu CloudWatch atau Kinesis Data Firehose untuk mempertahankan log. Jika Anda memiliki beberapa klaster, pertimbangkan log target yang berbeda untuk setiap klaster karena ini akan membantu mengisolasi data saat pemecahan masalah.

  [Sumber Daya]:

  - Pengiriman log: Pengiriman log
  - Tujuan pencatatan: Amazon CloudWatch Logs
  - Pengenalan Amazon CloudWatch Logs: Apa itu Amazon CloudWatch Logs?
  - Pengenalan Amazon Kinesis Data Firehose: Apa Itu Amazon Kinesis Data Firehose?
- [Terbaik] Jika menggunakan CloudWatch Log Amazon, pertimbangkan untuk memanfaatkan Amazon CloudWatch Logs Insights untuk menanyakan log mesin Valkey atau Redis OSS untuk informasi penting.

  Sebagai contoh, buat kueri terhadap grup CloudWatch Log yang berisi log mesin Valkey atau Redis OSS yang akan mengembalikan peristiwa dengan 'PERINGATAN', seperti: LogLevel

  ```
  fields @timestamp, LogLevel, Message
  | sort @timestamp desc
  | filter LogLevel = "WARNING"
  ```

  [Sumber Daya]: Menganalisis data log dengan Wawasan CloudWatch Log

Pilar Keamanan Lensa ElastiCache Well-Architected Amazon

Pilar keamanan berfokus pada perlindungan informasi dan sistem. Topik utama yang dibahas meliputi kerahasiaan dan integritas data, mengidentifikasi dan mengelola "siapa yang dapat

melakukan apa" dengan manajemen berbasis hak akses, melindungi sistem, dan menetapkan kontrol untuk mendeteksi peristiwa keamanan.

Topik

SEC 1: Langkah apa yang Anda ambil dalam mengendalikan akses resmi ke ElastiCache data?

Pengenalan tingkat pertanyaan: Semua ElastiCache cluster dirancang untuk diakses dari instans Amazon Elastic Compute Cloud dalam VPC, fungsi tanpa server (), atau wadah (Amazon Elastic Container Service AWS Lambda). Skenario yang paling sering ditemui adalah mengakses ElastiCache cluster dari instans Amazon Elastic Compute Cloud dalam Amazon Virtual Private Cloud yang sama (Amazon Virtual Private Cloud). Sebelum dapat terhubung ke cluster dari EC2 instans Amazon, Anda harus mengotorisasi EC2 instans Amazon untuk mengakses klaster. Untuk mengakses ElastiCache cluster yang berjalan di VPC, perlu untuk memberikan masuknya jaringan ke cluster.

Manfaat tingkat pertanyaan: Lalu lintas masuk jaringan ke dalam klaster dikendalikan melalui grup keamanan VPC. Grup keamanan bertindak sebagai firewall virtual untuk EC2 instans Amazon Anda untuk mengontrol lalu lintas masuk dan keluar. Aturan masuk mengontrol lalu lintas yang masuk ke instans Anda, dan aturan keluar mengontrol lalu lintas yang keluar dari instans Anda. Dalam kasus ElastiCache, ketika meluncurkan cluster, itu membutuhkan asosiasi grup keamanan. Hal ini memastikan bahwa aturan lalu lintas masuk dan keluar berlaku untuk semua simpul yang membentuk klaster. Selain itu, ElastiCache dikonfigurasi untuk menyebarkan pada subnet pribadi secara eksklusif sehingga mereka hanya dapat diakses dari melalui jaringan pribadi VPC.

- [Wajib] Grup keamanan yang dikaitkan dengan klaster Anda mengontrol lalu lintas masuk dan akses jaringan ke klaster. Secara default, grup keamanan tidak akan memiliki aturan masuk yang ditentukan dan, oleh karena itu, tidak ada jalur masuk ke. ElastiCache Untuk mengaktifkan ini, konfigurasikan aturan masuk pada grup keamanan yang menentukan alamat/rentang IP sumber, lalu lintas jenis TCP dan port untuk ElastiCache cluster Anda (port default 6379 untuk Valkey dan Redis OSS misalnya ElastiCache). Meskipun dimungkinkan untuk mengizinkan serangkaian sumber masuk yang sangat luas, seperti semua sumber daya dalam VPC (0.0.0.0/0), disarankan untuk sedetail mungkin dalam mendefinisikan aturan masuk seperti hanya mengotorisasi akses masuk ke klien Valkey atau Redis OSS yang berjalan di Amazon Amazon Amazon yang terkait dengan grup keamanan tertentu. EC2

  [Sumber Daya]:

  - Subnet dan grup subnet
  - Mengakses klaster atau grup replikasi Anda
  - Mengontrol lalu lintas ke sumber daya menggunakan grup keamanan
  - Grup keamanan Amazon Elastic Compute Cloud untuk instans Linux
- AWS Identity and Access Management Kebijakan [Wajib] dapat ditetapkan ke AWS Lambda fungsi yang memungkinkan mereka mengakses ElastiCache data. Untuk mengaktifkan fitur ini, buat peran eksekusi IAM dengan `AWSLambdaVPCAccessExecutionRole` izin, lalu tetapkan peran ke fungsi tersebut AWS Lambda .

  [Sumber Daya]: Mengonfigurasi fungsi Lambda untuk mengakses Amazon di VPC ElastiCache Amazon: Tutorial: Mengonfigurasi fungsi Lambda untuk mengakses Amazon di VPC Amazon ElastiCache

SEC 2: Apakah aplikasi Anda memerlukan otorisasi tambahan untuk kontrol berbasis jaringan di ElastiCache atas dan di atas?

Pengenalan tingkat pertanyaan: Dalam skenario di mana perlu untuk membatasi atau mengontrol akses ke cluster pada tingkat klien individu, disarankan untuk mengautentikasi melalui perintah AUTH. ElastiCache token otentikasi, dengan manajemen pengguna dan grup pengguna opsional, memungkinkan ElastiCache untuk memerlukan kata sandi sebelum mengizinkan klien menjalankan perintah dan kunci akses, sehingga meningkatkan keamanan bidang data.

Manfaat tingkat pertanyaan: Untuk membantu menjaga keamanan data Anda, ElastiCache sediakan mekanisme untuk melindungi terhadap akses data Anda yang tidak sah. Ini termasuk menegakkan

ROLE-Based Access Control (RBAC) AUTH, atau token AUTH (kata sandi) yang digunakan oleh klien untuk terhubung sebelum melakukan perintah resmi. ElastiCache

- [Terbaik] Untuk ElastiCache versi 6.x dan lebih tinggi untuk Redis OSS, dan ElastiCache versi 7.2 dan lebih tinggi untuk Valkey, tentukan kontrol otentikasi dan otorisasi dengan mendefinisikan grup pengguna, pengguna, dan string akses. Tetapkan pengguna ke grup pengguna, lalu tetapkan grup pengguna ke klaster. Untuk memanfaatkan RBAC, pemilihan harus dilakukan pada pembuatan klaster, dan enkripsi bergerak harus diaktifkan. Pastikan Anda menggunakan klien Valkey atau Redis OSS yang mendukung TLS untuk dapat memanfaatkan RBAC.

  [Sumber Daya]:

  - Menerapkan RBAC ke Grup Replikasi untuk ElastiCache
  - Menentukan Izin Menggunakan String Akses
  - ACL
  - ElastiCache Versi yang didukung
- [Terbaik] Untuk ElastiCache versi sebelum 6.x untuk Redis OSS, selain menetapkan kuat token/password dan mempertahankan kebijakan kata sandi yang ketat untuk AUTH, praktik terbaik adalah memutar kata sandi/token. ElastiCache dapat mengelola hingga dua (2) token otentikasi pada waktu tertentu. Anda juga dapat mengubah klaster untuk secara eksplisit mewajibkan penggunaan token autentikasi.

  [Sumber Daya]: Memodifikasi token AUTH pada cluster yang ada ElastiCache

SEC 3: Apakah ada risiko bahwa perintah dapat dijalankan secara tidak sengaja yang menyebabkan kehilangan atau kegagalan data?

Pengenalan tingkat pertanyaan: Ada sejumlah perintah Valkey atau Redis OSS yang dapat berdampak buruk pada operasi jika dijalankan secara tidak sengaja atau oleh aktor jahat. Perintah ini dapat memiliki konsekuensi yang tidak diinginkan dari perspektif performa dan keamanan data. Misalnya developer dapat secara rutin memanggil perintah FLUSHALL di lingkungan pengembangan, dan karena kesalahan mungkin secara tidak sengaja mencoba memanggil perintah ini pada sistem produksi, yang mengakibatkan kehilangan data yang tidak disengaja.

Manfaat tingkat pertanyaan: Dimulai dengan ElastiCache versi 5.0.3 untuk Redis OSS, Anda memiliki kemampuan untuk mengganti nama perintah tertentu yang mungkin mengganggu beban kerja Anda. Mengganti nama perintah dapat membantu mencegahnya dieksekusi secara tidak sengaja di klaster.

- [Wajib]

  [Sumber Daya]:

  - ElastiCache versi 5.0.3 untuk Redis OSS (usang, gunakan versi 5.0.6)
  - ElastiCache versi 5.0.3 untuk perubahan parameter Redis OSS
  - Keamanan Redis OSS

SEC 4: Bagaimana Anda memastikan enkripsi data saat istirahat ElastiCache

Pengenalan tingkat pertanyaan: Meskipun ElastiCache merupakan penyimpanan data dalam memori, dimungkinkan untuk mengenkripsi data apa pun yang mungkin disimpan (pada penyimpanan) sebagai bagian dari operasi standar cluster. Hal ini termasuk pencadangan terjadwal dan manual yang ditulis ke Amazon S3, serta data yang disimpan ke penyimpanan disk sebagai hasil dari operasi sinkronisasi dan swap. Jenis instans dalam keluarga M6g dan R6g juga memiliki fitur enkripsi dalam memori yang selalu aktif.

Manfaat tingkat pertanyaan: ElastiCache menyediakan enkripsi opsional saat istirahat untuk meningkatkan keamanan data.

- [Diperlukan] Enkripsi AT-rest dapat diaktifkan pada ElastiCache cluster (grup replikasi) hanya ketika dibuat. Klaster yang ada tidak dapat diubah untuk mulai mengenkripsi data diam. Secara default, ElastiCache akan menyediakan dan mengelola kunci yang digunakan dalam enkripsi saat istirahat.

  [Sumber Daya]:

  - Kendala Enkripsi At-Rest
  - Mengaktifkan Enkripsi Diam
- [Terbaik] Manfaatkan jenis EC2 instans Amazon yang mengenkripsi data saat berada di memori (seperti M6g atau R6g). Jika memungkinkan, pertimbangkan untuk mengelola kunci Anda sendiri untuk enkripsi diam. Untuk lingkungan keamanan data yang lebih ketat, AWS Key Management Service (KMS) dapat digunakan untuk mengelola sendiri Customer Master Keys (CMK). Melalui ElastiCache integrasi dengan AWS Key Management Service, Anda dapat membuat, memiliki, dan mengelola kunci yang digunakan untuk enkripsi data saat istirahat untuk ElastiCache cluster Anda.

  [Sumber Daya]:

  - Menggunakan kunci yang dikelola pelanggan dari AWS Key Management Service
  - AWS Layanan Manajemen Kunci

- Konsep AWS KMS

SEC 5: Bagaimana Anda mengenkripsi data dalam transit? ElastiCache

Pengantar tingkat pertanyaan: Adalah persyaratan umum untuk memitigasi risiko data disusupi saat bergerak. Ini mewakili data dalam komponen sistem terdistribusi, serta antara klien aplikasi dan node cluster. ElastiCache mendukung persyaratan ini dengan memungkinkan untuk mengenkripsi data dalam transit antara klien dan cluster, dan antara node cluster itu sendiri. Jenis instans dalam keluarga M6g dan R6g juga memiliki fitur enkripsi dalam memori yang selalu aktif.

Manfaat tingkat pertanyaan: Enkripsi ElastiCache in-transit Amazon adalah fitur opsional yang memungkinkan Anda meningkatkan keamanan data Anda di titik-titik yang paling rentan, ketika sedang dalam perjalanan dari satu lokasi ke lokasi lain.

- [Wajib] Enkripsi dalam transit hanya dapat diaktifkan pada klaster (grup replikasi) setelah pembuatan. Perlu diketahui bahwa karena pemrosesan tambahan yang diperlukan untuk mengenkripsi/mendekripsi data, penerapan enkripsi bergerak akan memiliki beberapa dampak terhadap performa. Untuk memahami dampaknya, disarankan untuk membandingkan beban kerja Anda sebelum dan sesudah mengaktifkan. encryption-in-transit

  [Sumber Daya]:

  - Gambaran umum enkripsi bergerak

SEC 6: Bagaimana Anda membatasi akses untuk bidang kontrol sumber daya?

Pengenalan tingkat pertanyaan: Kebijakan IAM dan ARN memungkinkan kontrol akses berbutir halus ElastiCache untuk Valkey dan Redis OSS, memungkinkan kontrol yang lebih ketat untuk mengelola pembuatan, modifikasi, dan penghapusan cluster.

Manfaat tingkat pertanyaan: Pengelolaan ElastiCache sumber daya Amazon, seperti grup replikasi, node, dll. Dapat dibatasi ke AWS akun yang memiliki izin khusus berdasarkan kebijakan IAM, meningkatkan keamanan dan keandalan sumber daya.

- [Diperlukan] Kelola akses ke ElastiCache sumber daya Amazon dengan menetapkan AWS Identity and Access Management kebijakan khusus kepada AWS pengguna, memungkinkan kontrol yang lebih baik atas akun mana yang dapat melakukan tindakan apa pada klaster.

  [Sumber Daya]:

- Ikhtisar mengelola izin akses ke sumber daya Anda ElastiCache
- Menggunakan kebijakan berbasis identitas (kebijakan IAM) untuk Amazon ElastiCache

SEC 7: Bagaimana Anda mendeteksi dan menanggapi peristiwa keamanan?

Pengenalan tingkat pertanyaan:ElastiCache, saat diterapkan dengan RBAC diaktifkan, mengekspor CloudWatch metrik untuk memberi tahu pengguna tentang peristiwa keamanan. Metrik ini membantu mengidentifikasi percobaan gagal untuk melakukan autentikasi, mengakses kunci, atau menjalankan perintah yang tidak diizinkan bagi pengguna RBAC yang melakukan koneksi.

Selain itu, sumber daya AWS produk dan layanan membantu mengamankan beban kerja Anda secara keseluruhan dengan mengotomatiskan penerapan dan mencatat semua tindakan dan modifikasi untuk peninjauan/audit nanti.

Manfaat tingkat pertanyaan: Dengan memantau peristiwa, Anda memungkinkan organisasi Anda untuk merespons sesuai dengan persyaratan, kebijakan, dan prosedur Anda. Mengotomatiskan pemantauan dan respons terhadap peristiwa keamanan ini memperkuat postur keamanan Anda secara keseluruhan.

- [Wajib] Biasakan diri Anda dengan CloudWatch Metrik yang diterbitkan yang berkaitan dengan otentikasi RBAC dan kegagalan otorisasi.
  - AuthenticationFailures = Upaya gagal untuk mengautentikasi ke Valkey atau Redis OSS
  - KeyAuthorizationFailures = Upaya gagal oleh pengguna untuk mengakses kunci tanpa izin
  - CommandAuthorizationFailures = Upaya gagal oleh pengguna untuk menjalankan perintah tanpa izin

  [Sumber Daya]:
  - Metrik untuk Valkey atau Redis OSS
- [Terbaik] Sebaiknya siapkan peringatan dan notifikasi pada metrik ini dan respons sesuai kebutuhan.

  [Sumber Daya]:
  - Menggunakan CloudWatch alarm Amazon
- [Terbaik] Gunakan perintah Valkey atau Redis OSS ACL LOG untuk mengumpulkan rincian lebih lanjut

  [Sumber Daya]:

- ACL LOG

- [Terbaik] Biasakan diri Anda dengan kemampuan AWS produk dan layanan yang berkaitan dengan pemantauan, pencatatan, dan analisis ElastiCache penyebaran dan peristiwa

  [Sumber Daya]:

  - Mencatat panggilan ElastiCache API Amazon dengan AWS CloudTrail
  - elasticache-redis-cluster-automatic-backup-periksa
  - Pemantauan penggunaan dengan CloudWatch Metrik

Pilar Keandalan Lensa ElastiCache Well-Architected Amazon

Pilar keandalan berfokus pada beban kerja yang menjalankan fungsi yang dimaksudkan dan bagaimana memulihkan dengan cepat dari kegagalan untuk memenuhi tuntutan. Topik utama meliputi desain sistem terdistribusi, perencanaan pemulihan, dan beradaptasi dengan perubahan persyaratan.

Topik

- REL 1: Bagaimana Anda mendukung deployment arsitektur ketersediaan tinggi (HA)?
- REL 2: Bagaimana Anda memenuhi Tujuan Titik Pemulihan (RPOs) AndaElastiCache?
- REL 3: Bagaimana Anda mendukung persyaratan pemulihan bencana (DR)?
- REL 4: Bagaimana Anda merencanakan failover secara efektif?
- REL 5: Apakah ElastiCache komponen Anda dirancang untuk skala?

REL 1: Bagaimana Anda mendukung deployment arsitektur ketersediaan tinggi (HA)?

Pengenalan tingkat pertanyaan: Memahami arsitektur ketersediaan tinggi Amazon ElastiCache akan memungkinkan Anda beroperasi dalam keadaan tangguh selama acara ketersediaan.

Manfaat tingkat pertanyaan: Merancang ElastiCache cluster Anda agar tahan terhadap kegagalan memastikan ketersediaan yang lebih tinggi untuk penerapan Anda. ElastiCache

- [Wajib] Tentukan tingkat keandalan yang Anda butuhkan untuk ElastiCache klaster Anda. Beban kerja yang berbeda memiliki standar ketahanan yang berbeda, dari beban kerja yang sepenuhnya sementara hingga beban kerja krusial. Tentukan kebutuhan untuk setiap jenis lingkungan yang Anda operasikan seperti dev, test, dan production.

Mesin caching: ElastiCache untuk Memcached vs ElastiCache untuk Valkey dan Redis OSS

1. ElastiCache untuk Memcached tidak menyediakan mekanisme replikasi apa pun dan digunakan terutama untuk beban kerja sementara.
2. ElastiCache untuk Valkey dan Redis OSS menawarkan fitur HA dibahas di bawah ini

- [Terbaik] Untuk beban kerja yang membutuhkan HA, gunakan ElastiCache dalam mode cluster dengan minimal dua replika per pecahan, bahkan untuk beban kerja persyaratan throughput kecil yang hanya membutuhkan satu pecahan.
  1. Untuk mode klaster diaktifkan, Multi-AZ diaktifkan secara otomatis.

     Multi-AZ meminimalkan waktu henti dengan melakukan failover otomatis dari simpul primer ke replika, jika terjadi pemeliharaan yang direncanakan atau tidak direncanakan serta mengurangi kegagalan AZ.
  2. Untuk beban kerja sharded, minimal tiga pecahan memberikan pemulihan yang lebih cepat selama peristiwa failover karena Protokol Cluster Valkey atau Redis OSS memerlukan mayoritas node primer tersedia untuk mencapai kuorum.
  3. Siapkan dua replika atau lebih di seluruh Zona Ketersediaan.

     Memiliki dua replika memberikan peningkatan skalabilitas baca dan juga ketersediaan baca dalam skenario di mana satu replika sedang menjalani pemeliharaan.
  4. Gunakan jenis simpul berbasis Graviton2 (simpul default di sebagian besar wilayah).

     ElastiCache telah menambahkan kinerja yang dioptimalkan pada node ini. Hasilnya, Anda mendapatkan performa replikasi dan sinkronisasi yang lebih baik, sehingga meningkatkan ketersediaan secara keseluruhan.
  5. Monitor dan ukuran yang tepat untuk menangani puncak lalu lintas yang diantisipasi: di bawah beban berat, mesin mungkin menjadi tidak responsif, yang memengaruhi ketersediaan. `BytesUsedForCache`dan `DatabaseMemoryUsagePercentage` merupakan indikator yang baik dari penggunaan memori Anda, sedangkan `ReplicationLag` merupakan indikator kesehatan replikasi Anda berdasarkan tingkat tulis Anda. Anda dapat menggunakan metrik ini untuk memicu penskalaan klaster.
  6. Pastikan ketahanan sisi klien dengan melakukan pengujian dengan API Failover sebelum peristiwa failover di lingkungan produksi.

  [Sumber Daya]:

  - Konfigurasikan ElastiCache untuk Redis OSS untuk ketersediaan yang lebih tinggi

- Ketersediaan tinggi menggunakan grup replikasi

REL 2: Bagaimana Anda memenuhi Tujuan Titik Pemulihan (RPOs) AndaElastiCache?

Pengenalan tingkat pertanyaan: Memahami RPO beban kerja untuk menginformasikan keputusan tentang ElastiCache strategi pencadangan dan pemulihan.

Manfaat tingkat pertanyaan: Dengan menerapkan strategi RPO, Anda dapat meningkatkan kelangsungan bisnis jika terjadi skenario pemulihan bencana. Merancang kebijakan pencadangan dan pemulihan dapat membantu Anda memenuhi Tujuan Titik Pemulihan (RPO) untuk ElastiCache data Anda. ElastiCache menawarkan kemampuan snapshot yang disimpan di Amazon S3, bersama dengan kebijakan retensi yang dapat dikonfigurasi. Snapshot ini diambil selama periode pencadangan yang ditentukan dan ditangani oleh layanan secara otomatis. Jika beban kerja Anda memerlukan perincian pencadangan tambahan, Anda memiliki opsi untuk membuat hingga 20 pencadangan manual per hari. Pencadangan yang dibuat secara manual tidak memiliki kebijakan retensi layanan dan dapat disimpan tanpa batas waktu.

- [Wajib] Memahami dan mendokumentasikan RPO ElastiCache penerapan Anda.
  - Perlu diketahui bahwa Memcached tidak menawarkan proses pencadangan apa pun.
  - Tinjau kemampuan fitur ElastiCache Backup dan Restore.
- [Terbaik] Terapkan proses yang dikomunikasikan dengan baik untuk mencadangkan klaster Anda.
  - Mulai pencadangan manual sesuai kebutuhan.
  - Tinjau kebijakan retensi untuk pencadangan otomatis.
  - Perhatikan bahwa cadangan manual akan dipertahankan tanpa batas waktu.
  - Jadwalkan pencadangan otomatis Anda selama periode penggunaan rendah.
  - Lakukan operasi pencadangan terhadap replika baca untuk memastikan Anda meminimalkan dampak pada performa klaster.
- [Bagus] Manfaatkan fitur pencadangan terjadwal ElastiCache untuk mencadangkan data Anda secara teratur selama jendela yang ditentukan.
  - Tes secara berkala pemulihan dari cadangan Anda.
- [Sumber Daya]:
  - Redis OSS
  - Backup dan restore untuk ElastiCache
  - Membuat pencadangan manual

- Menjadwalkan pencadangan otomatis
- Cadangkan dan Kembalikan ElastiCache Cluster

REL 3: Bagaimana Anda mendukung persyaratan pemulihan bencana (DR)?

Pengenalan tingkat pertanyaan: Pemulihan bencana adalah aspek penting dari setiap perencanaan beban kerja. ElastiCache menawarkan beberapa opsi untuk menerapkan pemulihan bencana berdasarkan persyaratan ketahanan beban kerja. Dengan Amazon ElastiCache Global Datastore, Anda dapat menulis ke klaster Anda di satu Wilayah dan memiliki data yang tersedia untuk dibaca dari dua klaster replika Lintas wilayah lainnya, sehingga memungkinkan pembacaan latensi rendah dan pemulihan bencana di seluruh wilayah.

Manfaat tingkat pertanyaan: Memahami dan merencanakan berbagai skenario bencana dapat memastikan kelangsungan bisnis. Strategi DR harus seimbang dengan biaya, dampak performa, dan potensi kehilangan data.

- [Wajib] Kembangkan dan dokumentasikan strategi DR untuk semua ElastiCache komponen Anda berdasarkan persyaratan beban kerja. ElastiCache unik karena beberapa kasus penggunaan sepenuhnya fana dan tidak memerlukan strategi DR apa pun, sedangkan yang lain berada di ujung spektrum yang berlawanan dan memerlukan strategi DR yang sangat kuat. Semua opsi harus ditimbang dalam kaitannya dengan Optimalisasi Biaya - ketahanan yang lebih kuat membutuhkan jumlah infrastruktur yang lebih besar.

  Memahami opsi DR yang tersedia di tingkat regional dan multi-wilayah.

  - Deployment Multi-AZ direkomendasikan untuk mencegah kegagalan AZ. Pastikan untuk menerapkan dengan Cluster-Mode diaktifkan dalam arsitektur Multi-AZ, dengan minimal 3 tersedia. AZs
  - Penyimpanan Data Global direkomendasikan untuk memberikan perlindungan terhadap kegagalan regional.
- [Terbaik] Aktifkan Penyimpanan Data Global untuk beban kerja yang membutuhkan ketahanan tingkat wilayah.
  - Siapkan rencana untuk melakukan failover ke wilayah sekunder jika terjadi degradasi primer.
  - Uji proses failover multi-wilayah sebelum melakukan failover dalam lingkungan produksi.
  - Pantau metrik `ReplicationLag` untuk memahami potensi dampak kehilangan data selama peristiwa failover.
- [Sumber Daya]:

- Memitigasi Kegagalan
- Replikasi lintas AWS Wilayah menggunakan datastores global
- Memulihkan dari cadangan dengan opsi perubahan ukuran klaster
- Meminimalkan waktu henti ElastiCache untuk Valkey dan Redis OSS dengan Multi-AZ

REL 4: Bagaimana Anda merencanakan failover secara efektif?

Pengenalan tingkat pertanyaan: Mengaktifkan Multi-AZ dengan failover otomatis adalah praktik terbaik. ElastiCache Dalam kasus tertentu, ElastiCache untuk Valkey dan Redis OSS menggantikan node primer sebagai bagian dari operasi layanan. Contohnya termasuk peristiwa pemeliharaan yang direncanakan dan kasus yang tidak diharapkan seperti kegagalan simpul atau masalah zona ketersediaan. Failover yang berhasil bergantung pada keduanya ElastiCache dan konfigurasi pustaka klien Anda.

Manfaat tingkat pertanyaan: Mengikuti praktik terbaik untuk ElastiCache kegagalan bersama dengan pustaka ElastiCache klien spesifik Anda membantu Anda meminimalkan potensi waktu henti selama peristiwa failover.

- [Wajib] Untuk mode klaster dinonaktifkan, gunakan waktu tunggu sehingga klien dapat mendeteksi jika perlu memutuskan koneksi dari simpul primer lama dan terhubung kembali ke simpul primer baru, menggunakan alamat IP titik akhir primer yang diperbarui. Untuk mode klaster diaktifkan, pustaka klien bertanggung jawab untuk mendeteksi perubahan dalam topologi klaster yang mendasarinya. Ini paling sering dilakukan dengan pengaturan konfigurasi di pustaka ElastiCache klien, yang juga memungkinkan Anda untuk mengkonfigurasi frekuensi dan metode penyegaran. Setiap pustaka klien menawarkan pengaturannya sendiri dan detail lebih lanjut tersedia dalam dokumentasi yang sesuai.

  [Sumber Daya]:

  - Meminimalkan waktu henti ElastiCache untuk Valkey dan Redis OSS dengan Multi-AZ
  - Tinjau praktik terbaik perpustakaan ElastiCache klien Anda.
- [Wajib] Failover yang berhasil bergantung pada lingkungan replikasi yang berkondisi baik antara simpul primer dan replika. Tinjau dan pahami sifat asinkron replikasi Valkey dan Redis OSS, serta CloudWatch metrik yang tersedia untuk melaporkan jeda replikasi antara node primer dan replika. Untuk kasus penggunaan yang membutuhkan keamanan data yang lebih besar, manfaatkan perintah WAIT untuk memaksa replika mengakui penulisan sebelum menanggapi klien yang terhubung.

[Sumber Daya]:

- Metrik untuk Valkey atau Redis OSS
- Memantau praktik terbaik dengan ElastiCache menggunakan Amazon CloudWatch

- [Terbaik] Secara teratur memvalidasi respons aplikasi Anda selama failover menggunakan ElastiCache Test Failover API.

  [Sumber Daya]:

  - Menguji Failover Otomatis ke Read Replica ElastiCache
  - Menguji failover otomatis

REL 5: Apakah ElastiCache komponen Anda dirancang untuk skala?

Pengenalan tingkat pertanyaan: Dengan memahami kemampuan penskalaan dan topologi penerapan yang tersedia, ElastiCache komponen Anda dapat menyesuaikan dari waktu ke waktu untuk memenuhi persyaratan beban kerja yang berubah. ElastiCachemenawarkan penskalaan 4 arah: in/out (horizontal) serta up/down (vertikal).

Manfaat tingkat pertanyaan: Mengikuti praktik terbaik untuk ElastiCache penerapan memberikan fleksibilitas penskalaan terbesar, serta memenuhi prinsip penskalaan yang dirancang dengan baik secara horizontal untuk meminimalkan dampak kegagalan.

- [Wajib] Memahami perbedaan antara topologi Mode Klaster Diaktifkan dan Mode Klaster Dinonaktifkan. Dalam hampir semua kasus, sebaiknya lakukan deployment dengan mode Klaster diaktifkan karena memungkinkan skalabilitas yang lebih besar dari waktu ke waktu. Komponen mode klaster dinonaktifkan memiliki kemampuan terbatas untuk menskalakan secara horizontal dengan menambahkan replika baca.
- [Wajib] Memahami kapan dan bagaimana melakukan penskalaan.
  - Untuk meningkatkan READIOPS: tambahkan replika
  - Untuk meningkatkan WRITEOPS: tambahkan serpihan (menskalakan ke luar)
  - Untuk meningkatkan IO jaringan - gunakan instans yang dioptimalkan jaringan (menaikkan skala)
- [Terbaik] Terapkan ElastiCache komponen Anda dengan mode Cluster diaktifkan, dengan bias terhadap lebih banyak node yang lebih kecil daripada lebih sedikit, node yang lebih besar. Opsi ini secara efektif membatasi radius ledakan kegagalan simpul.

- [Terbaik] Sertakan replika di klaster Anda untuk meningkatkan respons selama peristiwa penskalaan
- [Bagus] Untuk mode cluster dinonaktifkan, manfaatkan replika baca untuk meningkatkan kapasitas baca secara keseluruhan. ElastiCache memiliki dukungan hingga 5 replika baca dalam mode cluster dinonaktifkan, serta penskalaan vertikal.
- [Sumber Daya]:
  - Cluster penskalaan ElastiCache
  - Menaikkan skala secara online

Pilar ElastiCache Efisiensi Kinerja Lensa Well-Architected Amazon

Pilar efisiensi performa berfokus pada penggunaan sumber daya TI dan komputasi secara efisien. Topik utamanya meliputi pemilihan jenis dan ukuran sumber daya yang tepat berdasarkan persyaratan beban kerja, pemantauan performa, dan pembuatan keputusan berdasarkan informasi untuk mempertahankan efisiensi seiring dengan berkembangnya kebutuhan bisnis.

Topik

- PE 1: Bagaimana Anda memantau kinerja ElastiCache cluster Amazon Anda?
- PE 2: Bagaimana Anda mendistribusikan pekerjaan di seluruh node ElastiCache Cluster Anda?
- PE 3: Untuk beban kerja caching, bagaimana cara melacak dan melaporkan efektivitas dan performa cache Anda?
- PE 4: Bagaimana beban kerja Anda mengoptimalkan penggunaan sumber daya dan koneksi jaringan?
- PE 5: Bagaimana Anda mengelola penggusuran penghapusan and/or kunci?
- PE 6: Bagaimana Anda memodelkan dan berinteraksi dengan data ElastiCache?
- PE 7: Bagaimana Anda mencatat perintah yang berjalan lambat di ElastiCache cluster Amazon Anda?
- PE8: Bagaimana Auto Scaling membantu dalam meningkatkan kinerja cluster? ElastiCache

PE 1: Bagaimana Anda memantau kinerja ElastiCache cluster Amazon Anda?

Pengantar tingkat pertanyaan: Dengan memahami metrik pemantauan yang ada, Anda dapat mengidentifikasi pemanfaatan saat ini. Pemantauan yang tepat dapat membantu mengidentifikasi potensi hambatan yang memengaruhi performa klaster Anda.

Manfaat tingkat pertanyaan: Memahami metrik yang terkait dengan klaster Anda dapat membantu memandu teknik pengoptimalan yang dapat mengurangi latensi dan meningkatkan throughput.

- [Wajib] Pengujian performa dasar menggunakan subset beban kerja Anda.
  - Anda harus memantau performa beban kerja aktual menggunakan mekanisme seperti pengujian beban.
  - Pantau CloudWatch metrik saat menjalankan tes ini untuk mendapatkan pemahaman tentang metrik yang tersedia, dan untuk menetapkan garis dasar kinerja.
- [Terbaik] ElastiCache Untuk beban kerja Valkey dan Redis OSS, ganti nama perintah yang mahal secara komputasi, sepertiKEYS, untuk membatasi kemampuan pengguna menjalankan perintah pemblokiran pada cluster produksi.
  - ElastiCache beban kerja yang menjalankan engine 6.x untuk Redis OSS dapat memanfaatkan kontrol akses berbasis peran untuk membatasi perintah tertentu. Akses ke perintah dapat dikontrol dengan membuat Pengguna dan Grup Pengguna dengan AWS Konsol atau CLI, dan mengaitkan Grup Pengguna ke klaster. Di Redis OSS 6, ketika RBAC diaktifkan, kita dapat menggunakan “- @dangerous" dan itu akan melarang perintah mahal seperti KEYS, MONITOR, SORT, dll. untuk pengguna itu.
  - Untuk engine versi 5.x, ganti nama perintah menggunakan `rename-commands` parameter pada grup parameter cluster.
- [Lebih Baik] Analisis kueri lambat dan cari teknik pengoptimalan.
  - ElastiCache Untuk beban kerja Valkey dan Redis OSS, pelajari lebih lanjut tentang kueri Anda dengan menganalisis Log Lambat. Misalnya, Anda dapat menggunakan perintah berikut, `valkey-cli slowlog get 10` untuk menampilkan 10 perintah terakhir yang melebihi ambang batas latensi (10 detik secara default).
  - Kueri tertentu dapat dilakukan lebih efisien menggunakan kompleks ElastiCache untuk struktur data Valkey dan Redis OSS. Sebagai contoh, untuk pencarian rentang gaya numerik, aplikasi dapat mengimplementasikan indeks numerik sederhana dengan Sorted Set. Mengelola indeks ini dapat mengurangi pemindaian yang dilakukan pada set data, dan menampilkan data dengan efisiensi performa yang lebih baik.
  - ElastiCache Untuk beban kerja Valkey dan Redis OSS, `redis-benchmark` menyediakan antarmuka sederhana untuk menguji kinerja perintah yang berbeda menggunakan input yang ditentukan pengguna seperti jumlah klien, dan ukuran data.
  - Karena Memcached hanya mendukung perintah tingkat kunci sederhana, pertimbangkan untuk membangun kunci tambahan sebagai indeks untuk menghindari iterasi melalui ruang kunci untuk melayani kueri klien.

- [Sumber Daya]:
  - Pemantauan penggunaan dengan CloudWatch Metrik
  - Menggunakan CloudWatch alarm Amazon
  - Parameter spesifik Valkey dan Redis OSS
  - SLOWLOG
  - tolok ukur

PE 2: Bagaimana Anda mendistribusikan pekerjaan di seluruh node ElastiCache Cluster Anda?

Pengenalan tingkat pertanyaan: Cara aplikasi Anda terhubung ke ElastiCache node Amazon dapat memengaruhi kinerja dan skalabilitas klaster.

Manfaat tingkat pertanyaan: Memanfaatkan simpul yang tersedia di klaster dengan tepat akan memastikan bahwa pekerjaan didistribusikan ke seluruh sumber daya yang tersedia. Teknik-teknik berikut juga membantu menghindari sumber daya idle.

- [Wajib] Minta klien terhubung ke ElastiCache titik akhir yang tepat.
  - ElastiCache untuk Valkey dan Redis OSS mengimplementasikan titik akhir yang berbeda berdasarkan mode cluster yang digunakan. Untuk mode cluster diaktifkan, ElastiCache akan menyediakan titik akhir konfigurasi. Untuk mode cluster dinonaktifkan, ElastiCache menyediakan titik akhir utama, biasanya digunakan untuk menulis, dan titik akhir pembaca untuk menyeimbangkan pembacaan di seluruh replika. Menerapkan titik akhir ini dengan tepat akan menghasilkan performa yang lebih baik, dan operasi penskalaan yang lebih mudah. Hindari menghubungkan ke titik akhir simpul individual kecuali jika ada persyaratan khusus untuk melakukan tindakan ini.
  - Untuk cluster Memcached multi-node, ElastiCache menyediakan titik akhir konfigurasi yang memungkinkan Auto Discovery. Sebaiknya gunakan algoritma hashing untuk mendistribusikan pekerjaan secara merata di seluruh simpul cache. Banyak pustaka klien Memcached menerapkan hashing yang konsisten. Periksa dokumentasi untuk pustaka yang Anda gunakan untuk melihat apakah pustaka tersebut mendukung hashing konsisten dan cara menerapkannya. Anda dapat menemukan informasi selengkapnya tentang penerapan fitur-fitur ini di sini.
- [Lebih baik] Manfaatkan ElastiCache untuk cluster yang mengaktifkan mode cluster Valkey dan Redis OSS untuk meningkatkan skalabilitas.

  - ElastiCache untuk cluster Valkey dan Redis OSS (mode cluster enabled) mendukung operasi penskalaan online (out/in and up/down) untuk membantu mendistribusikan data secara dinamis di seluruh pecahan. Menggunakan Titik Akhir Konfigurasi akan memastikan klaster Anda yang sadar klien dapat menyesuaikan perubahan dalam topologi klaster.
  - Anda juga dapat menyeimbangkan kembali cluster dengan memindahkan hashslots antara pecahan yang tersedia di cluster ElastiCache for Valkey dan Redis OSS (mode cluster enabled) Anda. Tindakan ini membantu mendistribusikan pekerjaan secara lebih efisien di seluruh serpihan yang tersedia.
- [Lebih Baik] Terapkan strategi untuk mengidentifikasi dan memulihkan hot key dalam beban kerja Anda.
  - Pertimbangkan dampak struktur data Valkey atau Redis OSS multi-dimensi seperti daftar, aliran, set, dll. Struktur data ini disimpan dalam Keys tunggal, yang berada pada satu node. Kunci multi-dimensi yang sangat besar memiliki potensi untuk memanfaatkan lebih banyak kapasitas jaringan dan memori daripada jenis data lainnya dan dapat menyebabkan penggunaan simpul yang tidak proporsional. Jika memungkinkan, rancang beban kerja Anda untuk menyebarkan akses data di banyak Kunci diskrit.
  - Hot key dalam beban kerja dapat memengaruhi performa simpul yang digunakan. ElastiCache Untuk beban kerja Valkey dan Redis OSS, Anda dapat mendeteksi tombol pintas menggunakan `valkey-cli --hotkeys` jika kebijakan memori maksimum LFU diberlakukan.
  - Pertimbangkan untuk mereplikasi hot key di beberapa simpul untuk mendistribusikan akses ke sana secara lebih merata. Pendekatan ini mengharuskan klien untuk menulis ke beberapa node primer (node Valkey atau Redis OSS itu sendiri tidak akan menyediakan fungsi ini) dan untuk mempertahankan daftar nama kunci untuk dibaca, selain nama kunci asli.
  - ElastiCache engine 7.2 untuk Valkey dan di atasnya, dan ElastiCache versi 6 untuk Redis OSS dan yang lebih tinggi, semuanya mendukung caching sisi klien yang dibantu server. Hal ini memungkinkan aplikasi untuk menunggu perubahan pada kunci sebelum membuat panggilan jaringan kembali keElastiCache.
- [Sumber Daya]:
  - Konfigurasikan ElastiCache untuk Valkey dan Redis OSS untuk ketersediaan yang lebih tinggi
  - Menemukan titik akhir koneksi di ElastiCache
  - Praktik terbaik penyeimbangan beban
  - Resharding online untuk Valkey atau Redis OSS (mode cluster diaktifkan)
  - Caching sisi klien di Valkey dan Redis OSS

PE 3: Untuk beban kerja caching, bagaimana cara melacak dan melaporkan efektivitas dan performa cache Anda?

Pengenalan tingkat pertanyaan: Caching adalah beban kerja yang umum ditemui ElastiCache dan penting bagi Anda untuk memahami cara mengelola efektivitas dan kinerja cache Anda.

Manfaat tingkat pertanyaan: Aplikasi Anda mungkin menunjukkan tanda-tanda performa yang lamban. Kemampuan Anda untuk menggunakan metrik khusus cache untuk menginformasikan keputusan Anda tentang cara meningkatkan performa aplikasi sangat penting untuk beban kerja cache Anda.

- [Wajib] Ukur dan lacak dari waktu ke waktu rasio hit cache. Efisiensi cache Anda ditentukan oleh 'rasio cache hit'. Rasio cache hit ditentukan oleh total hit kunci dibagi dengan total hit dan miss. Semakin dekat ke 1 rasionya, semakin efektif cache Anda. Rasio cache hit yang rendah disebabkan oleh volume cache miss. Cache miss terjadi ketika kunci yang diminta tidak ditemukan di cache. Kunci tidak ada dalam cache karena telah dikosongkan atau dihapus, telah habis masa berlakunya, atau tidak pernah ada. Pahami mengapa kunci tidak ada dalam cache dan kembangkan strategi yang tepat untuk menyimpan kunci dalam cache.

  [Sumber Daya]:

  - Metrik untuk Valkey dan Redis OSS
- [Wajib] Ukur dan kumpulkan kinerja cache aplikasi Anda bersama dengan nilai latensi dan pemanfaatan CPU untuk memahami apakah Anda perlu melakukan penyesuaian pada komponen aplikasi Anda time-to-live atau lainnya. ElastiCache menyediakan satu set CloudWatch metrik untuk latensi agregat untuk setiap struktur data. Metrik latensi ini dihitung menggunakan statistik commandstats dari perintah INFO dan tidak menyertakan jaringan dan waktu I/O. Ini hanya waktu yang dihabiskan ElastiCache untuk memproses operasi.

  [Sumber Daya]:

  - Metrik untuk Valkey dan Redis OSS
  - Memantau praktik terbaik dengan ElastiCache menggunakan Amazon CloudWatch
- [Terbaik] Pilih strategi caching yang tepat untuk kebutuhan Anda. Rasio cache hit yang rendah disebabkan oleh volume cache miss. Jika beban kerja Anda dirancang untuk memiliki volume cache miss yang rendah (seperti komunikasi waktu nyata), sebaiknya tinjau strategi caching Anda dan terapkan solusi yang paling tepat untuk beban kerja Anda, seperti instrumentasi kueri untuk mengukur memori dan performa. Strategi aktual yang Anda gunakan untuk mengisi dan memelihara cache Anda akan tergantung pada data apa yang perlu di-cache oleh klien dan pola

akses ke data tersebut. Misalnya, Anda cenderung tidak akan menggunakan strategi yang sama untuk rekomendasi yang dipersonalisasi pada aplikasi streaming, dan untuk berita yang sedang tren.

[Sumber Daya]:

- Strategi cache untuk Memcached
- Praktik Terbaik Caching
- Kinerja dalam Skala dengan ElastiCache Whitepaper Amazon

PE 4: Bagaimana beban kerja Anda mengoptimalkan penggunaan sumber daya dan koneksi jaringan?

Pengenalan tingkat pertanyaan: ElastiCache untuk Valkey, Memcached, dan Redis OSS didukung oleh banyak klien aplikasi, dan implementasi dapat bervariasi. Anda perlu memahami manajemen jaringan dan koneksi yang diterapkan untuk menganalisis potensi dampak performa.

Manfaat tingkat pertanyaan: Penggunaan sumber daya jaringan yang efisien dapat meningkatkan efisiensi performa klaster Anda. Rekomendasi berikut dapat mengurangi tuntutan jaringan, dan meningkatkan latensi dan throughput klaster.

- [Wajib] Secara proaktif mengelola koneksi ke ElastiCache klaster Anda.
  - Pooling koneksi dalam aplikasi mengurangi jumlah overhead pada klaster yang dibuat dengan membuka dan menutup koneksi. Pantau perilaku koneksi di Amazon CloudWatch menggunakan `CurrConnections` dan`NewConnections`.
  - Hindari kebocoran koneksi dengan menutup koneksi klien dengan benar sesuai keperluan. Strategi manajemen koneksi termasuk menutup dengan benar koneksi yang tidak digunakan, dan pengaturan waktu habis koneksi.
  - Untuk beban kerja Memcached, ada jumlah memori yang dapat dikonfigurasi yang dicadangkan untuk menangani koneksi yang disebut, `memcached_connections_overhead`.
- [Lebih Baik] Kompres objek besar untuk mengurangi memori, dan meningkatkan throughput jaringan.
  - Kompresi data dapat mengurangi jumlah throughput jaringan yang diperlukan (Gbps), tetapi meningkatkan jumlah pekerjaan pada aplikasi untuk mengompres dan mendekompresi data.
  - Kompresi juga mengurangi jumlah memori yang dikonsumsi oleh kunci

  - Berdasarkan kebutuhan aplikasi Anda, pertimbangkan kompromi antara rasio kompresi dan kecepatan kompresi.
- [Sumber Daya]:
  - ElastiCache - Datastore Global
  - Parameter spesifik Memcached
  - ElastiCache versi 5.0.3 untuk Redis OSS meningkatkan penanganan untuk meningkatkan kinerja I/O
  - Metrik untuk Valkey dan Redis OSS
  - Konfigurasikan ElastiCache untuk ketersediaan yang lebih tinggi

PE 5: Bagaimana Anda mengelola penggusuran penghapusan and/or kunci?

Pengenalan tingkat pertanyaan: Beban kerja memiliki persyaratan yang berbeda, dan perilaku yang diharapkan saat node cluster mendekati batas konsumsi memori. ElastiCache memiliki kebijakan yang berbeda untuk menangani situasi ini.

Manfaat tingkat pertanyaan: Manajemen yang tepat atas memori yang tersedia, dan pemahaman tentang kebijakan pengosongan akan membantu memastikan kesadaran perilaku klaster ketika batas memori instans terlampaui.

- [Wajib] Instrumentasi akses data untuk mengevaluasi kebijakan mana yang akan diterapkan. Identifikasi kebijakan max-memory yang sesuai untuk mengontrol apakah dan bagaimana pengosongan dilakukan di klaster.
  - Pengosongan terjadi ketika max-memory pada klaster dikonsumsi dan kebijakan diberlakukan untuk memungkinkan pengosongan. Perilaku klaster dalam situasi ini tergantung pada kebijakan pengosongan yang ditentukan. Kebijakan ini dapat dikelola menggunakan grup parameter `maxmemory-policy` pada klaster.
  - Kebijakan default `volatile-lru` mengosongkan memori dengan mengosongkan kunci dengan waktu kedaluwarsa yang ditetapkan (nilai TTL). Kebijakan least frequently used (LFU) dan least recently used (LRU) menghapus kunci berdasarkan penggunaan.
  - Untuk beban kerja Memcached, ada kebijakan LRU default yang mengendalikan pengosongan di setiap simpul. Jumlah penggusuran di ElastiCache klaster Amazon Anda dapat dipantau menggunakan metrik Penggusuran di Amazon. CloudWatch
- [Lebih Baik] Lakukan standarisasi perilaku penghapusan untuk mengontrol dampak performa pada klaster Anda untuk menghindari kemacetan performa yang tidak terduga.

  - ElastiCache Untuk beban kerja Valkey dan Redis OSS, ketika secara eksplisit menghapus kunci dari cluster, sepertiDEL: menghapus kunci yang UNLINK ditentukan. Namun, perintah ini mengklaim kembali memori yang sebenarnya di thread yang berbeda, sehingga tidak memblokir, sedangkan DEL memblokir. Penghapusan yang sebenarnya akan terjadi nanti secara asinkron.
  - Untuk ElastiCache versi 6.x untuk beban kerja Redis OSS, perilaku DEL perintah dapat dimodifikasi dalam grup parameter menggunakan parameter. lazyfree-lazy-user-del
- [Sumber Daya]:
  - Mengkonfigurasi parameter mesin menggunakan grup ElastiCache parameter
  - UNLINK
  - Manajemen Keuangan Cloud dengan AWS

PE 6: Bagaimana Anda memodelkan dan berinteraksi dengan data ElastiCache?

Pengenalan tingkat pertanyaan: ElastiCache sangat bergantung pada struktur data dan model data yang digunakan, tetapi juga perlu mempertimbangkan penyimpanan data yang mendasarinya (jika ada). Pahami struktur data yang tersedia dan pastikan Anda menggunakan struktur data yang paling tepat untuk kebutuhan Anda.

Manfaat tingkat pertanyaan: Pemodelan data ElastiCache memiliki beberapa lapisan, termasuk kasus penggunaan aplikasi, tipe data, dan hubungan antar elemen data. Selain itu, setiap tipe data dan perintah memiliki tanda tangan kinerja mereka sendiri yang terdokumentasi dengan baik.

- [Terbaik] Praktik terbaiknya adalah mengurangi penimpaan data yang tidak disengaja. Gunakan konvensi penamaan yang meminimalkan tumpang-tindih nama kunci. Penamaan struktur data secara konvensional menggunakan metode hierarkis seperti: APPNAME:CONTEXT:ID, misalnya ORDER-APP:CUSTOMER:123.

  [Sumber Daya]:

  - Key naming
- [Terbaik] ElastiCache untuk perintah Valkey dan Redis OSS memiliki kompleksitas waktu yang ditentukan oleh notasi Big O. Kompleksitas perintah kali ini merupakan algorithmic/mathematical representasi dari dampaknya. Saat memperkenalkan jenis data baru dalam aplikasi, Anda perlu meninjau kompleksitas waktu perintah terkait dengan cermat. Perintah dengan kompleksitas waktu O(1) bersifat konstan dalam waktu dan tidak bergantung pada ukuran input, namun perintah dengan kompleksitas waktu O(N) bersifat linier dalam waktu dan menyesuaikan ukuran input. Karena desain ulir tunggal ElastiCache untuk Valkey dan Redis OSS, volume besar operasi

kompleksitas waktu tinggi akan menghasilkan kinerja yang lebih rendah dan potensi batas waktu operasi.

[Sumber Daya]:

- Commands

- [Terbaik] Gunakan APIs untuk mendapatkan visibilitas GUI ke dalam model data di cluster Anda.

[Sumber Daya]:

- Komandan Redis OSS
- Browser Redis OSS
- Redsmin

PE 7: Bagaimana Anda mencatat perintah yang berjalan lambat di ElastiCache cluster Amazon Anda?

Pengantar tingkat pertanyaan: Manfaat penyesuaian performa melalui pengambilan, agregasi, dan notifikasi perintah yang berjalan lama. Dengan memahami berapa lama waktu yang dibutuhkan untuk menjalankan perintah, Anda dapat menentukan perintah mana yang menghasilkan kinerja yang buruk serta perintah yang menghalangi mesin agar tidak berkinerja optimal. ElastiCache juga memiliki kemampuan untuk meneruskan informasi ini ke Amazon CloudWatch atau Amazon Kinesis Data Firehose.

Manfaat tingkat pertanyaan: Pencatatan log ke lokasi permanen khusus dan penyediaan peristiwa notifikasi untuk perintah lambat dapat membantu analisis performa terperinci dan dapat digunakan untuk memicu peristiwa otomatis.

- [Diperlukan] ElastiCache menjalankan mesin Valkey 7.2 atau yang lebih baru, atau menjalankan mesin Redis OSS versi 6.0 atau yang lebih baru, grup parameter yang dikonfigurasi dengan benar dan logging SLOWLOG diaktifkan di cluster.
  - Parameter yang diperlukan hanya tersedia ketika kompatibilitas versi mesin diatur ke Valkey 7.2 dan lebih tinggi, atau Redis OSS versi 6.0 atau lebih tinggi.
  - Pencatatan log SLOWLOG terjadi ketika waktu eksekusi server terhadap suatu perintah membutuhkan waktu lebih lama dari nilai yang ditentukan. Perilaku klaster tergantung pada parameter Grup Parameter terkait yaitu `slowlog-log-slower-than` dan `slowlog-max-len`.
  - Perubahan akan diterapkan segera.
- [Terbaik] Manfaatkan CloudWatch atau kemampuan Kinesis Data Firehose.

- Gunakan kemampuan pemfilteran dan alarm, Wawasan CloudWatch Log CloudWatch, dan Layanan Pemberitahuan Sederhana Amazon untuk mencapai pemantauan kinerja dan pemberitahuan peristiwa.
- Gunakan kemampuan streaming Kinesis Data Firehose untuk mengarsipkan log SLOWLOG ke penyimpanan permanen atau untuk memicu penyesuaian parameter klaster otomatis.
- Tentukan apakah JSON atau format TEXT biasa yang paling sesuai dengan kebutuhan Anda.
- Berikan izin IAM untuk mempublikasikan ke CloudWatch atau Kinesis Data Firehose.

- [Lebih Baik] Konfigurasikan `slowlog-log-slower-than` ke nilai selain default.
  - Parameter ini menentukan berapa lama perintah dapat dijalankan untuk dalam mesin Valkey atau Redis OSS sebelum dicatat sebagai perintah berjalan lambat. Nilai default-nya adalah 10.000 mikrodetik (10 milidetik). Nilai default mungkin terlalu tinggi untuk beberapa beban kerja.
  - Tentukan nilai yang lebih sesuai untuk beban kerja Anda berdasarkan kebutuhan aplikasi dan hasil pengujian; namun, nilai yang terlalu rendah dapat menghasilkan data yang berlebihan.
- [Lebih Baik] Biarkan `slowlog-max-len` pada nilai default.
  - Parameter ini menentukan batas atas berapa banyak perintah yang berjalan lambat ditangkap dalam memori Valkey atau Redis OSS pada waktu tertentu. Nilai 0 secara efektif menonaktifkan penangkapan. Semakin tinggi nilainya, semakin banyak entri yang akan disimpan dalam memori, sehingga mengurangi kemungkinan informasi penting dikosongkan sebelum dapat ditinjau. Nilai default-nya adalah 128.
  - Nilai default tersebut sesuai untuk sebagian besar beban kerja. Jika ada kebutuhan untuk menganalisis data dalam jendela waktu yang diperluas dari valkey-cli melalui perintah SLOWLOG, pertimbangkan untuk meningkatkan nilai ini. Ini memungkinkan lebih banyak perintah untuk tetap berada di memori Valkey atau Redis OSS.

    Jika Anda memancarkan data SLOWLOG ke Log atau Kinesis Data Firehose, data akan bertahan dan dapat dianalisis di luar sistem, mengurangi kebutuhan untuk menyimpan sejumlah besar perintah yang berjalan lambat di memori Valkey atau Redis OSS. CloudWatch ElastiCache
- [Sumber Daya]:
  - Bagaimana cara mengaktifkan log lambat di cluster cache?
  - Pengiriman log
  - Parameter khusus Redis OS
  - https://aws.amazon.com/cloudwatch/Amazon CloudWatch
  - Amazon Kinesis Data Firehose

PE8: Bagaimana Auto Scaling membantu dalam meningkatkan kinerja cluster? ElastiCache

Pengenalan tingkat pertanyaan: Dengan menerapkan fitur penskalaan otomatis Valkey atau Redis OSS, ElastiCache komponen Anda dapat menyesuaikan dari waktu ke waktu untuk menambah atau mengurangi pecahan atau replika yang diinginkan secara otomatis. Ini dapat dilakukan dengan menerapkan kebijakan pelacakan target atau penskalaan terjadwal.

Manfaat tingkat pertanyaan: Memahami dan merencanakan lonjakan beban kerja dapat memastikan peningkatan kinerja caching dan kelangsungan bisnis. ElastiCache Auto Scaling terus memantau pemanfaatan CPU/memori Anda untuk memastikan klaster Anda beroperasi pada tingkat kinerja yang Anda inginkan.

- [Diperlukan] Saat meluncurkan cluster ElastiCache untuk Valkey atau Redis OSS:
  1. Pastikan mode Klaster diaktifkan
  2. Pastikan instans berasal dari keluarga jenis dan ukuran tertentu yang mendukung Auto Scaling
  3. Pastikan klaster tidak berjalan di Penyimpanan Data Global, Outposts, atau Zona Lokal

  [Sumber Daya]:
  - Penskalaan cluster di Valkey dan Redis OSS (Mode Cluster Diaktifkan)
  - Menggunakan Auto Scaling dengan serpihan
  - Menggunakan Auto Scaling dengan replika
- [Terbaik] Identifikasi apakah beban kerja Anda menjalankan operasi baca atau tulis yang berat untuk menentukan kebijakan penskalaan. Untuk performa terbaik, gunakan hanya satu metrik pelacakan. Sebaiknya hindari beberapa kebijakan untuk setiap dimensi, karena kebijakan Auto Scaling akan menskalakan keluar saat target tercapai, namun menskalakan masuk hanya jika semua kebijakan pelacakan target siap untuk diskalakan.

  [Sumber Daya]:
  - Kebijakan Auto Scaling
  - Menetapkan kebijakan penskalaan
- [Terbaik] Memantau performa dari waktu ke waktu dapat membantu Anda mendeteksi perubahan beban kerja yang tidak akan terdeteksi jika dipantau pada titik waktu tertentu. Anda dapat menganalisis CloudWatch metrik yang sesuai untuk pemanfaatan klaster selama periode empat minggu untuk menentukan ambang nilai target. Jika Anda masih tidak yakin nilai apa yang harus dipilih, sebaiknya mulai dengan nilai metrik standar minimum yang didukung.

[Sumber Daya]:

- Pemantauan penggunaan dengan CloudWatch Metrik

- [Lebih baik] Kami menyarankan untuk menguji aplikasi Anda dengan beban kerja minimum dan maksimum yang diharapkan, untuk mengidentifikasi jumlah pasti yang shards/replicas diperlukan kluster untuk mengembangkan kebijakan penskalaan dan mengurangi masalah ketersediaan.

  [Sumber Daya]:

  - Mendaftarkan Target yang Dapat Diskalakan
  - Mendaftarkan Target yang Dapat Diskalakan menggunakan AWS CLI

Pilar ElastiCache Pengoptimalan Biaya Lensa Well-Architected Amazon

Pilar optimisasi biaya berfokus untuk menghindari biaya yang tidak perlu. Topik utamanya termasuk memahami dan mengendalikan ke mana biaya dihabiskan, memilih jenis simpul yang paling tepat (menggunakan instans yang mendukung tingkatan data berdasarkan kebutuhan beban kerja), jumlah jenis sumber daya yang tepat (berapa banyak replika baca), menganalisis pengeluaran dari waktu ke waktu, dan penskalaan untuk memenuhi kebutuhan bisnis tanpa pengeluaran berlebihan.

Topik

- BIAYA 1: Bagaimana Anda mengidentifikasi dan melacak biaya yang terkait dengan ElastiCache sumber daya Anda? Bagaimana Anda mengembangkan mekanisme untuk memungkinkan pengguna membuat, mengelola, dan menghapus sumber daya yang dibuat?
- BIAYA 2: Bagaimana Anda menggunakan alat pemantauan berkelanjutan untuk membantu Anda mengoptimalkan biaya yang terkait dengan ElastiCache sumber daya Anda?
- BIAYA 3: Haruskah Anda menggunakan jenis instans yang mendukung tingkatan data? Apa keuntungan dari tingkatan data? Kapan sebaiknya tidak menggunakan instans tingkatan data?

BIAYA 1: Bagaimana Anda mengidentifikasi dan melacak biaya yang terkait dengan ElastiCache sumber daya Anda? Bagaimana Anda mengembangkan mekanisme untuk memungkinkan pengguna membuat, mengelola, dan menghapus sumber daya yang dibuat?

Pengantar tingkat pertanyaan: Untuk memahami metrik biaya, diperlukan partisipasi dan kolaborasi berbagai tim: rekayasa perangkat lunak, manajemen data, pemilik produk, keuangan, dan pimpinan.

Untuk mengidentifikasi pendorong biaya utama, semua pihak yang terlibat harus memahami pemicu penggunaan layanan dan kompromi manajemen biaya. Hal ini juga sering kali menjadi perbedaan utama antara upaya optimisasi biaya yang berhasil dan kurang berhasil. Memastikan Anda memiliki proses dan alat untuk melacak sumber daya yang dibuat dari pengembangan hingga produksi dan pensiun membantu Anda mengelola biaya yang terkait dengannyaElastiCache.

Manfaat tingkat pertanyaan: Pelacakan terus menerus dari semua biaya yang terkait dengan beban kerja Anda membutuhkan pemahaman mendalam tentang arsitektur yang termasuk ElastiCache sebagai salah satu komponennya. Selain itu, Anda harus memiliki rencana manajemen biaya untuk mengumpulkan dan membandingkan data penggunaan dengan anggaran Anda.

- [Diperlukan] Lengkapi Cloud Center of Excellence (CCoE) dengan salah satu piagam pendirinya untuk menentukan, melacak, dan mengambil tindakan pada metrik seputar penggunaan organisasi Anda. ElastiCache Jika CCo E ada dan berfungsi, pastikan ia tahu cara membaca dan melacak biaya yang terkait dengannya ElastiCache. Saat sumber daya dibuat, gunakan peran dan kebijakan IAM untuk memvalidasi bahwa hanya tim dan grup tertentu yang dapat melakukan instansiasi sumber daya. Hal ini memastikan bahwa biaya yang terkait dengan hasil bisnis dan garis akuntabilitas yang jelas ditetapkan, dari perspektif biaya.
  1. CCoE harus mengidentifikasi, mendefinisikan, dan mempublikasikan metrik biaya yang diperbarui secara reguler -bulanan seputar ElastiCache penggunaan utama di seluruh data kategoris seperti:
     a. Jenis simpul yang digunakan dan atributnya: standar vs. memori dioptimalkan, instans sesuai permintaan vs. terpesan, wilayah, dan zona ketersediaan
     b. Jenis lingkungan: gratis, developer, pengujian, dan produksi
     c. Strategi penyimpanan dan retensi cadangan
     d. Transfer data dalam dan lintas wilayah
     e. Instans yang berjalan di Amazon Outposts
  2. CCoE terdiri dari tim lintas fungsi dengan representasi non-eksklusif dari rekayasa perangkat lunak, manajemen data, tim produk, keuangan, dan tim kepemimpinan di organisasi Anda.

  [Sumber Daya]:
  - Buat Pusat Keunggulan Cloud
  - ElastiCacheHarga Amazon
- [Wajib] Gunakan tag alokasi biaya untuk memantau biaya pada tingkat granularitas yang rendah. Gunakan Manajemen AWS Biaya untuk memvisualisasikan, memahami, dan mengelola AWS biaya dan penggunaan Anda dari waktu ke waktu.

1. Gunakan tag untuk mengatur sumber daya Anda, dan tag alokasi biaya untuk melacak AWS biaya Anda pada tingkat terperinci. Setelah Anda mengaktifkan tag alokasi biaya, AWS gunakan tag alokasi biaya untuk mengatur biaya sumber daya Anda pada laporan alokasi biaya Anda, untuk memudahkan Anda mengkategorikan dan melacak biaya Anda. AWS AWS menyediakan dua jenis tag alokasi biaya, tag yang AWS dihasilkan dan tag yang ditentukan pengguna. AWS mendefinisikan, membuat, dan menerapkan tag yang AWS dihasilkan untuk Anda, dan Anda menentukan, membuat, dan menerapkan tag yang ditentukan pengguna. Anda harus mengaktifkan kedua jenis tag ini secara terpisah sebelum tag tersebut dapat muncul di Manajemen Biaya atau laporan alokasi biaya.
2. Gunakan tag alokasi biaya untuk mengatur AWS tagihan Anda untuk mencerminkan struktur biaya Anda sendiri. Ketika Anda menambahkan tag alokasi biaya ke sumber daya Anda di Amazon ElastiCache, Anda akan dapat melacak biaya dengan mengelompokkan pengeluaran pada faktur Anda berdasarkan nilai tag sumber daya. Anda sebaiknya mengombinasikan beberapa tag untuk melacak biaya secara lebih mendetail.

   [Sumber Daya]:

   - Menggunakan AWS tag alokasi biaya
   - Memantau biaya dengan tag alokasi biaya
   - AWS Cost Explorer

- [Terbaik] Connect ElastiCache biaya ke metrik yang menjangkau seluruh organisasi.
  1. Pertimbangkan metrik bisnis serta metrik operasional seperti latensi - konsep apa dalam model bisnis Anda yang dapat dimengerti di seluruh peran? Metrik harus dapat dipahami oleh sebanyak mungkin peran dalam organisasi.
  2. Contoh - pengguna yang dilayani secara simultan, latensi maks dan rata-rata per operasi dan pengguna, skor keterlibatan pengguna, pengembalian pengguna, tingkat pengabaianrates/week, session length/user, tingkat hit cache, dan kunci yang dilacak

  [Sumber Daya]:

  - Pemantauan penggunaan dengan CloudWatch Metrik
- [Baik] Pertahankan visibilitas up-to-date arsitektur dan operasional pada metrik dan biaya di seluruh beban kerja yang digunakan. ElastiCache
  1. Pahami seluruh ekosistem solusi Anda, ElastiCache cenderung menjadi bagian dari ekosistem AWS layanan lengkap dalam rangkaian teknologinya, dari klien hingga API Gateway, Redshift, dan QuickSight untuk alat pelaporan (misalnya).

2. Petakan komponen solusi Anda dari klien, koneksi, keamanan, operasi dalam memori, penyimpanan, otomatisasi sumber daya, akses, dan manajemen data, di diagram arsitektur Anda. Setiap lapisan terhubung ke seluruh solusi dan memiliki kebutuhan dan kemampuan sendiri yang menambah untuk and/or membantu Anda mengelola biaya keseluruhan.
3. Diagram Anda harus mencakup penggunaan komputasi, jaringan, penyimpanan, kebijakan siklus hidup, pengumpulan metrik serta elemen operasional dan fungsional ElastiCache aplikasi Anda
4. Persyaratan beban kerja Anda cenderung berubah dari waktu ke waktu dan penting bagi Anda untuk terus memelihara dan mendokumentasikan pemahaman Anda tentang komponen yang mendasarinya serta tujuan fungsional utama Anda agar tetap proaktif dalam manajemen biaya beban kerja Anda.
5. Dukungan eksekutif untuk visibilitas, akuntabilitas, prioritas, dan sumber daya sangat penting bagi Anda untuk memiliki strategi manajemen biaya yang efektif untuk Anda. ElastiCache

BIAYA 2: Bagaimana Anda menggunakan alat pemantauan berkelanjutan untuk membantu Anda mengoptimalkan biaya yang terkait dengan ElastiCache sumber daya Anda?

Pengenalan tingkat pertanyaan: Anda perlu membidik keseimbangan yang tepat antara metrik ElastiCache biaya dan kinerja aplikasi Anda. Amazon CloudWatch menyediakan visibilitas ke metrik operasional utama yang dapat membantu Anda menilai apakah ElastiCache sumber daya Anda terlalu banyak atau kurang digunakan, relatif terhadap kebutuhan Anda. Dari perspektif pengoptimalan biaya, Anda perlu memahami kapan Anda dilebih-lebihkan dan dapat mengembangkan mekanisme yang tepat untuk mengubah ukuran ElastiCache sumber daya Anda sambil mempertahankan kebutuhan operasional, ketersediaan, ketahanan, dan kinerja Anda.

Manfaat tingkat pertanyaan: Dalam keadaan yang ideal, Anda akan menyediakan sumber daya yang cukup untuk memenuhi kebutuhan operasional beban kerja Anda dan tidak memiliki sumber daya yang kurang dimanfaatkan yang dapat mengakibatkan situasi biaya yang kurang optimal. Anda harus dapat mengidentifikasi dan menghindari pengoperasian ElastiCache sumber daya yang terlalu besar untuk jangka waktu yang lama.

- [Wajib] Gunakan CloudWatch untuk memantau ElastiCache klaster Anda dan menganalisis bagaimana metrik ini berhubungan dengan dasbor AWS Cost Explorer Anda.

1. ElastiCache menyediakan metrik tingkat host (misalnya, penggunaan CPU) dan metrik yang khusus untuk perangkat lunak mesin cache (misalnya, cache mendapat dan kehilangan cache). Metrik ini diukur dan dipublikasikan untuk setiap simpul cache dalam interval 60 detik.
2. ElastiCache metrik kinerja (CPUUtilization,, EngineUtilization, SwapUsage CurrConnections, dan Penggusuran) dapat menunjukkan bahwa Anda perlu menskalakan up/down (menggunakan jenis node larger/smaller cache) atau in/out (add more/less pecahan). Pahami implikasi biaya dari keputusan penskalaan dengan membuat matriks buku pedoman yang memperkirakan biaya tambahan dan lama waktu min dan maks yang diperlukan untuk mencapai ambang batas performa aplikasi Anda.

[Sumber Daya]:

- Pemantauan penggunaan dengan CloudWatch Metrik
- Metrik Apa yang Harus Saya Pantau?
- ElastiCacheHarga Amazon

- [Wajib] Pahami dan dokumentasikan strategi pencadangan dan implikasi biaya Anda.
  1. Dengan ElastiCache, cadangan disimpan di Amazon S3, yang menyediakan penyimpanan tahan lama. Anda perlu memahami implikasi biaya dalam kaitannya dengan kemampuan Anda untuk pulih dari kegagalan.
  2. Aktifkan pencadangan otomatis yang akan menghapus file cadangan yang melewati batas retensi.

  [Sumber Daya]:

  - Menjadwalkan pencadangan otomatis
  - Harga Amazon Simple Storage Service
- [Terbaik] Gunakan Simpul Terpesan untuk instans Anda sebagai strategi yang disengaja untuk mengelola biaya beban kerja yang dipahami dan didokumentasikan dengan baik. Simpul terpesan dikenai biaya di muka dan bergantung pada jenis simpul dan durasi pemesanan—satu atau tiga tahun. Biaya ini jauh lebih kecil daripada biaya penggunaan per jam yang dikenakan untuk simpul sesuai permintaan.
  1. Anda mungkin perlu mengoperasikan ElastiCache klaster Anda menggunakan node sesuai permintaan sampai Anda mengumpulkan data yang cukup untuk memperkirakan persyaratan instance yang dicadangkan. Rencanakan dan dokumentasikan sumber daya yang diperlukan untuk memenuhi kebutuhan Anda dan membandingkan biaya yang diharapkan di seluruh jenis instans (sesuai permintaan vs. terpesan)

2. Evaluasi secara rutin jenis simpul cache baru yang tersedia dan nilai apakah masuk akal, dari perspektif metrik biaya dan operasional, untuk memigrasikan armada instans Anda ke jenis simpul cache baru

BIAYA 3: Haruskah Anda menggunakan jenis instans yang mendukung tingkatan data? Apa keuntungan dari tingkatan data? Kapan sebaiknya tidak menggunakan instans tingkatan data?

Pengantar tingkat pertanyaan: Memilih jenis instans yang sesuai tidak hanya dapat memiliki dampak performa dan tingkat layanan, tetapi juga dampak finansial. Jenis instans memiliki biaya berbeda yang terkait dengannya. Memilih satu atau beberapa jenis instans besar yang dapat mengakomodasi semua kebutuhan penyimpanan dalam memori mungkin merupakan keputusan yang wajar. Namun, hal ini dapat memiliki dampak biaya yang signifikan seiring proyek berkembang. Memastikan bahwa jenis instance yang benar dipilih memerlukan pemeriksaan periodik waktu idle ElastiCache objek.

Manfaat tingkat pertanyaan: Anda harus memiliki pemahaman yang jelas tentang bagaimana berbagai jenis instans memengaruhi biaya Anda saat ini dan di masa depan. Perubahan beban kerja marginal atau berkala seharusnya tidak menyebabkan perubahan biaya yang tidak proporsional. Jika beban kerja mengizinkannya, jenis instans yang mendukung tingkatan data menawarkan harga yang lebih baik per penyimpanan yang tersedia. Karena penyimpanan SSD yang tersedia per instans, instans tingkatan data mendukung kemampuan total data per instans yang jauh lebih tinggi.

- [Wajib] Pahami batasan instans tingkatan data
  1. Hanya tersedia ElastiCache untuk klaster Valkey atau Redis OSS.
  2. Hanya jenis instans tertentu yang mendukung tingkatan data.
  3. Hanya ElastiCache versi 6.2 untuk Redis OSS dan di atasnya yang didukung
  4. Item besar tidak ditukar ke SSD. Objek di atas 128 MiB dipertahankan dalam memori.

  [Sumber Daya]:

  - Tingkatan data
  - ElastiCacheHarga Amazon
- [Wajib] Pahami berapa persentase basis data Anda yang secara teratur diakses oleh beban kerja Anda.

1. Instans tingkatan data ideal untuk beban kerja yang sering mengakses sebagian kecil dari keseluruhan set data Anda, tetapi masih memerlukan akses cepat ke data yang tersisa. Dengan kata lain, rasio data panas ke hangat adalah sekitar 20:80.
2. Kembangkan pelacakan tingkat klaster terhadap waktu idle objek.
3. Implementasi besar lebih dari 500 Gb data merupakan kandidat yang baik

- [Wajib] Pahami bahwa instans tingkatan data tidak bersifat opsional untuk beban kerja tertentu.
  1. Ada biaya performa kecil untuk mengakses objek yang jarang digunakan karena objek ini ditukar ke SSD lokal. Jika aplikasi Anda sensitif terhadap waktu respons, uji dampaknya terhadap beban kerja Anda.
  2. Tidak cocok untuk cache yang menyimpan sebagian besar objek berukuran lebih dari 128 MiB.

  [Sumber Daya]:

  - Batasan
- [Terbaik] Jenis instans terpesan mendukung tingkatan data. Hal ini menjamin biaya terendah dalam hal jumlah penyimpanan data per instans.
  1. Anda mungkin perlu mengoperasikan ElastiCache cluster Anda menggunakan instance tiering non-data sampai Anda memiliki pemahaman yang lebih baik tentang kebutuhan Anda.
  2. Analisis pola penggunaan data ElastiCache cluster Anda.
  3. Buat pekerjaan otomatis yang secara berkala mengumpulkan waktu idle objek.
  4. Jika Anda melihat bahwa persentase besar (sekitar 80%) objek berada dalam kondisi idle untuk jangka waktu yang dianggap sesuai untuk beban kerja Anda, dokumentasikan temuan tersebut dan sarankan untuk memigrasikan klaster ke instans yang mendukung tingkatan data.
  5. Evaluasi secara rutin jenis simpul cache baru yang tersedia dan nilai apakah masuk akal, dari perspektif metrik biaya dan operasional, untuk memigrasikan armada instans Anda ke jenis simpul cache baru.

  [Sumber Daya]:

  - OBJECT IDLETIME
  - ElastiCacheHarga Amazon

Langkah-langkah pemecahan masalah umum dan praktik terbaik dengan ElastiCache

Topik berikut memberikan saran pemecahan masalah untuk kesalahan dan masalah yang mungkin Anda temui saat menggunakan. ElastiCache Jika Anda menemukan masalah yang tidak tercantum di sini, Anda dapat menggunakan tombol umpan balik di halaman ini untuk melaporkannya.

Untuk saran pemecahan masalah lainnya dan jawaban atas pertanyaan dukungan umum, kunjungi Pusat Pengetahuan AWS

Topik

Masalah koneksi

Jika Anda tidak dapat terhubung ke ElastiCache cache Anda, pertimbangkan salah satu dari berikut ini:

1. Menggunakan TLS: Jika Anda mengalami koneksi macet saat mencoba terhubung ke ElastiCache titik akhir Anda, Anda mungkin tidak menggunakan TLS di klien Anda. Jika Anda menggunakan ElastiCache Tanpa Server, enkripsi dalam perjalanan selalu diaktifkan. Pastikan klien Anda menggunakan TLS untuk terhubung ke cache. Pelajari lebih lanjut tentang menghubungkan ke cache yang diaktifkan TLS.
2. VPC: ElastiCache cache hanya dapat diakses dari dalam VPC. Pastikan bahwa EC2 instance dari mana Anda mengakses cache dan ElastiCache cache dibuat dalam VPC yang sama. Atau, Anda harus mengaktifkan peering VPC antara VPC tempat instance Anda EC2 berada dan VPC tempat Anda membuat cache.
3. Grup keamanan: ElastiCache menggunakan grup keamanan untuk mengontrol akses ke cache Anda. Pertimbangkan hal berikut:

a. Pastikan bahwa grup keamanan yang digunakan oleh ElastiCache cache Anda memungkinkan akses masuk ke sana dari EC2 instance Anda. Lihat di sini untuk mempelajari cara mengatur aturan masuk di grup keamanan Anda dengan benar.

b. Pastikan bahwa grup keamanan yang digunakan oleh ElastiCache cache Anda memungkinkan akses ke port cache Anda (6379 dan 6380 untuk tanpa server, dan 6379 secara default untuk dirancang sendiri). ElastiCache menggunakan port ini untuk menerima perintah Valkey atau Redis OSS. Pelajari lebih lanjut tentang cara mengatur akses port di sini.

Jika koneksi terus sulit, lihat Masalah koneksi persisten langkah-langkah lain.

Kesalahan klien Valkey atau Redis OSS

ElastiCache Tanpa server hanya dapat diakses menggunakan klien yang mendukung protokol mode cluster Valkey atau Redis OSS. Cluster yang dirancang sendiri dapat diakses dari klien dalam mode mana pun, tergantung pada konfigurasi cluster.

Jika Anda mengalami kesalahan pada klien Anda, pertimbangkan hal berikut:

1. Mode cluster: Jika Anda mengalami kesalahan atau kesalahan CROSSLOT dengan perintah SELECT, Anda mungkin mencoba mengakses cache Cluster Mode Enabled dengan klien Valkey atau Redis OSS yang tidak mendukung protokol Cluster. ElastiCache Tanpa server hanya mendukung klien yang mendukung protokol cluster Valkey atau Redis OSS. Jika Anda ingin menggunakan Valkey atau Redis OSS di “Cluster Mode Disabled” (CMD), maka Anda harus mendesain cluster Anda sendiri.
2. Kesalahan CROSSLOT: Jika Anda mengalami `ERR CROSSLOT Keys in request don't hash to the same slot` kesalahan, Anda mungkin mencoba mengakses kunci yang bukan milik slot yang sama dalam cache mode Cluster. Sebagai pengingat, ElastiCache Serverless selalu beroperasi dalam Mode Cluster. Operasi multi-kunci, transaksi, atau skrip Lua yang melibatkan beberapa kunci hanya diperbolehkan jika semua kunci yang terlibat berada dalam slot hash yang sama.

Untuk praktik terbaik tambahan seputar mengonfigurasi klien Valkey atau Redis OSS, silakan tinjau posting blog ini.

Memecahkan masalah latensi tinggi di Tanpa Server ElastiCache

Jika beban kerja Anda tampaknya mengalami latensi tinggi, Anda dapat menganalisis CloudWatch `SuccessfulReadRequestLatency` dan `SuccessfulWriteRequestLatency` metrik untuk memeriksa apakah latensi terkait dengan Tanpa Server. ElastiCache Metrik ini mengukur latensi yang internal ke ElastiCache Tanpa Server - latensi sisi klien dan waktu perjalanan jaringan antara klien Anda dan titik akhir Tanpa ElastiCache Server tidak disertakan.

Memecahkan masalah latensi sisi klien

Jika Anda melihat peningkatan latensi di sisi klien tetapi tidak ada peningkatan CloudWatch `SuccessfulReadRequestLatency` dan `SuccessfulWriteRequestLatency` metrik yang sesuai yang mengukur latensi sisi server, pertimbangkan hal berikut:

- Pastikan grup keamanan memungkinkan akses ke port 6379 dan 6380: ElastiCache Tanpa server menggunakan port 6379 untuk titik akhir primer dan port 6380 untuk titik akhir pembaca. Beberapa klien membuat konektivitas ke kedua port untuk setiap koneksi baru, bahkan jika aplikasi Anda tidak menggunakan fitur Baca dari Replika. Jika grup keamanan Anda tidak mengizinkan akses masuk ke kedua port, maka pembentukan koneksi dapat memakan waktu lebih lama. Pelajari lebih lanjut tentang cara mengatur akses port di sini.

Memecahkan masalah latensi sisi server

Beberapa variabilitas dan lonjakan sesekali seharusnya tidak menjadi perhatian. Namun, jika `Average` statistik menunjukkan peningkatan tajam dan berlanjut, Anda harus memeriksa AWS Health Dashboard dan Dashboard Kesehatan Pribadi Anda untuk informasi lebih lanjut. Jika perlu, pertimbangkan untuk membuka kasus dukungan dengan Dukungan.

Pertimbangkan praktik dan strategi terbaik berikut untuk mengurangi latensi:

- Aktifkan Baca dari Replika: Jika aplikasi Anda mengizinkannya, sebaiknya aktifkan fitur “Baca dari Replika” di klien Valkey atau Redis OSS Anda untuk menskalakan pembacaan dan mencapai latensi yang lebih rendah. Saat diaktifkan, ElastiCache Serverless mencoba merutekan permintaan baca Anda ke replika node cache yang berada di Availability Zone (AZ) yang sama dengan klien Anda, sehingga menghindari latensi jaringan lintas-AZ. Perhatikan, bahwa mengaktifkan fitur Baca dari Replika di klien Anda menandakan bahwa aplikasi Anda akhirnya menerima konsistensi data. Aplikasi Anda mungkin menerima data lama untuk beberapa waktu jika Anda mencoba membaca setelah menulis ke kunci.

- Pastikan aplikasi Anda di-deploy AZs sama dengan cache Anda: Anda dapat mengamati latensi sisi klien yang lebih tinggi jika aplikasi Anda tidak di-deploy AZs sama dengan cache Anda. Saat Anda membuat cache tanpa server, Anda dapat menyediakan subnet dari mana aplikasi Anda akan mengakses cache, dan ElastiCache Tanpa Server membuat VPC Endpoint di subnet tersebut. Pastikan aplikasi Anda digunakan dalam hal yang sama AZs. Jika tidak, aplikasi Anda mungkin mengalami lompatan lintas-AZ saat mengakses cache yang menghasilkan latensi sisi klien yang lebih tinggi.
- Gunakan kembali koneksi: Permintaan ElastiCache tanpa server dibuat melalui koneksi TCP yang diaktifkan TLS menggunakan protokol RESP. Memulai koneksi (termasuk mengautentikasi koneksi, jika dikonfigurasi) membutuhkan waktu sehingga latensi permintaan pertama lebih tinggi dari biasanya. Permintaan melalui koneksi yang sudah diinisialisasi memberikan ElastiCache latensi rendah yang konsisten. Untuk alasan ini, Anda harus mempertimbangkan untuk menggunakan penyatuan koneksi atau menggunakan kembali koneksi Valkey atau Redis OSS yang ada.
- Kecepatan penskalaan: ElastiCache Tanpa server secara otomatis menskalakan seiring dengan bertambahnya tingkat permintaan Anda. Peningkatan besar secara tiba-tiba dalam tingkat permintaan, lebih cepat daripada kecepatan skala ElastiCache Tanpa Server, dapat mengakibatkan peningkatan latensi untuk beberapa waktu. ElastiCache Tanpa server biasanya dapat meningkatkan tingkat permintaan yang didukung dengan cepat, membutuhkan waktu hingga 10-12 menit untuk menggandakan tingkat permintaan.
- Periksa perintah yang berjalan lama: Beberapa perintah Valkey atau Redis OSS, termasuk skrip Lua atau perintah pada struktur data besar, dapat berjalan untuk waktu yang lama. Untuk mengidentifikasi perintah ini, ElastiCache menerbitkan metrik tingkat perintah. Dengan ElastiCache Tanpa Server Anda dapat menggunakan metrik. `BasedECPUs`
- Permintaan Terbatas: Ketika permintaan dibatasi di ElastiCache Tanpa Server, Anda mungkin mengalami peningkatan latensi sisi klien dalam aplikasi Anda. Saat permintaan dibatasi di ElastiCache Tanpa Server, Anda akan melihat peningkatan metrik Tanpa Server. `ThrottledRequests` ElastiCache Tinjau bagian di bawah ini untuk memecahkan masalah permintaan yang dibatasi.
- Distribusi kunci dan permintaan yang seragam: ElastiCache Untuk Valkey dan Redis OSS, distribusi kunci atau permintaan per slot yang tidak merata dapat menghasilkan slot panas yang dapat menghasilkan latensi tinggi. ElastiCache Tanpa server mendukung hingga 30.000 ECPUs/second (90.000 ECPUs/second saat menggunakan Read from Replica) pada satu slot, dalam beban kerja yang menjalankan perintah sederhana. SET/GET Sebaiknya evaluasi distribusi kunci dan permintaan Anda di seluruh slot dan memastikan distribusi yang seragam jika tingkat permintaan Anda melebihi batas ini.

Memecahkan masalah pembatasan di Tanpa Server ElastiCache

Dalam arsitektur berorientasi layanan dan sistem terdistribusi, membatasi kecepatan pemrosesan panggilan API oleh berbagai komponen layanan disebut throttling. Ini menghaluskan lonjakan, mengontrol ketidakcocokan dalam throughput komponen, dan memungkinkan pemulihan yang lebih dapat diprediksi ketika ada peristiwa operasional yang tidak terduga. ElastiCache Tanpa server dirancang untuk jenis arsitektur ini, dan sebagian besar klien Valkey atau Redis OSS memiliki percobaan ulang bawaan untuk permintaan yang dibatasi. Throttling pada tingkat tertentu belum tentu menjadi masalah bagi aplikasi Anda, tetapi throttling yang terus-menerus pada bagian alur kerja data Anda yang sensitif terhadap latensi dapat berdampak negatif terhadap pengalaman pengguna dan mengurangi efisiensi sistem secara keseluruhan.

Saat permintaan dibatasi di ElastiCache Tanpa Server, Anda akan melihat peningkatan metrik Tanpa Server. `ThrottledRequests` ElastiCache Jika Anda melihat sejumlah besar permintaan terbatas, pertimbangkan hal berikut:

- Kecepatan penskalaan: ElastiCache Tanpa server secara otomatis menskalakan saat Anda menelan lebih banyak data atau meningkatkan tingkat permintaan Anda. Jika aplikasi Anda menskalakan lebih cepat daripada kecepatan skala Tanpa Server, permintaan Anda mungkin terhambat saat ElastiCache Tanpa Server menskalakan untuk mengakomodasi beban kerja Anda. ElastiCache ElastiCache Tanpa server biasanya dapat meningkatkan ukuran penyimpanan dengan cepat, membutuhkan waktu hingga 10-12 menit untuk menggandakan ukuran penyimpanan di cache Anda.
- Distribusi kunci dan permintaan yang seragam: ElastiCache Untuk Valkey dan Redis OSS, distribusi kunci atau permintaan per slot yang tidak merata dapat menghasilkan slot panas. Slot panas dapat mengakibatkan pembatasan permintaan, jika tingkat permintaan ke satu slot melebihi 30.000 ECPUs/second dan berada dalam beban kerja yang mengeksekusi perintah sederhana. SET/GET Demikian pula, dengan ElastiCache untuk Memcached, tombol pintas dapat mengakibatkan pembatasan permintaan jika tingkat permintaan melebihi 30.000 /detik. ECPUs
- Baca dari Replika: Jika aplikasi Anda mengizinkannya, pertimbangkan untuk menggunakan fitur “Baca dari Replika”. Sebagian besar klien Valkey atau Redis OSS dapat dikonfigurasi ke” pembacaan skala “untuk mengarahkan pembacaan ke node replika. Fitur ini memungkinkan Anda untuk menskalakan lalu lintas baca. Selain itu ElastiCache Tanpa Server secara otomatis merutekan pembacaan dari permintaan replika ke node di Availability Zone yang sama dengan aplikasi Anda sehingga latensi lebih rendah. Ketika Read from Replica diaktifkan, Anda dapat mencapai hingga 90.000 ECPUs/second pada satu slot, untuk beban kerja dengan perintah sederhana. SET/GET

Masalah koneksi persisten

Item berikut harus diverifikasi saat memecahkan masalah konektivitas persisten dengan: ElastiCache

Topik

Grup keamanan

Grup Keamanan adalah firewall virtual yang melindungi ElastiCache klien Anda (EC2 misalnya, AWS Lambda fungsi, wadah Amazon ECS, dll.) dan ElastiCache cache. Grup keamanan bersifat stateful, artinya bahwa setelah lalu lintas masuk atau keluar diizinkan, maka tanggapan untuk lalu lintas itu akan secara otomatis mendapatkan otorisasi dalam konteks grup keamanan tertentu itu.

Fitur stateful mensyaratkan grup keamanan melacak semua koneksi yang diberikan otorisasi, dan terdapat batasan untuk koneksi dilacak. Jika batas itu tercapai, maka koneksi baru akan gagal. Silakan merujuk ke bagian pemecahan masalah untuk bantuan tentang cara mengidentifikasi apakah batas telah tercapai pada klien atau ElastiCache sisi.

Anda dapat memiliki satu grup keamanan yang ditetapkan secara bersamaan ke klien dan ElastiCache klaster, atau grup keamanan individual untuk masing-masing grup.

Untuk kedua kasus, Anda perlu mengizinkan lalu lintas keluar TCP di ElastiCache port dari sumber dan lalu lintas masuk pada port yang sama. ElastiCache Port default adalah 11211 untuk

Memcached, dan 6379 untuk Valkey atau Redis OSS. Secara default, grup keamanan mengizinkan semua lalu lintas ke luar. Dalam kasus ini, hanya aturan masuk di target grup keamanan yang diperlukan.

Untuk informasi selengkapnya, lihat Pola akses untuk mengakses ElastiCache klaster di VPC Amazon.

Jaringan ACLs

Network Access Control Lists (ACLs) adalah aturan stateless. Lalu lintas harus diizinkan di kedua arah (masuk dan keluar) agar koneksi berhasil. Jaringan ACLs ditugaskan ke subnet, bukan sumber daya khusus. Dimungkinkan untuk memiliki ACL yang sama yang ditugaskan ke ElastiCache dan sumber daya klien, terutama jika mereka berada di subnet yang sama.

Secara default, jaringan ACLs memungkinkan semua lintasan. Namun, dimungkinkan untuk menyesuaikan ACL agar menolak atau mengizinkan lalu lintas. Selain itu, evaluasi aturan ACL adalah berurutan, yang berarti bahwa aturan dengan nomor terendah yang cocok dengan lalu lintas yang akan mengizinkan atau menolak lalu lintas itu. Konfigurasi minimum untuk memungkinkan lalu lintas Valkey atau Redis OSS adalah:

ACL Jaringan sisi klien:

- Aturan Masuk:
- Nomor aturan: sebaiknya lebih rendah dari semua aturan menolak;
- Jenis: Aturan TCP Kustom;
- Protokol: TCP
- Rentang Port: 1024-65535
- Sumber: 0.0.0.0/0 (atau buat aturan individual untuk subnet cluster) ElastiCache
- Izinkan/Tolak: Izinkan

- Aturan keluar:
- Nomor aturan: sebaiknya lebih rendah dari semua aturan menolak;
- Jenis: Aturan TCP Kustom;
- Protokol: TCP
- Rentang Port: 6379

- Sumber: 0.0.0.0/0 (atau subnet cluster. ElastiCache Ingatlah bahwa menggunakan spesifik IPs dapat menimbulkan masalah jika terjadi failover atau penskalaan cluster)
- Izinkan/Tolak: Izinkan

ElastiCache Jaringan ACL:

- Aturan Masuk:
- Nomor aturan: sebaiknya lebih rendah dari semua aturan menolak;
- Jenis: Aturan TCP Kustom;
- Protokol: TCP
- Rentang Port: 6379
- Sumber: 0.0.0.0/0 (atau buat aturan individual untuk subnet cluster) ElastiCache
- Izinkan/Tolak: Izinkan

- Aturan keluar:
- Nomor aturan: sebaiknya lebih rendah dari semua aturan menolak;
- Jenis: Aturan TCP Kustom;
- Protokol: TCP
- Rentang Port: 1024-65535
- Sumber: 0.0.0.0/0 (atau subnet cluster. ElastiCache Ingatlah bahwa menggunakan spesifik IPs dapat menimbulkan masalah jika terjadi failover atau penskalaan cluster)
- Izinkan/Tolak: Izinkan

Untuk informasi selengkapnya, lihat Jaringan ACLs.

Tabel rute

Sama halnya dengan Network ACLs, setiap subnet dapat memiliki tabel rute yang berbeda. Jika klien dan ElastiCache cluster berada dalam subnet yang berbeda, pastikan bahwa tabel rute mereka memungkinkan mereka untuk mencapai satu sama lain.

Lingkungan yang lebih kompleks, yang melibatkan beberapa VPCs, perutean dinamis, atau firewall jaringan, mungkin menjadi sulit untuk dipecahkan. Lihat Validasi konektivitas jaringan untuk mengonfirmasi bahwa pengaturan jaringan Anda sesuai.

Resolusi DNS

ElastiCache menyediakan titik akhir layanan berdasarkan nama DNS. Titik akhir yang tersedia adalah `Configuration`, `Primary`, `Reader`, dan titik akhir `Node`. Untuk informasi selengkapnya, lihat Menemukan Titik Akhir Koneksi.

Dalam hal failover atau perubahan klaster, alamat yang terkait dengan nama titik akhir mungkin berubah dan akan diperbarui secara otomatis.

Pengaturan DNS khusus (yaitu, tidak menggunakan layanan DNS VPC) mungkin tidak mengetahui nama DNS ElastiCache yang disediakan. Pastikan sistem Anda berhasil menyelesaikan ElastiCache titik akhir menggunakan alat sistem seperti `dig` (seperti yang ditunjukkan berikut) atau`unslookup`.

```
$ dig +short example.xxxxxx.ng.0001.use1.cache.amazonaws.com
example-001.xxxxxx.0001.use1.cache.amazonaws.com.
1.2.3.4
```

Anda juga dapat memaksakan resolusi nama melalui layanan DNS VPC:

```
$ dig +short example.xxxxxx.ng.0001.use1.cache.amazonaws.com @169.254.169.253
example-001.tihewd.0001.use1.cache.amazonaws.com.
1.2.3.4
```

Mengidentifikasi masalah dengan diagnostik sisi server

CloudWatch metrik dan informasi run-time dari ElastiCache mesin adalah sumber umum atau informasi untuk mengidentifikasi potensi sumber masalah koneksi. Analisis yang baik biasanya dimulai dengan item berikut:

- Penggunaan CPU: Valkey dan Redis OSS adalah aplikasi multi-threaded. Namun, pelaksanaan dari setiap perintah terjadi dalam satu thread tunggal (utama). Untuk alasan ini, ElastiCache berikan metrik `CPUUtilization` dan`EngineCPUUtilization`. `EngineCPUUtilization`menyediakan pemanfaatan CPU yang didedikasikan untuk proses Valkey atau Redis OSS, dan penggunaan `CPUUtilization` di semua v. CPUs Node dengan lebih dari satu vCPU biasanya memiliki nilai yang berbeda untuk `CPUUtilization` dan`EngineCPUUtilization`, yang kedua biasanya lebih tinggi. `EngineCPUUtilization` yang tinggi dapat disebabkan oleh peningkatan jumlah permintaan atau operasi kompleks yang membutuhkan waktu CPU yang besar untuk diselesaikan. Anda dapat mengidentifikasi keduanya dengan berikut ini:

- Peningkatan jumlah permintaan: Periksa peningkatan pada metrik lain yang cocok dengan pola `EngineCPUUtilization`. Metrik yang berguna adalah:
  - `CacheHits` dan `CacheMisses`: jumlah permintaan berhasil atau permintaan yang tidak menemukan item yang valid dalam cache. Jika rasio dari yang meleset lebih tinggi dibandingkan yang berhasil, maka aplikasi memboroskan waktu dan sumber daya dengan permintaan yang tidak membuahkan hasil.
  - `SetTypeCmds` dan `GetTypeCmds`: Kedua metrik ini yang berhubungan dengan `EngineCPUUtilization` dapat membantu untuk memahami jika beban secara signifikan lebih tinggi untuk permintaan tulis, diukur oleh `SetTypeCmds`, atau lebih tinggi untuk permintaan baca, diukur oleh `GetTypeCmds`. Jika yang lebih banyak adalah beban baca, maka menggunakan beberapa replika baca dapat menyeimbangkan permintaan di beberapa simpul sehingga simpul primer dapat melayani permintaan tulis saja. Dalam cluster yang dinonaktifkan mode cluster, penggunaan replika baca dapat dilakukan dengan membuat konfigurasi koneksi tambahan dalam aplikasi menggunakan endpoint pembaca. ElastiCache Untuk informasi selengkapnya, lihat Menemukan Titik Akhir Koneksi. Operasi baca harus dikirimkan ke koneksi tambahan ini. Operasi tulis akan dilakukan melalui titik akhir primer biasa. Pada mode klaster diaktifkan, sebaiknya menggunakan pustaka yang mendukung replika baca secara native. Dengan flag yang tepat, perpustakaan akan dapat secara otomatis menemukan topologi cluster, node replika, mengaktifkan operasi baca melalui perintah READONLY Valkey atau Redis OSS, dan mengirimkan permintaan baca ke replika.
- Jumlah koneksi yang meningkat:
  - `CurrConnections` dan `NewConnections`: `CurrConnection` adalah jumlah koneksi yang dibuat pada saat pengumpulan titik data, sementara `NewConnections` menunjukkan jumlah koneksi yang dibuat pada periode itu.

    Membuat dan menangani koneksi menyebabkan overhead CPU yang cukup besar. Selain itu, proses handshake tiga arah TCP yang diperlukan untuk membuat koneksi baru akan berdampak secara negatif pada waktu respons secara keseluruhan.

    Sebuah ElastiCache node dengan ribuan `NewConnections` per menit menunjukkan bahwa koneksi dibuat dan digunakan hanya dengan beberapa perintah, yang tidak optimal. Menjaga koneksi selalu tersedia dan menggunakan kembali koneksi itu untuk operasi baru adalah praktik terbaik. Hal ini dimungkinkan jika aplikasi klien mendukung dan menerapkan dengan baik pooling koneksi atau koneksi yang persisten. Dengan pooling koneksi, angka `currConnections` tidak akan memiliki variasi besar, dan `NewConnections` seharusnya menjadi serendah mungkin. Valkey dan Redis OSS memberikan kinerja optimal dengan

sejumlah kecil CurrConnections. Menjaga currConnection dalam kelompok puluhan atau ratusan meminimalkan penggunaan sumber daya untuk mendukung koneksi tersendiri seperti buffer klien dan siklus CPU untuk melayani koneksi.

- Throughput jaringan:
  - Tentukan bandwidth: ElastiCache node memiliki bandwidth jaringan yang sebanding dengan ukuran node. Karena aplikasi memiliki karakteristik yang berbeda, hasilnya dapat bervariasi sesuai dengan beban kerja. Sebagai contoh, aplikasi dengan permintaan kecil dengan laju yang tinggi cenderung menyebabkan pemanfaatan CPU lebih besar daripada throughput jaringan sementara kunci yang lebih besar akan menyebabkan pemanfaatan jaringan yang lebih tinggi. Untuk alasan itu, sebaiknya menguji simpul dengan beban kerja yang sebenarnya untuk pemahaman yang lebih baik tentang batasan.

    Mensimulasi beban dari aplikasi akan memberikan hasil yang lebih akurat. Namun, alat tolok ukur dapat memberikan gambaran yang baik tentang batasan.
  - Untuk kasus di mana permintaan didominasi oleh operasi baca, menggunakan replika untuk operasi baca akan mengurangi beban pada simpul primer. Jika kasus penggunaan didominasi operasi tulis, digunakannya banyak replika akan memperkuat penggunaan jaringan. Untuk setiap byte yang ditulis ke simpul primer, N byte akan dikirim ke replika, di mana N adalah jumlah replika. Praktik terbaik untuk menulis beban kerja intensif digunakan ElastiCache untuk Redis OSS dengan mode cluster yang diaktifkan sehingga penulisan dapat diseimbangkan di beberapa pecahan, atau ditingkatkan ke tipe node dengan lebih banyak kemampuan jaringan.
  - CloudWatchmetrics `NetworkBytesIn`Dan `NetworkBytesOut` memberikan jumlah data yang masuk atau keluar dari node, masing-masing. `ReplicationBytes`adalah lalu lintas yang didedikasikan untuk replikasi data.

  Untuk informasi selengkapnya, lihat Batas terkait jaringan.
- Perintah kompleks: Perintah Redis OSS disajikan pada satu utas, yang berarti bahwa permintaan disajikan secara berurutan. Perintah tunggal yang lambat dapat memengaruhi permintaan lain dan koneksi, yang berpuncak pada terjadinya waktu habis. Penggunaan perintah yang bertindak pada beberapa nilai, kunci, atau jenis data harus dilakukan dengan hati-hati. Koneksi dapat diblokir atau dihentikan tergantung pada jumlah parameter, atau ukuran nilai input atau output-nya.

  Contoh yang terkenal buruk adalah perintah `KEYS`. Perintah ini menyapu seluruh ruang kunci untuk mencari pola tertentu dan memblokir pelaksanaan dari perintah lain selama

pelaksanaannya. Redis OSS menggunakan notasi “Big O” untuk menggambarkan kompleksitas perintahnya.

Perintah keys memiliki O(N) kali kompleksitas, N menjadi jumlah kunci dalam basis data. Oleh karena itu, semakin besar jumlah kunci, semakin lambat perintahnya. KEYS dapat menyebabkan masalah dengan cara yang berbeda: Jika tidak menggunakan pola pencarian, perintah ini akan menghasilkan semua nama kunci yang tersedia. Dalam basis data dengan ribuan atau jutaan item, output yang besar akan tercipta dan membanjiri buffer jaringan.

Jika pola pencarian digunakan, hanya kunci yang cocok dengan pola yang akan dikembalikan ke klien. Namun, mesin masih akan menyapu ke seluruh ruang kunci untuk mencarinya, dan waktu untuk menyelesaikan perintah akan sama.

Alternatif untuk KEYS adalah perintah SCAN. Perintah ini melakukan iterasi atas ruang kunci dan membatasi iterasi dalam jumlah tertentu dari item, menghindari pemblokiran yang berkepanjangan pada mesin.

Scan memiliki parameter COUNT, digunakan untuk mengatur ukuran dari blok iterasi. Nilai default-nya adalah 10 (10 item per iterasi).

Tergantung pada jumlah item dalam basis data, blok dengan nilai COUNT yang kecil akan membutuhkan lebih banyak iterasi untuk menyelesaikan perintah scan penuh, dan nilai yang lebih besar ini akan membuat mesin sibuk lebih lama di setiap iterasi. Sementara nilai count yang kecil akan membuat SCAN lebih lambat pada basis data yang besar, nilai yang lebih besar dapat menyebabkan masalah yang sama seperti disebutkan untuk KEYS.

Sebagai contoh, menjalankan perintah SCAN dengan nilai count sebesar 10 akan membutuhkan 100.000 pengulangan pada basis data dengan 1 juta kunci. Jika waktu pulang pergi jaringan rata-rata adalah 0,5 milidetik, sekitar 50.000 milidetik (50 detik) akan dihabiskan untuk mengirimkan permintaan ini.

Di sisi lain, jika nilai count adalah 100,0000, iterasi tunggal akan diperlukan dan hanya 0,5 milidetik yang akan dihabiskan untuk mengirimnya. Namun, mesin akan sepenuhnya terblokir untuk operasi lain sampai perintah itu selesai menyapu semua ruang kunci.

Selain KEYS, beberapa perintah lain berpotensi membahayakan jika tidak digunakan dengan benar. Untuk melihat daftar semua perintah dan kompleksitas waktu masing-masing, buka perintah Valkey dan Redis OSS.

Contoh potensi masalah:

- Skrip Lua: Valkey dan Redis OSS menyediakan penerjemah Lua tertanam, memungkinkan eksekusi skrip di sisi server. Skrip Lua pada Valkey dan Redis OSS dieksekusi pada tingkat mesin dan bersifat atomik menurut definisi, yang berarti bahwa tidak ada perintah atau skrip lain yang diizinkan untuk dijalankan saat skrip sedang dieksekusi. Skrip Lua memberikan kemungkinan menjalankan beberapa perintah, algoritme pengambilan keputusan, penguraian data, dan lainnya langsung di mesin. Meskipun sifat atom dari skrip dan kemungkinan melepaskan aplikasi cukup menarik, skrip harus digunakan dengan hati-hati dan untuk operasi yang kecil. ElastiCacheAktif, waktu eksekusi skrip Lua dibatasi hingga 5 detik. Skrip yang belum ditulis ke ruang kunci akan secara otomatis dihentikan setelah periode 5 detik. Untuk menghindari kerusakan dan inkonsistensi data, simpul akan melakukan failover jika pelaksanaan skrip belum selesai dalam 5 detik dan tetap menjalankan operasi tulis apa pun selama pelaksanaan skrip. Transaksi adalah alternatif untuk menjamin konsistensi beberapa modifikasi kunci terkait di Redis OSS. Perintah transaction memungkinkan eksekusi dari suatu blok perintah, memperhatikan perubahan pada kunci yang sudah ada. Jika salah satu kunci yang diperhatikan berubah sebelum selesainya transaksi, maka semua perubahan akan dibuang.
- Penghapusan item secara massal: perintah `DEL` menerima beberapa parameter, yang merupakan nama kunci yang akan dihapus. Operasi penghapusan bersifat sinkron dan akan mengambil waktu CPU yang signifikan jika daftar parameter besar, atau berisi daftar yang panjang, set, sorted set, atau hash yang besar (struktur data memegang beberapa sub-item). Dengan kata lain, bahkan penghapusan satu kunci pun dapat memakan waktu lama jika kunci itu memiliki banyak elemen. Alternatifnya `DEL` adalah`UNLINK`, yang merupakan perintah asinkron yang tersedia sejak Redis OSS 4. `UNLINK`harus lebih disukai daripada `DEL` bila memungkinkan. Mulai ElastiCache untuk Redis OSS 6x, `lazyfree-lazy-user-del` parameter membuat `DEL` perintah berperilaku seperti `UNLINK` saat diaktifkan. Untuk informasi selengkapnya, lihat Redis OSS 6.0 Perubahan Parameter.
- Perintah yang bekerja pada beberapa kunci: `DEL` disebutkan sebelumnya sebagai perintah yang menerima beberapa argumen dan waktu pelaksanaannya akan berbanding lurus dengan itu. Namun, Redis OSS menyediakan lebih banyak perintah yang bekerja sama. Sebagai contoh, `MSET` dan `MGET` memungkinkan penyisipan atau pengambilan beberapa kunci String sekaligus. Penggunaannya mungkin bermanfaat untuk mengurangi latensi jaringan yang terjadi pada beberapa perintah tersendiri `SET` atau `GET`. Namun, daftar parameter yang panjang akan memengaruhi penggunaan CPU.

Sementara pemanfaatan CPU sendiri bukan penyebab untuk masalah konektivitas, menghabiskan terlalu banyak waktu untuk memproses satu atau beberapa perintah atas beberapa kunci dapat menyebabkan kegagalan permintaan lain dan meningkatkan pemanfaatan CPU secara keseluruhan.

Jumlah kunci dan ukurannya akan memengaruhi kompleksitas perintah dan karena itu berpengaruh pada waktu penyelesaian.

Contoh lain dari perintah yang dapat bertindak atas beberapa kunci: `HMGET`, `HMSET`, `MSETNX`, `PFCOUNT`, `PFMERGE`, `SDIFF`, `SDIFFSTORE`, `SINTER`, `SINTERSTORE`, `SUNION`, `SUNIONSTORE`, `TOUCH`, `ZDIFF`, `ZDIFFSTORE`, `ZINTER` atau `ZINTERSTORE`.

- Perintah yang bekerja pada beberapa tipe data: Redis OSS juga menyediakan perintah yang bertindak atas satu atau beberapa kunci, terlepas dari tipe datanya. ElastiCache untuk Redis OSS menyediakan metrik `KeyBasedCmds` untuk memantau perintah tersebut. Metrik ini menjumlahkan eksekusi dari perintah berikut pada periode yang dipilih:
  - Kompleksitas O(N):
    - `KEYS`
  - O(1)
    - `EXISTS`
    - `OBJECT`
    - `PTTL`
    - `RANDOMKEY`
    - `TTL`
    - `TYPE`
    - `EXPIRE`
    - `EXPIREAT`
    - `MOVE`
    - `PERSIST`
    - `PEXPIRE`
    - `PEXPIREAT`
    - `UNLINK (O(N)` untuk mengeklaim kembali memori. Namun tugas mengeklaim kembali memori itu terjadi di thread yang terpisah dan tidak memblokir mesin
  - Waktu kompleksitas yang berbeda akan tergantung pada jenis data:

- DEL
- DUMP
- RENAME dianggap perintah dengan kompleksitas O(1), tetapi menjalankan DEL secara internal. Waktu pelaksanaan akan bervariasi tergantung pada ukuran kunci yang diganti namanya.
- RENAMENX
- RESTORE
- SORT

- Hash besar: Hash adalah jenis data yang memungkinkan satu kunci dengan beberapa sub-item nilai kunci. Setiap hash dapat menyimpan 4.294.967.295 item dan operasi pada hash yang besar dapat menghabiskan banyak daya komputasi. Sama dengan KEYS, hash mempunyai perintah HKEYS dengan kompleksitas waktu O(N), N adalah jumlah item dalam hash. HSCAN seharusnya lebih dipilih dibandingkan HKEYS untuk menghindari perintah yang berjalan lama. HDEL, HGETALL, HMGET, HMSET dan HVALS adalah perintah yang harus digunakan dengan hati-hati pada hash besar.

- Struktur big data lainnya: Selain hash, struktur data lainnya dapat memakan waktu CPU. Set, List, Sorted Set, dan Hyperloglog juga dapat memakan waktu yang lama untuk dimanipulasi tergantung pada ukuran dan perintah yang digunakan. Untuk informasi lebih lanjut tentang perintah tersebut, lihat perintah Valkey dan Redis OSS.

Validasi konektivitas jaringan

Setelah meninjau konfigurasi jaringan yang terkait dengan resolusi DNS, grup keamanan, jaringan ACLs, dan tabel rute, konektivitas dapat divalidasi dengan VPC Reachability Analyzer dan alat sistem.

Reachability Analyzer akan menguji konektivitas jaringan dan mengonfirmasi jika semua persyaratan dan izin terpenuhi. Untuk tes di bawah ini Anda akan memerlukan ID ENI (Elastic Network Interface Identification) dari salah satu ElastiCache node yang tersedia di VPC Anda. Anda dapat menemukannya dengan melakukan hal berikut:

1. Pergi ke https://console.aws.amazon.com/ec2/v2/home? #NIC:
2. Filter daftar antarmuka dengan nama ElastiCache cluster Anda atau alamat IP yang didapat dari validasi DNS sebelumnya.

3. Tuliskan atau simpan ENI ID. Jika beberapa antarmuka ditampilkan, tinjau deskripsi untuk mengonfirmasi bahwa mereka termasuk dalam ElastiCache cluster yang tepat dan pilih salah satunya.
4. Lanjutkan ke langkah berikutnya.
5. Buat jalur analisis di https://console.aws.amazon.com/vpc/rumah? # ReachabilityAnalyzer dan pilih opsi berikut:

   - Jenis Sumber: Pilih instance jika ElastiCache klien Anda berjalan pada EC2 instans Amazon atau Antarmuka Jaringan jika menggunakan layanan lain, seperti AWS Fargate Amazon ECS dengan jaringan awsvpc, dll) AWS Lambda, dan ID sumber daya masing-masing (EC2 instance atau ID ENI);
   - Jenis Tujuan: Pilih Antarmuka Jaringan dan pilih ElastiCache ENI dari daftar.
   - Port tujuan: tentukan 6379 untuk Redis OSS atau 11211 ElastiCache untuk untuk Memcached. ElastiCache Itu adalah port yang ditetapkan dengan konfigurasi default dan contoh ini mengasumsikan bahwa port tersebut tidak berubah.
   - Protokol: TCP

Buat jalur analisis dan tunggu beberapa saat untuk hasilnya. Jika status tidak terjangkau, buka detail analisis dan tinjau Penjelajah analisis untuk detail di mana permintaan diblokir.

Jika tes penjangkauan sudah lulus, lanjutkan ke verifikasi di tingkat OS.

Untuk memvalidasi konektivitas TCP pada port ElastiCache layanan: Di Amazon Linux, `Nping` tersedia dalam paket `nmap` dan dapat menguji konektivitas TCP pada ElastiCache port, serta menyediakan waktu pulang-pergi jaringan untuk membuat koneksi. Gunakan ini untuk memvalidasi konektivitas jaringan dan latensi saat ini ke ElastiCache cluster, seperti yang ditunjukkan berikut:

```
$ sudo nping --tcp -p 6379 example.xxxxxx.ng.0001.use1.cache.amazonaws.com

Starting Nping 0.6.40 ( http://nmap.org/nping ) at 2020-12-30 16:48 UTC
SENT (0.0495s) TCP ...
(Output suppressed )

Max rtt: 0.937ms | Min rtt: 0.318ms | Avg rtt: 0.449ms
Raw packets sent: 5 (200B) | Rcvd: 5 (220B) | Lost: 0 (0.00%)
Nping done: 1 IP address pinged in 4.08 seconds
```

Secara default, `nping` mengirimkan 5 paket penyelidikan dengan waktu tunda 1 detik di antara paket itu. Anda dapat menggunakan opsi "-c" untuk menambah jumlah paket penyelidikan dan "--delay " untuk mengubah waktu untuk mengirim pengujian baru.

Jika pengujian dengan `nping` gagal dan pengujian VPC Reachability Analyzer lulus, mintalah administrator sistem Anda untuk meninjau kemungkinan aturan firewall berbasis Host, aturan perutean asimetris, atau batasan lain yang dimungkinkan di tingkat sistem operasi.

Di ElastiCache konsol, periksa apakah Enkripsi dalam transit diaktifkan di detail ElastiCache klaster Anda. Jika enkripsi bergerak diaktifkan, lakukan konfirmasi jika sesi TLS dapat dibuat dengan perintah berikut:

```
openssl s_client -connect example.xxxxxx.use1.cache.amazonaws.com:6379
```

Output yang panjang akan keluar jika koneksi dan negosiasi TLS berhasil. Periksa kode yang dihasilkan yang terdapat di baris terakhir, nilainya harus `0 (ok)`. Jika openssl mengembalikan sesuatu yang berbeda, periksa alasan kesalahan di https://www.openssl.org/docs/man1.0.2/man1/verify.html #DIAGNOSTICS.

Jika semua pengujian infrastruktur dan sistem operasi lulus tetapi aplikasi Anda masih tidak dapat terhubung ElastiCache, periksa apakah konfigurasi aplikasi sesuai dengan pengaturan. ElastiCache Kesalahan yang umum adalah:

- Aplikasi Anda tidak mendukung mode ElastiCache klaster, dan ElastiCache mengaktifkan mode cluster;
- Aplikasi Anda tidak mendukung TLS/SSL, dan ElastiCache memiliki enkripsi dalam transit yang diaktifkan;
- Aplikasi mendukung TLS/SSL tetapi tidak memiliki bendera konfigurasi yang tepat atau otoritas sertifikasi tepercaya;

Batas terkait jaringan

- Jumlah maksimum koneksi: Ada batas keras untuk koneksi secara serentak. Setiap ElastiCache node memungkinkan hingga 65.000 koneksi simultan di semua klien. Batas ini dapat dipantau melalui metrik `CurrConnections` di CloudWatch. Namun, klien juga memiliki batas untuk koneksi keluar. Pada Linux, periksa rentang port ephemeral yang diizinkan dengan perintah:

  ```
  # sysctl net.ipv4.ip_local_port_range
  ```

```
net.ipv4.ip_local_port_range = 32768 60999
```

Pada contoh sebelumnya, 28231 koneksi akan diizinkan dari sumber yang sama, ke IP tujuan (ElastiCache node) dan port yang sama. Perintah berikut menunjukkan berapa banyak koneksi yang ada untuk ElastiCache node tertentu (IP 1.2.3.4):

```
ss --numeric --tcp state connected "dst 1.2.3.4 and dport == 6379" | grep -vE
 '^State' | wc -l
```

Jika jumlahnya terlalu tinggi, sistem Anda mungkin menjadi kelebihan beban mencoba memproses permintaan koneksi. Sebaiknya mempertimbangkan menerapkan teknik seperti pooling koneksi atau koneksi persisten untuk menangani koneksi itu dengan lebih baik. Jika memungkinkan, lakukan konfigurasi pool koneksi untuk membatasi jumlah maksimum koneksi hanya beberapa ratus. Selain itu, logika back-off untuk menangani masalah waktu habis atau pengecualian koneksi lain juga dianjurkan untuk menghindari masalah koneksi churn.

- Batas lalu lintas jaringan: Periksa CloudWatch metrik berikut untuk Redis OSS untuk mengidentifikasi kemungkinan batas jaringan yang terkena pada node: ElastiCache
  - `NetworkBandwidthInAllowanceExceeded` / `NetworkBandwidthOutAllowanceExceeded`: Paket jaringan yang ditunda karena throughput melebihi batas agregasi bandwidth.

    Penting untuk dicatat bahwa setiap byte yang ditulis ke simpul primer akan direplikasi ke N replika, di mana N adalah jumlah replika. Klaster dengan jenis simpul kecil, beberapa replika, dan permintaan sarat operasi tulis mungkin tidak mampu mengatasi backlog replikasi. Untuk kasus seperti itu, praktik terbaiknya adalah menaikkan skala (mengubah jenis simpul), menskalakan ke luar (menambahkan serpihan dalam klaster dengan mode klaster diaktifkan), mengurangi jumlah replika, atau meminimalkan jumlah operasi tulis.
  - `NetworkConntrackAllowanceExceeded`: Paket yang ditunda karena telah terlampauinya jumlah maksimum koneksi yang dilacak di seluruh grup keamanan yang ditetapkan ke simpul. Koneksi baru kemungkinan akan gagal selama periode ini.
  - `NetworkPackets PerSecondAllowanceExceeded`: Jumlah maksimum paket per detik terlampaui. Beban kerja berdasarkan laju yang tinggi dari permintaan yang sangat kecil dapat mencapai batas ini sebelum bandwidth mencapai maksimum.

  Metrik di atas adalah cara ideal untuk mengonfirmasi simpul yang mencapai batas jaringannya. Namun, batas juga dapat diidentifikasi dengan bentuk plato pada metrik jaringan.

Jika dataran tinggi teramati untuk waktu yang lama, kemungkinan akan diikuti oleh lag replikasi, peningkatan byte yang Digunakan untuk cache, penurunan memori yang dapat dikosongkan, swap tinggi, dan penggunaan CPU. EC2 Instans Amazon juga memiliki batas jaringan yang dapat dilacak melalui metrik driver ENA. Instans Linux dengan dukungan jaringan yang ditingkatkan dan penggerak ENA 2.2.10 atau yang lebih baru dapat meninjau penghitung batas dengan perintah:

```
# ethtool -S eth0 | grep "allowance_exceeded"
```

Penggunaan CPU

Metrik penggunaan CPU adalah titik awal penyelidikan, dan item berikut dapat membantu mempersempit kemungkinan masalah di ElastiCache samping:

- Redis OSS SlowLogs: Konfigurasi ElastiCache default mempertahankan 128 perintah terakhir yang membutuhkan waktu lebih dari 10 milidetik untuk diselesaikan. Riwayat perintah lambat disimpan selama runtime mesin dan akan hilang jika terjadi kegagalan atau mulai ulang. Jika daftar mencapai 128 entri, maka peristiwa lama akan dihapus untuk memberikan tempat untuk peristiwa baru. Ukuran dari daftar peristiwa lambat dan waktu pelaksanaan yang dianggap lambat dapat diubah melalui parameter `slowlog-max-len` dan `slowlog-log-slower-than` dalam grup parameter kustom. Daftar slowlogs dapat diambil dengan menjalankan `SLOWLOG GET 128` pada mesin, 128 adalah 128 perintah lambat terakhir yang dilaporkan. Setiap entri memiliki bidang berikut:

```
1) 1) (integer) 1 -----------> Sequential ID
   2) (integer) 1609010767 --> Timestamp (Unix epoch time)of the Event
   3) (integer) 4823378 -----> Time in microseconds to complete the command.
   4) 1) "keys" -------------> Command
      2) "*" ----------------> Arguments
   5) "1.2.3.4:57004"-> Source
```

Peristiwa di atas terjadi pada tanggal 26 Desember pukul 19:26:07 UTC, membutuhkan 4,8 detik (4,823ms) untuk diselesaikan dan disebabkan oleh perintah `KEYS` yang diminta dari klien 1.2.3.4.

Di Linux, stempel waktu dapat dikonversi dengan perintah tanggal:

```
$ date --date='@1609010767'
Sat Dec 26 19:26:07 UTC 2020
```

Pada Python:

```
>>> from datetime import datetime
>>> datetime.fromtimestamp(1609010767)
datetime.datetime(2020, 12, 26, 19, 26, 7)
```

Atau di Windows dengan PowerShell:

```
PS D:\Users\user> [datetimeoffset]::FromUnixTimeSeconds('1609010767')
DateTime      : 12/26/2020 7:26:07 PM
UtcDateTime
                : 12/26/2020 7:26:07 PM
LocalDateTime : 12/26/2020 2:26:07 PM
Date          : 12/26/2020 12:00:00 AM
Day           : 26
DayOfWeek
                : Saturday
DayOfYear     : 361
Hour          : 19
Millisecond   : 0
Minute        : 26
Month
                : 12
Offset        : 00:00:00Ticks         : 637446075670000000
UtcTicks
                : 637446075670000000
TimeOfDay     : 19:26:07
Year          : 2020
```

Banyak perintah lambat dalam waktu singkat (menit yang sama atau kurang) menjadi hal yang dikhawatirkan. Tinjau sifat dari perintah dan bagaimana perintah itu dapat dioptimalkan (lihat contoh sebelumnya). Jika perintah dengan kompleksitas waktu O(1) sering dilaporkan, periksa faktor lain yang menyebabkan penggunaan CPU tinggi yang disebutkan sebelumnya.

- Metrik latensi: ElastiCache untuk Redis OSS menyediakan CloudWatch metrik untuk memantau latensi rata-rata untuk berbagai kelas perintah. Titik data dihitung dengan membagi jumlah pelaksanaan perintah dalam kategori dengan total waktu pelaksanaan pada periode tersebut. Penting untuk dipahami bahwa hasil metrik latensi merupakan kumpulan dari beberapa perintah. Satu perintah dapat menyebabkan hasil tidak terduga, seperti waktu habis, tanpa dampak signifikan pada metrik. Untuk kasus seperti ini, peristiwa slowlog akan menjadi sumber informasi

yang lebih akurat. Daftar berikut berisi metrik latensi yang tersedia dan perintah terkait yang memengaruhinya.

- EvalBasedCmdsLatency: terkait dengan perintah Lua Script,`eval`,`evalsha`;
- GeoSpatialBasedCmdsLatency: `geodist`, `geohash`, `geopos`, `georadius`, `georadiusbymember`, `geoadd`;
- GetTypeCmdsLatency: Baca perintah, terlepas dari tipe data;
- HashBasedCmdsLatency: `hexists`, `hget`, `hgetall`, `hkeys`, `hlen`, `hmget`, `hvals`, `hstrlen`, `hdel`, `hincrby`, `hincrbyfloat`, `hmset`, `hset`, `hsetnx`;
- HyperLogLogBasedCmdsLatency: `pfselftest`, `pfcount`, `pfdebug`, `pfadd`, `pfmerge`;
- KeyBasedCmdsLatency: Perintah yang dapat bertindak atas tipe data yang berbeda: dump `existskeys`,`object`,`pttl`,,`randomkey`,`ttl`,`type`,`del`,`expire`,`expireat`,`move`,`persist`,`pexpire`
- ListBasedCmdsLatency: lindex, llen, lrange, blpop, brpop, brpoplpush, linsert, lpop, lpush, lpushx, lrem, lset, ltrim, rpop, rpoplpush, rpush, rpushx;
- PubSubBasedCmdsLatency: psubscribe, publish, pubsub, punsubscribe, subscribe, unsubscribe;
- SetBasedCmdsLatency: `scard`, `sdiff`, `sinter`, `sismember`, `smembers`, `srandmember`, `sunion`, `sadd`, `sdiffstore`, `sinterstore`, `smove`, `spop`, `srem`, `sunionstore`;
- SetTypeCmdsLatency: Menulis perintah, terlepas dari tipe data;
- SortedSetBasedCmdsLatency: `zcard`, `zcount`, `zrange`, `zrangebyscore`, `zrank`, `zrevrange`, `zrevrangebyscore`, `zrevrank`, `zscore`, `zrangebylex`, `zrevrangebylex`, `zlexcount`, `zadd`. `zincrby`, `zinterstore`, `zrem`, `zremrangebyrank`, `zremrangebyscore`, `zunionstore`, `zremrangebylex`, `zpopmax`, `zpopmin`, `bzpopmin`, `bzpopmax`;
- StringBasedCmdsLatency: `bitcount`, `get`, `getbit`, `getrange`, `mget`, `strlen`, `substr`, `bitpos`, `append`, `bitop`, `bitfield`, `decr`, `decrby`, `getset`, `incr`, `incrby`, `incrbyfloat`, `mset`, `msetnx`, `psetex`, `set`, `setbit`, `setex`, `setnx`, `setrange`;
- StreamBasedCmdsLatency: `xrange`, `xrevrange`, `xlen`, `xread`, `xpending`, `xinfo`, `xadd`, `xgroup`, `readgroup`, `xack`, `xclaim`, `xdel`, `xtrim`, `xsetid`;

- Perintah runtime Redis OSS:
  - info commandstats: Menyediakan daftar perintah yang dijalankan sejak mesin dihidupkan, jumlah eksekusi kumulatifnya, total waktu eksekusi, dan waktu eksekusi rata-rata per perintah;
  - client list: Menyediakan daftar klien yang saat ini terhubung dan informasi yang relevan seperti penggunaan buffer, perintah yang dilaksanakan terakhir, dll;

- Backup dan replikasi: ElastiCache untuk versi Redis OSS lebih awal dari 2.8.22 gunakan proses bercabang untuk membuat cadangan dan memproses sinkronisasi penuh dengan replika. Metode ini mungkin menyebabkan overhead memori yang besar untuk kasus penggunaan sarat operasi tulis.

  Dimulai dengan ElastiCache Redis OSS 2.8.22, AWS memperkenalkan metode pencadangan dan replikasi tanpa garpu. Metode yang baru dapat menunda operasi tulis untuk mencegah kegagalan. Kedua metode dapat menyebabkan periode pemanfaatan CPU yang lebih tinggi, yang menyebabkan waktu respons lebih tinggi dan akibatnya menyebabkan klien mengalami waktu habis selama melaksanakan perintah. Selalu periksa apakah kegagalan klien terjadi selama periode pencadangan atau metrik `SaveInProgress` bernilai 1 pada periode tersebut. Sebaiknya menjadwalkan periode pencadangan untuk periode pemanfaatan yang rendah untuk meminimalkan kemungkinan masalah dengan klien atau kegagalan proses pencadangan.

Koneksi yang dihentikan dari sisi server

Default ElastiCache untuk konfigurasi Redis OSS membuat koneksi klien dibuat tanpa batas waktu. Namun, dalam beberapa kasus penghentian koneksi mungkin diinginkan. Misalnya:

- Bug dalam aplikasi klien dapat menyebabkan koneksi dilupakan dan tetap terhubung dengan keadaan idle. Ini disebut “kebocoran koneksi “ dan konsekuensinya adalah peningkatan yang tetap pada jumlah koneksi terhubung yang diamati pada metrik `CurrConnections`. Perilaku ini dapat mengakibatkan kejenuhan pada klien atau ElastiCache sisi. Ketika perbaikan langsung tidak dimungkinkan dari sisi klien, beberapa administrator menetapkan nilai” batas waktu “dalam grup ElastiCache parameter mereka. Waktu habis adalah waktu dalam detik yang diizinkan bagi koneksi idle untuk bertahan. Jika klien tidak mengirimkan permintaan apa pun dalam periode tersebut, mesin akan menghentikan koneksi segera setelah koneksi mencapai nilai batas waktu. Nilai waktu habis yang kecil dapat mengakibatkan pemutusan koneksi yang tidak perlu dan klien akan perlu menangani ini dengan tepat dan menghubungkan kembali, yang menyebabkan penundaan.
- Memori yang digunakan untuk menyimpan kunci dibagikan dengan buffer klien. Klien lambat dengan permintaan atau tanggapan besar mungkin menuntut sejumlah besar memori untuk menangani buffer. Default ElastiCache untuk konfigurasi Redis OSS tidak membatasi ukuran buffer keluaran klien biasa. Jika batas `maxmemory` tercapai, mesin akan mencoba mengosongkan item untuk memenuhi penggunaan buffer. Dalam kondisi memori yang sangat rendah, ElastiCache untuk Redis OSS mungkin memilih untuk memutuskan klien yang mengkonsumsi buffer keluaran klien besar untuk membebaskan memori dan mempertahankan kesehatan cluster.

Dimungkinkan untuk membatasi ukuran buffer klien dengan konfigurasi kustom dan klien yang mencapai batas ini akan terputus. Namun, klien harus dapat menangani pemutusan yang tidak terduga. Parameter untuk menangani ukuran buffer untuk klien biasa adalah sebagai berikut:

- client-query-buffer-limit: Ukuran maksimum permintaan input tunggal;
- client-output-buffer-limit-normal-soft-limit: Batas lunak untuk koneksi klien. Koneksi akan dihentikan jika tetap di atas batas lunak selama lebih dari waktu dalam detik yang ditentukan pada client-output-buffer-limit - normal-soft-seconds atau jika mencapai batas keras;
- client-output-buffer-limit-normal-soft-seconds: Waktu yang diizinkan untuk koneksi melebihi client-output-buffer-limit -normal-soft-limit;
- client-output-buffer-limit-normal-hard-limit: Koneksi yang mencapai batas ini akan segera dihentikan.

Selain buffer klien reguler, opsi berikut mengontrol buffer untuk node replika dan klien Pub/Sub (Publish/Subscribe):

- client-output-buffer-limit-replica-hard-limit;
- client-output-buffer-limit-replica-soft-seconds;
- client-output-buffer-limit-replica-hard-limit;
- client-output-buffer-limit-pubsub-soft-limit;
- client-output-buffer-limit-pubsub-soft-seconds;
- client-output-buffer-limit-pubsub-hard-limit;

Pemecahan masalah sisi klien untuk instans Amazon EC2

Beban dan daya tanggap di sisi klien juga dapat memengaruhi permintaan. ElastiCache EC2 instance dan batas sistem operasi perlu ditinjau dengan cermat saat memecahkan masalah konektivitas intermiten atau batas waktu. Beberapa poin penting untuk diamati:

- CPU:
  - EC2 penggunaan CPU misalnya: Pastikan CPU belum jenuh atau mendekati 100 persen. Analisis historis dapat dilakukan melalui CloudWatch, namun perlu diingat bahwa perincian titik data adalah 1 menit (dengan pemantauan terperinci diaktifkan) atau 5 menit;
  - Jika menggunakan EC2 instans burstable, pastikan saldo kredit CPU mereka belum habis. Informasi ini tersedia pada `CPUCreditBalance` CloudWatch metrik.

- Periode pendek penggunaan CPU yang tinggi dapat menyebabkan waktu habis tanpa mencerminkan 100 persen pemanfaatan pada CloudWatch. Kasus seperti itu memerlukan pemantauan waktu nyata dengan peralatan sistem operasi seperti `top`, `ps`, dan `mpstat`.

- Jaringan
  - Periksa apakah throughput Jaringan berada di bawah nilai yang dapat diterima sesuai dengan kemampuan instans. Untuk informasi selengkapnya, lihat Jenis EC2 Instans Amazon
  - Pada instans dengan `ena` penggerak Jaringan yang Ditingkatkan, periksa statistik ena untuk waktu habis habis atau batas yang terlampaui. Statistik berikut berguna untuk mengonfirmasi kejenuhan batas jaringan:
    - `bw_in_allowance_exceeded` / `bw_out_allowance_exceeded`: jumlah paket yang ditunda karena kelebihan throughput masuk atau keluar;
    - `conntrack_allowance_exceeded`: jumlah paket yang tidak diteruskan karena batas pelacakan koneksi dari grup keamanan. Koneksi baru akan gagal saat batas ini jenuh;
    - `linklocal_allowance_exceeded`: jumlah paket yang tidak diteruskan karena permintaan berlebihan untuk metadata dari instans, NTP melalui DNS VPC. Batasnya adalah 1024 paket per detik untuk semua layanan;
    - `pps_allowance_exceeded`: jumlah paket yang tidak diteruskan karena rasio paket per detik yang berlebihan. Batas paket per detik dapat tercapai jika lalu lintas jaringan terdiri dari ribuan atau jutaan permintaan yang sangat kecil per detik. Lalu lintas ElastiCache dapat dioptimalkan untuk memanfaatkan paket jaringan lebih baik melalui alur atau perintah yang melakukan beberapa operasi sekaligus seperti `MGET` alih-alih `GET`.

Membedah waktu yang dibutuhkan untuk menyelesaikan satu permintaan tunggal

- Di jaringan: `Tcpdump` dan `Wireshark` (tshark pada baris perintah) adalah alat yang berguna untuk memahami berapa banyak waktu yang dibutuhkan permintaan untuk melakukan perjalanan jaringan, menekan ElastiCache mesin dan mendapatkan pengembalian. Contoh berikut menyorot satu permintaan tunggal yang dibuat dengan perintah berikut:

```
$ echo ping | nc example.xxxxxx.ng.0001.use1.cache.amazonaws.com 6379
+PONG
```

  Sejajar dengan perintah di atas, tcpdump dijalankan dan menghasilkan:

```
$ sudo tcpdump -i any -nn port 6379 -tt
```

```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
1609428918.917869 IP 172.31.11.142.40966
    > 172.31.11.247.6379: Flags [S], seq 177032944, win 26883, options [mss
 8961,sackOK,TS val 27819440 ecr 0,nop,wscale 7], length 0
1609428918.918071 IP 172.31.11.247.6379 > 172.31.11.142.40966: Flags [S.], seq
 53962565, ack 177032945, win
    28960, options [mss 1460,sackOK,TS val 3788576332 ecr 27819440,nop,wscale 7],
 length 0
1609428918.918091 IP 172.31.11.142.40966 > 172.31.11.247.6379: Flags [.], ack 1, win
 211, options [nop,nop,TS val 27819440 ecr 3788576332], length 0
1609428918.918122
    IP 172.31.11.142.40966 > 172.31.11.247.6379: Flags [P.], seq 1:6, ack 1, win 211,
 options [nop,nop,TS val 27819440 ecr 3788576332], length 5: RESP "ping"
1609428918.918132 IP 172.31.11.142.40966 > 172.31.11.247.6379: Flags [F.], seq 6, ack
    1, win 211, options [nop,nop,TS val 27819440 ecr 3788576332], length 0
1609428918.918240 IP 172.31.11.247.6379 > 172.31.11.142.40966: Flags [.], ack 6, win
 227, options [nop,nop,TS val 3788576332 ecr 27819440], length 0
1609428918.918295
    IP 172.31.11.247.6379 > 172.31.11.142.40966: Flags [P.], seq 1:8, ack 7, win 227,
 options [nop,nop,TS val 3788576332 ecr 27819440], length 7: RESP "PONG"
1609428918.918300 IP 172.31.11.142.40966 > 172.31.11.247.6379: Flags [.], ack 8, win
    211, options [nop,nop,TS val 27819441 ecr 3788576332], length 0
1609428918.918302 IP 172.31.11.247.6379 > 172.31.11.142.40966: Flags [F.], seq 8, ack
 7, win 227, options [nop,nop,TS val 3788576332 ecr 27819440], length 0
1609428918.918307
    IP 172.31.11.142.40966 > 172.31.11.247.6379: Flags [.], ack 9, win 211, options
 [nop,nop,TS val 27819441 ecr 3788576332], length 0
^C
10 packets captured
10 packets received by filter
0 packets dropped by kernel
```

Dari output di atas kita dapat mengonfirmasi bahwa handshake tiga arah TCP diselesaikan dalam 222 mikrodetik (918091 - 917869) dan perintah ping dikirim dan dikembalikan dalam 173 mikrodetik (918295 - 918122).

Dibutuhkan waktu 438 mikrodetik (918307 - 917869) mulai dari membuat permintaan hingga koneksi ditutup. Hasil tersebut akan memastikan bahwa waktu respons jaringan dan mesin adalah baik dan penyelidikan dapat berfokus pada komponen lainnya.

- Pada sistem operasi: `Strace` dapat membantu mengidentifikasi kesenjangan waktu pada tingkat OS. Analisis aplikasi aktual akan jauh lebih luas dan dianjurkan menggunakan alat profil khusus

untuk aplikasi atau debugger. Contoh berikut hanya menunjukkan jika komponen sistem operasi dasar berfungsi seperti yang diharapkan, jika tidak, penyelidikan lebih lanjut mungkin diperlukan. Menggunakan `PING` perintah Redis OSS yang sama dengan `strace` kita mendapatkan:

```
$ echo ping | strace -f -tttt -r -e trace=execve,socket,open,recvfrom,sendto
 nc example.xxxxxx.ng.0001.use1.cache.amazonaws.com (http://
example.xxxxxx.ng.0001.use1.cache.amazonaws.com/)
    6379
1609430221.697712 (+ 0.000000) execve("/usr/bin/nc", ["nc",
 "example.xxxxxx.ng.0001.use"..., "6379"], 0x7fffede7cc38 /* 22 vars */) = 0
1609430221.708955 (+ 0.011231) socket(AF_UNIX, SOCK_STREAM|SOCK_CLOEXEC|
SOCK_NONBLOCK, 0) = 3
1609430221.709084
    (+ 0.000124) socket(AF_UNIX, SOCK_STREAM|SOCK_CLOEXEC|SOCK_NONBLOCK, 0) = 3
1609430221.709258 (+ 0.000173) open("/etc/nsswitch.conf", O_RDONLY|O_CLOEXEC) = 3
1609430221.709637 (+ 0.000378) open("/etc/host.conf", O_RDONLY|O_CLOEXEC) = 3
1609430221.709923
    (+ 0.000286) open("/etc/resolv.conf", O_RDONLY|O_CLOEXEC) = 3
1609430221.711365 (+ 0.001443) open("/etc/hosts", O_RDONLY|O_CLOEXEC) = 3
1609430221.713293 (+ 0.001928) socket(AF_INET, SOCK_DGRAM|SOCK_CLOEXEC|SOCK_NONBLOCK,
 IPPROTO_IP) = 3
1609430221.717419
    (+ 0.004126) recvfrom(3, "\362|
\201\200\0\1\0\2\0\0\0\0\rnotls20201224\6tihew"..., 2048, 0, {sa_family=AF_INET,
 sin_port=htons(53), sin_addr=inet_addr("172.31.0.2")}, [28->16]) = 155
1609430221.717890 (+ 0.000469) recvfrom(3,
 "\204\207\201\200\0\1\0\1\0\0\0\0\rnotls20201224\6tihew"...,
    65536, 0, {sa_family=AF_INET, sin_port=htons(53),
 sin_addr=inet_addr("172.31.0.2")}, [28->16]) = 139
1609430221.745659 (+ 0.027772) socket(AF_INET, SOCK_STREAM, IPPROTO_TCP) = 3
1609430221.747548 (+ 0.001887) recvfrom(0, 0x7ffcf2f2ca50, 8192,
    0, 0x7ffcf2f2c9d0, [128]) = -1 ENOTSOCK (Socket operation on non-socket)
1609430221.747858 (+ 0.000308) sendto(3, "ping\n", 5, 0, NULL, 0) = 5
1609430221.748048 (+ 0.000188) recvfrom(0, 0x7ffcf2f2ca50, 8192, 0, 0x7ffcf2f2c9d0,
 [128]) = -1 ENOTSOCK
    (Socket operation on non-socket)
1609430221.748330 (+ 0.000282) recvfrom(3, "+PONG\r\n", 8192, 0, 0x7ffcf2f2c9d0,
 [128->0]) = 7
+PONG
1609430221.748543 (+ 0.000213) recvfrom(3, "", 8192, 0, 0x7ffcf2f2c9d0, [128->0]) = 0
1609430221.752110
    (+ 0.003569) +++ exited with 0 +++
```

Pada contoh di atas, perintah ini membutuhkan waktu sekitar 54 milidetik untuk diselesaikan (752110 - 697712 = 54398 mikrodetik).

Lama waktu yang signifikan, sekitar 20 ms, dibutuhkan untuk membuat instans nc dan melakukan resolusi nama (dari 697712 hingga 717890), setelah itu, waktu 2 ms diperlukan untuk membuat soket TCP (745659 hingga 747858), dan waktu 0,4 ms (747858 hingga 748330) dibutuhkan untuk mengirimkan dan menerima respons untuk permintaan.

Topik Terkait

- the section called “Praktik terbaik dan strategi caching”

Keamanan di Amazon ElastiCache

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. Model tanggung jawab bersama menggambarkan hal ini sebagai keamanan dari cloud dan keamanan di cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga secara berkala menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari Program kepatuhan AWS. Untuk mempelajari tentang program kepatuhan yang berlaku untuk Amazon ElastiCache, lihat AWS Layanan dalam Lingkup berdasarkan Program Kepatuhan.
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain, yang mencakup sensitivitas data Anda, persyaratan perusahaan Anda, serta undang-undang dan peraturan yang berlaku.

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan Amazon ElastiCache. Topik berikut menunjukkan cara mengonfigurasi Amazon ElastiCache untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga mempelajari cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan ElastiCache sumber daya Amazon Anda.

Topik

Perlindungan data di Amazon ElastiCache

Model tanggung jawab AWS bersama model berlaku untuk perlindungan data di AWS ElastiCache (ElastiCache). Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Konten ini mencakup konfigurasi keamanan dan tugas manajemen untuk AWS layanan yang Anda gunakan. Untuk informasi selengkapnya tentang privasi data, lihat FAQ privasi data.

Untuk tujuan perlindungan data, kami menyarankan Anda untuk melindungi kredensyal AWS akun dan menyiapkan akun individual dengan AWS Identity and Access Management (IAM). Dengan cara ini, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugas mereka. Kami juga merekomendasikan agar Anda mengamankan data Anda dengan cara-cara berikut ini:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan TLS untuk berkomunikasi dengan AWS sumber daya.
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail.
- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default dalam AWS layanan.
- Gunakan layanan keamanan terkelola lanjutan seperti Amazon Macie, yang membantu menemukan dan mengamankan data pribadi yang disimpan di Amazon S3.

Sebaiknya jangan pernah memasukkan informasi identitas yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam bidang isian bebas seperti bidang Nama. Ini termasuk ketika Anda bekerja dengan ElastiCache atau AWS layanan lain menggunakan konsol, API AWS CLI, atau AWS SDKs. Data apa pun yang Anda masukkan ke ElastiCache atau layanan lain mungkin akan diambil untuk dimasukkan dalam log diagnostik. Saat Anda memberikan URL ke server eksternal, jangan sertakan informasi kredensial di URL untuk memvalidasi permintaan Anda ke server tersebut.

Topik

- Keamanan data di Amazon ElastiCache

Keamanan data di Amazon ElastiCache

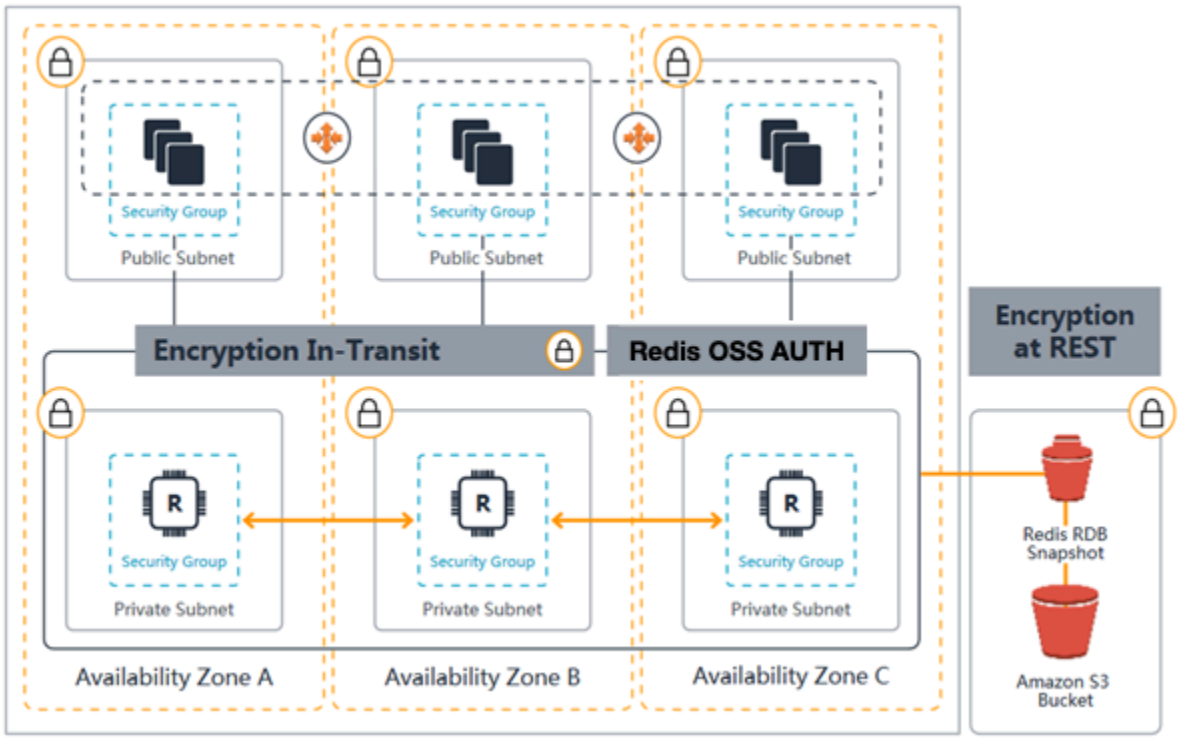
Untuk membantu menjaga keamanan data Anda, Amazon ElastiCache dan Amazon EC2 menyediakan mekanisme untuk mencegah akses data Anda yang tidak sah di server.

Amazon ElastiCache untuk Memcached menyediakan fitur enkripsi untuk data pada cache yang menjalankan Memcached versi 1.6.12 atau yang lebih baru.

Amazon ElastiCache dengan Valkey dan Redis OSS menyediakan fitur enkripsi untuk data pada cache yang menjalankan Valkey 7.2 atau yang lebih baru, dan Redis OSS versi 3.2.6 (dijadwalkan untuk EOL, lihat jadwal akhir masa pakai versi Redis OSS), 4.0.10 atau yang lebih baru. Amazon ElastiCache juga mendukung otentikasi pengguna dengan IAM atau Valkey dan Redis OSS AUTH, dan mengotorisasi operasi pengguna menggunakan Role-Based Access Control (RBAC).

- Enkripsi bergerak mengenkripsi data Anda setiap kali data bergerak dari satu tempat ke tempat lain, misalnya antara simpul di klaster atau antara cache dan aplikasi Anda.
- Enkripsi diam mengenkripsi data pada disk Anda selama operasi sinkronisasi dan pencadangan.

ElastiCache mendukung otentikasi pengguna menggunakan IAM dan perintah Valkey dan Redis OSS AUTH, dan mengotorisasi operasi pengguna menggunakan Role-Based Access Control (RBAC).



ElastiCache untuk Diagram Keamanan Valkey dan Redis OSS

Topik

ElastiCache enkripsi dalam transit (TLS)

Untuk membantu menjaga keamanan data Anda, Amazon ElastiCache dan Amazon EC2 menyediakan mekanisme untuk mencegah akses data Anda yang tidak sah di server. Dengan menyediakan kemampuan enkripsi dalam perjalanan, ElastiCache memberi Anda alat yang dapat Anda gunakan untuk membantu melindungi data Anda saat berpindah dari satu lokasi ke lokasi lain.

Semua cache tanpa server Valkey atau Redis OSS memiliki enkripsi dalam transit yang diaktifkan. Untuk klaster yang dirancang sendiri, Anda dapat mengaktifkan enkripsi bergerak pada grup replikasi dengan menetapkan parameter `TransitEncryptionEnabled` ke `true` (CLI: `--transit-encryption-enabled`) saat Anda membuat grup replikasi. Anda dapat melakukan ini apakah Anda membuat grup replikasi menggunakan AWS Management Console, the AWS CLI, atau ElastiCache API.

Semua cache nirserver memiliki enkripsi bergerak yang aktif. Untuk klaster yang dirancang sendiri, Anda dapat mengaktifkan enkripsi bergerak pada klaster cache dengan mengatur parameter `TransitEncryptionEnabled` ke `true` (CLI: `--transit-encryption-enabled`) saat Anda membuat klaster cache menggunakan operasi `CreateCacheCluster` (CLI: `create-cache-cluster`).

Topik

- Menghubungkan ke ElastiCache (Valkey) atau Amazon ElastiCache untuk Redis OSS dengan enkripsi dalam transit menggunakan valkey-cli
- Mengaktifkan enkripsi dalam transit pada Redis OSS Cluster yang dirancang sendiri menggunakan Python
- Praktik terbaik saat mengaktifkan enkripsi bergerak
- Menghubungkan ke node diaktifkan dengan enkripsi dalam transit menggunakan Openssl (Memcached)
- Membuat klien Memcached TLS menggunakan Java
- Membuat klien Memcached TLS menggunakan PHP

Gambaran umum enkripsi bergerak

Enkripsi ElastiCache in-transit Amazon adalah fitur yang memungkinkan Anda meningkatkan keamanan data Anda di titik yang paling rentan — saat transit dari satu lokasi ke lokasi lain. Karena diperlukan beberapa pemrosesan untuk mengenkripsi dan mendekripsi data di titik akhir, mengaktifkan enkripsi bergerak dapat memberikan beberapa dampak pada performa. Anda harus menolok ukur data Anda dengan dan tanpa enkripsi bergerak untuk menentukan dampak terhadap performa pada kasus penggunaan Anda.

ElastiCache enkripsi in-transit mengimplementasikan fitur-fitur berikut:

- Koneksi klien terenkripsi—koneksi klien ke simpul cache dienkripsi TLS.
- Koneksi server terenkripsi—data yang bergerak antarsimpul dalam klaster dienkripsi.
- Autentikasi server—Klien dapat mengautentikasi bahwa koneksinya dilakukan ke server yang benar.
- Otentikasi klien —menggunakan fitur Valkey dan Redis OSS AUTH, server dapat mengautentikasi klien.

Kondisi enkripsi dalam transit (Valkey dan Redis OSS)

Kendala berikut pada enkripsi ElastiCache in-transit Amazon harus diingat ketika Anda merencanakan implementasi cluster yang dirancang sendiri:

- Enkripsi dalam transit didukung pada grup replikasi yang menjalankan Valkey 7.2 dan yang lebih baru, dan Redis OSS versi 3.2.6, 4.0.10 dan yang lebih baru.

- Memodifikasi pengaturan enkripsi dalam transit, untuk klaster yang ada, didukung pada grup replikasi yang menjalankan Valkey 7.2 dan yang lebih baru, dan Redis OSS versi 7 dan yang lebih baru.
- Enkripsi bergerak didukung hanya untuk grup replikasi yang berjalan di Amazon VPC.
- Enkripsi dalam transit tidak didukung untuk grup replikasi yang menjalankan tipe node berikut: M1, M2.

  Untuk informasi selengkapnya, lihat Jenis simpul yang didukung.
- Enkripsi in-transit diaktifkan dengan menetapkan parameter `TransitEncryptionEnabled` menjadi `true` secara eksplisit.
- Pastikan klien cache Anda mendukung konektivitas TLS dan Anda telah mengaktifkannya dalam konfigurasi klien.
- Mulai 26 Januari 2026, AWS akan memperbarui versi TLS minimum yang didukung ke 1.2 aktif ElastiCache untuk Valkey versi 7.2 ke atas, dan ElastiCache untuk Redis OSS versi 6 ke atas. Pelanggan harus memperbarui perangkat lunak klien mereka sebelum tanggal tersebut. Pembaruan ini membantu Anda memenuhi kebutuhan keamanan, kepatuhan, dan peraturan.

Kondisi enkripsi dalam transit (Memcached)

Kendala berikut pada enkripsi ElastiCache in-transit Amazon harus diingat ketika Anda merencanakan implementasi cluster yang dirancang sendiri:

- Enkripsi bergerak didukung pada klaster yang menjalankan Memcached versi 1.6.12 atau yang lebih baru.
- Enkripsi bergerak mendukung Keamanan Lapisan Pengangkutan (TLS) versi 1.2 dan 1.3.
- Enkripsi bergerak didukung hanya untuk klaster yang berjalan di Amazon VPC.
- Enkripsi dalam transit tidak didukung untuk grup replikasi yang menjalankan jenis node berikut: M1, M2, M3, R3, T2.

  Untuk informasi selengkapnya, lihat Jenis simpul yang didukung.
- Enkripsi bergerak diaktifkan dengan menetapkan parameter `TransitEncryptionEnabled` ke `true` secara eksplisit.
- Anda dapat mengaktifkan enkripsi bergerak pada klaster hanya saat pembuatan klaster. Anda tidak dapat mengaktifkan dan menonaktifkan enkripsi bergerak dengan mengubah klaster.

- Pastikan klien cache Anda mendukung konektivitas TLS dan Anda telah mengaktifkannya dalam konfigurasi klien.

Praktik terbaik enkripsi bergerak

- Karena pemrosesan diperlukan untuk mengenkripsi dan mendekripsi data di titik akhir, menerapkan enkripsi bergerak dapat mengurangi performa. Lakukan tolok ukur terhadap enkripsi bergerak dibandingkan dengan tanpa enkripsi pada data Anda sendiri untuk menentukan dampaknya terhadap performa untuk implementasi Anda.
- Karena membuat koneksi baru bisa menghabiskan banyak daya komputasi, Anda dapat mengurangi dampak enkripsi bergerak pada performa dengan mempersistensi koneksi TLS Anda.

Opsi Valkey dan Redis OSS lebih lanjut

Untuk informasi lebih lanjut tentang opsi yang tersedia untuk Valkey dan Redis OSS, lihat tautan berikut.

- Enkripsi At-Rest di ElastiCache
- Mengautentikasi dengan perintah Valkey dan Redis OSS AUTH
- Kontrol Akses Berbasis Peran (RBAC)
- Amazon VPCs dan ElastiCache keamanan
- Identity and Access Management untuk Amazon ElastiCache

Mengaktifkan enkripsi dalam transit untuk Memcached

Untuk mengaktifkan enkripsi bergerak saat membuat klaster Memcached menggunakan Konsol Manajemen AWS , buat pilihan berikut:

- Pilih Memcached sebagai mesin Anda.
- Pilih versi mesin 1.6.12 atau yang lebih baru.
- Pada Enkripsi bergerak, pilih Aktifkan.

Untuk step-by-step prosesnya, lihatMembuat cluster untuk Valkey atau Redis OSS.

Mengaktifkan enkripsi bergerak

Semua cache nirserver memiliki enkripsi bergerak yang aktif. Pada cluster yang dirancang sendiri, Anda dapat mengaktifkan enkripsi dalam transit menggunakan AWS Management Console, AWS CLI, atau API. ElastiCache

Mengaktifkan enkripsi dalam transit menggunakan AWS Management Console

Mengaktifkan enkripsi dalam transit untuk kluster baru yang dirancang sendiri menggunakan AWS Management Console

Saat merancang klaster Anda sendiri, konfigurasi 'Dev/Test' dan 'Produksi' dengan metode 'Mudah dibuat' akan mengaktifkan enkripsi bergerak. Saat memilih konfigurasi sendiri, buat pilihan berikut:

- Pilih versi mesin 3.2.6, 4.0.10 atau yang lebih baru.
- Klik kotak centang di samping Aktifkan untuk opsi Enkripsi bergerak.

Untuk step-by-step prosesnya, lihat yang berikut ini:

- Membuat cluster Valkey (mode cluster dinonaktifkan) (Konsol)
- Membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) (Konsol)

Mengaktifkan enkripsi dalam transit untuk kluster yang dirancang sendiri yang ada menggunakan AWS Management Console

Mengaktifkan enkripsi bergerak, adalah proses dua langkah, Anda harus terlebih dahulu mengatur mode enkripsi bergerak ke `preferred`. Mode ini memungkinkan klien Valkey atau Redis OSS Anda untuk terhubung menggunakan koneksi terenkripsi dan tidak terenkripsi. Setelah Anda memigrasikan semua klien Valkey atau Redis OSS Anda untuk menggunakan koneksi terenkripsi, Anda kemudian dapat memodifikasi konfigurasi cluster Anda untuk mengatur mode enkripsi transit ke. `required` Mengatur mode enkripsi bergerak ke `required` akan menghentikan semua koneksi yang tidak terenkripsi dan hanya akan mengizinkan koneksi terenkripsi.

Setel mode enkripsi Transit Anda ke Preferred

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Pilih cache Valkey atau cache Redis OSS dari ElastiCache Sumber daya yang tercantum di panel navigasi, hadir di sebelah kiri.

3. Pilih cache yang ingin Anda perbarui.
4. Pilih drop-down Tindakan, lalu pilih Ubah.
5. Pilih Aktifkan pada Enkripsi bergerak di bagian Keamanan.
6. Klik Pilihan sebagai Mode enkripsi bergerak.
7. Pilih Pratinjau perubahan dan simpan perubahan Anda.

Setelah Anda memigrasikan semua klien Valkey atau Redis OSS Anda untuk menggunakan koneksi terenkripsi:

Setel mode enkripsi Transit Anda ke Wajib

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Pilih cache Valkey atau cache Redis OSS dari ElastiCache Sumber daya yang tercantum di panel navigasi, hadir di sebelah kiri.
3. Pilih cache yang ingin Anda perbarui.
4. Pilih drop-down Tindakan, lalu pilih Ubah.
5. Pilih Wajib sebagai Mode enkripsi bergerak, di bagian Keamanan.
6. Pilih Pratinjau perubahan dan simpan perubahan Anda.

Mengaktifkan enkripsi dalam transit menggunakan AWS CLI

Untuk mengaktifkan enkripsi dalam transit saat membuat grup replikasi Valkey atau Redis OSS menggunakan, gunakan parameter. AWS CLI`transit-encryption-enabled`

Mengaktifkan enkripsi dalam transit pada cluster baru yang dirancang sendiri untuk Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) (CLI)

Gunakan AWS CLI operasi `create-replication-group` dan parameter berikut untuk membuat grup replikasi Valkey atau Redis OSS dengan replika yang mengaktifkan enkripsi in-transit:

Parameter kunci:

- **`--engine`**—Harus `valkey` atau`redis`.
- **`--engine-version`**—Jika mesinnya Redis OSS, ini harus 3.2.6, 4.0.10 atau yang lebih baru.

- **`--transit-encryption-enabled`**—Wajib. Jika Anda mengaktifkan enkripsi bergerak, Anda juga harus menyediakan nilai untuk parameter `--cache-subnet-group`.
- **`--num-cache-clusters`**—Minimal bernilai 1. Nilai maksimum untuk parameter ini adalah enam.

Untuk informasi selengkapnya, lihat berikut ini:

- Membuat grup replikasi Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) dari awal ()AWS CLI
- create-replication-group

Mengaktifkan enkripsi dalam transit pada cluster baru yang dirancang sendiri untuk Valkey atau Redis OSS (Mode Cluster Diaktifkan) (CLI)

Gunakan AWS CLI operasi `create-replication-group` dan parameter berikut untuk membuat grup replikasi Valkey atau Redis OSS (mode cluster enabled) yang mengaktifkan enkripsi in-transit:

Parameter kunci:

- **`--engine`**—Harus `valkey` atau`redis`.
- **`--engine-version`**—Jika mesinnya Redis OSS, ini harus 3.2.6, 4.0.10 atau yang lebih baru.
- **`--transit-encryption-enabled`**—Wajib. Jika Anda mengaktifkan enkripsi bergerak, Anda juga harus menyediakan nilai untuk parameter `--cache-subnet-group`.
- Gunakan salah satu set parameter berikut untuk menentukan konfigurasi grup simpul dari grup replikasi:
  - **`--num-node-groups`**—Menentukan jumlah serpihan (grup simpul) dalam grup replikasi ini. Nilai maksimum untuk parameter ini adalah 500.

    `--replicas-per-node-group`—Menentukan jumlah simpul replika di setiap grup simpul. Nilai yang ditentukan di sini berlaku untuk semua serpihan dalam grup replikasi ini. Nilai maksimum untuk parameter ini adalah 5.
  - **`--node-group-configuration`**—Menentukan konfigurasi dari setiap serpihan secara independen.

Untuk informasi selengkapnya, lihat berikut ini:

- Membuat grup replikasi Valkey atau Redis OSS (Cluster Mode Enabled) dari awal ()AWS CLI
- create-replication-group

Mengaktifkan enkripsi bergerak pada klaster yang sudah ada menggunakan AWS CLI

Mengaktifkan enkripsi bergerak, adalah proses dua langkah, Anda harus terlebih dahulu mengatur mode enkripsi bergerak ke `preferred`. Mode ini memungkinkan klien Valkey atau Redis OSS Anda untuk terhubung menggunakan koneksi terenkripsi dan tidak terenkripsi. Setelah Anda memigrasikan semua klien Valkey atau Redis OSS Anda untuk menggunakan koneksi terenkripsi, Anda kemudian dapat memodifikasi konfigurasi cluster Anda untuk mengatur mode enkripsi transit ke. `required` Mengatur mode enkripsi bergerak ke `required` akan menghentikan semua koneksi yang tidak terenkripsi dan hanya akan mengizinkan koneksi terenkripsi.

Gunakan AWS CLI operasi `modify-replication-group` dan parameter berikut untuk memperbarui grup replikasi Valkey atau Redis OSS (mode cluster enabled) yang menonaktifkan enkripsi in-transit.

Untuk mengaktifkan enkripsi bergerak

1. Setel transit-encryption-mode ke`preferred`, menggunakan parameter berikut
   - **`--transit-encryption-enabled`**—Wajib.
   - **`--transit-encryption-mode`**—Harus diatur ke `preferred`.
2. Setel transit-encryption-mode ke`required`, menggunakan parameter berikut:
   - **`--transit-encryption-enabled`**—Wajib.
   - **`--transit-encryption-mode`**—Harus diatur ke `required`.

Menghubungkan ke ElastiCache (Valkey) atau Amazon ElastiCache untuk Redis OSS dengan enkripsi dalam transit menggunakan valkey-cli

Untuk mengakses data dari ElastiCache cache Redis OSS yang diaktifkan dengan enkripsi dalam transit, Anda menggunakan klien yang bekerja dengan Secure Socket Layer (SSL). Anda juga dapat menggunakan valkey-cli dengan di TLS/SSL Amazon Linux dan Amazon Linux 2. Jika klien Anda tidak mendukung TLS, Anda dapat menggunakan `stunnel` perintah pada host klien Anda untuk membuat terowongan SSL ke node Redis OSS.

Koneksi terenkripsi dengan Linux

Untuk menggunakan valkey-cli untuk terhubung ke klaster Valkey atau Redis OSS yang diaktifkan dengan enkripsi dalam transit di Amazon Linux 2 atau Amazon Linux, ikuti langkah-langkah berikut.

1. Unduh dan kompilasi utilitas valkey-cli. Utilitas ini termasuk dalam distribusi perangkat lunak Valkey.
2. Pada prompt perintah EC2 instance Anda, ketikkan perintah yang sesuai untuk versi Linux yang Anda gunakan.

   Amazon Linux 2

   Jika menggunakan Amazon Linux 2, masukkan ini:

   ```
   sudo yum -y install openssl-devel gcc
   wget https://github.com/valkey-io/valkey/archive/refs/tags/7.2.6.tar.gz
   tar xvzf valkey-7.2.6.tar.gz
   cd valkey-7.2.6
   make distclean
   make valkey-cli BUILD_TLS=yes
   sudo install -m 755 src/valkey-cli /usr/local/bin/
   ```

   Amazon Linux

   Jika menggunakan Amazon Linux, masukkan ini:

   ```
   sudo yum install gcc jemalloc-devel openssl-devel tcl tcl-devel clang wget
   wget https://github.com/valkey-io/valkey/archive/refs/tags/8.0.0.tar.gz
   tar xvzf valkey-8.0.0.tar.gz
   cd valkey-8.0.0
   make valkey-cli CC=clang BUILD_TLS=yes
   sudo install -m 755 src/valkey-cli /usr/local/bin/
   ```

   Di Amazon Linux, Anda mungkin perlu menjalankan langkah tambahan berikut:

   ```
   sudo yum install clang
   CC=clang make
   sudo make install
   ```
3. Setelah Anda mengunduh dan menginstal utilitas valkey-cli, Anda disarankan untuk menjalankan perintah opsional. `make-test`
4. Untuk terhubung ke cluster dengan enkripsi dan otentikasi diaktifkan, masukkan perintah ini:

   ```
   valkey-cli -h Primary or Configuration Endpoint --tls -a 'your-password' -p 6379
   ```

> **Note**
>
> Jika Anda menginstal redis6 di Amazon Linux 2023, Anda sekarang dapat menggunakan `redis6-cli` perintah alih-alih: `valkey-cli`
>
> ```
> redis6-cli -h Primary or Configuration Endpoint --tls -p 6379
> ```

Koneksi terenkripsi dengan stunnel

Untuk menggunakan valkey-cli untuk terhubung ke cluster Redis OSS yang diaktifkan dengan enkripsi dalam transit menggunakan stunnel, ikuti langkah-langkah berikut.

1. Gunakan SSH untuk menghubungkan ke klien Anda dan instal `stunnel`.

   ```
   sudo yum install stunnel
   ```

2. Jalankan perintah berikut untuk membuat dan mengedit file `'/etc/stunnel/valkey-cli.conf'` secara bersamaan untuk menambahkan titik akhir cluster Redis OSS untuk satu atau lebih parameter koneksi, menggunakan output yang disediakan di bawah ini sebagai template. ElastiCache

   ```
   vi /etc/stunnel/valkey-cli.conf


   fips = no
   setuid = root
   setgid = root
   pid = /var/run/stunnel.pid
   debug = 7
   delay = yes
   options = NO_SSLv2
   options = NO_SSLv3
   [valkey-cli]
      client = yes
      accept = 127.0.0.1:6379
      connect = primary.ssltest.wif01h.use1.cache.amazonaws.com:6379
   [valkey-cli-replica]
      client = yes
      accept = 127.0.0.1:6380
   ```

```
   connect = ssltest-02.ssltest.wif01h.use1.cache.amazonaws.com:6379
```

Dalam contoh ini, file config memiliki dua koneksi, `valkey-cli` dan `valkey-cli-replica`. Parameternya ditetapkan sebagai berikut:

- client ditetapkan ke ya untuk menentukan bahwa instans stunnel ini adalah klien.
- accept ditetapkan ke IP klien. Dalam contoh ini, primer diatur ke default Redis OSS 127.0.0.1 pada port 6379. Replika harus memanggil port yang berbeda dan ditetapkan ke 6380. Anda dapat menggunakan port sementara 1024–65535. Untuk informasi selengkapnya, lihat Port sementara dalam Panduan Pengguna Amazon VPC.
- connect diatur ke titik akhir server Redis OSS. Untuk informasi selengkapnya, lihat Menemukan titik akhir koneksi di ElastiCache.

3. Mulai `stunnel`.

```
sudo stunnel /etc/stunnel/valkey-cli.conf
```

Gunakan perintah `netstat` untuk mengonfirmasi bahwa tunnel dimulai.

```
sudo netstat -tulnp | grep -i stunnel

tcp        0      0 127.0.0.1:6379              0.0.0.0:*                   LISTEN
     3189/stunnel
tcp        0      0 127.0.0.1:6380              0.0.0.0:*                   LISTEN
     3189/stunnel
```

4. Connect ke node Redis OSS terenkripsi menggunakan endpoint lokal terowongan.

- Jika tidak ada kata sandi AUTH yang digunakan ElastiCache selama pembuatan klaster Redis OSS, contoh ini menggunakan valkey-cli untuk terhubung ke server Redis OSS untuk menggunakan jalur lengkap ElastiCache untuk valkey-cli, di Amazon Linux:

```
/home/ec2-user/redis-7.2.5/src/valkey-cli -h localhost -p 6379
```

Jika kata sandi AUTH digunakan selama pembuatan klaster Redis OSS, contoh ini menggunakan valkey-cli untuk terhubung ke server Redis OSS menggunakan jalur lengkap untuk valkey-cli, di Amazon Linux:

```
 /home/ec2-user/redis-7.2.5/src/valkey-cli -h localhost -p 6379 -a my-secret-password
```

ATAU

- Ubah direktori ke redis-7.2.5 dan lakukan hal berikut:

  Jika tidak ada kata sandi AUTH yang digunakan ElastiCache selama pembuatan klaster Redis OSS, contoh ini menggunakan valkey-cli untuk terhubung ke server Redis OSS untuk menggunakan jalur lengkap ElastiCache untuk valkey-cli, di Amazon Linux:

  ```
  src/valkey-cli -h localhost -p 6379
  ```

  Jika kata sandi AUTH digunakan selama pembuatan cluster Redis OSS, contoh ini menggunakan valkey-cli untuk terhubung ke server Valkey atau Redis OSS menggunakan jalur lengkap untuk valkey-cli, di Amazon Linux:

  ```
  src/valkey-cli -h localhost -p 6379 -a my-secret-password
  ```

Contoh ini menggunakan Telnet untuk terhubung ke server Valkey Redis OSS.

```
telnet localhost 6379

Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
auth MySecretPassword
+OK
get foo
$3
bar
```

5. Untuk menghentikan dan menutup tunnel SSL, jalankan `pkill` terhadap proses stunnel.

```
sudo pkill stunnel
```

Mengaktifkan enkripsi dalam transit pada Redis OSS Cluster yang dirancang sendiri menggunakan Python

Panduan berikut akan menunjukkan cara mengaktifkan enkripsi dalam transit pada cluster Redis OSS 7.0 yang awalnya dibuat dengan enkripsi in-transit dinonaktifkan. Klien TCP dan TLS akan terus berkomunikasi dengan klaster selama proses ini tanpa waktu henti.

Boto3 akan mendapatkan kredensial yang dibutuhkannya (`aws_access_key_id`, `aws_secret_access_key`, dan `aws_session_token`) dari variabel lingkungan. Kredensial tersebut akan ditempelkan terlebih dahulu di terminal bash yang sama di tempat kita menjalankan `python3` untuk memproses kode Python yang ditunjukkan dalam panduan ini. Kode dalam contoh di bawah ini adalah proses dari sebuah EC2 instance yang diluncurkan di VPC yang sama yang akan digunakan untuk membuat ElastiCache Redis OSS Cluster di dalamnya.

> **Note**
>
> - Contoh berikut menggunakan boto3 SDK untuk operasi ElastiCache manajemen (kluster atau pembuatan pengguna) dan redis-py-cluster redis-py/ untuk penanganan data.
> - Anda harus menggunakan setidaknya boto3 versi (=~) 1.26.39 untuk menggunakan migrasi TLS online dengan API perubahan klaster.
> - ElastiCache mendukung migrasi TLS online hanya untuk cluster dengan Valkey versi 7.2 ke atas atau Redis OSS versi 7.0 atau lebih tinggi. Jadi jika Anda memiliki cluster yang menjalankan versi Redis OSS lebih awal dari 7.0, Anda harus memutakhirkan versi Redis OSS dari cluster Anda. Untuk informasi selengkapnya tentang perbedaan versi, lihat Perilaku versi mesin utama dan perbedaan kompatibilitas dengan Redis OSS.

Topik

Tentukan konstanta string yang akan meluncurkan ElastiCache Valkey atau Redis OSS Cluster

Pertama, mari kita mendefinisikan beberapa konstanta string Python sederhana yang akan menampung nama-nama AWS entitas yang diperlukan untuk membuat ElastiCache cluster seperti`security-group`,`Cache Subnet group`, dan a. `default parameter group` Semua AWS entitas ini harus dibuat terlebih dahulu di AWS akun Anda di Wilayah yang ingin Anda gunakan.

```
#Constants definitions
SECURITY_GROUP = "sg-0492aa0a29c558427"
CLUSTER_DESCRIPTION = "This cluster has been launched as part of the online TLS
 migration user guide"
EC_SUBNET_GROUP = "client-testing"
DEFAULT_PARAMETER_GROUP_REDIS_7_CLUSTER_MODE_ENABLED = "default.redis7.cluster.on"
```

Tentukan kelas untuk konfigurasi klaster

Sekarang, mari kita definisikan beberapa kelas Python sederhana yang akan mewakili konfigurasi cluster, yang akan menyimpan metadata tentang cluster seperti versi Valkey atau Redis OSS, jenis instance, dan apakah enkripsi in-transit (TLS) diaktifkan atau dinonaktifkan.

```
#Class definitions

class Config:
    def __init__(
        self,
        instance_type: str = "cache.t4g.small",
        version: str = "7.0",
        multi_az: bool = True,
        TLS: bool = True,
        name: str = None,
    ):
        self.instance_type = instance_type
        self.version = version
        self.multi_az = multi_az
        self.TLS = TLS
        self.name = name or f"tls-test"

    def create_base_launch_request(self):
        return {
            "ReplicationGroupId": self.name,
            "TransitEncryptionEnabled": self.TLS,
            "MultiAZEnabled": self.multi_az,
```

```
            "CacheNodeType": self.instance_type,
            "Engine": "redis",
            "EngineVersion": self.version,
            "CacheSubnetGroupName": EC_SUBNET_GROUP ,
            "CacheParameterGroupName":
 DEFAULT_PARAMETER_GROUP_REDIS_7_CLUSTER_MODE_ENABLED ,
            "ReplicationGroupDescription": CLUSTER_DESCRIPTION,
            "SecurityGroupIds": [SECURITY_GROUP],
        }

class ConfigCME(Config):
    def __init__(
        self,
        instance_type: str = "cache.t4g.small",
        version: str = "7.0",
        multi_az: bool = True,
        TLS: bool = True,
        name: str = None,
        num_shards: int = 2,
        num_replicas_per_shard: int = 1,
    ):
        super().__init__(instance_type, version, multi_az, TLS, name)
        self.num_shards = num_shards
        self.num_replicas_per_shard = num_replicas_per_shard

    def create_launch_request(self) -> dict:
        launch_request = self.create_base_launch_request()
        launch_request["NumNodeGroups"] = self.num_shards
        launch_request["ReplicasPerNodeGroup"] = self.num_replicas_per_shard
        return launch_request
```

Mendefinisikan kelas yang akan merepresentasikan klaster itu sendiri

Sekarang, mari kita mendefinisikan beberapa kelas Python sederhana yang akan mewakili ElastiCache Valkey atau Redis OSS Cluster itu sendiri. Kelas ini akan memiliki bidang klien yang akan menampung klien boto3 untuk operasi ElastiCache manajemen seperti membuat cluster dan query API. ElastiCache

```
import botocore.config
import boto3

# Create boto3 client
def init_client(region: str = "us-east-1"):
```

```
    config = botocore.config.Config(retries={"max_attempts": 10, "mode": "standard"})
    init_request = dict()
    init_request["config"] = config
    init_request["service_name"] = "elasticache"
    init_request["region_name"] = region
    return boto3.client(**init_request)


class ElastiCacheClusterBase:
    def __init__(self, name: str):
        self.name = name
        self.elasticache_client = init_client()

    def get_first_replication_group(self):
        return self.elasticache_client.describe_replication_groups(
        ReplicationGroupId=self.name
        )["ReplicationGroups"][0]

    def get_status(self) -> str:
        return self.get_first_replication_group()["Status"]

    def get_transit_encryption_enabled(self) -> bool:
        return self.get_first_replication_group()["TransitEncryptionEnabled"]

    def is_available(self) -> bool:
        return self.get_status() == "available"

    def is_modifying(self) -> bool:
        return self.get_status() == "modifying"

    def wait_for_available(self):
        while True:
            if self.is_available():
                break
            else:
                time.sleep(5)

    def wait_for_modifying(self):
        while True:
            if self.is_modifying():
                break
            else:
                time.sleep(5)
```

```
    def delete_cluster(self) -> bool:
        self.elasticache_client.delete_replication_group(
            ReplicationGroupId=self.name, RetainPrimaryCluster=False
        )

    def modify_transit_encryption_mode(self, new_transit_encryption_mode: str):
        # generate api call to migrate the cluster to TLS preffered or to TLS required
            self.elasticache_client.modify_replication_group(
                ReplicationGroupId=self.name,
                TransitEncryptionMode=new_transit_encryption_mode,
                TransitEncryptionEnabled=True,
                ApplyImmediately=True,
            )
        self.wait_for_modifying()

 class ElastiCacheClusterCME(ElastiCacheClusterBase):
    def __init__(self, name: str):
        super().__init__(name)

    @classmethod
    def launch(cls, config: ConfigCME = None) -> ElastiCacheClusterCME:
        config = config or ConfigCME()
        print(config)
        new_cluster = ElastiCacheClusterCME(config.name)
        launch_request = config.create_launch_request()
        new_cluster.elasticache_client.create_replication_group(**launch_request)
        new_cluster.wait_for_available()
        return new_cluster

    def get_configuration_endpoint(self) -> str:
        return self.get_first_replication_group()["ConfigurationEndpoint"]["Address"]

#Since the code can throw exceptions, we define this class to make the code more
 readable and
#so we won't forget to delete the cluster
class ElastiCacheCMEManager:
    def __init__(self, config: ConfigCME = None):
        self.config = config or ConfigCME()

    def __enter__(self) -> ElastiCacheClusterCME:
        self.cluster = ElastiCacheClusterCME.launch(self.config)
        return self.cluster

    def __exit__(self, exc_type, exc_val, exc_tb):
```

```
        self.cluster.delete_cluster()
```

(Opsional) Buat kelas pembungkus untuk demo koneksi klien ke Valkey atau Redis OSS cluster

Sekarang, mari kita buat kelas wrapper untuk klien `redis-py-cluster`. Kelas wrapper ini akan mendukung pra-pengisian klaster dengan beberapa kunci lalu melakukan perintah `get` acak berulang.

> **Note**
>
> Ini adalah langkah opsional tetapi menyederhanakan kode fungsi utama yang muncul pada langkah selanjutnya.

```
import redis
improt random
from time import perf_counter_ns, time


class DowntimeTestClient:
    def __init__(self, client):
        self.client = client

        # num of keys prefilled
        self.prefilled = 0
        # percent of get above prefilled
        self.percent_get_above_prefilled = 10 # nil result expected when get hit above
 prefilled
        # total downtime in nano seconds
        self.downtime_ns = 0
        # num of success and fail operations
        self.success_ops = 0
        self.fail_ops = 0
        self.connection_errors = 0
        self.timeout_errors = 0


    def replace_client(self, client):
        self.client = client

    def prefill_data(self, timelimit_sec=60):
        end_time = time() + timelimit_sec
```

```
        while time() < end_time:
            self.client.set(self.prefilled, self.prefilled)
            self.prefilled += 1

    # unsuccesful operations throw exceptions
    def _exec(self, func):
        try:
            start_ns = perf_counter_ns()
            func()
            self.success_ops += 1
            elapsed_ms = (perf_counter_ns() - start_ns) // 10 ** 6
            # upon succesful execution of func
            # reset random_key to None so that the next command
            # will use a new random key
            self.random_key = None

        except Exception as e:
            elapsed_ns = perf_counter_ns() - start_ns
            self.downtime_ns += elapsed_ns
            # in case of failure- increment the relevant counters so that we will keep
track
            # of how many connection issues we had while trying to communicate with
            # the cluster.
            self.fail_ops += 1
            if e.__class__ is redis.exceptions.ConnectionError:
                self.connection_errors += 1
            if e.__class__ is redis.exceptions.TimeoutError:
                self.timeout_errors += 1

    def _repeat_exec(self, func, seconds):
        end_time = time() + seconds
        while time() < end_time:
            self._exec(func)

    def _new_random_key_if_needed(self, percent_above_prefilled):
        if self.random_key is None:
            max = int((self.prefilled * (100 + percent_above_prefilled)) / 100)
            return random.randint(0, max)
        return self.random_key

    def _random_get(self):
        key = self._new_random_key_if_needed(self.percent_get_above_prefilled)
        result = self.client.get(key)
        # we know the key was set for sure only in the case key < self.prefilled
```

```
        if key < self.prefilled:
            assert result.decode("UTF-8") == str(key)


    def repeat_get(self, seconds=60):
        self._repeat_exec(self._random_get, seconds)

    def get_downtime_ms(self) -> int:
        return self.downtime_ns // 10 ** 6


    def do_get_until(self, cond_check):
        while not cond_check():
            self.repeat_get()
        # do one more get cycle once condition is met
        self.repeat_get()
```

Membuat fungsi utama yang mendemonstrasikan proses pengubahan konfigurasi enkripsi bergerak

Sekarang, mari kita definisikan fungsi utama, yang akan melakukan hal berikut:

1. Buat cluster menggunakan klien boto3 ElastiCache .
2. Inisialisasi klien `redis-py-cluster` yang akan terhubung ke klaster dengan koneksi TCP yang jelas tanpa TLS.
3. klien `redis-py-cluster` mengisi klaster dengan beberapa data.
4. Klien boto3 akan memicu migrasi TLS dari tanpa TLS ke TLS diutamakan.
5. Sementara klaster sedang dimigrasikan ke TLS `Preferred`, klien `redis-py-cluster` TCP akan mengirim operasi `get` berulang ke klaster sampai migrasi selesai.
6. Setelah migrasi ke mode TLS `Preferred` selesai, kita akan memastikan bahwa klaster mendukung enkripsi bergerak. Setelah itu, kita akan membuat klien `redis-py-cluster` yang akan terhubung ke klaster dengan TLS.
7. Kita akan mengirim beberapa perintah `get` menggunakan klien TLS baru dan klien TCP lama.
8. Klien boto3 akan memicu migrasi TLS dari TLS `Preferred` ke TLS diperlukan.
9. Sementara cluster sedang dimigrasikan ke TLS diperlukan, klien redis-py-cluster TLS akan mengirim `get` operasi berulang ke cluster sampai migrasi selesai.

```
import redis
```

```
def init_cluster_client(
    cluster: ElastiCacheClusterCME, prefill_data: bool, TLS: bool = True) ->
 DowntimeTestClient:
    # we must use for the host name the cluster configuration endpoint.
    redis_client = redis.RedisCluster(
        host=cluster.get_configuration_endpoint(), ssl=TLS, socket_timeout=0.25,
 socket_connect_timeout=0.1
    )
    test_client = DowntimeTestClient(redis_client)
    if prefill_data:
        test_client.prefill_data()
    return test_client

if __name__ == '__main__':
    config = ConfigCME(TLS=False, instance_type="cache.m5.large")

    with ElastiCacheCMEManager(config) as cluster:
        # create a client that will connect to the cluster with clear tcp connection
        test_client_tcp = init_cluster_client(cluster, prefill_data=True, TLS=False)

       # migrate the cluster to TLS Preferred
        cluster.modify_transit_encryption_mode(new_transit_encryption_mode="preferred")

        # do repeated get commands until the cluster finishes the migration to TLS
 Preferred
        test_client_tcp.do_get_until(cluster.is_available)

      # verify that in transit encryption is enabled so that clients will be able to
 connect to the cluster with TLS
        assert cluster.get_transit_encryption_enabled() == True

       # create a client that will connect to the cluster with TLS connection.
        # we must first make sure that the cluster indeed supports TLS
        test_client_tls = init_cluster_client(cluster, prefill_data=True, TLS=True)

        # by doing get commands with the tcp client for 60 more seconds
       # we can verify that the existing tcp connection to the cluster still works
        test_client_tcp.repeat_get(seconds=60)

        # do get commands with the new TLS client for 60 more seconds
        test_client_tcp.repeat_get(seconds=60)

       # migrate the cluster to TLS required
        cluster.modify_transit_encryption_mode(new_transit_encryption_mode="required")
```

```
      # from this point the tcp clients will be disconnected and we must not use them
anymore.
      # do get commands with the TLS client until the cluster finishes migartion to
TLS required mode.
       test_client_tls.do_get_until(cluster.is_available)
```

Praktik terbaik saat mengaktifkan enkripsi bergerak

Sebelum mengaktifkan enkripsi bergerak: pastikan Anda memiliki penanganan catatan DNS yang tepat

> **Note**
>
> Kita akan mengubah dan menghapus titik akhir lama selama proses ini. Penggunaan titik akhir yang salah dapat mengakibatkan klien Valkey atau Redis OSS menggunakan titik akhir lama dan yang dihapus yang akan mencegahnya terhubung ke cluster.

Sementara cluster sedang dimigrasikan dari No-TLS ke TLS-preferen, catatan DNS titik akhir konfigurasi cluster lama disimpan dan catatan DNS titik akhir konfigurasi cluster baru sedang dihasilkan dalam format yang berbeda. Cluster yang mendukung TLS menggunakan format catatan DNS yang berbeda dari kluster yang dinonaktifkan TLS. ElastiCache akan menyimpan kedua catatan DNS ketika cluster dikonfigurasi `encryption mode: Preferred` sehingga Aplikasi dan klien Valkey atau Redis OSS lainnya dapat beralih di antara mereka. Perubahan dalam catatan DNS berikut terjadi selama proses migrasi TLS:

Deskripsi perubahan dalam catatan DNS yang terjadi saat mengaktifkan enkripsi bergerak

Untuk klaster CME

Ketika sebuah klaster diatur ke 'mode enkripsi bergerak: pilihan':

- Titik akhir konfigurasi cluster asli untuk cluster No-TLS akan tetap aktif. Tidak akan ada waktu henti ketika klaster dikonfigurasi ulang dari mode enkripsi TLS 'tidak ada' ke 'diutamakan'.
- Titik akhir TLS Valkey atau Redis OSS baru akan dihasilkan saat cluster diatur ke mode pilihan TLS. Titik akhir baru ini akan menyelesaikan IPs sama dengan yang lama (non-TLS).
- Titik akhir konfigurasi TLS Valkey atau Redis OSS yang baru akan diekspos di ElastiCache Konsol dan sebagai respons terhadap API. `describe-replication-group`

Ketika sebuah klaster diatur ke 'mode enkripsi bergerak: wajib':

- Titik akhir lama non-TLS akan dihapus. Tidak akan ada waktu henti pada titik akhir klaster TLS.
- Anda dapat mengambil yang baru `cluster-configuration-endpoint` dari ElastiCache Console atau dari `describe-replication-group` API.

Untuk klaster CMD dengan Failover Otomatis aktif atau Failover Otomatis nonaktif

Ketika grup replikasi diatur ke 'mode enkripsi bergerak: pilihan':

- Titik akhir primer asli dan titik akhir pembaca untuk klaster non-TLS akan tetap aktif.
- Titik akhir primer dan pembaca TLS baru akan dihasilkan saat klaster diatur ke mode TLS `Preferred`. Titik akhir baru ini akan diresolusi ke IP yang sama dengan yang lama (non-TLS).
- Titik akhir utama dan titik akhir pembaca yang baru akan diekspos di ElastiCache Konsol dan sebagai respons terhadap API. `describe-replication-group`

Ketika grup replikasi diatur ke 'mode enkripsi bergerak: wajib':

- Titik akhir primer dan pembaca non-TLS lama akan dihapus. Tidak akan ada waktu henti pada titik akhir klaster TLS.
- Anda dapat mengambil titik akhir primer dan pembaca baru dari ElastiCache Console atau dari API. `describe-replication-group`

Penggunaan catatan DNS yang disarankan

Untuk klaster CME

- Gunakan titik akhir konfigurasi klaster, bukan catatan DNS per simpul dalam kode aplikasi Anda. Menggunakan nama DNS per-node secara langsung tidak disarankan karena selama migrasi mereka akan berubah dan kode aplikasi akan memutuskan koneksi ke cluster.
- Jangan hardcode endpoint konfigurasi cluster di aplikasi Anda, karena akan berubah selama proses ini.
- Memiliki endpoint konfigurasi cluster yang di-hardcode dalam aplikasi Anda adalah praktik yang buruk, karena dapat diubah selama proses ini. Setelah enkripsi dalam transit selesai, kueri titik akhir konfigurasi cluster dengan `describe-replication-group` API (seperti yang ditunjukkan di atas (dalam huruf tebal)) dan gunakan DNS yang Anda dapatkan sebagai respons sejak saat itu.

Untuk klaster CMD dengan Failover Otomatis aktif

- Gunakan titik akhir primer dan titik akhir pembaca, bukan nama DNS per simpul dalam kode aplikasi Anda karena nama DNS per simpul yang lama dihapus dan yang baru akan dihasilkan saat memigrasikan klaster dari mode tanpa TLS ke mode TLS diutamakan. Penggunaan nama DNS per simpul secara langsung tidak direkomendasikan karena Anda mungkin akan menambahkan replika ke klaster Anda nanti. Selain itu, ketika Failover Otomatis diaktifkan, peran klaster utama dan replika diubah secara otomatis oleh ElastiCache layanan, menggunakan titik akhir utama dan titik akhir pembaca disarankan untuk membantu Anda melacak perubahan tersebut. Terakhir, menggunakan titik akhir pembaca akan membantu Anda mendistribusikan operasi baca Anda dari replika secara merata di antara replika di klaster.
- Memiliki titik akhir primer dan titik akhir pembaca yang di-hardcoding di aplikasi Anda adalah praktik yang buruk karena titik akhir tersebut dapat berubah selama proses migrasi TLS. Setelah perubahan migrasi ke pilihan TLS selesai, kueri titik akhir utama dan titik akhir pembaca dengan describe-replication-group API dan gunakan DNS yang Anda dapatkan sebagai tanggapan mulai saat ini. Dengan cara ini, Anda dapat melacak perubahan titik akhir secara dinamis.

Untuk klaster CMD dengan Failover Otomatis nonaktif

- Gunakan titik akhir primer dan titik akhir pembaca, bukan nama DNS per simpul dalam kode aplikasi Anda. Saat Failover Otomatis dinonaktifkan, penskalaan, penambalan, failover, dan prosedur lain yang dikelola secara otomatis oleh ElastiCache layanan saat Failover Otomatis diaktifkan akan dilakukan oleh Anda. Hal ini memudahkan Anda untuk melacak titik akhir yang berbeda-beda secara manual. Karena nama DNS per simpul yang lama dihapus dan yang baru akan dihasilkan saat memigrasikan klaster dari mode tanpa TLS ke mode TLS diutamakan, jangan gunakan nama DNS per simpul secara langsung. Hal ini wajib dilakukan agar klien dapat terhubung ke klaster selama migrasi TLS. Selain itu, Anda akan mendapatkan manfaat dengan menyebarkan permintaan baca secara merata di antara replika saat menggunakan titik akhir pembaca dan melacak catatan DNS saat menambahkan atau menghapus replika dari klaster.
- Memiliki titik akhir konfigurasi klaster yang di-hardcoding di aplikasi Anda adalah praktik yang buruk karena titik akhir tersebut dapat berubah selama proses migrasi TLS.

Selama enkripsi bergerak: perhatikan kapan proses migrasi selesai

Perubahan mode enkripsi bergerak tidak diterapkan segera dan dapat memakan waktu. Hal ini terutama berlaku untuk klaster besar. Hanya setelah menyelesaikan migrasi ke mode TLS diutamakan, klaster ini dapat menerima dan melayani koneksi TCP dan TLS. Oleh karena itu, Anda

sebaiknya tidak membuat klien yang akan mencoba membuat koneksi TLS ke klaster sampai enkripsi bergerak selesai.

Ada beberapa cara untuk mendapatkan notifikasi ketika enkripsi bergerak berhasil diselesaikan atau gagal: (Tidak ditampilkan dalam contoh kode di atas):

- Menggunakan layanan SNS untuk mendapatkan notifikasi saat enkripsi selesai
- Menggunakan API `describe-events` yang akan memublikasikan peristiwa saat enkripsi selesai
- Melihat pesan di ElastiCache Konsol bahwa enkripsi selesai

Anda juga dapat menerapkan logika dalam aplikasi Anda untuk mengetahui apakah enkripsi selesai. Pada contoh di atas, kita melihat beberapa cara untuk memastikan klaster menyelesaikan migrasi:

- Menunggu hingga proses migrasi dimulai (status klaster berubah menjadi "mengubah"), dan menunggu hingga perubahan selesai (status klaster berubah kembali ke "tersedia")
- Memastikan bahwa klaster telah mengatur `transit_encryption_enabled` ke True dengan mengueri API `describe-replication-group`.

Setelah mengaktifkan enkripsi bergerak: pastikan klien yang Anda gunakan dikonfigurasi dengan benar

Saat klaster berada dalam mode TLS diutamakan, aplikasi Anda harus membuka koneksi TLS ke klaster dan hanya menggunakan koneksi tersebut. Dengan begitu, aplikasi Anda tidak akan mengalami waktu henti saat sedang mengaktifkan enkripsi bergerak. Anda dapat memastikan bahwa tidak ada koneksi TCP yang lebih jelas ke mesin Valkey atau Redis OSS menggunakan perintah info di bawah bagian SSL.

```
# SSL
ssl_enabled:yes
ssl_current_certificate_not_before_date:Mar 20 23:27:07 2017 GMT
ssl_current_certificate_not_after_date:Feb 24 23:27:07 2117 GMT
ssl_current_certificate_serial:D8C7DEA91E684163
tls_mode_connected_tcp_clients:0   (should be zero)
tls_mode_connected_tls_clients:100
```

Menghubungkan ke node diaktifkan dengan enkripsi dalam transit menggunakan Openssl (Memcached)

Untuk mengakses data dari ElastiCache node Memcached yang diaktifkan dengan enkripsi dalam transit, Anda perlu menggunakan klien yang bekerja dengan Secure Socket Layer (SSL). Anda juga dapat menggunakan s_client Openssl di Amazon Linux dan Amazon Linux 2.

Untuk menggunakan s_client Openssl untuk terhubung ke klaster Memcached dengan enkripsi bergerak diaktifkan di Amazon Linux 2 atau Amazon Linux:

```
/usr/bin/openssl s_client -connect memcached-node-endpoint:memcached-port
```

Membuat klien Memcached TLS menggunakan Java

Untuk membuat klien dalam mode TLS, lakukan hal berikut untuk menginisialisasi klien dengan yang sesuai: SSLContext

```
import java.security.KeyStore;
import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManagerFactory;
import net.spy.memcached.AddrUtil;
import net.spy.memcached.ConnectionFactoryBuilder;
import net.spy.memcached.MemcachedClient;
public class TLSDemo {
    public static void main(String[] args) throws Exception {
        ConnectionFactoryBuilder connectionFactoryBuilder = new
 ConnectionFactoryBuilder();
        // Build SSLContext
        TrustManagerFactory tmf =
 TrustManagerFactory.getInstance(TrustManagerFactory.getDefaultAlgorithm());
        tmf.init((KeyStore) null);
        SSLContext sslContext = SSLContext.getInstance("TLS");
        sslContext.init(null, tmf.getTrustManagers(), null);
        // Create the client in TLS mode
        connectionFactoryBuilder.setSSLContext(sslContext);
        MemcachedClient client = new MemcachedClient(connectionFactoryBuilder.build(),
 AddrUtil.getAddresses("mycluster.fnjyzo.cfg.use1.cache.amazonaws.com:11211"));

        // Store a data item for an hour.
        client.set("theKey", 3600, "This is the data value");
    }
}
```

Membuat klien Memcached TLS menggunakan PHP

Untuk membuat klien dalam mode TLS, lakukan hal berikut untuk menginisialisasi klien dengan yang sesuai: SSLContext

```
<?php

/**
 * Sample PHP code to show how to create a TLS Memcached client. In this example we
 * will use the Amazon ElastiCache Auto Descovery feature, but TLS can also be
 * used with a Static mode client.
 * See Using the ElastiCache Cluster Client for PHP (https://docs.aws.amazon.com/
AmazonElastiCache/latest/dg/AutoDiscovery.Using.ModifyApp.PHP.html) for more
 information
 * about Auto Discovery and persistent-id.
 */

/* Configuration endpoint to use to initialize memcached client.
 * this is only an example */
$server_endpoint = "mycluster.fnjyzo.cfg.use1.cache.amazonaws.com";

/* Port for connecting to the cluster.
 * This is only an example     */
$server_port = 11211;

/* Initialize a persistent Memcached client and configure it with the Dynamic client
 mode  */
$tls_client =  new Memcached('persistent-id');
$tls_client->setOption(Memcached::OPT_CLIENT_MODE, Memcached::DYNAMIC_CLIENT_MODE);

/* Add the memcached's cluster server/s */
$tls_client->addServer($server_endpoint, $server_port);

/* Configure the client to use TLS */
if(!$tls_client->setOption(Memcached::OPT_USE_TLS, 1)) {
    echo $tls_client->getLastErrorMessage(), "\n";
    exit(1);
}

/* Set your TLS context configurations values.
 * See MemcachedTLSContextConfig in memcached-api.php for all configurations */
$tls_config = new MemcachedTLSContextConfig();
$tls_config->hostname = '*.mycluster.fnjyzo.use1.cache.amazonaws.com';
$tls_config->skip_cert_verify = false;
```

```
$tls_config->skip_hostname_verify = false;

/* Use the created TLS context configuration object to create OpenSSL's SSL_CTX and set
 it to your client.
 * Note:  These TLS context configurations will be applied to all the servers connected
 to this client. */
$tls_client->createAndSetTLSContext((array)$tls_config);

/* test the TLS connection with set-get scenario: */

 /* store the data for 60 seconds in the cluster.
 * The client will decide which cache host will store this item.
 */
if($tls_client->set('key', 'value', 60)) {
    print "Successfully stored key\n";
} else {
    echo "Failed to set key: ", $tls_client->getLastErrorMessage(), "\n";
    exit(1);
}

/* retrieve the key */
if ($tls_client->get('key') === 'value') {
    print "Successfully retrieved key\n";
} else {
    echo "Failed to get key: ", $tls_client->getLastErrorMessage(), "\n";
    exit(1);
}
```

Untuk informasi selengkapnya tentang menggunakan klien PHP, lihat Menginstal klien ElastiCache cluster untuk PHP.

Enkripsi At-Rest di ElastiCache

Untuk membantu menjaga keamanan data Anda, Amazon ElastiCache dan Amazon S3 menyediakan berbagai cara untuk membatasi akses ke data di cache Anda. Untuk informasi selengkapnya, lihat Amazon VPCs dan ElastiCache keamanan dan Identity and Access Management untuk Amazon ElastiCache.

ElastiCache enkripsi at-rest adalah fitur untuk meningkatkan keamanan data dengan mengenkripsi data on-disk. Fitur ini selalu diaktifkan di cache nirserver. Saat diaktifkan, fitur ini mengenkripsi aspek-aspek berikut:

- Disk selama operasi sinkronisasi, pencadangan, dan swap
- Cadangan yang disimpan di Amazon S3

Data yang disimpan pada SSDs (solid-state drive) dalam cluster yang diaktifkan tiering data selalu dienkripsi.

ElastiCache menawarkan enkripsi default (dikelola layanan) saat istirahat, serta kemampuan untuk menggunakan kunci KMS yang dikelola pelanggan simetris Anda sendiri di Layanan Manajemen AWS Kunci (AWS KMS). Saat cache dicadangkan, di bagian opsi enkripsi, pilih apakah akan menggunakan kunci enkripsi default atau kunci yang dikelola pelanggan. Untuk informasi selengkapnya, lihat Mengaktifkan Enkripsi Diam.

> **Important**
>
> Mengaktifkan Enkripsi At-Rest pada cluster Valkey atau Redis OSS yang dirancang sendiri melibatkan penghapusan grup replikasi Anda yang ada, setelah menjalankan pencadangan dan pemulihan pada grup replikasi.

Enkripsi diam dapat diaktifkan di cache hanya pada saat pembuatannya. Karena diperlukan beberapa pemrosesan untuk mengenkripsi dan mendekripsi data, mengaktifkan enkripsi diam dapat berdampak pada performa selama operasi ini. Anda harus membandingkan data Anda menggunakan dan tidak menggunakan enkripsi diam untuk menentukan dampaknya terhadap performa untuk kasus penggunaan Anda.

Topik

- Kondisi Enkripsi Diam

- Menggunakan kunci yang dikelola pelanggan dari AWS KMS
- Mengaktifkan Enkripsi Diam
- Lihat Juga

Kondisi Enkripsi Diam

Kendala berikut pada enkripsi ElastiCache saat istirahat harus diingat ketika Anda merencanakan implementasi enkripsi saat istirahat: ElastiCache

- Enkripsi AT-rest didukung pada grup replikasi yang menjalankan Valkey 7.2 dan yang lebih baru, dan versi Redis OSS (3.2.6 dijadwalkan untuk EOL, lihat jadwal akhir masa pakai versi Redis OSS), 4.0.10 atau yang lebih baru.
- Enkripsi diam didukung hanya untuk grup replikasi yang berjalan di Amazon VPC.
- Enkripsi diam hanya didukung untuk grup replikasi yang menjalankan jenis simpul berikut.
  - R7g, R6gd, R6g, R5, R4, R3
  - M7g, M6g, M5, M4, M3
  - T4g, T3, T2
  - C7gN

  Untuk informasi selengkapnya, lihat Jenis simpul yang didukung
- Enkripsi diam diaktifkan dengan menetapkan parameter `AtRestEncryptionEnabled` ke `true` secara eksplisit.
- Anda dapat mengaktifkan enkripsi diam pada grup replikasi hanya saat membuat grup replikasi. Anda tidak dapat mengaktifkan dan menonaktifkan enkripsi diam dengan mengubah grup replikasi. Untuk informasi tentang cara menerapkan enkripsi diam pada grup replikasi yang sudah ada, lihat Mengaktifkan Enkripsi Diam.
- Jika klaster menggunakan jenis simpul dari keluarga r6gd, data yang disimpan di SSD dienkripsi baik apakah enkripsi diam diaktifkan atau tidak.
- Opsi untuk menggunakan kunci terkelola pelanggan untuk enkripsi saat istirahat tidak tersedia di AWS GovCloud (us-gov-east-1 dan us-gov-west -1) wilayah.
- Jika cluster menggunakan tipe node dari keluarga r6gd, data yang disimpan di SSD dienkripsi dengan kunci AWS KMS terkelola pelanggan yang dipilih (atau enkripsi yang dikelola layanan di Wilayah). AWS GovCloud

- Dengan Memcached, enkripsi saat istirahat hanya didukung pada cache tanpa server.
- Saat menggunakan Memcached, opsi untuk menggunakan kunci terkelola pelanggan untuk enkripsi saat istirahat tidak tersedia di AWS GovCloud (us-gov-east-1 dan us-gov-west -1) wilayah.

Menerapkan enkripsi diam dapat menurunkan performa selama operasi pencadangan dan sinkronisasi simpul. Lakukan tolok ukur enkripsi diam dibandingkan dengan tanpa enkripsi pada data Anda sendiri untuk menentukan dampaknya terhadap performa untuk implementasi Anda.

Menggunakan kunci yang dikelola pelanggan dari AWS KMS

ElastiCache mendukung kunci AWS KMS yang dikelola pelanggan simetris (kunci KMS) untuk enkripsi saat istirahat. Kunci KMS yang dikelola pelanggan adalah kunci enkripsi yang Anda buat, miliki, dan kelola di akun Anda. AWS Untuk informasi selengkapnya, lihat Kunci AWS KMS dalam Panduan Developer untuk AWS Key Management Service. Kunci harus dibuat di AWS KMS sebelum dapat digunakan. ElastiCache

Untuk mempelajari cara membuat kunci root AWS KMS, lihat Membuat Kunci di Panduan Pengembang Layanan Manajemen AWS Kunci.

ElastiCache memungkinkan Anda untuk berintegrasi dengan AWS KMS. Untuk informasi selengkapnya, lihat Menggunakan Grant dalam Panduan Developer AWS Key Management Service. Tidak diperlukan tindakan pelanggan untuk mengaktifkan ElastiCache integrasi Amazon dengan AWS KMS.

Kunci `kms:ViaService` kondisi membatasi penggunaan kunci AWS KMS (kunci KMS) untuk permintaan dari layanan tertentu AWS . Untuk digunakan `kms:ViaService` dengan ElastiCache, sertakan kedua ViaService nama dalam nilai kunci kondisi: `elasticache.AWS_region.amazonaws.com` dan`dax.AWS_region.amazonaws.com`. Untuk informasi lebih lanjut, lihat kms: ViaService.

Anda dapat menggunakan AWS CloudTrailuntuk melacak permintaan yang ElastiCache dikirimkan AWS Key Management Service Amazon atas nama Anda. Semua panggilan API yang AWS Key Management Service terkait dengan kunci yang dikelola pelanggan memiliki CloudTrail log yang sesuai. Anda juga dapat melihat hibah yang ElastiCache dibuat dengan memanggil panggilan API ListGrantsKMS.

Setelah grup replikasi dienkripsi menggunakan kunci yang dikelola pelanggan, semua cadangan untuk grup replikasi akan dienkripsi sebagai berikut:

- Cadangan harian otomatis dienkripsi menggunakan kunci yang dikelola pelanggan yang terkait dengan klaster.
- Cadangan akhir yang dibuat saat grup replikasi dihapus, juga dienkripsi menggunakan kunci yang dikelola pelanggan yang terkait dengan grup replikasi.
- Cadangan yang dibuat secara manual dienkripsi secara default untuk menggunakan kunci KMS yang terkait dengan grup replikasi. Anda dapat menggantinya dengan memilih kunci dikelola pelanggan yang lain.
- Jika cadangan disalin, kunci yang dikelola pelanggan yang terkait dengan cadangan sumber akan secara default digunakan. Anda dapat menggantinya dengan memilih kunci dikelola pelanggan yang lain.

> **Note**
>
> - Kunci yang dikelola pelanggan tidak dapat digunakan saat mengekspor cadangan ke bucket Amazon S3 pilihan Anda. Namun, semua cadangan yang diekspor ke Amazon S3 akan dienkripsi menggunakan Enkripsi sisi server. Anda dapat memilih untuk menyalin file cadangan ke objek S3 baru dan mengenkripsi menggunakan kunci KMS yang dikelola pelanggan, menyalin file ke bucket S3 lain yang diatur dengan enkripsi default menggunakan kunci KMS atau mengubah opsi enkripsi dalam file itu sendiri.
> - Anda juga dapat menggunakan kunci yang dikelola pelanggan untuk mengenkripsi cadangan yang dibuat secara manual untuk grup replikasi yang tidak menggunakan kunci yang dikelola pelanggan untuk enkripsi. Dengan opsi ini, file cadangan yang disimpan di Amazon S3 akan dienkripsi menggunakan kunci KMS, meskipun data tersebut tidak dienkripsi pada grup replikasi yang asli.
>
> Memulihkan dari cadangan memungkinkan Anda memilih opsi enkripsi yang tersedia, mirip dengan pilihan enkripsi yang tersedia saat membuat grup replikasi baru.

- Jika Anda menghapus kunci atau menonaktifkan kunci dan mencabut grant untuk kunci yang digunakan untuk mengenkripsi cache, cache menjadi tidak dapat dipulihkan. Dengan kata lain, itu tidak dapat dimodifikasi atau dipulihkan setelah kegagalan perangkat keras. AWS KMS menghapus kunci root hanya setelah masa tunggu setidaknya tujuh hari. Setelah kunci dihapus, Anda dapat menggunakan kunci yang dikelola pelanggan yang berbeda untuk membuat cadangan untuk tujuan pengarsipan.

- Rotasi kunci otomatis mempertahankan properti kunci root AWS KMS Anda, sehingga rotasi tidak berpengaruh pada kemampuan Anda untuk mengakses data Anda ElastiCache . ElastiCache Cache Amazon terenkripsi tidak mendukung rotasi kunci manual, yang melibatkan pembuatan kunci root baru dan memperbarui referensi apa pun ke kunci lama. Untuk mempelajari selengkapnya, lihat Memutar kunci AWS KMS di Panduan Pengembang Layanan Manajemen AWS Kunci.
- Mengenkripsi ElastiCache cache menggunakan kunci KMS memerlukan satu hibah per cache. Grant ini digunakan sepanjang masa pakai cache. Selain itu, satu grant per cadangan digunakan selama pembuatan cadangan. Grant ini dipensiunkan setelah cadangan dibuat.
- Untuk informasi selengkapnya tentang AWS hibah dan batasan KMS, lihat Batas dalam Panduan Pengembang Layanan Manajemen AWS Utama.

Mengaktifkan Enkripsi Diam

Semua cache nirserver memiliki enkripsi diam yang aktif.

Saat membuat klaster yang dirancang sendiri, Anda dapat mengaktifkan enkripsi diam dengan mengatur parameter `AtRestEncryptionEnabled` ke `true`. Anda tidak dapat mengaktifkan enkripsi diam di grup replikasi yang ada.

Anda dapat mengaktifkan enkripsi saat Anda membuat ElastiCache cache. Anda dapat melakukannya dengan menggunakan AWS Management Console, the AWS CLI, atau ElastiCache API.

Saat membuat cache, Anda dapat memilih salah satu opsi berikut:

- Default – Opsi ini menggunakan enkripsi diam yang dikelola layanan.
- Kunci terkelola pelanggan - Opsi ini memungkinkan Anda untuk memberikan Kunci ID/ARN dari AWS KMS untuk enkripsi saat istirahat.

Untuk mempelajari cara membuat kunci root AWS KMS, lihat Membuat Kunci di Panduan Pengembang Layanan Manajemen AWS Kunci

Daftar Isi

Mengaktifkan Enkripsi At-Rest pada Cluster Valkey atau Redis OSS yang Dirancang Sendiri

Anda hanya dapat mengaktifkan enkripsi saat Anda membuat grup replikasi Valkey atau Redis OSS. Jika Anda memiliki grup replikasi yang ada tempat Anda ingin mengaktifkan enkripsi diam, lakukan hal berikut.

Untuk mengaktifkan enkripsi diam pada grup replikasi yang ada

1. Buat cadangan manual dari grup replikasi yang ada. Untuk informasi selengkapnya, lihat Membuat cadangan manual.
2. Buat grup replikasi baru dengan memulihkan dari cadangan. Pada grup replikasi baru, aktifkan enkripsi diam. Untuk informasi selengkapnya, lihat Melakukan pemulihan dari cadangan ke dalam cache baru.
3. Perbarui titik akhir dalam aplikasi Anda untuk mengarah ke grup replikasi baru.
4. Hapus grup replikasi lama. Untuk informasi selengkapnya, lihat Menghapus cluster di ElastiCache atau Menghapus grup replikasi.

Mengaktifkan Enkripsi At-Rest Menggunakan AWS Management Console

Mengaktifkan Enkripsi Diam di Klaster Nirserver (Konsol)

Semua cache nirserver memiliki enkripsi diam yang aktif. Secara default, kunci KMS yang AWS dimiliki digunakan untuk mengenkripsi data. Untuk memilih AWS KMS kunci Anda sendiri, buat pilihan berikut:

- Perluas bagian Pengaturan default.
- Pilih Sesuaikan pengaturan default di bagian Pengaturan default.
- Pilih Sesuaikan pengaturan keamanan Anda di bagian Keamanan.
- Pilih CMK yang dikelola pelanggan di bagian pengaturan Kunci enkripsi.
- Pilih kunci di bagian pengaturan Kunci AWS KMS .

Mengaktifkan Enkripsi Diam di Klaster yang Dirancang Sendiri (Konsol)

Saat merancang cache Anda sendiri, konfigurasi 'Dev/Test' dan 'Produksi' dengan metode 'Mudah dibuat' akan menjadikan enkripsi diam aktif menggunakan kunci Default. Saat memilih konfigurasi sendiri, buat pilihan berikut:

- Pilih versi 3.2.6, 4.0.10 atau yang lebih baru sebagai versi mesin Anda.

- Klik kotak centang di sebelah Aktifkan untuk opsi Enkripsi diam.
- Pilih Kunci default atau CMK yang dikelola pelanggan.

Untuk step-by-step prosedurnya, lihat yang berikut ini:

- Membuat cluster Valkey (mode cluster dinonaktifkan) (Konsol)
- Membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) (Konsol)

Mengaktifkan Enkripsi At-Rest Menggunakan AWS CLI

Untuk mengaktifkan enkripsi saat membuat klaster Valkey atau Redis OSS menggunakan AWS CLI, gunakan at-rest-encryption-enabled parameter -- saat membuat grup replikasi.

Mengaktifkan Enkripsi At-Rest pada Cluster (CLI) Valkey atau Redis OSS (Mode Cluster Dinonaktifkan)

Operasi berikut membuat grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) `my-classic-rg` dengan tiga node (-- num-cache-clusters), replika utama dan dua baca. Enkripsi AT-rest diaktifkan untuk grup replikasi ini (-- at-rest-encryption-enabled).

Parameter berikut dan nilainya diperlukan untuk mengaktifkan enkripsi pada grup replikasi ini:

Parameter Kunci

- **`--engine`**—Harus `valkey` atau`redis`.
- **`--engine-version`**—Jika mesinnya Redis OSS, ini harus 3.2.6, 4.0.10 atau yang lebih baru.
- **`--at-rest-encryption-enabled`**—Wajib untuk mengaktifkan enkripsi diam.

Example 1: Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) Cluster dengan Replika

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
    --replication-group-id my-classic-rg \
    --replication-group-description "3 node replication group" \
    --cache-node-type cache.m4.large \
    --engine redis \
    --at-rest-encryption-enabled \
    --num-cache-clusters 3
```

Untuk Windows:

```
aws elasticache create-replication-group ^
    --replication-group-id my-classic-rg ^
    --replication-group-description "3 node replication group" ^
    --cache-node-type cache.m4.large ^
    --engine redis ^
    --at-rest-encryption-enabled ^
    --num-cache-clusters 3 ^
```

Untuk informasi tambahan, lihat hal berikut:

- Membuat grup replikasi Valkey atau Redis OSS (Mode Cluster Dinonaktifkan) dari awal ()AWS CLI
- create-replication-group

Mengaktifkan Enkripsi At-Rest pada Cluster untuk Valkey atau Redis OSS (Mode Cluster Diaktifkan) (CLI)

Operasi berikut membuat grup replikasi Valkey atau Redis OSS (mode cluster enabled) *my-clustered-rg* dengan tiga grup node atau pecahan (--). num-node-groups Masing-masing memiliki tiga node, primer dan dua replika baca (-- replicas-per-node-group). Enkripsi AT-rest diaktifkan untuk grup replikasi ini (-- at-rest-encryption-enabled).

Parameter berikut dan nilainya diperlukan untuk mengaktifkan enkripsi pada grup replikasi ini:

Parameter Kunci

- **`--engine`**—Harus `valkey` atau`redis`.
- **`--engine-version`**—Jika mesinnya Redis OSS, ini harus 4.0.10 atau lebih baru.
- **`--at-rest-encryption-enabled`**—Wajib untuk mengaktifkan enkripsi diam.
- **`--cache-parameter-group`**—Harus `default-redis4.0.cluster.on` atau turunannya untuk membuat grup replikasi dengan mode klaster diaktifkan.

Example 2: Cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan)

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
   --replication-group-id my-clustered-rg \
   --replication-group-description "redis clustered cluster" \
   --cache-node-type cache.m3.large \
   --num-node-groups 3 \
   --replicas-per-node-group 2 \
   --engine redis \
   --engine-version 6.2 \
   --at-rest-encryption-enabled \
   --cache-parameter-group default.redis6.x.cluster.on
```

Untuk Windows:

```
aws elasticache create-replication-group ^
   --replication-group-id my-clustered-rg ^
   --replication-group-description "redis clustered cluster" ^
   --cache-node-type cache.m3.large ^
   --num-node-groups 3 ^
   --replicas-per-node-group 2 ^
   --engine redis ^
   --engine-version 6.2 ^
   --at-rest-encryption-enabled ^
   --cache-parameter-group default.redis6.x.cluster.on
```

Untuk informasi tambahan, lihat hal berikut:

- Membuat grup replikasi Valkey atau Redis OSS (Cluster Mode Enabled) dari awal ()AWS CLI
- create-replication-group

Lihat Juga

- Amazon VPCs dan ElastiCache keamanan
- Identity and Access Management untuk Amazon ElastiCache

Autentikasi dan Otorisasi

AWS Identity and Access Management (IAM) adalah layanan web yang membantu Anda mengontrol akses ke sumber daya dengan aman. AWS ElastiCache mendukung otentikasi pengguna

menggunakan IAM dan perintah Valkey dan Redis OSS AUTH, dan mengotorisasi operasi pengguna menggunakan Role-Based Access Control (RBAC).

Topik

Kontrol Akses Berbasis Peran (RBAC)

Dengan perintah Valkey dan Redis OSS AUTH seperti yang dijelaskan dalam Mengautentikasi dengan perintah Valkey dan Redis OSS AUTH Anda dapat menggunakan Role-Based Access Control (RBAC). RBAC juga merupakan satu-satunya cara untuk mengontrol akses ke cache nirserver. Ini tersedia untuk Valkey 7.2 dan seterusnya, dan Redis OSS 6.0 hingga 7.2.

RBAC memungkinkan Anda untuk:

- Kontrol akses cache melalui grup pengguna. Grup pengguna ini dirancang sebagai cara untuk mengatur akses ke cache.
- Dengan AuthN, miliki kata sandi per pengguna sebagai lawan dari token autentikasi per cluster.
- Dengan AuthZ, miliki izin pengguna yang berbutir halus.
- Dasarkan akses klaster Anda ACLs.

Tidak seperti Valkey dan Redis OSS AUTH, di mana semua klien yang diautentikasi memiliki akses cache penuh jika token mereka diautentikasi, RBAC memungkinkan Anda untuk menetapkan pengguna ke set tergantung pada peran yang diinginkan pengguna. Set ini dirancang sebagai cara untuk mengatur akses ke cache.

Dengan RBAC, Anda membuat pengguna dan memberi mereka izin tertentu menggunakan string akses, seperti yang dijelaskan berikut. Anda menetapkan pengguna untuk menetapkan selaras dengan peran tertentu (administrator, sumber daya manusia) yang kemudian disebarkan ke satu atau beberapa cache. ElastiCache Dengan melakukan ini, Anda dapat menetapkan batas keamanan antara klien menggunakan cache atau cache Valkey atau Redis OSS yang sama, dan mencegah klien mengakses data satu sama lain.

RBAC dirancang untuk mendukung pengenalan ACL di Redis OSS 6. Saat Anda menggunakan RBAC dengan cache ElastiCache Valkey atau Redis OSS Anda, ada beberapa batasan:

- Grup pengguna yang dikonfigurasi untuk mesin “VALKEY” hanya dapat berisi pengguna yang menggunakan mekanisme otentikasi (baik kata sandi atau IAM). Ini berarti semua pengguna dengan mesin “VALKEY”, dan pengguna lain dengan mesin “Redis” yang memiliki pengaturan mereka dikonfigurasi untuk mengautentikasi dengan kata sandi atau IAM, dapat berada di grup pengguna ini.
- Saat menggunakan RBAC dengan cluster Valkey, kedua grup pengguna dengan mesin “VALKEY” dan dengan mesin “REDIS” dapat digunakan.
- Saat menggunakan RBAC dengan cluster Redis OSS, hanya grup pengguna dengan mesin “REDIS” yang dapat digunakan.
- Anda tidak dapat menentukan kata sandi dalam string akses. Anda mengatur kata sandi dengan CreateUseratau ModifyUserpanggilan.
- Untuk hak pengguna, Anda mengaktifkan atau menonaktifkan pengguna dengan on dan `off` sebagai bagian dari string akses. Jika salah satu nilai tersebut tidak ditentukan dalam string akses, pengguna ditetapkan ke `off` dan tidak memiliki hak akses ke cache.
- Anda tidak dapat menggunakan perintah terlarang dan berganti nama sebagai bagian dari string akses. Jika Anda menentukan perintah terlarang atau yang diubah namanya, pengecualian akan terjadi. Jika Anda ingin menggunakan daftar kontrol akses (ACLs) untuk perintah yang diganti namanya, tentukan nama asli perintah tersebut, dengan kata lain nama perintah sebelum diganti namanya.
- Anda tidak dapat menggunakan perintah `reset` sebagai bagian dari string akses. Anda menentukan kata sandi dengan parameter API, dan ElastiCache untuk Valkey dan Redis OSS mengelola kata sandi. Dengan demikian, Anda tidak dapat menggunakan `reset` karena akan menghapus semua kata sandi untuk pengguna.
- Redis OSS 6 memperkenalkan perintah ACL LIST. Perintah ini mengembalikan daftar pengguna bersama dengan aturan ACL yang diterapkan untuk setiap pengguna. ElastiCache mendukung `ACL LIST` perintah, tetapi tidak menyertakan dukungan untuk hash kata sandi seperti yang dilakukan Redis OSS. Dengan ElastiCache, Anda dapat menggunakan DescribeUsersoperasi untuk mendapatkan informasi serupa, termasuk aturan yang terkandung dalam string akses. Namun, DescribeUserstidak mengambil kata sandi pengguna.
- Perintah read-only lainnya yang didukung oleh ElastiCache untuk Valkey dan Redis OSS termasuk ACL WHOAMI, ACL USERS, dan ACL CAT. ElastiCache untuk Valkey dan Redis OSS tidak mendukung perintah ACL berbasis penulisan lainnya.
- Batasan berikut berlaku:

Sumber Daya	Maksimum yang diizinkan
Pengguna per grup pengguna	100
Jumlah pengguna	1000
Jumlah grup pengguna	100

RBAC dengan Valkey

Saat menggunakan Kontrol Akses Berbasis Peran dengan Valkey, pengguna dan grup pengguna dibuat dengan tipe mesin “VALKEY”. Ini direkomendasikan, karena secara default Valkey dengan RBAC memberikan peningkatan keamanan dibandingkan dengan Redis OSS. Kluster Valkey yang disediakan dan tanpa server mendukung asosiasi pengguna dan grup pengguna VALKEY.

Fitur utama dari Valkey Access Control meliputi:

- Pengguna Valkey dibatasi hanya untuk asosiasi grup pengguna Valkey.
- Grup pengguna Valkey dapat berisi pengguna Valkey, dan pengguna Redis OSS yang dilindungi kata sandi atau autentikasi IAM diaktifkan.
- Pengguna Valkey harus menggunakan perlindungan kata sandi atau otentikasi IAM.
- Grup pengguna VALKEY hanya dapat dikaitkan dengan cluster cache VALKEY
- Tidak ada persyaratan pengguna default. Ketika grup pengguna Valkey dilampirkan ke cluster cache, persyaratan pengguna default secara otomatis dinonaktifkan. Pelanggan akan melihat bahwa pengguna default dimatikan saat menggunakan perintah ACL LIST.

Informasi lebih lanjut tentang penggunaan RBAC dengan ElastiCache untuk Valkey dan Redis OSS berikut.

Topik

- Menentukan Izin Menggunakan String Akses
- Menerapkan RBAC ke Cache ElastiCache untuk Valkey atau Redis OSS
- Migrasi dari AUTH ke RBAC
- Migrasi dari RBAC ke AUTH
- Merotasi kata sandi untuk pengguna secara otomatis

- Autentikasi dengan IAM

Menentukan Izin Menggunakan String Akses

Untuk menentukan izin ke cache ElastiCache Valkey atau Redis OSS, Anda membuat string akses dan menetapkannya ke pengguna melalui atau. AWS CLI AWS Management Console

String akses didefinisikan sebagai daftar aturan yang dipisahkan spasi yang diterapkan pada pengguna. String akses menentukan perintah yang dapat dijalankan oleh pengguna dan kunci yang dapat dioperasikan oleh pengguna. Agar dapat menjalankan perintah, pengguna harus memiliki akses ke perintah yang dijalankan dan semua kunci yang diakses oleh perintah tersebut. Aturan diterapkan dari kiri ke kanan secara kumulatif, dan string yang lebih sederhana dapat digunakan sebagai pengganti yang disediakan jika ada kelebihan dalam string yang disediakan.

Untuk informasi tentang sintaksis aturan ACL, lihat ACL.

Pada contoh berikut, string akses merepresentasikan pengguna aktif dengan akses ke semua kunci dan perintah yang tersedia.

```
on ~* +@all
```

Sintaksis string akses diuraikan sebagai berikut:

- `on` – Pengguna adalah pengguna yang aktif.
- `~*` – Akses diberikan ke semua kunci yang tersedia.
- `+@all` – Akses diberikan ke semua perintah yang tersedia.

Pengaturan di atas adalah pengaturan yang tidak terlalu membatasi. Anda dapat mengubah pengaturan ini untuk membuatnya lebih aman.

Pada contoh berikut, string akses merepresentasikan pengguna dengan akses yang dibatasi untuk akses baca pada kunci yang diawali dengan ruang kunci "app::"

```
on ~app::* -@all +@read
```

Anda dapat mempersempit izin ini lebih lanjut dengan menampilkan daftar perintah yang dapat diakses pengguna:

`+`*`command1`* – Akses pengguna ke perintah dibatasi pada *`command1`*.

`+@category` – Akses pengguna dibatasi pada kategori perintah.

Untuk informasi tentang cara menetapkan string akses ke pengguna, lihat Membuat Pengguna dan Grup Pengguna dengan Konsol dan CLI.

Jika Anda memigrasikan beban kerja yang ada ke ElastiCache, Anda dapat mengambil string akses dengan meneleponACL LIST, tidak termasuk pengguna dan hash kata sandi apa pun.

Untuk Redis OSS versi 6.2 dan di atas sintaks string akses berikut juga didukung:

- `&*` – Akses diberikan ke semua saluran yang tersedia.

Untuk Redis OSS versi 7.0 dan di atas sintaks string akses berikut juga didukung:

- `|` – Dapat digunakan untuk memblokir subperintah (misalnya "-config|set").
- `%R~<pattern>` – Menambahkan pola kunci baca yang ditentukan. Sintaksis ini berperilaku mirip dengan pola kunci biasa, tetapi hanya memberikan izin untuk membaca dari kunci yang cocok dengan pola yang diberikan. Lihat izin kunci untuk informasi selengkapnya.
- `%W~<pattern>` – Menambahkan pola kunci tulis yang ditentukan. Sintaksis ini berperilaku mirip dengan pola kunci biasa, tetapi hanya memberikan izin untuk menulis dari kunci yang cocok dengan pola yang diberikan. Lihat izin kunci ACL untuk informasi selengkapnya.
- `%RW~<pattern>` – Alias untuk `~<pattern>`.
- `(<rule list>)` – Membuat pemilih baru yang akan digunakan untuk mencocokkan aturan. Pemilih dievaluasi setelah izin pengguna, dan dievaluasi sesuai dengan urutan penentuannya. Perintah akan diizinkan jika cocok dengan izin pengguna atau pemilih apa pun. Lihat pemilih ACL untuk informasi selengkapnya.
- `clearselectors` – Menghapus semua pemilih yang dilampirkan pada pengguna.

Menerapkan RBAC ke Cache ElastiCache untuk Valkey atau Redis OSS

Untuk digunakan ElastiCache untuk Valkey atau Redis OSS RBAC, Anda mengambil langkah-langkah berikut:

1. Buat satu atau beberapa pengguna.
2. Buat grup pengguna dan tambahkan pengguna ke grup tersebut.
3. Tentukan grup pengguna ke cache yang memiliki enkripsi bergerak yang aktif.

Langkah-langkah ini dijelaskan secara rinci sebagai berikut.

Topik

Membuat Pengguna dan Grup Pengguna dengan Konsol dan CLI

Informasi pengguna untuk pengguna RBAC adalah ID pengguna, nama pengguna, serta secara opsional, kata sandi dan string akses. String akses menyediakan tingkat izin pada kunci dan perintah. ID pengguna bersifat unik untuk pengguna, dan nama pengguna adalah data yang diteruskan ke mesin.

Pastikan bahwa izin pengguna yang Anda berikan sesuai dengan tujuan yang dimaksud untuk grup pengguna. Misalnya, jika Anda membuat grup pengguna bernama `Administrators`, setiap pengguna yang Anda tambahkan ke grup tersebut akan memiliki string akses yang ditetapkan menjadi akses penuh ke kunci dan perintah. Untuk pengguna di grup pengguna `e-commerce`, Anda dapat mengatur string aksesnya menjadi akses hanya-baca.

ElastiCache secara otomatis mengkonfigurasi pengguna default dengan ID pengguna dan nama pengguna `"default"` dan menambahkannya ke semua grup pengguna. Anda tidak dapat mengubah atau menghapus pengguna ini. Pengguna ini dimaksudkan untuk kompatibilitas dengan perilaku default versi Redis OSS sebelumnya dan memiliki string akses yang memungkinkannya memanggil semua perintah dan mengakses semua kunci.

Untuk menambahkan kontrol akses yang tepat ke cache, ganti pengguna default ini dengan pengguna baru yang tidak diaktifkan atau menggunakan kata sandi yang kuat. Untuk mengubah pengguna default, buat pengguna baru dengan nama pengguna yang ditetapkan ke `default`. Anda kemudian dapat menukarnya dengan pengguna default yang asli.

Prosedur berikut menunjukkan cara untuk menukar pengguna `default` asli dengan pengguna `default` lain yang memiliki string akses yang sudah diubah.

Untuk mengubah pengguna default di konsol

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Pilih Manajemen grup pengguna dari panel navigasi.

3. Untuk ID grup pengguna, pilih ID yang ingin Anda ubah. Pastikan bahwa Anda memilih tautan, bukan kotak centang.
4. Pilih Ubah.
5. Di jendela Modify, pilih Manage. Untuk “pilih pengguna yang Anda inginkan”, pilih pengguna dengan nama Pengguna sebagai default.
6. Pilih Tutup.
7. Pilih Ubah. Saat Anda melakukannya, koneksi apa pun yang ada ke cache yang dimiliki oleh pengguna default asli akan dihentikan.

Untuk memodifikasi pengguna default dengan AWS CLI

1. Buat pengguna baru dengan nama pengguna `default` menggunakan perintah berikut.

   Untuk Linux, macOS, atau Unix:

   ```
   aws elasticache create-user \
    --user-id "new-default-user" \
    --user-name "default" \
    --engine "VALKEY" \
    --passwords "a-str0ng-pa))word" \
    --access-string "off +get ~keys*"
   ```

   Untuk Windows:

   ```
   aws elasticache create-user ^
    --user-id "new-default-user" ^
    --user-name "default" ^
    --engine "VALKEY" ^
    --passwords "a-str0ng-pa))word" ^
    --access-string "off +get ~keys*"
   ```

2. Buat grup pengguna dan tambahkan pengguna yang telah Anda buat sebelumnya.

   Untuk Linux, macOS, atau Unix:

   ```
   aws elasticache create-user-group \
     --user-group-id "new-group-2" \
     --engine "VALKEY" \
     --user-ids "new-default-user"
   ```

Untuk Windows:

```
aws elasticache create-user-group ^
  --user-group-id "new-group-2" ^
  --engine "VALKEY" ^
  --user-ids "new-default-user"
```

Saat membuat pengguna, Anda dapat menetapkan hingga dua kata sandi. Saat Anda mengubah kata sandi, koneksi apa pun yang ada ke cache akan dipertahankan.

Secara khusus, perhatikan kendala kata sandi pengguna ini saat menggunakan RBAC ElastiCache untuk Valkey dan Redis OSS:

- Kata sandi harus terdiri dari 16–128 karakter yang dapat dicetak.
- Karakter non-alfanumerik berikut tidak diizinkan: `, "" / @`.

Mengelola Pengguna dengan Konsol dan CLI

Gunakan prosedur berikut untuk mengelola pengguna di konsol.

Mengelola pengguna di konsol

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di ElastiCache dasbor Amazon, pilih Manajemen pengguna. Pilihan berikut tersedia:

   - Buat pengguna – Saat membuat pengguna, Anda memasukkan ID pengguna, nama pengguna, mode autentikasi, dan string akses. String akses menetapkan tingkat izin untuk kunci dan perintah yang boleh digunakan pengguna.

     Saat membuat pengguna, Anda dapat menetapkan hingga dua kata sandi. Saat Anda mengubah kata sandi, koneksi apa pun yang ada ke cache akan dipertahankan.
   - Ubah pengguna – Memungkinkan Anda memperbarui pengaturan autentikasi pengguna atau mengubah string aksesnya.
   - Hapus pengguna – Akun akan dihapus dari Grup Pengguna yang memiliki akun tersebut.

Gunakan prosedur berikut untuk mengelola pengguna dengan AWS CLI.

Untuk mengubah pengguna menggunakan CLI

- Gunakan perintah `modify-user` untuk memperbarui satu atau beberapa kata sandi pengguna atau mengubah izin akses pengguna.

  Saat pengguna diubah, grup pengguna yang terkait dengan pengguna akan diperbarui, beserta cache apa pun yang terkait dengan grup pengguna. Semua koneksi yang ada akan dipertahankan. Berikut ini adalah beberapa contohnya.

  Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-user \
  --user-id user-id-1 \
  --access-string "~objects:* ~items:* ~public:*" \
  --authentication-mode Type=iam
```

  Untuk Windows:

```
aws elasticache modify-user ^
  --user-id user-id-1 ^
  --access-string "~objects:* ~items:* ~public:*" ^
  --authentication-mode Type=iam
```

> **Note**
>
> Sebaiknya jangan menggunakan opsi `nopass`. Jika Anda menggunakannya, sebaiknya tetapkan izin pengguna menjadi hanya-baca dengan akses ke kumpulan kunci yang terbatas.

Untuk menghapus pengguna menggunakan CLI

- Gunakan perintah `delete-user` untuk menghapus pengguna. Akun dihapus dan dihilangkan dari grup pengguna yang memiliki akun tersebut. Berikut adalah contohnya.

  Untuk Linux, macOS, atau Unix:

```
aws elasticache delete-user \
  --user-id user-id-2
```

Untuk Windows:

```
aws elasticache delete-user ^
  --user-id user-id-2
```

Untuk melihat daftar pengguna, panggil operasi describe-users.

```
aws elasticache describe-users
```

Mengelola Grup Pengguna dengan Konsol dan CLI

Anda dapat membuat grup pengguna untuk mengatur dan mengontrol akses pengguna ke satu atau beberapa cache, seperti yang ditunjukkan berikut.

Gunakan prosedur berikut untuk mengelola grup pengguna menggunakan konsol.

Mengelola grup pengguna menggunakan konsol

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Di ElastiCache dasbor Amazon, pilih Manajemen grup pengguna.

   Operasi berikut tersedia untuk membuat grup pengguna baru:

   - Buat – Saat Anda membuat grup pengguna, Anda menambahkan pengguna, lalu menetapkan grup pengguna ke cache. Misalnya, Anda dapat membuat grup pengguna `Admin` untuk pengguna yang memiliki peran administratif pada cache.

     > **Important**
     >
     > Jika Anda tidak menggunakan grup pengguna Valkey atau Redis OSS maka Anda harus menyertakan pengguna default saat membuat grup pengguna.

   - Tambahkan Pengguna – Menambahkan pengguna ke grup pengguna.
   - Hapus Pengguna – Menghapus pengguna dari grup pengguna. Saat pengguna dihapus dari grup pengguna, koneksi apa pun yang dimiliki grup tersebut ke sebuah cache akan dihentikan.
   - Hapus – Gunakan operasi ini untuk menghapus grup pengguna. Perhatikan bahwa grup pengguna itu sendiri, bukan pengguna yang terdapat dalam grup, yang akan dihapus.

Untuk grup pengguna yang ada, Anda dapat melakukan hal berikut:

- Tambahkan Pengguna – Menambahkan pengguna yang ada ke grup pengguna.
- Hapus Pengguna – Menghapus pengguna yang ada dari grup pengguna.

> **Note**
>
> Pengguna dihilangkan dari grup pengguna, tetapi tidak dihapus dari sistem.

Gunakan prosedur berikut untuk mengelola grup pengguna menggunakan CLI.

Untuk membuat grup pengguna baru dan menambahkan pengguna menggunakan CLI

- Gunakan perintah `create-user-group` seperti berikut ini.

  Untuk Linux, macOS, atau Unix:

```
aws elasticache create-user-group \
  --user-group-id "new-group-1" \
  --engine "VALKEY" \
  --user-ids user-id-1, user-id-2
```

  Untuk Windows:

```
aws elasticache create-user-group ^
  --user-group-id "new-group-1" ^
  --engine "VALKEY" ^
  --user-ids user-id-1, user-id-2
```

Untuk mengubah grup pengguna dengan menambahkan pengguna baru atau menghapus anggota saat ini menggunakan CLI

- Gunakan perintah `modify-user-group` seperti berikut ini.

  Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-user-group --user-group-id new-group-1 \
```

```
--user-ids-to-add user-id-3 \
--user-ids-to-remove user-id-2
```

Untuk Windows:

```
aws elasticache modify-user-group --user-group-id new-group-1 ^
--user-ids-to-add user-id-3 ^
--user-ids-to-removere user-id-2
```

> **Note**
>
> Koneksi terbuka milik pengguna yang dihapus dari sebuah grup pengguna akan diakhiri dengan perintah ini.

Untuk menghapus grup pengguna menggunakan CLI

- Gunakan perintah `delete-user-group` seperti berikut ini. Grup pengguna itu sendiri, bukan pengguna yang terdapat dalam grup, yang akan dihapus.

  Untuk Linux, macOS, atau Unix:

  ```
  aws elasticache delete-user-group /
     --user-group-id
  ```

  Untuk Windows:

  ```
  aws elasticache delete-user-group ^
     --user-group-id
  ```

Untuk melihat daftar grup pengguna, Anda dapat memanggil describe-user-groupsoperasi.

```
aws elasticache describe-user-groups \
  --user-group-id test-group
```

Menetapkan Grup Pengguna ke Cache Nirserver

Setelah Anda membuat grup pengguna dan menambahkan pengguna, langkah terakhir dalam menerapkan RBAC adalah menetapkan grup pengguna ke cache nirserver.

Menetapkan Grup Pengguna ke Cache Nirserver Menggunakan Konsol

Untuk menambahkan grup pengguna ke cache tanpa server menggunakan AWS Management Console, lakukan hal berikut:

- Untuk mode klaster dinonaktifkan, lihat Membuat cluster Valkey (mode cluster dinonaktifkan) (Konsol)
- Untuk mode klaster diaktifkan, lihat Membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) (Konsol)

Menetapkan Grup Pengguna ke Cache Tanpa Server Menggunakan AWS CLI

AWS CLI Operasi berikut membuat cache tanpa server menggunakan user-group-id parameter dengan nilai. *my-user-group-id* Ganti grup subnet `sng-test` dengan grup subnet yang ada.

Parameter Kunci

- **`--engine`**— Harus `VALKEY` atau`REDIS`.
- **`--user-group-id`** – Nilai ini menyediakan ID grup pengguna, yang terdiri dari pengguna dengan izin akses yang ditentukan untuk cache.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-serverless-cache \
    --serverless-cache-name "new-serverless-cache" \
    --description "new-serverless-cache" \
    --engine "VALKEY" \
    --user-group-id "new-group-1"
```

Untuk Windows:

```
aws elasticache create-serverless-cache ^
    --serverless-cache-name "new-serverless-cache" ^
```

```
    --description "new-serverless-cache" ^
    --engine "VALKEY" ^
    --user-group-id "new-group-1"
```

AWS CLI Operasi berikut memodifikasi cache tanpa server dengan user-group-id parameter dengan nilai. *my-user-group-id*

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-serverless-cache \
    --serverless-cache-name serverless-cache-1 \
    --user-group-id "new-group-2"
```

Untuk Windows:

```
aws elasticache modify-serverless-cache ^
    --serverless-cache-name serverless-cache-1 ^
    --user-group-id "new-group-2"
```

Perhatikan bahwa perubahan apa pun yang dibuat pada cache akan diperbarui secara asinkron. Anda dapat memantau progres ini dengan melihat peristiwa. Untuk informasi selengkapnya, lihat Melihat ElastiCache acara.

Menetapkan Grup Pengguna ke Grup Replikasi

Setelah Anda membuat grup pengguna dan menambahkan pengguna, langkah terakhir dalam menerapkan RBAC adalah menetapkan grup pengguna ke grup replikasi.

Menetapkan Grup Pengguna ke Grup Replikasi Menggunakan Konsol

Untuk menambahkan grup pengguna ke replikasi menggunakan AWS Management Console, lakukan hal berikut:

- Untuk mode klaster dinonaktifkan, lihat Membuat cluster Valkey (mode cluster dinonaktifkan) (Konsol)
- Untuk mode klaster diaktifkan, lihat Membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) (Konsol)

Menetapkan Grup Pengguna ke Grup Replikasi Menggunakan AWS CLI

AWS CLI Operasi berikut membuat grup replikasi dengan enkripsi dalam transit (TLS) diaktifkan dan user-group-ids parameter dengan nilai. *my-user-group-id* Ganti grup subnet `sng-test` dengan grup subnet yang ada.

Parameter Kunci

- **`--engine`**— Harus `valkey` atau`redis`.
- **`--engine-version`** – Harus 6.0 atau yang lebih baru.
- **`--transit-encryption-enabled`** – Wajib untuk autentikasi dan untuk mengaitkan grup pengguna.
- **`--user-group-ids`** – Nilai ini menyediakan ID grup pengguna, yang terdiri dari pengguna dengan izin akses yang ditentukan untuk cache.
- **`--cache-subnet-group`** – Wajib untuk mengaitkan grup pengguna.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
    --replication-group-id "new-replication-group" \
    --replication-group-description "new-replication-group" \
    --engine "VALKEY" \
    --cache-node-type cache.m5.large \
    --transit-encryption-enabled \
    --user-group-ids "new-group-1" \
    --cache-subnet-group "cache-subnet-group"
```

Untuk Windows:

```
aws elasticache create-replication-group ^
    --replication-group-id "new-replication-group" ^
    --replication-group-description "new-replication-group" ^
    --engine "VALKEY" ^
    --cache-node-type cache.m5.large ^
    --transit-encryption-enabled ^
    --user-group-ids "new-group-1" ^
    --cache-subnet-group "cache-subnet-group"
```

AWS CLI Operasi berikut memodifikasi grup replikasi dengan enkripsi dalam transit (TLS) diaktifkan dan user-group-ids parameter dengan nilai. *my-user-group-id*

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
    --replication-group-id replication-group-1 \
    --user-group-ids-to-remove "new-group-1" \
    --user-group-ids-to-add "new-group-2"
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
    --replication-group-id replication-group-1 ^
    --user-group-ids-to-remove "new-group-1" ^
    --user-group-ids-to-add "new-group-2"
```

Catat `PendingChanges` dalam respons. Perubahan apa pun yang dibuat pada cache akan diperbarui secara asinkron. Anda dapat memantau progres ini dengan melihat peristiwa. Untuk informasi selengkapnya, lihat Melihat ElastiCache acara.

Migrasi dari AUTH ke RBAC

Jika Anda menggunakan AUTH seperti yang dijelaskan Mengautentikasi dengan perintah Valkey dan Redis OSS AUTH dan ingin bermigrasi menggunakan RBAC, gunakan prosedur berikut.

Gunakan prosedur berikut untuk bermigrasi dari AUTH ke RBAC menggunakan konsol.

Untuk bermigrasi dari Valkey atau Redis OSS AUTH ke RBAC menggunakan konsol

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari daftar di sudut kanan atas, pilih AWS Wilayah tempat cache yang ingin Anda ubah berada.
3. Di panel navigasi, pilih mesin yang berjalan pada cache yang ingin Anda modifikasi.

   Daftar cache mesin yang dipilih akan muncul.
4. Di daftar cache, pilih nama cache yang ingin diubah.
5. Untuk Tindakan, pilih Ubah.

Jendela Ubah muncul.

6. Untuk Kontrol akses, pilih Daftar kontrol akses grup pengguna.
7. Untuk Daftar kontrol akses grup pengguna, pilih grup pengguna.
8. Pilih Pratinjau perubahan, lalu di layar berikutnya, pilih Ubah.

Gunakan prosedur berikut untuk bermigrasi dari Valkey atau Redis OSS AUTH ke RBAC menggunakan CLI.

Untuk bermigrasi dari AUTH ke RBAC menggunakan CLI

- Gunakan perintah `modify-replication-group` seperti berikut ini.

  Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group --replication-group-id test \
    --auth-token-update-strategy DELETE \
    --user-group-ids-to-add user-group-1
```

  Untuk Windows:

```
aws elasticache modify-replication-group --replication-group-id test ^
    --auth-token-update-strategy DELETE ^
    --user-group-ids-to-add user-group-1
```

Migrasi dari RBAC ke AUTH

Jika Anda menggunakan RBAC dan ingin bermigrasi ke Redis OSS AUTH, lihat. Migrasi dari RBAC ke AUTH

> **Note**
>
> Jika Anda perlu menonaktifkan kontrol akses pada ElastiCache cache, Anda harus melakukannya melalui file AWS CLI. Lihat informasi yang lebih lengkap di the section called “Menonaktifkan kontrol akses pada cache ElastiCache Valkey atau Redis OSS”.

Merotasi kata sandi untuk pengguna secara otomatis

Dengan AWS Secrets Manager, Anda dapat mengganti kredensi hardcode dalam kode Anda (termasuk kata sandi) dengan panggilan API ke Secrets Manager untuk mengambil rahasia secara terprogram. Ini membantu memastikan bahwa rahasia tidak dapat dikompromikan oleh seseorang yang memeriksa kode Anda, karena rahasianya tidak ada di sana. Selain itu, Anda dapat mengonfigurasi Secrets Manager untuk merotasi rahasia secara otomatis sesuai dengan jadwal yang Anda tentukan. Ini memungkinkan Anda mengganti rahasia jangka panjang dengan rahasia jangka pendek, yang membantu mengurangi risiko kompromi secara signifikan.

Menggunakan Secrets Manager, Anda dapat secara otomatis memutar kata sandi Redis OSS Anda ElastiCache (yaitu, rahasia) menggunakan AWS Lambda fungsi yang disediakan Secrets Manager.

Untuk informasi lebih lanjut tentang AWS Secrets Manager, lihat Apa itu AWS Secrets Manager?

Bagaimana ElastiCache menggunakan rahasia

Valkey 7.2 dan di atasnya memiliki fitur yang setara dengan Redis OSS 7.0. Dalam Redis OSS 6, ElastiCache diperkenalkan Kontrol Akses Berbasis Peran (RBAC) untuk mengamankan cluster Valkey atau Redis OSS. Fitur ini memungkinkan koneksi tertentu dibatasi dalam hal perintah yang dapat dijalankan dan kunci yang dapat diakses. Dengan RBAC, saat pelanggan membuat pengguna dengan kata sandi, nilai kata sandi harus dimasukkan secara manual dalam teks biasa dan terlihat oleh operator.

Dengan Secrets Manager, aplikasi mengambil kata sandi dari Secrets Manager bukannya memasukkannya secara manual dan menyimpannya dalam konfigurasi aplikasi. Untuk informasi tentang cara melakukannya, lihat Bagaimana ElastiCache pengguna dikaitkan dengan rahasia.

Ada biaya yang dikeluarkan untuk menggunakan rahasia. Untuk informasi harga, lihat Harga AWS Secrets Manager.

Bagaimana ElastiCache pengguna dikaitkan dengan rahasia

Secrets Manager akan menyimpan referensi untuk pengguna terkait di bidang `SecretString` rahasia. Tidak akan ada referensi ke rahasia dari ElastiCache samping.

```
{
    "password": "strongpassword",
    "username": "user1",
    "user_arn": "arn:aws:elasticache:us-east-1:xxxxxxxxxx918:user:user1" //this is the
 bond between the secret and the user
```

```
}
```

Fungsi rotasi Lambda

Untuk mengaktifkan rotasi kata sandi otomatis Secrets Manager, Anda akan membuat fungsi Lambda yang akan berinteraksi dengan API modify-user untuk memperbarui kata sandi pengguna.

Untuk informasi tentang cara kerjanya, lihat Cara kerja rotasi.

> **Note**
>
> Untuk beberapa AWS layanan, untuk menghindari skenario wakil yang membingungkan, AWS merekomendasikan agar Anda menggunakan kunci kondisi `aws:SourceAccount` global `aws:SourceArn` dan global. Namun, jika Anda menyertakan kondisi `aws:SourceArn` dalam kebijakan fungsi rotasi Anda, fungsi rotasi hanya dapat digunakan untuk merotasi rahasia yang ditentukan oleh ARN tersebut. Sebaiknya hanya sertakan kunci konteks `aws:SourceAccount` agar Anda dapat menggunakan fungsi rotasi untuk beberapa rahasia.

Untuk masalah apa pun yang mungkin Anda temui, lihat Memecahkan masalah rotasi AWS Secrets Manager.

Cara membuat ElastiCache pengguna dan mengaitkannya dengan Secrets Manager

Langkah-langkah berikut menggambarkan cara membuat pengguna dan mengaitkannya dengan Secrets Manager:

1. Buat pengguna yang tidak aktif

   Untuk Linux, macOS, atau Unix:

   ```
   aws elasticache create-user \
    --user-id user1 \
    --user-name user1 \
    --engine "REDIS" \
    --no-password \ // no authentication is required
    --access-string "*off* +get ~keys*" // this disables the user
   ```

   Untuk Windows:

```
aws elasticache create-user ^
 --user-id user1 ^
 --user-name user1 ^
 --engine "REDIS" ^
 --no-password ^ // no authentication is required
 --access-string "*off* +get ~keys*" // this disables the user
```

Anda akan melihat respons yang mirip dengan berikut ini:

```
{
    "UserId": "user1",
    "UserName": "user1",
    "Status": "active",
    "Engine": "redis",
    "AccessString": "off ~keys* -@all +get",
    "UserGroupIds": [],
    "Authentication": {
        "Type": "no_password"
    },
    "ARN": "arn:aws:elasticache:us-east-1:xxxxxxxxxx918:user:user1"
}
```

2. Buat Rahasia

   Untuk Linux, macOS, atau Unix:

```
aws secretsmanager create-secret \
--name production/ec/user1 \
--secret-string \
'{
   "user_arn": "arn:aws:elasticache:us-east-1:123456xxxx:user:user1",
    "username":"user1"
 }'
```

   Untuk Windows:

```
aws secretsmanager create-secret ^
--name production/ec/user1 ^
--secret-string ^
'{
   "user_arn": "arn:aws:elasticache:us-east-1:123456xxxx:user:user1",
```

```
    "username":"user1"
 }'
```

Anda akan melihat respons yang mirip dengan berikut ini:

```
{
 "ARN": "arn:aws:secretsmanager:us-east-1:123456xxxx:secret:production/ec/user1-
eaFois",
 "Name": "production/ec/user1",
 "VersionId": "aae5b963-1e6b-4250-91c6-ebd6c47d0d95"
}
```

3. Konfigurasikan fungsi Lambda untuk merotasi kata sandi Anda

   a. Masuk ke AWS Management Console dan buka konsol Lambda di https://console.aws.amazon.com/lambda/

   b. Pilih Fungsi pada panel navigasi lalu pilih fungsi yang Anda buat. Pilih nama fungsi, bukan kotak centang di sebelah kirinya.

   c. Pilih tab Konfigurasi.

   d. Dalam konfigurasi Umum, pilih Edit lalu atur Waktu habis setidaknya 12 menit.

   e. Pilih Simpan.

   f. Pilih Variabel lingkungan lalu atur yang berikut:

      i. SECRETS_MANAGER_ENDPOINT - https://secretsmanager.**REGION**.amazonaws.com

      ii. SECRET_ARN – Amazon Resource Name (ARN) dari rahasia yang Anda buat pada Langkah 2.

      iii. USER_NAME — Nama pengguna pengguna, ElastiCache

      iv. Pilih Simpan.

   g. Pilih Izin.

   h. Pada Peran eksekusi, pilih nama peran fungsi Lambda untuk dilihat di konsol IAM.

   i. Fungsi Lambda akan memerlukan izin berikut untuk mengubah pengguna dan mengatur kata sandi:

      ElastiCache

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "elasticache:DescribeUsers",
                "elasticache:ModifyUser"
            ],
            "Resource": "arn:aws:elasticache:us-
east-1:111122223333:user:user1"
        }
    ]
}
```

Secrets Manager

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:PutSecretValue",
                "secretsmanager:UpdateSecretVersionStage"
            ],
            "Resource": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:XXXX"
        },
        {
            "Effect": "Allow",
            "Action": "secretsmanager:GetRandomPassword",
            "Resource": "*"
```

```
        }
    ]
}
```

4. Mengatur rotasi rahasia Secrets Manager

    a. Menggunakan AWS Management Console, lihat Mengatur rotasi otomatis untuk AWS rahasia Secrets Manager menggunakan konsol

    Untuk informasi selengkapnya tentang mengatur jadwal rotasi, lihat Menjadwalkan ekspresi di rotasi Secrets Manager.

    b. Menggunakan AWS CLI, lihat Mengatur rotasi otomatis untuk AWS Secrets Manager menggunakan AWS Command Line Interface

Autentikasi dengan IAM

Topik

- Gambaran Umum
- Batasan
- Pengaturan
- Terhubung

Gambaran Umum

Dengan Autentikasi IAM Anda dapat mengautentikasi koneksi ElastiCache untuk Valkey atau Redis OSS menggunakan identitas AWS IAM, ketika cache Anda dikonfigurasi untuk menggunakan Valkey atau Redis OSS versi 7 atau lebih tinggi. Hal ini memungkinkan Anda memperkuat model keamanan Anda dan menyederhanakan banyak tugas keamanan administratif. Anda juga dapat menggunakan Autentikasi IAM untuk mengonfigurasi kontrol akses berbutir halus untuk setiap ElastiCache cache dan ElastiCache pengguna individual, mengikuti prinsip izin hak istimewa paling sedikit. Otentikasi IAM untuk ElastiCache bekerja dengan menyediakan token otentikasi IAM berumur pendek alih-alih kata sandi ElastiCache pengguna yang berumur panjang di Valkey atau Redis OSS atau perintah. `AUTH HELLO` Untuk informasi selengkapnya tentang token autentikasi IAM, lihat proses penandatanganan Signature Version 4 di Panduan Referensi AWS Umum dan contoh kode di bawah ini.

Anda dapat menggunakan identitas IAM dan kebijakan terkait mereka untuk lebih membatasi akses Valkey atau Redis OSS. Anda juga dapat memberikan akses ke pengguna dari penyedia Identitas federasi mereka langsung ke cache Valkey atau Redis OSS.

Untuk menggunakan AWS IAM dengan ElastiCache, Anda harus terlebih dahulu membuat ElastiCache pengguna dengan mode otentikasi diatur ke IAM. Kemudian Anda dapat membuat atau menggunakan kembali identitas IAM. Identitas IAM memerlukan kebijakan terkait untuk memberikan `elasticache:Connect` tindakan ke ElastiCache cache dan ElastiCache pengguna. Setelah dikonfigurasi, Anda dapat membuat token otentikasi IAM menggunakan AWS kredensi pengguna atau peran IAM. Akhirnya Anda perlu memberikan token otentikasi IAM berumur pendek sebagai kata sandi di Klien Valkey atau Redis OSS Anda saat menghubungkan ke cache Anda. Klien Valkey atau Redis OSS dengan dukungan untuk penyedia kredensi dapat secara otomatis menghasilkan kredensi sementara secara otomatis untuk setiap koneksi baru. ElastiCache akan melakukan otentikasi IAM untuk permintaan koneksi ElastiCache pengguna yang mendukung IAM dan akan memvalidasi permintaan koneksi dengan IAM.

Batasan

Saat menggunakan autentikasi IAM, batasan berikut berlaku:

- Autentikasi IAM tersedia saat menggunakan ElastiCache untuk Valkey 7.2 dan di atasnya atau Redis OSS versi 7.0 dan di atasnya.
- Untuk pengguna yang mendukung IAM, nama ElastiCache pengguna dan properti id pengguna harus identik.
- Token autentikasi IAM berlaku selama 15 menit. Untuk koneksi yang berumur panjang, sebaiknya gunakan klien Valkey atau Redis OSS yang mendukung antarmuka penyedia kredensial.
- Koneksi yang diautentikasi IAM ke ElastiCache Valkey atau Redis OSS akan secara otomatis terputus setelah 12 jam. Koneksi dapat diperpanjang selama 12 jam dengan mengirim perintah `AUTH` atau `HELLO` dengan token autentikasi IAM baru.
- Autentikasi IAM tidak didukung dalam perintah `MULTI EXEC`.
- Saat ini, autentikasi IAM mendukung kunci konteks kondisi global berikut ini:
  - Saat menggunakan autentikasi IAM dengan cache nirserver, `aws:VpcSourceIp`, `aws:SourceVpc`, `aws:SourceVpce`, `aws:CurrentTime`, `aws:EpochTime`, dan `aws:ResourceTag/%s` (dari cache nirserver dan pengguna terkait) didukung.
  - Saat menggunakan autentikasi IAM dengan grup replikasi, `aws:SourceIp` dan `aws:ResourceTag/%s` (dari grup replikasi dan pengguna terkait) didukung.

Untuk informasi selengkapnya tentang kunci konteks kondisi global, lihat Kunci konteks kondisi global AWS dalam Panduan Pengguna IAM.

Pengaturan

Untuk mengatur autentikasi IAM:

1. Buat cache

```
aws elasticache create-serverless-cache \
  --serverless-cache-name cache-01  \
  --description "ElastiCache IAM auth application" \
  --engine redis
```

2. Buat dokumen kebijakan kepercayaan IAM, seperti yang ditunjukkan di bawah ini, untuk peran Anda yang memungkinkan akun Anda mengambil peran baru. Simpan kebijakan ini ke file bernama trust-policy.json.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
        "Action": "sts:AssumeRole"
    }
}
```

3. Buat dokumen kebijakan IAM, seperti yang ditunjukkan di bawah ini. Simpan kebijakan ke file bernama policy.json.

   JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect" : "Allow",
```

```
      "Action" : [
        "elasticache:Connect"
      ],
      "Resource" : [
        "arn:aws:elasticache:us-
east-1:123456789012:serverlesscache:cache-01",
        "arn:aws:elasticache:us-east-1:123456789012:user:iam-user-01"
      ]
    }
  ]
}
```

4. Buat peran IAM.

```
aws iam create-role \
--role-name "elasticache-iam-auth-app" \
--assume-role-policy-document file://trust-policy.json
```

5. Buat kebijakan IAM.

```
aws iam create-policy \
  --policy-name "elasticache-allow-all" \
  --policy-document file://policy.json
```

6. Lampirkan kebijakan IAM ke peran tersebut.

```
aws iam attach-role-policy \
 --role-name "elasticache-iam-auth-app" \
 --policy-arn "arn:aws:iam::123456789012:policy/elasticache-allow-all"
```

7. Buat pengguna baru yang didukung IAM.

```
aws elasticache create-user \
  --user-name iam-user-01 \
  --user-id iam-user-01 \
  --authentication-mode Type=iam \
  --engine redis \
  --access-string "on ~* +@all"
```

8. Buat grup pengguna dan lampirkan pengguna.

```
aws elasticache create-user-group \
  --user-group-id iam-user-group-01 \
```

```
  --engine redis \
  --user-ids default iam-user-01

aws elasticache modify-serverless-cache \
  --serverless-cache-name cache-01  \
  --user-group-id iam-user-group-01
```

Terhubung

Terhubung dengan token sebagai kata sandi

Pertama-tama, Anda harus membuat token autentikasi IAM berumur pendek menggunakan permintaan AWS SigV4 yang telah ditandatangani sebelumnya. Setelah itu Anda memberikan token otentikasi IAM sebagai kata sandi saat menghubungkan ke cache Valkey atau Redis OSS, seperti yang ditunjukkan pada contoh di bawah ini.

```
String userId = "insert user id";
String cacheName = "insert cache name";
boolean isServerless = true;
String region = "insert region";

// Create a default AWS Credentials provider.
// This will look for AWS credentials defined in environment variables or system
 properties.
AWSCredentialsProvider awsCredentialsProvider = new
 DefaultAWSCredentialsProviderChain();

// Create an IAM authentication token request and signed it using the AWS credentials.
// The pre-signed request URL is used as an IAM authentication token for ElastiCache
 with Redis OSS.
IAMAuthTokenRequest iamAuthTokenRequest = new IAMAuthTokenRequest(userId, cacheName,
 region, isServerless);
String iamAuthToken =
 iamAuthTokenRequest.toSignedRequestUri(awsCredentialsProvider.getCredentials());

// Construct Redis OSS URL with IAM Auth credentials provider
RedisURI redisURI = RedisURI.builder()
    .withHost(host)
    .withPort(port)
    .withSsl(ssl)
    .withAuthentication(userId, iamAuthToken)
    .build();
```

```
// Create a new Lettuce Redis OSS client
RedisClient client = RedisClient.create(redisURI);
client.connect();
```

Di bawah ini adalah definisi untuk `IAMAuthTokenRequest`.

```
public class IAMAuthTokenRequest {
    private static final HttpMethodName REQUEST_METHOD = HttpMethodName.GET;
    private static final String REQUEST_PROTOCOL = "http://";
    private static final String PARAM_ACTION = "Action";
    private static final String PARAM_USER = "User";
    private static final String PARAM_RESOURCE_TYPE = "ResourceType";
    private static final String RESOURCE_TYPE_SERVERLESS_CACHE = "ServerlessCache";
    private static final String ACTION_NAME = "connect";
    private static final String SERVICE_NAME = "elasticache";
    private static final long TOKEN_EXPIRY_SECONDS = 900;

    private final String userId;
    private final String cacheName;
    private final String region;
    private final boolean isServerless;

    public IAMAuthTokenRequest(String userId, String cacheName, String region, boolean
 isServerless) {
        this.userId = userId;
        this.cacheName = cacheName;
        this.region = region;
        this.isServerless = isServerless;
    }

    public String toSignedRequestUri(AWSCredentials credentials) throws
 URISyntaxException {
        Request<Void> request = getSignableRequest();
        sign(request, credentials);
        return new URIBuilder(request.getEndpoint())
            .addParameters(toNamedValuePair(request.getParameters()))
            .build()
            .toString()
            .replace(REQUEST_PROTOCOL, "");
    }

    private <T> Request<T> getSignableRequest() {
```

```
        Request<T> request  = new DefaultRequest<>(SERVICE_NAME);
        request.setHttpMethod(REQUEST_METHOD);
        request.setEndpoint(getRequestUri());
        request.addParameters(PARAM_ACTION, Collections.singletonList(ACTION_NAME));
        request.addParameters(PARAM_USER, Collections.singletonList(userId));
        if (isServerless) {
            request.addParameters(PARAM_RESOURCE_TYPE,
 Collections.singletonList(RESOURCE_TYPE_SERVERLESS_CACHE));
        }
        return request;
    }

    private URI getRequestUri() {
        return URI.create(String.format("%s%s/", REQUEST_PROTOCOL, cacheName));
    }

    private <T> void sign(SignableRequest<T> request, AWSCredentials credentials) {
        AWS4Signer signer = new AWS4Signer();
        signer.setRegionName(region);
        signer.setServiceName(SERVICE_NAME);

        DateTime dateTime = DateTime.now();
        dateTime = dateTime.plus(Duration.standardSeconds(TOKEN_EXPIRY_SECONDS));

        signer.presignRequest(request, credentials, dateTime.toDate());
    }

    private static List<NameValuePair> toNamedValuePair(Map<String, List<String>> in) {
        return in.entrySet().stream()
            .map(e -> new BasicNameValuePair(e.getKey(), e.getValue().get(0)))
            .collect(Collectors.toList());
    }
}
```

Terhubung dengan penyedia kredensial

Kode di bawah ini menunjukkan cara mengautentikasi dengan ElastiCache menggunakan penyedia kredensi otentikasi IAM.

```
String userId = "insert user id";
String cacheName = "insert cache name";
boolean isServerless = true;
String region = "insert region";
```

```
// Create a default AWS Credentials provider.
// This will look for AWS credentials defined in environment variables or system
 properties.
AWSCredentialsProvider awsCredentialsProvider = new
 DefaultAWSCredentialsProviderChain();

// Create an IAM authentication token request. Once this request is signed it can be
 used as an
// IAM authentication token for ElastiCache with Redis OSS.
IAMAuthTokenRequest iamAuthTokenRequest = new IAMAuthTokenRequest(userId, cacheName,
 region, isServerless);

// Create a Redis OSS credentials provider using IAM credentials.
RedisCredentialsProvider redisCredentialsProvider = new
 RedisIAMAuthCredentialsProvider(
    userId, iamAuthTokenRequest, awsCredentialsProvider);

// Construct Redis OSS URL with IAM Auth credentials provider
RedisURI redisURI = RedisURI.builder()
    .withHost(host)
    .withPort(port)
    .withSsl(ssl)
    .withAuthentication(redisCredentialsProvider)
    .build();

// Create a new Lettuce Redis OSS client
RedisClient client = RedisClient.create(redisURI);
client.connect();
```

Di bawah ini adalah contoh klien OSS Lettuce Redis yang membungkus penyedia kredensi IAMAuth TokenRequest dalam untuk menghasilkan kredensi sementara secara otomatis bila diperlukan.

```
public class RedisIAMAuthCredentialsProvider implements RedisCredentialsProvider {
    private static final long TOKEN_EXPIRY_SECONDS = 900;

    private final AWSCredentialsProvider awsCredentialsProvider;
    private final String userId;
    private final IAMAuthTokenRequest iamAuthTokenRequest;
    private final Supplier<String> iamAuthTokenSupplier;

    public RedisIAMAuthCredentialsProvider(String userId,
        IAMAuthTokenRequest iamAuthTokenRequest,
```

```
        AWSCredentialsProvider awsCredentialsProvider) {
        this.userName = userName;
        this.awsCredentialsProvider = awsCredentialsProvider;
        this.iamAuthTokenRequest = iamAuthTokenRequest;
        this.iamAuthTokenSupplier =
 Suppliers.memoizeWithExpiration(this::getIamAuthToken, TOKEN_EXPIRY_SECONDS,
 TimeUnit.SECONDS);
    }

    @Override
    public Mono<RedisCredentials> resolveCredentials() {
        return Mono.just(RedisCredentials.just(userId, iamAuthTokenSupplier.get()));
    }

    private String getIamAuthToken() {
        return
 iamAuthTokenRequest.toSignedRequestUri(awsCredentialsProvider.getCredentials());
    }
}
```

Mengautentikasi dengan perintah Valkey dan Redis OSS AUTH

> **Note**
>
> Yang AUTH telah digantikan oleh. the section called “Kontrol Akses Berbasis Peran (RBAC)” Semua cache nirserver harus menggunakan RBAC untuk autentikasi.

Token atau kata sandi otentikasi Valkey dan Redis OSS memungkinkan Valkey dan Redis OSS untuk meminta kata sandi sebelum mengizinkan klien menjalankan perintah, sehingga meningkatkan keamanan data. AUTHIni tersedia untuk cluster yang dirancang sendiri saja.

Topik

- Ikhtisar AUTH ElastiCache untuk Valkey dan Redis OSS
- Menerapkan otentikasi ke cluster ElastiCache untuk Valkey dan Redis OSS
- Memodifikasi token AUTH pada klaster yang telah ada
- Migrasi dari RBAC ke AUTH

Ikhtisar AUTH ElastiCache untuk Valkey dan Redis OSS

Saat Anda menggunakan cluster ElastiCache for Valkey dan Redis OSS Anda, ada beberapa penyempurnaan. AUTH

Secara khusus, perhatikan kendala token atau kata sandi AUTH ini saat menggunakan AUTH:

- Token, atau kata sandi, harus berisi 16–128 karakter yang dapat dicetak.
- Karakter non-alfanumerik yang diizinkan adalah (!, &, #, $, ^, <, >, -).
- AUTH hanya dapat diaktifkan untuk enkripsi in-transit yang diaktifkan Valkey atau Redis OSS cluster.

Untuk menyiapkan token yang kuat, sebaiknya ikuti kebijakan kata sandi yang ketat, seperti yang mewajibkan hal berikut:

- Token atau kata sandi harus mencakup setidaknya tiga dari jenis karakter berikut:
  - Karakter huruf besar
  - Karakter huruf kecil
  - Angka
  - Karakter non-alfanumerik (`!`, &, #, $, ^, <, >, `-`)
- Token atau kata sandi tidak boleh mengandung kata kamus atau kata kamus yang sedikit dimodifikasi.
- Token atau kata sandi tidak boleh sama dengan atau mirip dengan token yang baru digunakan.

Menerapkan otentikasi ke cluster ElastiCache untuk Valkey dan Redis OSS

Anda dapat meminta pengguna memasukkan token (kata sandi) pada server Valkey atau Redis OSS yang dilindungi token. Untuk melakukannya, sertakan parameter `--auth-token` (API: `AuthToken`) dengan token yang tepat ketika Anda membuat grup replikasi atau klaster Anda. Sertakan juga ke dalamnya semua perintah berikutnya untuk grup replikasi atau klaster.

AWS CLI Operasi berikut membuat grup replikasi dengan enkripsi dalam transit (TLS) diaktifkan dan token. AUTH *`This-is-a-sample-token`* Ganti grup subnet `sng-test` dengan grup subnet yang ada.

Parameter Kunci

- **`--engine`**Harus `valkey` atau`redis`.

- **`--engine-version`**— Jika mesin Redis OSS, harus 3.2.6, 4.0.10, atau yang lebih baru.
- **`--transit-encryption-enabled`** – Wajib untuk autentikasi dan kelayakan HIPAA.
- **`--auth-token`** – Wajib untuk kelayakan HIPAA. Nilai ini harus menjadi token yang benar untuk server Valkey atau Redis OSS yang dilindungi token ini.
- **`--cache-subnet-group`** – Wajib untuk kelayakan HIPAA.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
    --replication-group-id authtestgroup \
    --replication-group-description authtest \
    --engine redis \
    --cache-node-type cache.m4.large \
    --num-node-groups 1 \
    --replicas-per-node-group 2 \
    --transit-encryption-enabled \
    --auth-token This-is-a-sample-token \
    --cache-subnet-group sng-test
```

Untuk Windows:

```
aws elasticache create-replication-group ^
    --replication-group-id authtestgroup ^
    --replication-group-description authtest ^
    --engine redis ^
    --cache-node-type cache.m4.large ^
    --num-node-groups 1 ^
    --replicas-per-node-group 2 ^
    --transit-encryption-enabled ^
    --auth-token This-is-a-sample-token ^
    --cache-subnet-group sng-test
```

Memodifikasi token AUTH pada klaster yang telah ada

Untuk mempermudah memperbarui otentikasi Anda, Anda dapat memodifikasi AUTH token yang digunakan pada klaster. Anda dapat melakukan modifikasi ini jika versi mesinnya Valkey 7.2 atau lebih tinggi atau Redis 5.0.6 atau lebih tinggi. ElastiCache juga harus mengaktifkan enkripsi saat transit.

Mengubah token auth mendukung dua strategi: ROTATE dan SET. Strategi ROTATE menambahkan token AUTH tambahan ke server sambil mempertahankan token sebelumnya. Strategi SET memperbarui server untuk mendukung hanya satu token AUTH. Buat panggilan perubahan ini dengan parameter `--apply-immediately` untuk menerapkan perubahan dengan segera.

Merotasi token AUTH

Untuk memperbarui server Valkey atau Redis OSS dengan token AUTH baru, panggil `ModifyReplicationGroup` API dengan `--auth-token` parameter sebagai AUTH token baru dan `--auth-token-update-strategy` dengan nilai ROTATE. Setelah modifikasi ROTATE selesai, cluster akan mendukung token AUTH sebelumnya selain yang ditentukan dalam `auth-token` parameter. Jika tidak ada token AUTH yang dikonfigurasi pada grup replikasi sebelum rotasi token AUTH, cluster mendukung token AUTH yang ditentukan dalam `--auth-token` parameter selain mendukung koneksi tanpa otentikasi. Lihat Mengatur token AUTH untuk memperbarui token AUTH agar diperlukan menggunakan strategi pembaruan SET.

> **Note**
>
> Jika Anda tidak mengonfigurasi token AUTH sebelumnya, setelah perubahan selesai, klaster akan mendukung token no AUTH selain token yang ditentukan dalam parameter auth-token.

Jika modifikasi ini dilakukan pada server yang sudah mendukung dua token AUTH, token AUTH tertua juga akan dihapus selama operasi ini. Ini memungkinkan server untuk mendukung hingga dua token AUTH terbaru pada waktu tertentu.

Pada titik ini, Anda dapat melanjutkan dengan memperbarui klien untuk menggunakan token AUTH terbaru. Setelah klien diperbarui, Anda dapat menggunakan strategi SET untuk rotasi token AUTH (dijelaskan pada bagian berikut) untuk secara khusus mulai menggunakan token baru.

AWS CLI Operasi berikut memodifikasi grup replikasi untuk memutar token. AUTH *`This-is-the-rotated-token`*

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
--replication-group-id authtestgroup \
--auth-token This-is-the-rotated-token \
--auth-token-update-strategy ROTATE \
```

```
--apply-immediately
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
--replication-group-id authtestgroup ^
--auth-token This-is-the-rotated-token ^
--auth-token-update-strategy ROTATE ^
--apply-immediately
```

Mengatur token AUTH

Untuk memperbarui server Valkey atau Redis OSS untuk mendukung satu AUTH token yang diperlukan, panggil operasi `ModifyReplicationGroup` API dengan `--auth-token` parameter dengan nilai yang sama dengan token AUTH terakhir dan `--auth-token-update-strategy` parameter dengan nilai. `SET` Strategi SET hanya dapat digunakan dengan cluster yang memiliki 2 token AUTH atau 1 token AUTH opsional dari menggunakan strategi ROTATE sebelumnya. Setelah modifikasi selesai, server hanya mendukung token AUTH yang ditentukan dalam parameter auth-token.

AWS CLI Operasi berikut memodifikasi grup replikasi untuk mengatur token AUTH ke. *This-is-the-set-token*

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
--replication-group-id authtestgroup \
--auth-token This-is-the-set-token \
--auth-token-update-strategy SET \
--apply-immediately
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
--replication-group-id authtestgroup ^
--auth-token This-is-the-set-token ^
--auth-token-update-strategy SET ^
--apply-immediately
```

Mengaktifkan autentikasi pada klaster yang telah ada

Untuk mengaktifkan otentikasi pada server Valkey atau Redis OSS yang ada, panggil operasi API. `ModifyReplicationGroup` Panggil `ModifyReplicationGroup` dengan parameter `--auth-token` sebagai token baru dan `--auth-token-update-strategy` dengan nilai ROTATE.

Setelah modifikasi ROTATE selesai, cluster mendukung AUTH token yang ditentukan dalam `--auth-token` parameter selain mendukung koneksi tanpa otentikasi. Setelah semua aplikasi klien diperbarui untuk mengautentikasi ke Valkey atau Redis OSS dengan token AUTH, gunakan strategi SET untuk menandai token AUTH sesuai kebutuhan. Mengaktifkan otentikasi hanya didukung pada server Valkey dan Redis OSS dengan enkripsi dalam transit (TLS) diaktifkan.

Migrasi dari RBAC ke AUTH

Jika Anda mengautentikasi pengguna dengan Valkey atau Redis OSS Role-Based Access Control (RBAC) seperti yang dijelaskan dalamKontrol Akses Berbasis Peran (RBAC), dan Anda ingin bermigrasi ke AUTH, gunakan prosedur berikut. Anda dapat bermigrasi menggunakan konsol atau CLI.

Untuk bermigrasi dari RBAC ke AUTH menggunakan konsol

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari daftar di sudut kanan atas, pilih AWS Wilayah tempat cluster yang ingin Anda modifikasi berada.
3. Di panel navigasi, pilih mesin yang berjalan pada klaster yang ingin diubah.

   Daftar klaster mesin yang dipilih akan muncul.
4. Dalam daftar klaster tersebut, pilih nama klaster yang ingin Anda ubah.
5. Untuk Tindakan, pilih Ubah.

   Jendela Ubah muncul.
6. Untuk kontrol Akses, pilih akses pengguna default Valkey AUTH atau akses pengguna default Redis OSS AUTH.
7. Di bawah token Valkey AUTH atau token AUTH Redis OSS, tetapkan token baru.
8. Pilih Pratinjau perubahan, lalu di layar berikutnya, pilih Ubah.

Untuk bermigrasi dari RBAC ke AUTH menggunakan AWS CLI

Gunakan salah satu perintah berikut untuk mengonfigurasi AUTH token opsional baru untuk grup replikasi Valkey atau Redis OSS Anda. Perhatikan bahwa token Auth opsional akan memungkinkan akses tidak diautentikasi ke grup replikasi hingga token Auth ditandai sesuai kebutuhan, menggunakan strategi `SET` pembaruan pada langkah berikut.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
    --replication-group-id test \
    --remove-user-groups \
    --auth-token This-is-a-sample-token \
    --auth-token-update-strategy ROTATE \
    --apply-immediately
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
    --replication-group-id test ^
    --remove-user-groups ^
    --auth-token This-is-a-sample-token ^
    --auth-token-update-strategy ROTATE ^
    --apply-immediately
```

Setelah menjalankan perintah di atas, Anda dapat memperbarui aplikasi Valkey atau Redis OSS Anda untuk mengautentikasi ke grup ElastiCache replikasi menggunakan token AUTH opsional yang baru dikonfigurasi. Untuk menyelesaikan rotasi token Auth, gunakan strategi pembaruan pada perintah berikutnya `SET` di bawah ini. Ini akan menandai token AUTH opsional sesuai kebutuhan. Ketika pembaruan token Auth selesai, status grup replikasi akan ditampilkan sebagai `ACTIVE` dan semua koneksi ke grup replikasi ini akan memerlukan otentikasi.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
   --replication-group-id test \
   --auth-token This-is-a-sample-token \
   --auth-token-update-strategy SET \
   --apply-immediately
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
```

```
    --replication-group-id test ^
    --remove-user-groups ^
    --auth-token This-is-a-sample-token ^
    --auth-token-update-strategy SET ^
    --apply-immediately
```

Untuk informasi selengkapnya, lihat Mengautentikasi dengan perintah Valkey dan Redis OSS AUTH.

> **Note**
>
> Jika Anda perlu menonaktifkan kontrol akses pada ElastiCache Cluster, lihatthe section called "Menonaktifkan kontrol akses pada cache ElastiCache Valkey atau Redis OSS".

Menonaktifkan kontrol akses pada cache ElastiCache Valkey atau Redis OSS

Ikuti petunjuk di bawah ini untuk menonaktifkan kontrol akses pada cache Valkey atau Redis OSS TLS-enabled. Cache Anda akan memiliki salah satu dari dua jenis konfigurasi yang berbeda: AUTH default user access atau User group access control list (RBAC). Jika cache Anda dibuat dengan konfigurasi AUTH, Anda harus mengubahnya ke konfigurasi RBAC sebelum Anda dapat menonaktifkan cache dengan menghapus grup pengguna. Jika cache Anda dibuat dengan konfigurasi RBAC, Anda dapat langsung menonaktifkannya.

Untuk menonaktifkan cache tanpa server Valkey atau Redis OSS yang dikonfigurasi dengan RBAC

1. Hapus grup pengguna untuk menonaktifkan kontrol akses.

   ```
   aws elasticache modify-serverless-cache --serverless-cache-name <serverless-cache>
    --remove-user-group
   ```

2. (Opsional) Pastikan bahwa tidak ada grup pengguna yang terkait dengan cache nirserver.

   ```
   aws elasticache describe-serverless-caches --serverless-cache-name <serverless-
   cache>
   {
       "..."
       "UserGroupId": ""
       "..."
   }
   ```

Untuk menonaktifkan cache Valkey atau Redis OSS dengan dikonfigurasi dengan token AUTH

1. Ubah token AUTH ke RBAC dan tentukan grup pengguna yang akan ditambahkan.

```
aws elasticache modify-replication-group --replication-group-id <replication-group-id-value> --auth-token-update-strategy DELETE --user-group-ids-to-add <user-group-value>
```

2. Verifikasi bahwa token AUTH dinonaktifkan dan grup pengguna telah ditambahkan.

```
aws elasticache describe-replication-groups --replication-group-id <replication-group-id-value>
{
    "..."
    "AuthTokenEnabled": false,
    "UserGroupIds": [
        "<user-group-value>"
    ]
    "..."
}
```

3. Hapus grup pengguna untuk menonaktifkan kontrol akses.

```
aws elasticache modify-replication-group --replication-group-id <replication-group-value> --user-group-ids-to-remove <user-group-value>
{
    "..."
    "PendingModifiedValues": {
    "UserGroups": {
      "UserGroupIdsToAdd": [],
      "UserGroupIdsToRemove": [
        "<user-group-value>"
      ]
    }
    "..."
}
```

4. (Opsional) Pastikan bahwa tidak ada grup pengguna yang terkait dengan klaster. Bidang `AuthTokenEnabled` juga harus menampilkan salah (false).

```
aws elasticache describe-replication-groups --replication-group-id <replication-group-value>
```

```
"AuthTokenEnabled": false
```

Untuk menonaktifkan cluster Valkey atau Redis OSS yang dikonfigurasi dengan RBAC

1. Hapus grup pengguna untuk menonaktifkan kontrol akses.

```
aws elasticache modify-replication-group --replication-group-id <replication-group-
value> --user-group-ids-to-remove <user-group-value>
{
    "..."
    "PendingModifiedValues": {
    "UserGroups": {
      "UserGroupIdsToAdd": [],
      "UserGroupIdsToRemove": [
        "<user-group-value>"
      ]
    }
    "..."
}
```

2. (Opsional) Pastikan bahwa tidak ada grup pengguna yang terkait dengan klaster. Bidang `AuthTokenEnabled` juga harus menampilkan salah (false).

```
aws elasticache describe-replication-groups --replication-group-id <replication-
group-value>
"AuthTokenEnabled": false
```

Privasi lalu lintas antarjaringan

Amazon ElastiCache menggunakan teknik berikut untuk mengamankan data cache Anda dan melindunginya dari akses yang tidak sah:

- Amazon VPCs dan ElastiCache keamanan menjelaskan jenis grup keamanan yang Anda butuhkan untuk instalasi Anda.
- Identity and Access Management untuk Amazon ElastiCache untuk memberikan dan membatasi tindakan pengguna, grup, dan peran.

Topik

- Amazon VPCs dan ElastiCache keamanan
- ElastiCache API dan antarmuka VPC endpoint ()AWS PrivateLink
- Subnet dan grup subnet

Amazon VPCs dan ElastiCache keamanan

Karena keamanan data itu penting, ElastiCache menyediakan cara bagi Anda untuk mengontrol siapa yang memiliki akses ke data Anda. Cara Anda mengontrol akses ke data tergantung pada apakah Anda meluncurkan cluster Anda di Amazon Virtual Private Cloud (Amazon VPC) atau Amazon -Classic atau tidak. EC2

> **Important**
>
> Kami telah mencela penggunaan Amazon EC2 -Classic untuk meluncurkan cluster. ElastiCache Semua simpul generasi terkini diluncurkan di Amazon Virtual Private Cloud saja.

Layanan Amazon Virtual Private Cloud (Amazon VPC) mendefinisikan jaringan virtual yang mirip dengan pusat data biasa. Saat Anda mengonfigurasi Amazon VPC, Anda dapat memilih rentang alamat IP, membuat subnet, serta mengonfigurasi tabel rute, gateway jaringan, dan pengaturan keamanan. Anda juga dapat menambahkan klaster cache ke jaringan virtual, dan mengontrol akses ke klaster cache menggunakan grup keamanan Amazon VPC.

Bagian ini menjelaskan cara mengonfigurasi ElastiCache cluster secara manual di VPC Amazon. Informasi ini ditujukan untuk pengguna yang menginginkan pemahaman yang lebih dalam tentang bagaimana ElastiCache dan Amazon VPC bekerja sama.

Topik

- Pengertian ElastiCache dan Amazon VPCs
- Pola Akses untuk Mengakses ElastiCache Cache di VPC Amazon
- Membuat Cloud Privat Virtual (VPC)
- Menghubungkan ke cache yang berjalan di Amazon VPC

Pengertian ElastiCache dan Amazon VPCs

ElastiCache sepenuhnya terintegrasi dengan Amazon Virtual Private Cloud (Amazon VPC). Untuk ElastiCache pengguna, ini berarti sebagai berikut:

- Jika AWS akun Anda hanya mendukung platform EC2 -VPC, ElastiCache selalu luncurkan cluster Anda di Amazon VPC.
- Jika Anda baru mengenal AWS, cluster Anda akan digunakan ke VPC Amazon. VPC default akan dibuat untuk Anda secara otomatis.
- Jika Anda memiliki VPC default dan tidak menentukan subnet saat Anda meluncurkan sebuah klaster, maka klaster itu akan diluncurkan ke Amazon VPC default Anda.

Untuk informasi selengkapnya, lihat Mendeteksi Platform Anda yang Didukung dan Apakah Anda Memiliki VPC Default.

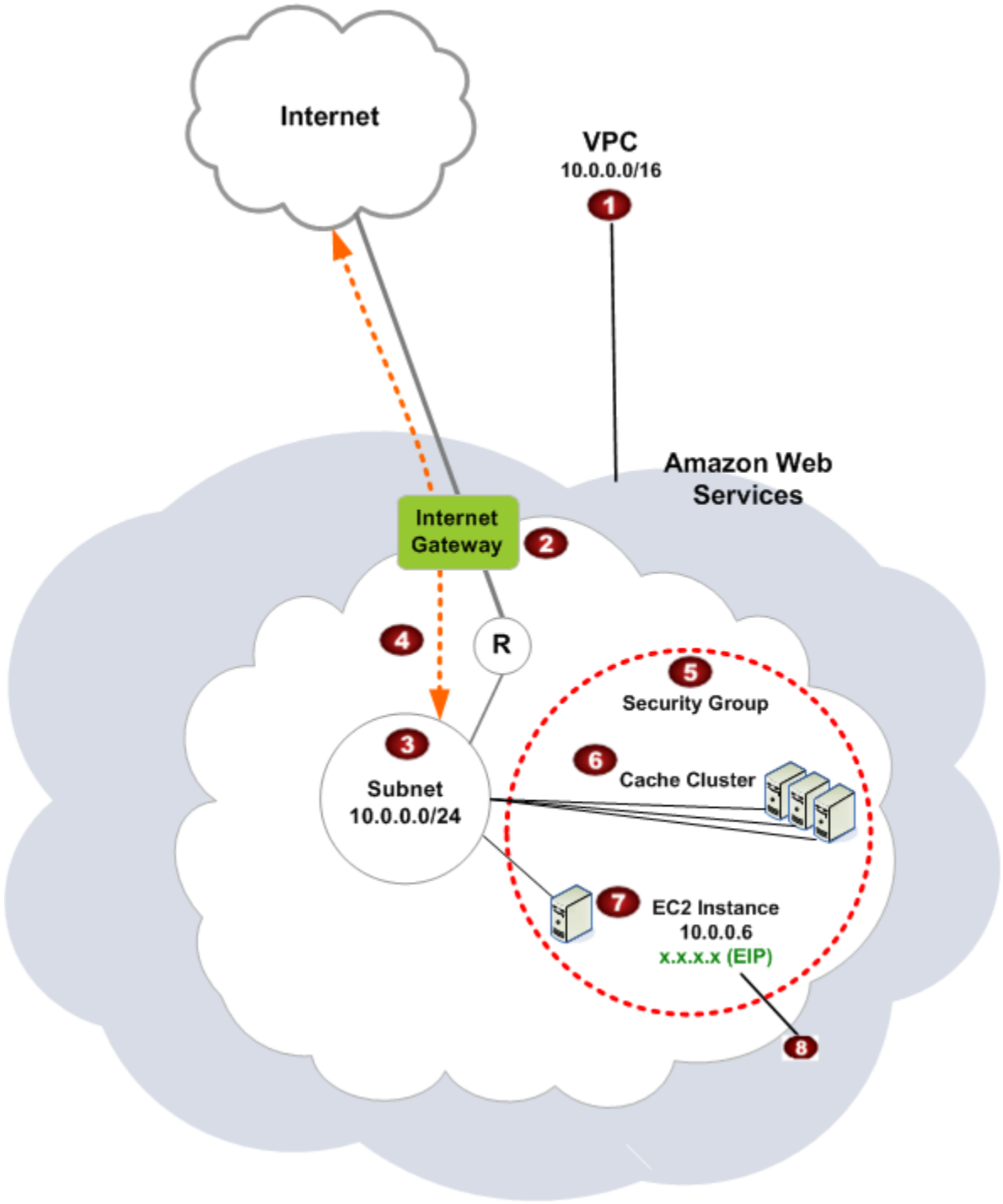
Dengan Amazon Virtual Private Cloud, Anda dapat membuat jaringan virtual di AWS cloud yang sangat mirip dengan pusat data tradisional. Anda dapat mengonfigurasi Amazon VPC Anda, termasuk memilih rentang alamat IP, membuat subnet, dan mengonfigurasi tabel rute, gateway jaringan, dan pengaturan keamanan.

Fungsionalitas ElastiCache dasarnya sama di cloud pribadi virtual; ElastiCache mengelola peningkatan perangkat lunak, penambalan, deteksi kegagalan, dan pemulihan apakah cluster Anda digunakan di dalam atau di luar VPC Amazon.

ElastiCache node cache yang digunakan di luar VPC Amazon diberi alamat IP yang namanya diselesaikan endpoint/DNS . Ini menyediakan konektivitas dari instans Amazon Elastic Compute Cloud (Amazon EC2). Saat Anda meluncurkan ElastiCache cluster ke subnet pribadi VPC Amazon, setiap node cache diberi alamat IP pribadi di dalam subnet tersebut.

Ikhtisar ElastiCache di VPC Amazon

Diagram dan tabel berikut menjelaskan lingkungan VPC Amazon, bersama dengan ElastiCache cluster dan EC2 instance Amazon yang diluncurkan di Amazon VPC.



1	Amazon VPC adalah bagian terisolasi dari AWS Cloud yang diberi blok alamat IP sendiri.
2	

	Gateway Internet menghubungkan VPC Amazon Anda langsung ke Internet dan menyediakan akses ke AWS sumber daya lain seperti Amazon Simple Storage Service (Amazon S3) yang berjalan di luar VPC Amazon Anda.
3	Subnet Amazon VPC adalah segmen dari rentang alamat IP VPC Amazon tempat Anda dapat mengisolasi AWS sumber daya sesuai dengan kebutuhan keamanan dan operasional Anda.
4	Tabel rute di Amazon VPC mengarahkan lalu lintas jaringan antara subnet dan Internet. Amazon VPC memiliki router tersirat, yang dilambangkan dalam diagram ini menggunakan lingkaran dengan R.
5	Grup keamanan Amazon VPC mengontrol lalu lintas masuk dan keluar untuk cluster ElastiCache dan instans Amazon Anda. EC2
6	Anda dapat meluncurkan ElastiCache cluster di subnet. Simpul cache memiliki alamat IP privat dari rentang alamat subnet.
7	Anda juga dapat meluncurkan EC2 instans Amazon di subnet. Setiap EC2 instans Amazon memiliki alamat IP pribadi dari rentang alamat subnet. EC2 Instans Amazon dapat terhubung ke node cache apa pun di subnet yang sama.
8	Agar EC2 instans Amazon di VPC Amazon Anda dapat dijangkau dari Internet, Anda perlu menetapkan alamat publik statis yang disebut alamat IP Elastis ke instans.

Prasyarat

Untuk membuat ElastiCache cluster dalam VPC Amazon, VPC Amazon Anda harus memenuhi persyaratan berikut:

- VPC Amazon harus mengizinkan instans Amazon EC2 yang tidak berdedikasi. Anda tidak dapat menggunakan ElastiCache VPC Amazon yang dikonfigurasi untuk penyewaan instans khusus.

- Grup subnet cache harus ditentukan untuk VPC Amazon Anda. ElastiCache menggunakan grup subnet cache tersebut untuk memilih subnet dan alamat IP dalam subnet tersebut untuk dikaitkan dengan titik akhir VPC atau node cache Anda.
- Blok CIDR untuk setiap subnet harus cukup besar untuk memberikan alamat IP cadangan ElastiCache untuk digunakan selama kegiatan pemeliharaan.

Perutean dan keamanan

Anda dapat mengonfigurasi perutean di Amazon VPC untuk mengendalikan tempat arus lalu lintas (misalnya, ke gateway Internet atau gateway privat virtual). Dengan gateway Internet, VPC Amazon Anda memiliki akses langsung ke AWS sumber daya lain yang tidak berjalan di VPC Amazon Anda. Jika Anda memilih untuk hanya memiliki gateway privat virtual dengan koneksi ke jaringan lokal dari organisasi Anda, maka Anda dapat merutekan lalu lintas dari dan ke Internet tersebut melalui VPN serta menggunakan kebijakan keamanan lokal dan firewall untuk mengontrol lalu lintas keluar. Dalam hal ini, Anda dikenakan biaya bandwidth tambahan ketika Anda mengakses AWS sumber daya melalui Internet.

Anda dapat menggunakan grup keamanan Amazon VPC untuk membantu mengamankan ElastiCache cluster dan EC2 instans Amazon di VPC Amazon Anda. Grup keamanan bertindak seperti firewall di tingkat instans, bukan di tingkat subnet.

> **Note**
>
> Sangat dianjurkan untuk menggunakan nama DNS untuk terhubung ke simpul cache Anda, karena alamat IP yang mendasarinya dapat berubah.

Dokumentasi Amazon VPC

Amazon VPC memiliki kumpulan dokumentasinya sendiri untuk menjelaskan cara membuat dan menggunakan Amazon VPC Anda. Tabel berikut memberikan tautan ke panduan Amazon VPC.

Deskripsi	Dokumentasi
Cara mulai menggunakan Amazon VPC	Mulai menggunakan Amazon VPC
Cara menggunakan Amazon VPC melalui AWS Management Console	Panduan Pengguna Amazon VPC

Deskripsi	Dokumentasi
Deskripsi lengkap dari semua perintah Amazon VPC	Referensi Baris EC2 Perintah Amazon (perintah Amazon VPC ditemukan di referensi Amazon EC2)
Deskripsi lengkap dari operasi API Amazon VPC, jenis data, dan kesalahan	Referensi Baris EC2 Perintah Amazon (operasi API VPC Amazon ditemukan di referensi Amazon EC2)
Informasi untuk administrator jaringan yang perlu mengkonfigurasi gateway di akhir koneksi IPsec VPN opsional	Apa itu AWS Site-to-Site VPN?

Untuk informasi lebih detail tentang Amazon Virtual Private Cloud, lihat Amazon Virtual Private Cloud.

Pola Akses untuk Mengakses ElastiCache Cache di VPC Amazon

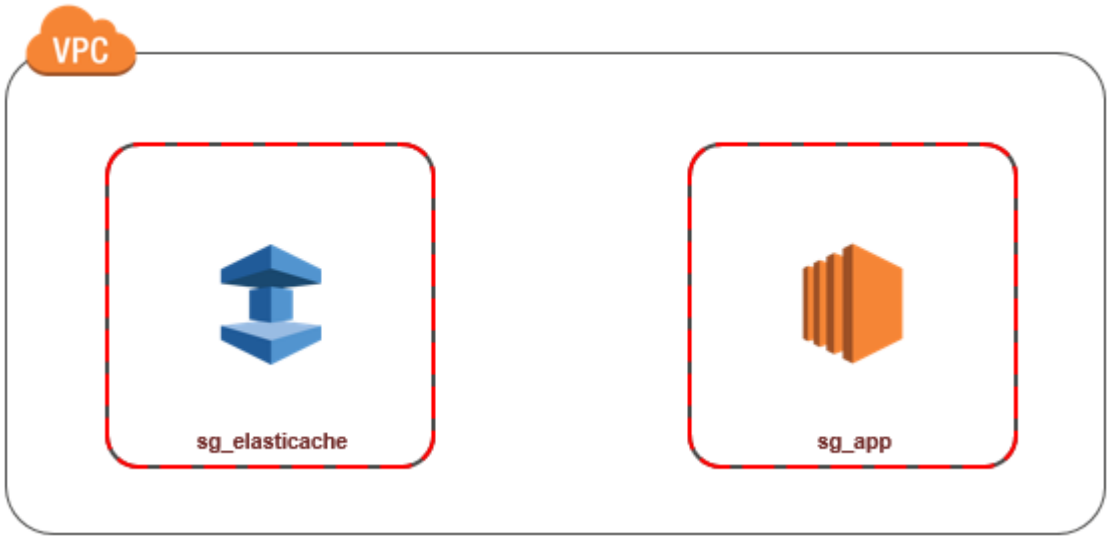
Amazon ElastiCache mendukung skenario berikut untuk mengakses cache di VPC Amazon:

Daftar Isi

Mengakses ElastiCache Cache saat dan EC2 Instans Amazon berada di VPC Amazon yang Sama

Kasus penggunaan yang paling umum adalah ketika aplikasi yang digunakan pada sebuah EC2 instance perlu terhubung ke cache di VPC yang sama.

Diagram berikut menggambarkan skenario ini.



Cara termudah untuk mengelola akses antara EC2 instance dan cache di VPC yang sama adalah dengan melakukan hal berikut:

1. Buat grup keamanan VPC untuk cache Anda. Grup keamanan ini dapat digunakan untuk membatasi akses ke cache. Contohnya, Anda dapat membuat aturan kustom untuk grup keamanan ini yang mengizinkan akses TCP menggunakan port yang Anda tetapkan untuk cache saat Anda membuatnya dan alamat IP yang Anda gunakan untuk mengakses cache tersebut.

   Port default untuk cache Memcached adalah 11211.

   Port default untuk cache Valkey dan Redis OSS adalah. 6379
2. Buat grup keamanan VPC untuk EC2 instance Anda (server web dan aplikasi). Grup keamanan ini dapat, jika diperlukan, mengizinkan akses ke EC2 instance dari Internet melalui tabel routing VPC. Misalnya, Anda dapat menetapkan aturan pada grup keamanan ini untuk mengizinkan akses TCP ke EC2 instance melalui port 22.
3. Buat aturan kustom di grup keamanan untuk cache Anda yang memungkinkan koneksi dari grup keamanan yang Anda buat untuk EC2 instance Anda. Hal ini akan mengizinkan semua anggota grup keamanan untuk mengakses cache.

> **Note**
>
> Jika Anda berencana untuk menggunakan Zona Lokal, pastikan Anda telah mengaktifkannya. Saat Anda membuat grup subnet di zona lokal, VPC Anda diperluas ke Zona Lokal tersebut

dan VPC Anda akan memperlakukan subnet itu seperti subnet lain di Zona Ketersediaan lainnya. Semua gateway dan tabel rute yang berkaitan akan disesuaikan secara otomatis.

Untuk membuat aturan dalam grup keamanan VPC yang memungkinkan koneksi dari grup keamanan lain

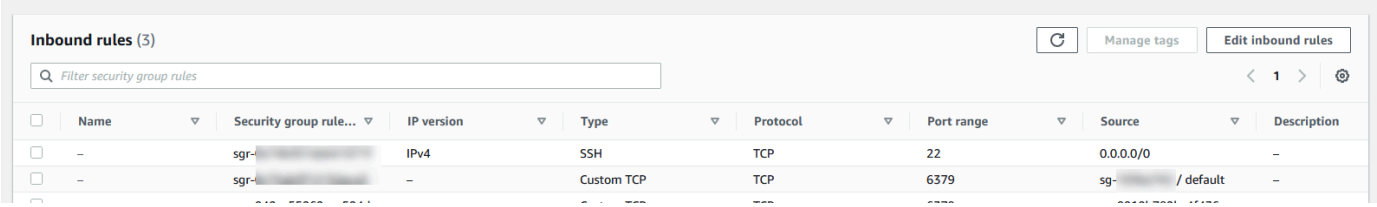
1. Masuk ke Konsol AWS Manajemen dan buka konsol VPC Amazon di https://console.aws.amazon.com /vpc.
2. Pada panel navigasi, pilih Grup Keamanan.
3. Pilih atau buat grup keamanan yang akan Anda gunakan untuk cache Anda. Pada Aturan Masuk, pilih Edit Aturan Masuk lalu pilih Tambahkan Aturan. Grup keamanan ini akan mengizinkan akses bagi anggota dari grup keamanan lain.
4. Dari Jenis, pilih Aturan TCP Kustom.

   a. Untuk Rentang Port, tentukan port yang Anda gunakan saat membuat cache.

      Port default untuk cache Memcached adalah `11211`.

      Port default untuk cache Valkey dan Redis OSS dan grup replikasi adalah. `6379`

   b. Pada kotak Sumber, masukkan ID dari grup keamanan. Dari daftar pilih grup keamanan yang akan Anda gunakan untuk EC2 instans Amazon Anda.
5. Pilih Simpan jika selesai.



Mengakses ElastiCache Cache saat itu dan EC2 Instans Amazon berada di Amazon yang Berbeda VPCs

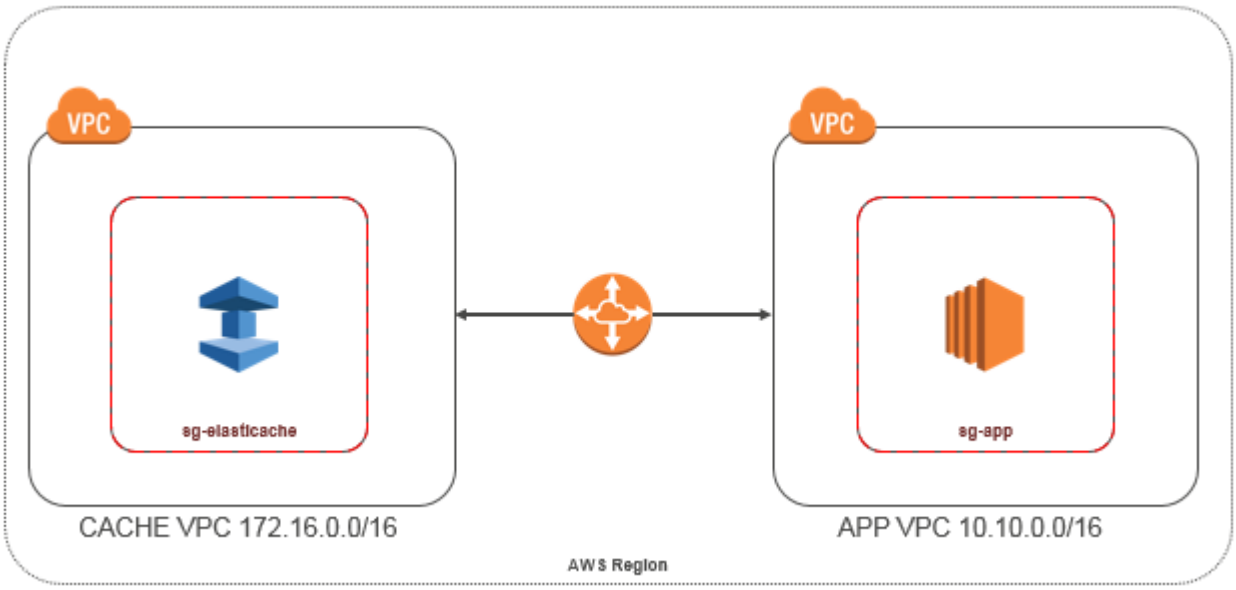
Ketika cache Anda berada di VPC yang berbeda dari EC2 instance yang Anda gunakan untuk mengaksesnya, ada beberapa cara untuk mengakses cache. Jika cache dan EC2 instance berbeda VPCs tetapi di wilayah yang sama, Anda dapat menggunakan peering VPC. Jika cache dan EC2 instance berada di wilayah yang berbeda, Anda dapat membuat konektivitas VPN antar wilayah.

Topik

- Mengakses ElastiCache Cache saat dan EC2 Instans Amazon berada di Amazon yang Berbeda VPCs di Wilayah yang Sama
- Mengakses ElastiCache Cache saat itu dan EC2 Instans Amazon berada di Amazon yang Berbeda di Wilayah VPCs yang Berbeda

Mengakses ElastiCache Cache saat dan EC2 Instans Amazon berada di Amazon yang Berbeda VPCs di Wilayah yang Sama

Diagram berikut menggambarkan mengakses cache oleh EC2 instans Amazon di VPC Amazon yang berbeda di wilayah yang sama menggunakan koneksi peering VPC Amazon.



Cache diakses oleh EC2 instans Amazon di VPC Amazon yang berbeda dalam Wilayah yang sama - VPC Peering Connection

Koneksi peering VPC adalah koneksi jaringan antara dua VPCs yang memungkinkan Anda untuk merutekan lalu lintas di antara mereka menggunakan alamat IP pribadi. Instans di kedua VPC tersebut dapat berkomunikasi satu sama lain seolah berada di jaringan yang sama. Anda dapat membuat koneksi peering VPC antara Amazon Anda sendiri VPCs, atau dengan VPC Amazon di AWS akun lain dalam satu wilayah. Untuk mempelajari selengkapnya peering Amazon VPC, lihat Dokumentasi VPC.

> **Note**
>
> Resolusi nama DNS mungkin gagal untuk peered VPCs, tergantung pada konfigurasi yang diterapkan ke VPC. ElastiCache Untuk mengatasi ini, keduanya VPCs harus diaktifkan untuk nama host DNS dan resolusi DNS. Untuk informasi selengkapnya, lihat Mengaktifkan resolusi DNS untuk koneksi peering VPC.

Untuk mengakses cache di Amazon VPC yang berbeda melalui peering

1. Pastikan keduanya VPCs tidak memiliki rentang IP yang tumpang tindih atau Anda tidak akan dapat mengintip mereka.
2. Peer keduanya VPCs. Untuk informasi selengkapnya, lihat Membuat dan Menerima Koneksi Peering VPC Amazon.
3. Perbarui tabel rute Anda. Untuk informasi selengkapnya, lihat Memperbarui Tabel Rute Anda untuk Koneksi Peering VPC

   Berikut adalah tampilan tabel rute untuk contoh pada diagram sebelumnya. Perhatikan bahwa pcx-a894f1c1 adalah koneksi peering.

Destination	Target
172.16.0.0/16	local
10.10.0.0/16	pcx-a894f1c1

Destination	Target
10.10.0.0/16	local
0.0.0.0/0	igw-bfdcccd8
172.16.0.0/16	pcx-a894f1c1

   Tabel Perutean VPC
4. Ubah Grup Keamanan ElastiCache cache Anda untuk mengizinkan koneksi masuk dari grup keamanan Aplikasi di VPC peered. Untuk informasi selengkapnya, lihat Grup Keamanan VPC Peer Referensi.

Akses cache melalui koneksi peering akan dikenai biaya transfer data tambahan.

Menggunakan Gateway Transit

Gateway transit memungkinkan Anda untuk melampirkan VPCs dan koneksi VPN di AWS Wilayah yang sama dan merutekan lalu lintas di antara mereka. Gateway transit berfungsi di seluruh AWS

akun, dan Anda dapat menggunakan AWS Resource Access Manager untuk berbagi gateway transit Anda dengan akun lain. Setelah Anda berbagi gateway transit dengan AWS akun lain, pemilik akun dapat melampirkannya VPCs ke gateway transit Anda. Pengguna dari kedua akun ini dapat menghapus lampiran VPC tersebut kapan saja.

Anda dapat mengaktifkan multicast pada gateway transit, lalu membuat domain multicast gateway transit yang memungkinkan lalu lintas multicast dikirim dari sumber multicast Anda untuk anggota grup multicast melalui lampiran VPC yang Anda kaitkan dengan domain.

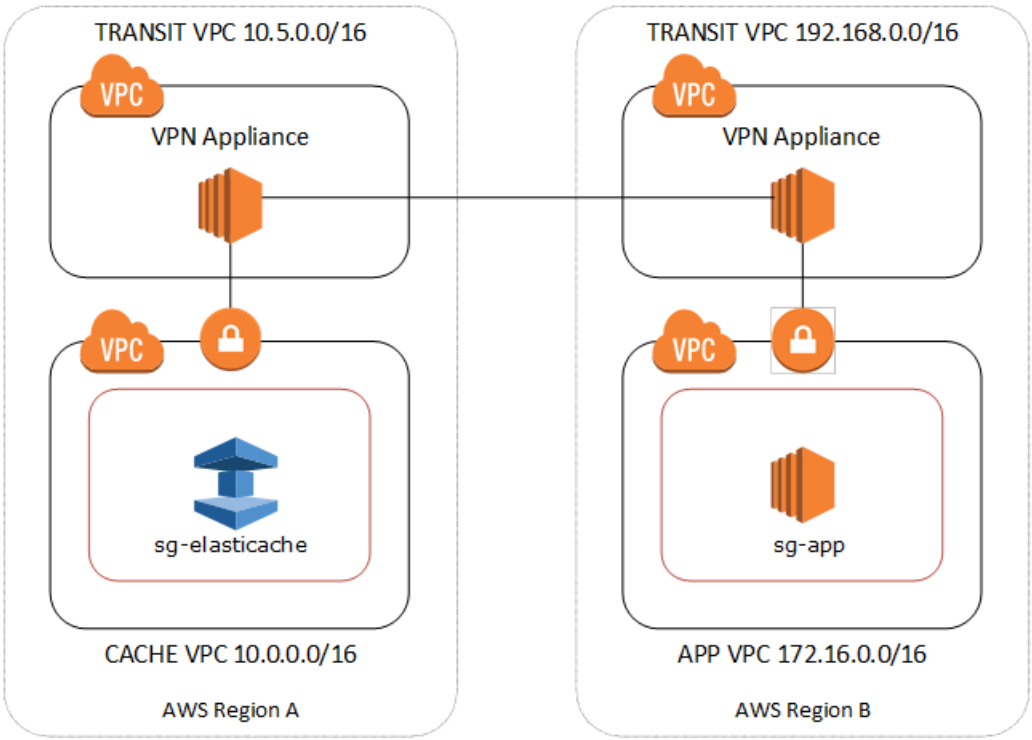
Anda juga dapat membuat lampiran koneksi peering antara gateway transit di Wilayah yang berbeda. AWS Hal ini memungkinkan Anda merutekan lalu lintas di antara beberapa lampiran gateway transit di Wilayah yang berbeda.

Untuk informasi selengkapnya, lihat Gateway transit.

Mengakses ElastiCache Cache saat itu dan EC2 Instans Amazon berada di Amazon yang Berbeda di Wilayah VPCs yang Berbeda

Menggunakan VPC Transit

Alternatif untuk menggunakan VPC peering, strategi umum lainnya untuk menghubungkan beberapa jaringan, tersebar secara geografis VPCs dan jarak jauh adalah dengan membuat VPC transit yang berfungsi sebagai pusat transit jaringan global. VPC transit menyederhanakan manajemen jaringan dan meminimalkan jumlah koneksi yang diperlukan untuk menghubungkan beberapa VPCs jaringan dan jarak jauh. Rancangan ini dapat menghemat waktu dan tenaga serta mengurangi biaya karena diimplementasikan secara virtual tanpa biaya yang biasa diperlukan untuk membangun kehadiran fisik di hub transit kolokasi atau men-deploy peralatan jaringan fisik.



Menghubungkan VPCs di berbagai wilayah

Setelah VPC Transit Amazon dibuat, aplikasi yang digunakan dalam VPC “spoke” di satu wilayah dapat terhubung ke ElastiCache cache di VPC “spoke” di wilayah lain.

Untuk mengakses cache di VPC yang berbeda dalam Wilayah yang berbeda AWS

1. Lakukan deployment Solusi VPC Transit. Untuk informasi selengkapnya, lihat AWS Transit Gateway.
2. Perbarui tabel perutean VPC di App dan Cache VPCs untuk merutekan lalu lintas melalui VGW (Virtual Private Gateway) dan VPN Appliance. Dalam kasus Perutean Dinamis dengan Protokol Gateway Batas (BGP), rute Anda dapat disebarkan secara otomatis.
3. Ubah Grup Keamanan ElastiCache cache Anda untuk memungkinkan koneksi masuk dari rentang IP instance Aplikasi. Perhatikan bahwa Anda tidak akan dapat mereferensikan Grup Keamanan server aplikasi dalam skenario ini.

Akses cache antar-wilayah akan menimbulkan latensi jaringan dan biaya transfer data lintas wilayah tambahan.

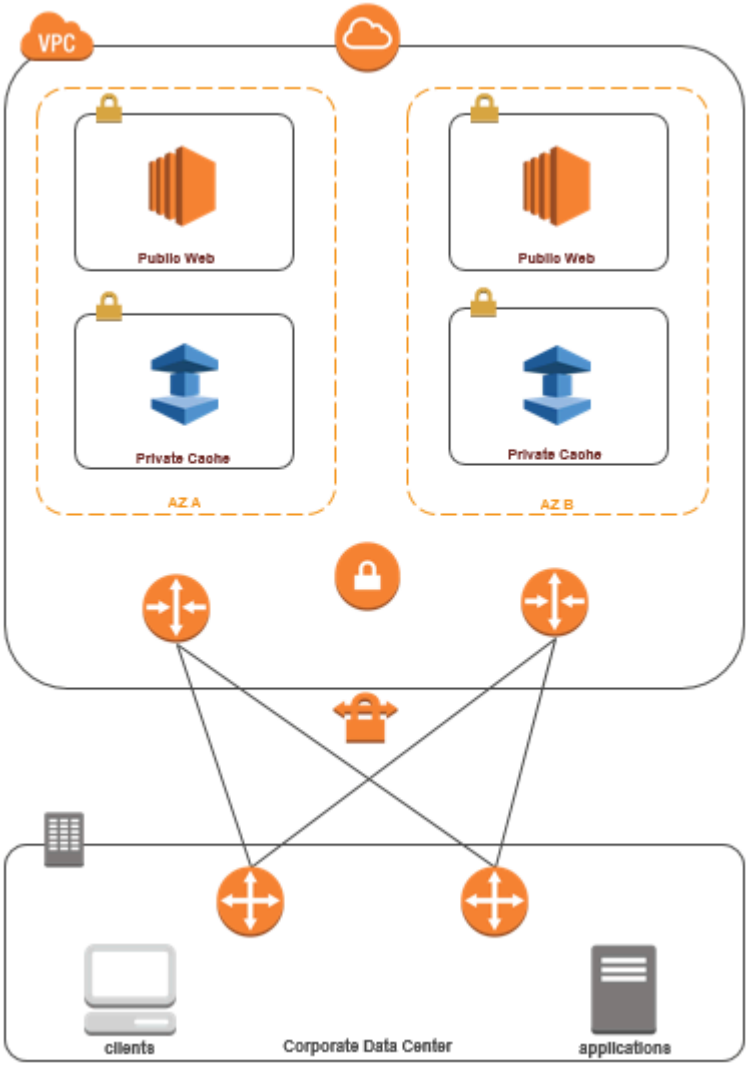
Mengakses ElastiCache Cache dari Aplikasi yang Berjalan di Pusat Data Pelanggan

Skenario lain yang mungkin adalah arsitektur Hybrid di mana klien atau aplikasi di pusat data pelanggan mungkin perlu mengakses ElastiCache cache di VPC. Skenario ini juga didukung jika ada konektivitas antara VPC pelanggan dan pusat data baik melalui VPN atau Direct Connect.

Topik

Mengakses ElastiCache Cache dari Aplikasi yang Berjalan di Pusat Data Pelanggan Menggunakan Konektivitas VPN

Diagram berikut menggambarkan mengakses ElastiCache cache dari aplikasi yang berjalan di jaringan perusahaan Anda menggunakan koneksi VPN.



Menghubungkan ke ElastiCache dari pusat data Anda melalui VPN

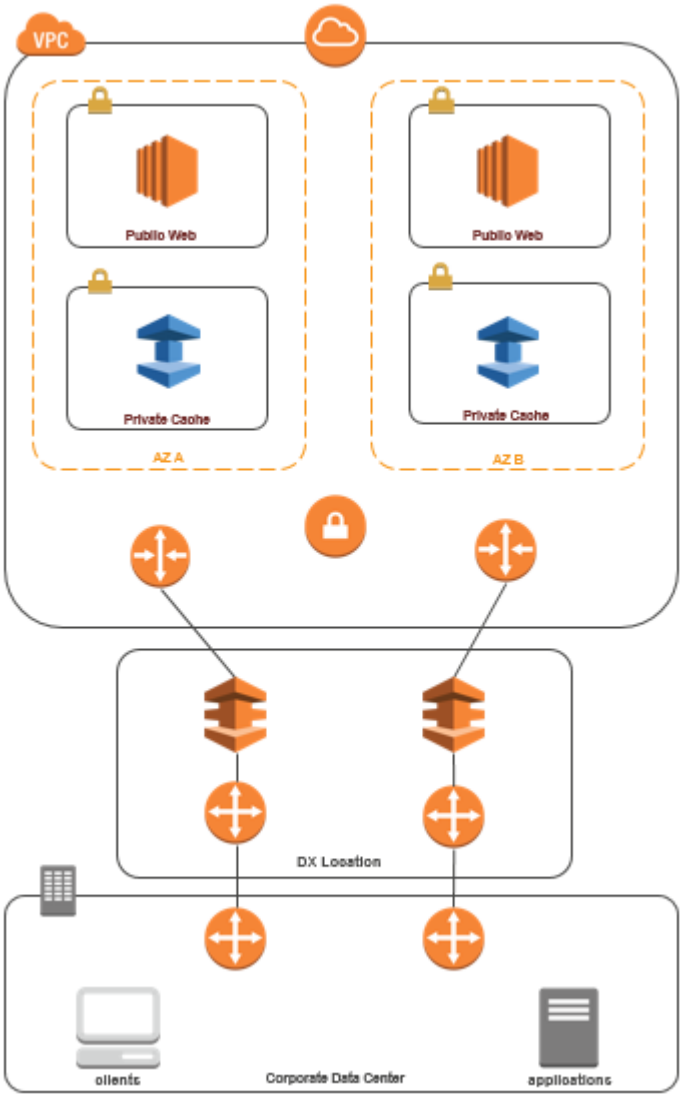
Untuk mengakses cache di VPC dari aplikasi on-premise melalui koneksi VPN

1. Buat Konektivitas VPN dengan menambahkan Gateway Privat Virtual perangkat keras ke VPC Anda. Untuk informasi selengkapnya, lihat Menambahkan Gateway Privat Virtual Perangkat Keras ke VPC Anda.
2. Perbarui tabel perutean VPC untuk subnet tempat ElastiCache cache Anda digunakan untuk memungkinkan lalu lintas dari server aplikasi lokal Anda. Dalam kasus Perutean Dinamis dengan BGP, rute Anda dapat disebarkan secara otomatis.
3. Ubah Grup Keamanan ElastiCache cache Anda untuk mengizinkan koneksi masuk dari server aplikasi lokal.

Akses cache melalui koneksi VPN akan menimbulkan latensi jaringan dan biaya transfer data tambahan.

Mengakses ElastiCache Cache dari Aplikasi yang Berjalan di Pusat Data Pelanggan Menggunakan Direct Connect

Diagram berikut menggambarkan mengakses ElastiCache cache dari aplikasi yang berjalan di jaringan perusahaan Anda menggunakan Direct Connect.



Menghubungkan ke ElastiCache dari pusat data Anda melalui Direct Connect

Untuk mengakses ElastiCache cache dari aplikasi yang berjalan di jaringan Anda menggunakan Direct Connect

1. Buat konektivitas Direct Connect. Untuk informasi selengkapnya, lihat Memulai dengan AWS Direct Connect.
2. Ubah Grup Keamanan ElastiCache cache Anda untuk mengizinkan koneksi masuk dari server aplikasi lokal.

Akses cache melalui koneksi DX dapat menimbulkan latensi jaringan dan biaya transfer data tambahan.

Membuat Cloud Privat Virtual (VPC)

Dalam contoh ini, Anda membuat Amazon VPC dengan subnet privat untuk setiap Zona Ketersediaan.

Membuat Amazon VPC (Konsol)

1. Masuk ke Konsol AWS Manajemen, dan buka konsol VPC Amazon di. https://console.aws.amazon.com/vpc/
2. Pada dasbor VPC, pilih Buat VPC.
3. Di bagian Sumber daya yang akan dibuat, pilih VPC dan lainnya.
4. Di bawah Jumlah Availability Zones (AZs), pilih jumlah Availability Zone yang ingin Anda luncurkan subnet.
5. Di bagian Jumlah subnet publik, pilih jumlah subnet publik yang ingin Anda tambahkan ke VPC Anda.
6. Di bagian Jumlah subnet privat, pilih jumlah subnet privat yang ingin Anda tambahkan ke VPC Anda.

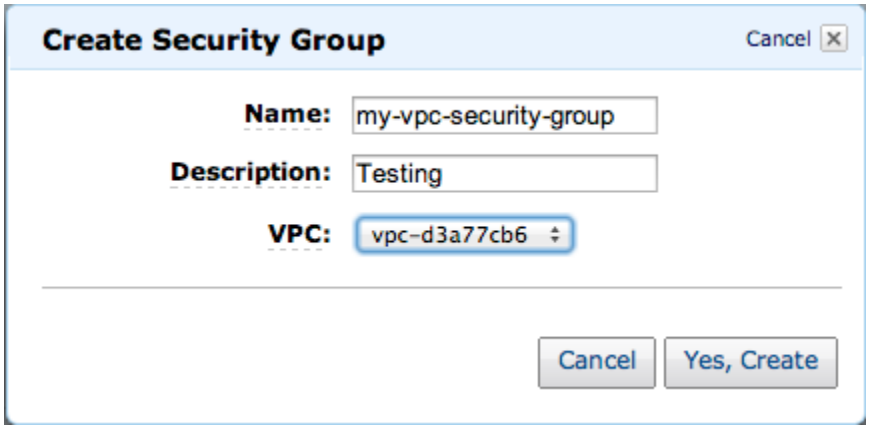
   > **Tip**
   >
   > Perhatikan pengidentifikasi subnet Anda, serta identifikasi mana yang publik dan privat. Anda akan memerlukan informasi ini nanti saat meluncurkan cluster dan menambahkan EC2 instance Amazon ke VPC Amazon Anda.

7. Buat grup keamanan Amazon VPC. Anda akan menggunakan grup ini untuk cluster cache dan EC2 instans Amazon Anda.

   a. Pada panel navigasi Konsol Manajemen Amazon VPC, pilih Grup Keamanan.

   b. Pilih Buat Grup Keamanan.

   c. Ketikkan nama dan deskripsi untuk grup keamanan Anda di kotak yang sesuai. Di kotak VPC, pilih pengidentifikasi untuk Amazon VPC Anda.



d. Jika pengaturan sudah sesuai keinginan Anda, pilih Ya, Buat.

8. Tentukan aturan masuk jaringan untuk grup keamanan Anda. Aturan ini akan memungkinkan Anda untuk terhubung ke EC2 instans Amazon Anda menggunakan Secure Shell (SSH).

   a. Di daftar navigasi, pilih Grup Keamanan.

   b. Temukan grup keamanan Anda dalam daftar, lalu pilih grup tersebut.

   c. Di bagian Grup Keamanan, pilih tab Masuk. Di kotak Buat aturan baru, pilih SSH, lalu pilih Tambahkan Aturan.

   d. Tetapkan nilai berikut untuk aturan masuk baru yang akan mengizinkan akses HTTP:

      - Jenis: HTTP
      - Sumber: 0.0.0.0/0

      Pilih Terapkan Perubahan Aturan.

Sekarang Anda sudah siap untuk membuat grup subnet cache dan meluncurkan klaster cache di Amazon VPC Anda.

- Membuat grup subnet
- Membuat klaster Memcached (konsol).
- Membuat cluster Valkey (mode cluster dinonaktifkan) (Konsol).

Menghubungkan ke cache yang berjalan di Amazon VPC

Contoh ini menunjukkan cara meluncurkan EC2 instans Amazon di VPC Amazon Anda. Anda kemudian dapat masuk ke instance ini dan mengakses ElastiCache cache yang berjalan di Amazon VPC.

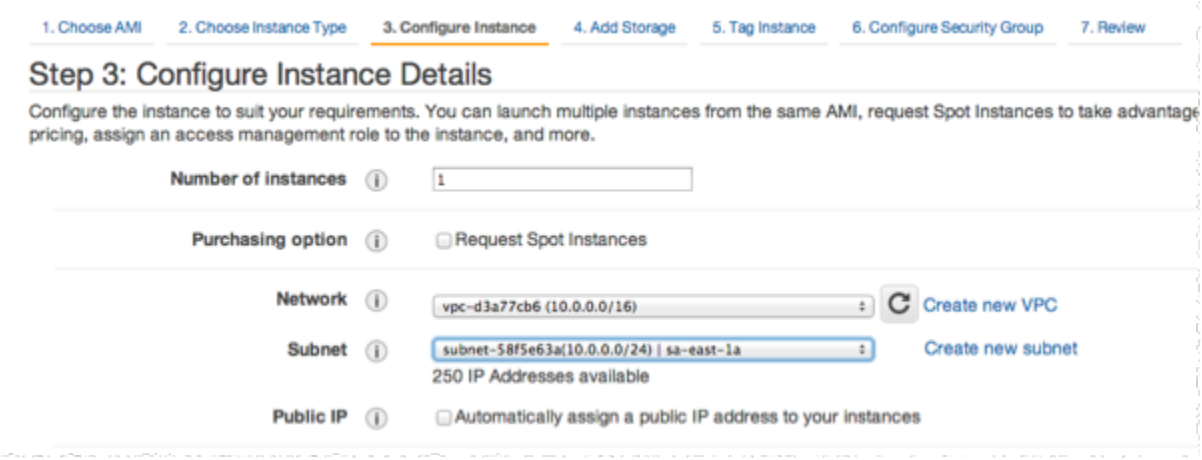
Menghubungkan ke cache yang berjalan di Amazon VPC (Konsol)

Dalam contoh ini, Anda membuat EC2 instance Amazon di VPC Amazon Anda. Anda dapat menggunakan EC2 instance Amazon ini untuk terhubung ke node cache yang berjalan di Amazon VPC.

> **Note**
>
> Untuk informasi tentang menggunakan Amazon EC2, lihat Panduan EC2 Memulai Amazon di EC2 dokumentasi Amazon.

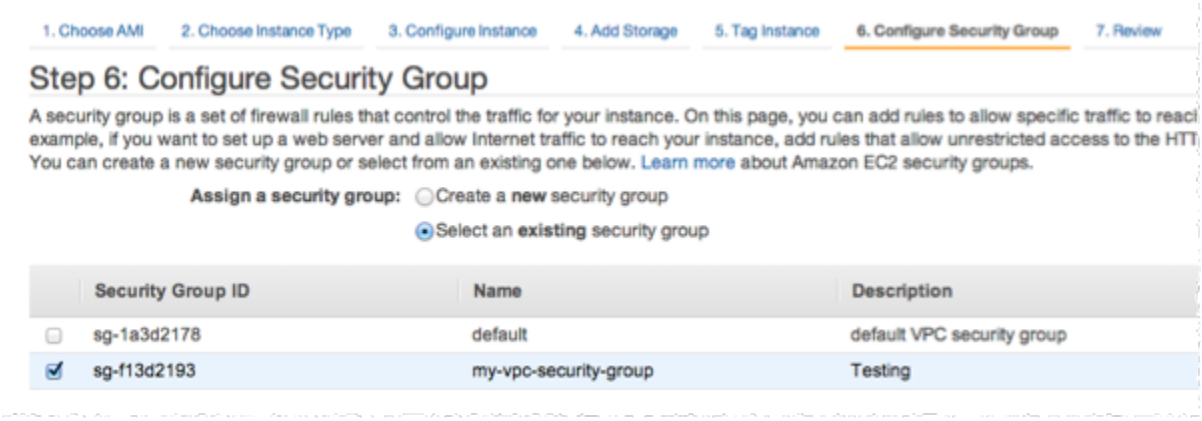
Untuk membuat EC2 instans Amazon di VPC Amazon Anda menggunakan konsol Amazon EC2

1. Masuk ke AWS Management Console dan buka EC2 konsol Amazon di https://console.aws.amazon.com/ec2/.
2. Pada konsol, pilih Luncurkan Instans dan ikuti langkah-langkah ini:
3. Di halaman Pilih Amazon Machine Image (AMI), pilih AMI Amazon Linux 64-bit, lalu klik Pilih.
4. Pada halaman Pilih Jenis Instans, pilih 3. Konfigurasi Instans.
5. Pada halaman Konfigurasi Detail Instans, buat pilihan berikut:
    a. Di daftar Jaringan, pilih Amazon VPC Anda.
    b. Di daftar Subnet, pilih subnet publik Anda.



Jika pengaturan sudah sesuai keinginan Anda, pilih 4. Tambahkan Penyimpanan.

6. Pada halaman Tambahkan Penyimpanan, pilih 5. Berikan Tag pada Instans.
7. Pada halaman Tag Instance, ketikkan nama untuk EC2 instans Amazon Anda, lalu pilih 6. Konfigurasi Grup Keamanan.
8. Pada halaman Konfigurasi Grup Keamanan, klik Pilih grup keamanan yang sudah ada. Untuk informasi selengkapnya tentang grup keamanan, lihat Grup EC2 keamanan Amazon untuk instans Linux.



Pilih nama grup keamanan Amazon VPC Anda, lalu pilih Tinjau dan Luncurkan.

9. Di halaman Tinjau Instans dan Luncurkan, pilih Luncurkan.
10. Di jendela Pilih pasangan kunci yang sudah ada atau buat pasangan kunci baru, tentukan pasangan kunci yang ingin Anda gunakan dengan instans ini.

> **Note**
>
> Untuk informasi tentang mengelola pasangan kunci, lihat Panduan EC2 Memulai Amazon.

11. Saat Anda siap meluncurkan EC2 instans Amazon Anda, pilih Luncurkan.

Anda sekarang dapat menetapkan alamat IP Elastis ke EC2 instans Amazon yang baru saja Anda buat. Anda perlu menggunakan alamat IP ini untuk terhubung ke EC2 instans Amazon.

Untuk menetapkan alamat IP elastis (Konsol)

1. Buka konsol VPC Amazon di. https://console.aws.amazon.com/vpc/
2. Dalam daftar navigasi, pilih Elastis IPs.
3. Pilih Alokasikan alamat IP Elastis.
4. Di kotak dialog Alokasi Alamat IP Elastis, terima Grup Perbatasan Jaringan default lalu pilih Alokasikan.
5. Pilih alamat IP Elastis yang baru saja Anda alokasikan dari daftar lalu pilih Kaitkan Alamat.
6. Di kotak dialog Alamat Asosiasi, di kotak Instans, pilih ID EC2 instans Amazon yang Anda luncurkan.

   Di kotak Alamat IP privat, pilih kotak untuk mendapatkan alamat IP privat lalu pilih Kaitkan.

   Anda sekarang dapat menggunakan SSH untuk terhubung ke EC2 instans Amazon menggunakan alamat IP Elastis yang Anda buat.

Untuk terhubung ke EC2 instans Amazon Anda

- Buka jendela perintah. Pada prompt perintah, keluarkan perintah berikut, dengan mengganti mykeypair.pem dengan nama file pasangan kunci Anda dan 54.207.55.251 dengan alamat IP Elastis Anda.

```
ssh -i mykeypair.pem ec2-user@54.207.55.251
```

> **Important**
>
> Jangan keluar dari EC2 instans Amazon Anda.

Anda sekarang siap untuk berinteraksi dengan ElastiCache cluster Anda. Sebelum Anda dapat melakukannya, Anda perlu menginstal utilitas telnet jika belum melakukannya.

Untuk menginstal telnet dan berinteraksi dengan klaster cache Anda (AWS CLI)

- Buka jendela perintah. Di prompt perintah, keluarkan perintah berikut. Di prompt konfirmasi, ketik y.

```
sudo yum install telnet
Loaded plugins: priorities, security, update-motd, upgrade-helper
Setting up Install Process
Resolving Dependencies
--> Running transaction check

...(output omitted)...

Total download size: 63 k
Installed size: 109 k
Is this ok [y/N]: y
Downloading Packages:
telnet-0.17-47.7.amzn1.x86_64.rpm                          |  63 kB     00:00

...(output omitted)...

Complete!
```

Anda sekarang dapat terhubung ke VPC dengan Memcached atau Redis.

Menghubungkan ke VPC dengan Memcached

1. Buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/dan dapatkan titik akhir untuk salah satu node di cluster cache Anda. Untuk informasi selengkapnya, lihat Menemukan Titik Akhir Koneksi.

2. Gunakan telnet untuk terhubung ke titik akhir simpul cache Anda melalui port 11211. Ganti nama host yang ditunjukkan di bawah ini dengan nama host dari simpul cache Anda.

```
telnet my-cache-cluster.7wufxa.0001.use1.cache.amazonaws.com 11211
```

Anda sekarang terhubung ke mesin cache dan dapat mengeluarkan perintah. Dalam contoh ini, Anda menambahkan item data ke cache lalu mendapatkannya segera sesudahnya. Terakhir, Anda akan memutuskan koneksi dari simpul cache.

Untuk menyimpan kunci dan nilai, ketik dua baris berikut:

```
add mykey 0 3600 28
This is the value for mykey
```

Mesin cache merespons dengan berikut ini:

```
OK
```

Untuk mengambil nilai untuk `mykey`, ketik berikut ini:

```
get mykey
```

Mesin cache merespons dengan berikut ini:

```
VALUE mykey 0 28
This is the value for my key
END
```

Untuk memutuskan koneksi dari mesin cache, ketik yang berikut ini:

```
quit
```

Menghubungkan ke VPC dengan Redis

1. Buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/dan dapatkan titik akhir untuk salah satu node di cluster cache Anda. Untuk informasi selengkapnya, lihat Menemukan titik akhir koneksi untuk Redis.

2. Gunakan telnet untuk terhubung ke titik akhir simpul cache Anda melalui port 6379. Ganti nama host yang ditunjukkan di bawah ini dengan nama host dari simpul cache Anda.

   ```
   telnet my-cache-cluster.7wufxa.0001.use1.cache.amazonaws.com 6379
   ```

   Anda sekarang terhubung ke mesin cache dan dapat mengeluarkan perintah. Dalam contoh ini, Anda menambahkan item data ke cache lalu mendapatkannya segera sesudahnya. Terakhir, Anda akan memutuskan koneksi dari simpul cache.

   Untuk menyimpan kunci dan nilai, ketik dua baris berikut:

   ```
   set mykey myvalue
   ```

   Mesin cache merespons dengan berikut ini:

   ```
   OK
   ```

   Untuk mengambil nilai untuk `mykey`, ketik berikut ini:

   ```
   get mykey
   ```

   Untuk memutuskan koneksi dari mesin cache, ketik yang berikut ini:

   ```
   quit
   ```

3. Buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/dan dapatkan titik akhir untuk salah satu node di cluster cache Anda. Untuk informasi lebih lanjut, Menemukan titik akhir koneksi untuk Redis OSS.
4. Gunakan telnet untuk terhubung ke titik akhir simpul cache Anda melalui port 6379. Ganti nama host yang ditunjukkan di bawah ini dengan nama host dari simpul cache Anda.

   ```
   telnet my-cache-cluster.7wufxa.0001.use1.cache.amazonaws.com 6379
   ```

   Anda sekarang terhubung ke mesin cache dan dapat mengeluarkan perintah. Dalam contoh ini, Anda menambahkan item data ke cache lalu mendapatkannya segera sesudahnya. Terakhir, Anda akan memutuskan koneksi dari simpul cache.

   Untuk menyimpan kunci dan nilai, ketik yang berikut ini:

```
set mykey myvalue
```

Mesin cache merespons dengan berikut ini:

```
OK
```

Untuk mengambil nilai untuk `mykey`, ketik berikut ini:

```
get mykey
```

Mesin cache merespons dengan berikut ini:

```
get mykey
myvalue
```

Untuk memutuskan koneksi dari mesin cache, ketik yang berikut ini:

```
quit
```

> **Important**
>
> Untuk menghindari biaya tambahan pada AWS akun Anda, pastikan untuk menghapus AWS sumber daya apa pun yang tidak lagi Anda inginkan setelah mencoba contoh-contoh ini.

ElastiCache API dan antarmuka VPC endpoint ()AWS PrivateLink

Anda dapat membuat koneksi pribadi antara titik akhir VPC dan Amazon ElastiCache API dengan membuat titik akhir VPC antarmuka. Endpoint antarmuka didukung oleh AWS PrivateLink. AWS PrivateLink memungkinkan Anda mengakses operasi Amazon ElastiCache API secara pribadi tanpa gateway internet, perangkat NAT, koneksi VPN, atau koneksi Direct AWS Connect.

Instans di VPC Anda tidak memerlukan alamat IP publik untuk berkomunikasi dengan titik akhir ElastiCache Amazon API. Instans Anda juga tidak memerlukan alamat IP publik untuk menggunakan salah satu operasi ElastiCache API yang tersedia. Lalu lintas antara VPC dan Amazon Anda

ElastiCache tidak meninggalkan jaringan Amazon. Setiap titik akhir antarmuka direpresentasikan oleh satu atau beberapa antarmuka jaringan elastis di subnet Anda. Untuk informasi selengkapnya tentang antarmuka jaringan elastis, lihat Antarmuka jaringan elastis di EC2 Panduan Pengguna Amazon.

- Untuk informasi selengkapnya tentang titik akhir VPC, lihat Titik akhir VPC Antarmuka () di AWS PrivateLink Panduan Pengguna Amazon VPC.
- Untuk informasi selengkapnya tentang operasi ElastiCache API, lihat Operasi ElastiCache API.

Setelah Anda membuat antarmuka VPC endpoint, jika Anda mengaktifkan nama host DNS pribadi untuk titik akhir, titik akhir default (https://elasticache. ElastiCache *Region*.amazonaws.com) menyelesaikan ke titik akhir VPC Anda. Jika Anda tidak mengaktifkan nama host DNS privat, Amazon VPC menyediakan nama titik akhir DNS yang dapat Anda gunakan dalam format berikut:

```
VPC_Endpoint_ID.elasticache.Region.vpce.amazonaws.com
```

Untuk informasi selengkapnya, lihat Titik Akhir VPC Antarmuka (AWS PrivateLink) di Panduan Pengguna Amazon VPC. ElastiCache mendukung panggilan ke semua Tindakan API di dalam VPC Anda.

> **Note**
>
> Nama host DNS privat dapat diaktifkan hanya untuk satu titik akhir VPC di VPC. Jika Anda ingin membuat titik akhir VPC tambahan maka nama host DNS pribadi harus dinonaktifkan untuk titik akhir tersebut.

Tersedia Wilayah Privatelink

Kode	Lokasi	Wilayah
CPT	Afrika (Cape Town)	AF-SELATAN-1
HKG	Asia Pasifik (Hong Kong)	AP-TIMUR-1
TPE	Asia Pasifik (Taipei)	AP-TIMUR-2
NRT	Asia Pasifik (Tokyo)	AP-TIMUR LAUT-1

Kode	Lokasi	Wilayah
ICN	Asia Pasifik (Seoul)	AP-TIMUR LAUT-2
KIX	Asia Pacific (Osaka)	AP-TIMUR LAUT-3
BOM	Asia Pasifik (Mumbai)	AP-SELATAN-1
HYD	Asia Pasifik (Hyderabad)	AP-SELATAN-2
DOSA	Asia Pasifik (Singapura)	AP-TENGGARA
SYD	Asia Pasifik (Sydney)	AP-TENGGARA 2
CGK	Asia Pasifik (Jakarta)	AP-TENGGARA 3
MEL	Asia Pasifik (Melbourne)	AP-TENGGARA 4
KUL	Asia Pasifik (Malaysia)	AP-TENGGARA 5
BKK	Asia Pasifik (Thailand)	AP-TENGGARA 7
YUL	Kanada (Pusat)	CA-PUSAT-1
YYC	Kanada Barat (Calgary)	CA-BARAT-1
BJS	Tiongkok (Beijing)	CN-UTARA-1
ZHY	Tiongkok (Ningxia)	CN-BARAT LAUT-1
FRA	Eropa (Frankfurt)	EU-SENTRAL-1
ZRH	Eropa (Zürich)	EU-SENTRAL-2
ARN	Eropa (Stockholm)	EU-UTARA-1
MXP	Eropa (Milan)	EU-SELATAN-1
ZAZ	Eropa (Spanyol)	EU-SELATAN-2
SULIH SUARA	Eropa (Irlandia)	EU-BARAT-1

Kode	Lokasi	Wilayah
LHR	Eropa (London)	EU-BARAT-2
CDG	Eropa (Paris)	EU-BARAT-3
TLV	Tel Aviv (Israel)	PUSAT-1
DXB	Timur Tengah (UEA)	SAYA-PUSAT-1
BAH	Timur Tengah (Bahrain)	SAYA-SELATAN-1
QRO	Meksiko (Tengah)	MX-PUSAT-1
GRU	Amerika Selatan (Sao Paulo)	SA-TIMUR-1
IAD	US East (Northern Virginia)	KAMI-TIMUR-1
CMH	AS Timur (Ohio)	KAMI-TIMUR-2
OSU	AWS GovCloud (AS-Timur)	KAMI-GOV-TIMUR-1
SFO	AS Barat (California Utara)	KAMI-BARAT-1
PDX	AS Barat (Oregon)	KAMI-BARAT-2
PDT	AWS GovCloud (AS-Barat)	KAMI-BARAT-1

Pertimbangan untuk titik akhir VPC

Sebelum menyiapkan titik akhir VPC antarmuka untuk titik akhir Amazon ElastiCache API, pastikan Anda meninjau properti dan batasan titik akhir Antarmuka di Panduan Pengguna Amazon VPC. Semua operasi ElastiCache API yang relevan untuk mengelola ElastiCache sumber daya Amazon tersedia dari VPC Anda menggunakan. AWS PrivateLink

Kebijakan titik akhir VPC didukung untuk ElastiCache titik akhir API. Secara default, akses penuh ke operasi ElastiCache API diizinkan melalui titik akhir. Untuk informasi selengkapnya, lihat Mengontrol Akses ke Layanan dengan titik akhir VPC dalam Panduan Pengguna Amazon VPC.

Membuat titik akhir VPC antarmuka untuk API ElastiCache

Anda dapat membuat titik akhir VPC untuk Amazon ElastiCache API menggunakan konsol VPC Amazon atau. AWS CLI Untuk informasi selengkapnya, lihat Membuat titik akhir antarmuka dalam Panduan Pengguna Amazon VPC.

Setelah membuat titik akhir VPC antarmuka, Anda dapat mengaktifkan nama host DNS privat untuk titik akhir. Ketika Anda melakukannya, ElastiCache titik akhir Amazon default (https://elasticache.*Region*.amazonaws.com) menyelesaikan ke titik akhir VPC Anda. Untuk Wilayah Tiongkok (Beijing) dan Tiongkok (Ningxia AWS), Anda dapat membuat permintaan API dengan titik akhir VPC dengan `elasticache.cn-north-1.amazonaws.com.cn` menggunakan untuk Beijing dan untuk Ningxia. `elasticache.cn-northwest-1.amazonaws.com.cn` Untuk informasi selengkapnya, lihat Mengakses layanan melalui titik akhir antarmuka dalam Panduan Pengguna Amazon VPC.

Membuat kebijakan titik akhir VPC untuk Amazon API ElastiCache

Anda dapat melampirkan kebijakan titik akhir ke titik akhir VPC yang mengontrol akses ke API. ElastiCache Kebijakan menentukan informasi berikut:

- Prinsipal yang dapat melakukan tindakan.
- Tindakan yang dapat dilakukan.
- Sumber daya yang menjadi target tindakan.

Untuk informasi selengkapnya, lihat Mengontrol Akses ke Layanan dengan titik akhir VPC dalam Panduan Pengguna Amazon VPC.

Example Kebijakan titik akhir VPC untuk tindakan ElastiCache API dengan Valkey atau Redis OSS

Berikut ini adalah contoh kebijakan endpoint untuk ElastiCache API. Saat dilampirkan ke titik akhir, kebijakan ini memberikan akses ke tindakan ElastiCache API yang terdaftar untuk semua prinsipal di semua sumber daya.

```
{
 "Statement": [{
  "Principal": "*",
  "Effect": "Allow",
  "Action": [
   "elasticache:CreateCacheCluster",
   "elasticache:ModifyCacheCluster",
```

```
   "elasticache:CreateSnapshot"
  ],
  "Resource": "*"
 }]
}
```

Example Kebijakan titik akhir VPC ElastiCache untuk tindakan API Memcached

Berikut ini adalah contoh kebijakan endpoint untuk ElastiCache API. Saat dilampirkan ke titik akhir, kebijakan ini memberikan akses ke tindakan ElastiCache API yang terdaftar untuk semua prinsipal di semua sumber daya.

```
{
 "Statement": [{
  "Principal": "*",
  "Effect": "Allow",
  "Action": [
   "elasticache:CreateCacheCluster",
   "elasticache:ModifyCacheCluster"
  ],
  "Resource": "*"
 }]
}
```

Example Kebijakan titik akhir VPC yang menolak semua akses dari akun tertentu AWS

Kebijakan titik akhir VPC berikut menolak *123456789012* semua akses AWS akun ke sumber daya menggunakan titik akhir. Kebijakan ini mengizinkan semua tindakan dari akun lainnya.

```
{
 "Statement": [{
   "Action": "*",
   "Effect": "Allow",
   "Resource": "*",
   "Principal": "*"
  },
  {
   "Action": "*",
   "Effect": "Deny",
   "Resource": "*",
   "Principal": {
    "AWS": [
```

```
      "123456789012"
     ]
    }
   }
  ]
}
```

Subnet dan grup subnet

Grup subnet adalah sekumpulan subnet (biasanya privat) yang dapat Anda tetapkan untuk klaster Anda yang dirancang sendiri di lingkungan Amazon Virtual Private Cloud (VPC).

Jika Anda membuat cluster yang dirancang sendiri di VPC Amazon, Anda harus menggunakan grup subnet. ElastiCache menggunakan grup subnet itu untuk memilih subnet dan alamat IP dalam subnet itu untuk dikaitkan dengan node Anda.

ElastiCache menyediakan grup IPv4 subnet default atau Anda dapat memilih untuk membuat yang baru. Untuk IPv6, Anda perlu membuat grup subnet dengan blok IPv6 CIDR. Jika Anda memilih tumpukan ganda, Anda harus memilih jenis IP Discovery, salah satu IPv6 atau IPv4.

ElastiCache Tanpa server tidak menggunakan sumber daya grup subnet, dan sebagai gantinya mengambil daftar subnet secara langsung selama pembuatan.

Bagian ini mencakup cara membuat dan memanfaatkan subnet dan grup subnet untuk mengelola akses ke sumber daya Anda ElastiCache .

Untuk informasi selengkapnya tentang penggunaan grup subnet di lingkungan Amazon VPC, lihat Mengakses ElastiCache klaster atau grup replikasi.

Topik

- Membuat grup subnet
- Menetapkan grup subnet ke cache
- Mengubah grup subnet
- Menghapus grup subnet

Membuat grup subnet

Grup subnet cache adalah kumpulan subnet yang dapat ditetapkan untuk cache Anda dalam VPC. Saat meluncurkan cache di VPC, Anda harus memilih grup subnet cache. Kemudian ElastiCache menggunakan kelompok subnet cache itu untuk menetapkan alamat IP dalam subnet itu ke setiap node cache di cache.

Saat Anda membuat grup subnet baru, perhatikan jumlah alamat IP yang tersedia. Jika subnet memiliki sangat sedikit alamat IP yang bebas, Anda akan dibatasi dalam hal jumlah simpul yang dapat ditambahkan ke klaster. Untuk mengatasi masalah ini, Anda dapat menetapkan satu atau beberapa subnet ke grup subnet sehingga Anda memiliki jumlah alamat IP yang cukup dalam Zona Ketersediaan dari klaster Anda. Setelah itu, Anda dapat menambahkan lebih banyak simpul ke klaster Anda.

Jika Anda memilih IPV4 sebagai jenis jaringan Anda, grup subnet default akan tersedia atau Anda dapat memilih untuk membuat yang baru. ElastiCache menggunakan grup subnet itu untuk memilih subnet dan alamat IP dalam subnet itu untuk dikaitkan dengan node Anda. Jika Anda memilih dual-stack atau IPV6, Anda akan diarahkan untuk membuat dual-stack atau subnet. IPV6 Untuk informasi selengkapnya tentang jenis jaringan, lihatMemilih jenis jaringan di ElastiCache. Untuk informasi selengkapnya, lihat Membuat subnet di VPC Anda.

Prosedur berikut menunjukkan cara membuat grup subnet yang disebut `mysubnetgroup` (console), the AWS CLI, dan ElastiCache API.

Membuat grup subnet (Konsol)

Prosedur berikut menunjukkan cara membuat grup subnet (konsol).

Untuk membuat grup subnet (Konsol)

1. Masuk ke AWS Management Console, dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di daftar navigasi, pilih Grup subnet.
3. Pilih Buat grup subnet.
4. Pada wizard Buat Grup Subnet, lakukan hal berikut. Jika semua pengaturan sudah sesuai keinginan Anda, pilih Buat.

    a. Pada kotak Nama, ketik nama grup subnet Anda.

    b. Pada kotak Deskripsi, ketik deskripsi untuk grup subnet Anda.

c. Di kotak ID VPC, pilih Amazon VPC Anda.

d. Semua subnet dipilih secara default. Di panel subnet Terpilih, klik Kelola dan pilih Availability Zones atau Local Zones dan IDs subnet pribadi Anda, lalu pilih Pilih.

5. Pada pesan konfirmasi yang muncul, pilih Tutup.

Grup subnet baru Anda muncul di daftar Grup Subnet konsol. ElastiCache Di bagian bawah jendela, Anda dapat memilih grup subnet untuk melihat detailnya, misalnya semua subnet yang terkait dengan grup ini.

Membuat grup subnet (AWS CLI)

Pada prompt perintah, gunakan perintah `create-cache-subnet-group` untuk membuat grup subnet.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-cache-subnet-group \
    --cache-subnet-group-name mysubnetgroup \
    --cache-subnet-group-description "Testing" \
    --subnet-ids subnet-53df9c3a
```

Untuk Windows:

```
aws elasticache create-cache-subnet-group ^
    --cache-subnet-group-name mysubnetgroup ^
    --cache-subnet-group-description "Testing" ^
    --subnet-ids subnet-53df9c3a
```

Perintah ini seharusnya menghasilkan output yang serupa dengan yang berikut:

```
{
    "CacheSubnetGroup": {
        "VpcId": "vpc-37c3cd17",
        "CacheSubnetGroupDescription": "Testing",
        "Subnets": [
            {
                "SubnetIdentifier": "subnet-53df9c3a",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2a"
                }
```

```
            }
        ],
        "CacheSubnetGroupName": "mysubnetgroup"
    }
}
```

Untuk informasi lebih lanjut, lihat AWS CLI topiknyacreate-cache-subnet-group.

Menetapkan grup subnet ke cache

Setelah Anda membuat grup subnet, Anda dapat meluncurkan cache di Amazon VPC. Untuk informasi selengkapnya, lihat hal berikut.

- Klaster Memcached – Untuk meluncurkan klaster Memcached, lihat Membuat klaster Memcached (konsol). Pada langkah 7.a (Pengaturan Memcached Lanjutan), pilih grup subnet VPC.
- Kluster Valkey atau Redis OSS mandiri — Untuk meluncurkan cluster Valkey atau Redis OSS simpul tunggal, lihat. Membuat cluster Valkey (mode cluster dinonaktifkan) (Konsol) Pada langkah 7.a (Pengaturan OSS Redis Lanjutan), pilih grup subnet VPC.
- Grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) - Untuk meluncurkan grup replikasi Valkey atau Redis OSS (mode cluster dinonaktifkan) di VPC, lihat. Membuat grup replikasi Valkey atau Redis OSS (Cluster Mode Disabled) dari awal Pada langkah 7.b (Pengaturan OSS Redis Lanjutan), pilih grup subnet VPC.
- Grup replikasi Valkey atau Redis OSS (mode cluster diaktifkan) —. Membuat cluster Valkey atau Redis OSS (Mode Cluster Diaktifkan) (Konsol) Pada langkah 6.i (Pengaturan OSS Redis Lanjutan), pilih grup subnet VPC.

Mengubah grup subnet

Anda dapat memodifikasi deskripsi grup subnet, atau memodifikasi daftar subnet yang IDs terkait dengan grup subnet. Anda tidak dapat menghapus ID subnet dari grup subnet jika cache saat ini menggunakan subnet tersebut.

Prosedur berikut menunjukkan cara mengubah grup subnet.

Mengubah grup subnet (Konsol)

Untuk mengubah grup subnet

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Grup Subnet.
3. Pada daftar grup subnet, pilih tombol radio yang ingin Anda ubah dan pilih Ubah.
4. Di panel Subnet yang dipilih, pilih Kelola.
5. Buat perubahan apa pun pada subnet yang dipilih dan klik Pilih.
6. Klik Simpan perubahan untuk menyimpan perubahan Anda.

Mengubah grup subnet (AWS CLI)

Pada prompt perintah, gunakan perintah `modify-cache-subnet-group` untuk mengubah grup subnet.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-cache-subnet-group \
    --cache-subnet-group-name mysubnetgroup \
    --cache-subnet-group-description "New description" \
    --subnet-ids "subnet-42df9c3a" "subnet-48fc21a9"
```

Untuk Windows:

```
aws elasticache modify-cache-subnet-group ^
    --cache-subnet-group-name mysubnetgroup ^
    --cache-subnet-group-description "New description" ^
    --subnet-ids "subnet-42df9c3a" "subnet-48fc21a9"
```

Perintah ini seharusnya menghasilkan output yang serupa dengan yang berikut:

```
{
    "CacheSubnetGroup": {
        "VpcId": "vpc-73cd3c17",
        "CacheSubnetGroupDescription": "New description",
        "Subnets": [
            {
                "SubnetIdentifier": "subnet-42dcf93a",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2a"
                }
            },
            {
                "SubnetIdentifier": "subnet-48fc12a9",
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2a"
                }
            }
        ],
        "CacheSubnetGroupName": "mysubnetgroup"
    }
}
```

Untuk informasi lebih lanjut, lihat AWS CLI topiknyamodify-cache-subnet-group.

Menghapus grup subnet

Jika Anda memutuskan bahwa Anda tidak lagi memerlukan grup subnet, Anda dapat menghapusnya. Anda tidak dapat menghapus grup subnet jika saat ini digunakan oleh cache.

Prosedur berikut menunjukkan cara menghapus grup subnet.

Menghapus grup subnet (Konsol)

Untuk menghapus grup subnet

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di panel navigasi, pilih Grup subnet.
3. Pada daftar grup subnet, pilih grup subnet yang ingin dihapus, lalu pilih Hapus.
4. Saat Anda diminta untuk mengonfirmasi operasi ini, ketik nama grup subnet di bidang input teks dan pilih Hapus.

Menghapus grup subnet (AWS CLI)

Dengan menggunakan AWS CLI, panggil perintah delete-cache-subnet-group dengan parameter berikut:

- `--cache-subnet-group-name` *`mysubnetgroup`*

Untuk Linux, macOS, atau Unix:

```
aws elasticache delete-cache-subnet-group \
    --cache-subnet-group-name mysubnetgroup
```

Untuk Windows:

```
aws elasticache delete-cache-subnet-group ^
    --cache-subnet-group-name mysubnetgroup
```

Perintah ini tidak menghasilkan output.

Untuk informasi lebih lanjut, lihat AWS CLI topiknyadelete-cache-subnet-group.

Identity and Access Management untuk Amazon ElastiCache

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk menggunakan sumber daya. ElastiCache IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Topik

Audiens

Cara Anda menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan. ElastiCache

Pengguna layanan — Jika Anda menggunakan ElastiCache layanan untuk melakukan pekerjaan Anda, maka administrator Anda memberi Anda kredensi dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak ElastiCache fitur untuk melakukan pekerjaan Anda, Anda mungkin memerlukan izin tambahan. Memahami cara mengelola akses dapat membantu Anda meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses fitur di ElastiCache, lihat Memecahkan masalah ElastiCache identitas dan akses Amazon.

Administrator layanan — Jika Anda bertanggung jawab atas ElastiCache sumber daya di perusahaan Anda, Anda mungkin memiliki akses penuh ke ElastiCache. Tugas Anda adalah menentukan ElastiCache fitur dan sumber daya mana yang harus diakses pengguna layanan Anda. Kemudian, Anda harus mengirimkan permintaan kepada administrator IAM untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep dasar IAM. Untuk

mempelajari lebih lanjut tentang bagaimana perusahaan Anda dapat menggunakan IAM ElastiCache, lihatBagaimana Amazon ElastiCache bekerja dengan IAM.

Administrator IAM – Jika Anda adalah administrator IAM, Anda mungkin ingin belajar dengan lebih detail tentang cara Anda menulis kebijakan untuk mengelola akses ke ElastiCache. Untuk melihat contoh kebijakan ElastiCache berbasis identitas yang dapat Anda gunakan di IAM, lihat. Contoh kebijakan berbasis identitas untuk Amazon ElastiCache

Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensi identitas Anda. Anda harus diautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengasumsikan peran IAM.

Anda dapat masuk AWS sebagai identitas federasi dengan menggunakan kredensil yang disediakan melalui sumber identitas. AWS IAM Identity Center Pengguna (IAM Identity Center), autentikasi masuk tunggal perusahaan Anda, dan kredensi Google atau Facebook Anda adalah contoh identitas federasi. Saat Anda masuk sebagai identitas terfederasi, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil peran.

Bergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal AWS akses. Untuk informasi selengkapnya tentang masuk AWS, lihat Cara masuk ke Panduan AWS Sign-In Pengguna Anda Akun AWS.

Jika Anda mengakses AWS secara terprogram, AWS sediakan kit pengembangan perangkat lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensil Anda. Jika Anda tidak menggunakan AWS alat, Anda harus menandatangani permintaan sendiri. Guna mengetahui informasi selengkapnya tentang penggunaan metode yang disarankan untuk menandatangani permintaan sendiri, lihat AWS Signature Version 4 untuk permintaan API dalam Panduan Pengguna IAM.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Misalnya, AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, lihat Autentikasi multi-faktor dalam Panduan Pengguna AWS IAM Identity Center dan Autentikasi multi-faktor AWS di IAM dalam Panduan Pengguna IAM.

Akun AWS pengguna root

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya di akun. Identitas ini disebut pengguna Akun AWS root dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihat Tugas yang memerlukan kredensial pengguna root dalam Panduan Pengguna IAM.

Identitas gabungan

Sebagai praktik terbaik, mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensi sementara.

Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, direktori Pusat Identitas AWS Directory Service, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensil yang disediakan melalui sumber identitas. Ketika identitas federasi mengakses Akun AWS, mereka mengambil peran, dan peran memberikan kredensi sementara.

Untuk manajemen akses terpusat, kami sarankan Anda menggunakan AWS IAM Identity Center. Anda dapat membuat pengguna dan grup di Pusat Identitas IAM, atau Anda dapat menghubungkan dan menyinkronkan ke sekumpulan pengguna dan grup di sumber identitas Anda sendiri untuk digunakan di semua aplikasi Akun AWS dan aplikasi Anda. Untuk informasi tentang Pusat Identitas IAM, lihat Apakah itu Pusat Identitas IAM? dalam Panduan Pengguna AWS IAM Identity Center .

Pengguna dan grup IAM

Pengguna IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, kami merekomendasikan untuk mengandalkan kredensial sementara, bukan membuat pengguna IAM yang memiliki kredensial jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan tertentu yang memerlukan kredensial jangka panjang dengan pengguna IAM, kami merekomendasikan Anda merotasi kunci akses. Untuk informasi selengkapnya, lihat Merotasi kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensial jangka panjang dalam Panduan Pengguna IAM.

Grup IAM adalah identitas yang menentukan sekumpulan pengguna IAM. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin bagi beberapa pengguna sekaligus. Grup mempermudah manajemen izin untuk sejumlah besar pengguna sekaligus. Misalnya, Anda dapat meminta kelompok untuk menyebutkan IAMAdmins dan memberikan izin kepada grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk mempelajari selengkapnya, lihat Kasus penggunaan untuk pengguna IAM dalam Panduan Pengguna IAM.

Peran IAM

Peran IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Untuk mengambil peran IAM sementara AWS Management Console, Anda dapat beralih dari pengguna ke peran IAM (konsol). Anda dapat mengambil peran dengan memanggil operasi AWS CLI atau AWS API atau dengan menggunakan URL kustom. Untuk informasi selengkapnya tentang cara menggunakan peran, lihat Metode untuk mengambil peran dalam Panduan Pengguna IAM.

Peran IAM dengan kredensial sementara berguna dalam situasi berikut:

- Akses pengguna terfederasi – Untuk menetapkan izin ke identitas terfederasi, Anda membuat peran dan menentukan izin untuk peran tersebut. Ketika identitas terfederasi mengautentikasi, identitas tersebut terhubung dengan peran dan diberi izin yang ditentukan oleh peran. Untuk informasi tentang peran untuk federasi, lihat Buat peran untuk penyedia identitas pihak ketiga dalam Panduan Pengguna IAM. Jika menggunakan Pusat Identitas IAM, Anda harus mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM akan mengorelasikan set izin ke peran dalam IAM. Untuk informasi tentang set izin, lihat Set izin dalam Panduan Pengguna AWS IAM Identity Center .
- Izin pengguna IAM sementara – Pengguna atau peran IAM dapat mengambil peran IAM guna mendapatkan berbagai izin secara sementara untuk tugas tertentu.
- Akses lintas akun – Anda dapat menggunakan peran IAM untuk mengizinkan seseorang (prinsipal tepercaya) di akun lain untuk mengakses sumber daya di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, dengan beberapa Layanan AWS, Anda dapat melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan peran sebagai

proxy). Untuk mempelajari perbedaan antara peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM dalam Panduan Pengguna IAM.

- Akses lintas layanan — Beberapa Layanan AWS menggunakan fitur lain Layanan AWS. Misalnya, saat Anda melakukan panggilan dalam suatu layanan, biasanya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Sebuah layanan mungkin melakukannya menggunakan izin prinsipal yang memanggil, menggunakan peran layanan, atau peran terkait layanan.
  - Sesi akses teruskan (FAS) — Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.
  - Peran layanan – Peran layanan adalah peran IAM yang dijalankan oleh layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.
  - Peran terkait layanan — Peran terkait layanan adalah jenis peran layanan yang ditautkan ke peran layanan. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.
- Aplikasi yang berjalan di Amazon EC2 — Anda dapat menggunakan peran IAM untuk mengelola kredensi sementara untuk aplikasi yang berjalan pada EC2 instance dan membuat AWS CLI atau AWS permintaan API. Ini lebih baik untuk menyimpan kunci akses dalam EC2 instance. Untuk menetapkan AWS peran ke EC2 instance dan membuatnya tersedia untuk semua aplikasinya, Anda membuat profil instans yang dilampirkan ke instance. Profil instance berisi peran dan memungkinkan program yang berjalan pada EC2 instance untuk mendapatkan kredensi sementara. Untuk informasi selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di EC2 instans Amazon di Panduan Pengguna IAM.

Mengelola akses menggunakan kebijakan

Anda mengontrol akses AWS dengan membuat kebijakan dan melampirkannya ke AWS identitas atau sumber daya. Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika prinsipal (pengguna, pengguna root, atau sesi peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang struktur dan isi dokumen kebijakan JSON, lihat Gambaran umum kebijakan JSON dalam Panduan Pengguna IAM.

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk melakukan operasinya. Misalnya, anggaplah Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan tersebut bisa mendapatkan informasi peran dari AWS Management Console, API AWS CLI, atau AWS API.

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Tentukan izin IAM kustom dengan kebijakan terkelola pelanggan dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan yang dikelola. Kebijakan inline disematkan langsung ke satu pengguna, grup, atau peran. Kebijakan terkelola adalah kebijakan mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran dalam. Akun AWS Kebijakan AWS terkelola mencakup kebijakan terkelola dan kebijakan yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan yang dikelola atau kebijakan inline, lihat Pilih antara kebijakan yang dikelola dan kebijakan inline dalam Panduan Pengguna IAM.

Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan AWS terkelola dari IAM dalam kebijakan berbasis sumber daya.

Daftar kontrol akses (ACLs)

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung. ACLs Untuk mempelajari selengkapnya ACLs, lihat Ringkasan daftar kontrol akses (ACL) di Panduan Pengembang Layanan Penyimpanan Sederhana Amazon.

Jenis-jenis kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Jenis-jenis kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda oleh jenis kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan tempat Anda mengatur izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas ke entitas IAM (pengguna IAM atau peran IAM). Anda dapat menetapkan batasan izin untuk suatu entitas. Izin yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas milik entitas dan batasan izinnya. Kebijakan berbasis sumber daya yang menentukan pengguna atau peran dalam bidang `Principal` tidak dibatasi oleh batasan izin. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya tentang batasan izin, lihat Batasan izin untuk entitas IAM dalam Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCPs) — SCPs adalah kebijakan JSON yang menentukan izin maksimum untuk organisasi atau unit organisasi (OU) di. AWS Organizations AWS Organizations

adalah layanan untuk mengelompokkan dan mengelola secara terpusat beberapa Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur dalam suatu organisasi, maka Anda dapat menerapkan kebijakan kontrol layanan (SCPs) ke salah satu atau semua akun Anda. SCP membatasi izin untuk entitas di akun anggota, termasuk masing-masing. Pengguna root akun AWS Untuk informasi selengkapnya tentang Organizations dan SCPs, lihat Kebijakan kontrol layanan di Panduan AWS Organizations Pengguna.

- Kebijakan kontrol sumber daya (RCPs) — RCPs adalah kebijakan JSON yang dapat Anda gunakan untuk menetapkan izin maksimum yang tersedia untuk sumber daya di akun Anda tanpa memperbarui kebijakan IAM yang dilampirkan ke setiap sumber daya yang Anda miliki. RCP membatasi izin untuk sumber daya di akun anggota dan dapat memengaruhi izin efektif untuk identitas, termasuk Pengguna root akun AWS, terlepas dari apakah itu milik organisasi Anda. Untuk informasi selengkapnya tentang Organizations dan RCPs, termasuk daftar dukungan Layanan AWS tersebut RCPs, lihat Kebijakan kontrol sumber daya (RCPs) di Panduan AWS Organizations Pengguna.
- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter ketika Anda membuat sesi sementara secara programatis untuk peran atau pengguna terfederasi. Izin sesi yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas pengguna atau peran dan kebijakan sesi. Izin juga bisa datang dari kebijakan berbasis sumber daya. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya, lihat Kebijakan sesi dalam Panduan Pengguna IAM.

Berbagai jenis kebijakan

Ketika beberapa jenis kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat Logika evaluasi kebijakan di Panduan Pengguna IAM.

Bagaimana Amazon ElastiCache bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ElastiCache, pelajari fitur IAM yang tersedia untuk digunakan. ElastiCache

Fitur IAM yang dapat Anda gunakan dengan Amazon ElastiCache

Fitur IAM	ElastiCache dukungan
Kebijakan berbasis identitas	Ya
Kebijakan berbasis sumber daya	Tidak
Tindakan kebijakan	Ya
Sumber daya kebijakan	Ya
Kunci kondisi kebijakan	Ya
ACLs	Ya
ABAC (tanda dalam kebijakan)	Ya
Kredensial sementara	Ya
Izin principal	Ya
Peran layanan	Ya
Peran terkait layanan	Ya

Untuk mendapatkan tampilan tingkat tinggi tentang cara ElastiCache dan AWS layanan lain bekerja dengan sebagian besar fitur IAM, lihat AWS layanan yang bekerja dengan IAM di Panduan Pengguna IAM.

Kebijakan berbasis identitas untuk ElastiCache

Mendukung kebijakan berbasis identitas: Ya

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Tentukan izin IAM kustom dengan kebijakan terkelola pelanggan dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan secara spesifik apakah tindakan dan sumber daya diizinkan atau ditolak, serta kondisi yang menjadi dasar dikabulkan atau ditolaknya tindakan tersebut. Anda tidak dapat menentukan secara spesifik prinsipal dalam sebuah kebijakan berbasis identitas karena prinsipal berlaku bagi pengguna atau peran yang melekat kepadanya. Untuk mempelajari semua elemen yang dapat Anda gunakan dalam kebijakan JSON, lihat Referensi elemen kebijakan IAM JSON dalam Panduan Pengguna IAM.

Contoh kebijakan berbasis identitas untuk ElastiCache

Untuk melihat contoh kebijakan ElastiCache berbasis identitas, lihat. Contoh kebijakan berbasis identitas untuk Amazon ElastiCache

Kebijakan berbasis sumber daya dalam ElastiCache

Mendukung kebijakan berbasis sumber daya: Tidak

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan secara spesifik seluruh akun atau entitas IAM di akun lain sebagai prinsipal dalam kebijakan berbasis sumber daya. Menambahkan prinsipal akun silang ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika prinsipal dan sumber daya berbeda Akun AWS, administrator IAM di akun tepercaya juga harus memberikan izin entitas utama (pengguna atau peran) untuk mengakses sumber daya. Mereka memberikan izin dengan melampirkan kebijakan berbasis identitas kepada entitas. Namun, jika kebijakan berbasis sumber daya memberikan akses ke principal dalam akun yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk informasi selengkapnya, lihat Akses sumber daya lintas akun di IAM dalam Panduan Pengguna IAM.

Tindakan kebijakan untuk ElastiCache

Mendukung tindakan kebijakan: Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama dengan operasi AWS API terkait. Ada beberapa pengecualian, misalnya tindakan hanya izin yang tidak memiliki operasi API yang cocok. Ada juga beberapa operasi yang memerlukan beberapa tindakan dalam suatu kebijakan. Tindakan tambahan ini disebut tindakan dependen.

Sertakan tindakan dalam kebijakan untuk memberikan izin untuk melakukan operasi terkait.

Untuk melihat daftar ElastiCache tindakan, lihat Tindakan yang Ditentukan oleh Amazon ElastiCache di Referensi Otorisasi Layanan.

Tindakan kebijakan ElastiCache menggunakan awalan berikut sebelum tindakan:

```
elasticache
```

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan tersebut dengan koma.

```
"Action": [
      "elasticache:action1",
      "elasticache:action2"
         ]
```

Anda juga dapat menentukan beberapa tindakan menggunakan wildcard (*). Sebagai contoh, untuk menentukan semua tindakan yang dimulai dengan kata `Describe`, sertakan tindakan berikut:

```
"Action": "elasticache:Describe*"
```

Untuk melihat contoh kebijakan ElastiCache berbasis identitas, lihat. Contoh kebijakan berbasis identitas untuk Amazon ElastiCache

Sumber daya kebijakan untuk ElastiCache

Mendukung sumber daya kebijakan: Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek yang menjadi target penerapan tindakan. Pernyataan harus menyertakan elemen `Resource` atau `NotResource`. Praktik terbaiknya, tentukan sumber daya menggunakan Amazon Resource Name (ARN). Anda dapat melakukan ini untuk tindakan yang mendukung jenis sumber daya tertentu, yang dikenal sebagai izin tingkat sumber daya.

Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, misalnya operasi pencantuman, gunakan wildcard (*) untuk menunjukkan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

```
"Resource": "*"
```

Untuk melihat daftar jenis sumber daya dan jenis ElastiCache sumber daya ARNs, lihat Sumber Daya yang Ditentukan oleh Amazon ElastiCache di Referensi Otorisasi Layanan. Untuk mempelajari tindakan mana yang dapat Anda tentukan ARN dari setiap sumber daya, lihat Tindakan yang Ditentukan oleh Amazon. ElastiCache

Untuk melihat contoh kebijakan ElastiCache berbasis identitas, lihat. Contoh kebijakan berbasis identitas untuk Amazon ElastiCache

Kunci kondisi kebijakan untuk ElastiCache

Mendukung kunci kondisi kebijakan khusus layanan: Yes

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Condition` (atau blok `Condition`) akan memungkinkan Anda menentukan kondisi yang menjadi dasar suatu pernyataan berlaku. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi bersyarat yang menggunakan operator kondisi, misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta.

Jika Anda menentukan beberapa elemen `Condition` dalam sebuah pernyataan, atau beberapa kunci dalam elemen `Condition` tunggal, maka AWS akan mengevaluasinya menggunakan operasi

AND logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS mengevaluasi kondisi menggunakan OR operasi logis. Semua kondisi harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan kondisi. Sebagai contoh, Anda dapat memberikan izin kepada pengguna IAM untuk mengakses sumber daya hanya jika izin tersebut mempunyai tanda yang sesuai dengan nama pengguna IAM mereka. Untuk informasi selengkapnya, lihat Elemen kebijakan IAM: variabel dan tanda dalam Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi khusus layanan. Untuk melihat semua kunci kondisi AWS global, lihat kunci konteks kondisi AWS global di Panduan Pengguna IAM.

Untuk melihat daftar kunci ElastiCache kondisi, lihat Kunci Kondisi untuk Amazon ElastiCache di Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat Anda gunakan kunci kondisi, lihat Tindakan yang Ditentukan oleh Amazon ElastiCache.

Untuk melihat contoh kebijakan ElastiCache berbasis identitas, lihat. Contoh kebijakan berbasis identitas untuk Amazon ElastiCache

Daftar kontrol akses (ACLs) di ElastiCache

Mendukung ACLs: Ya

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan JSON.

Kontrol akses berbasis atribut (ABAC) dengan ElastiCache

Mendukung ABAC (tanda dalam kebijakan): Ya

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang menentukan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke entitas IAM (pengguna atau peran) dan ke banyak AWS sumber daya. Penandaan ke entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian rancanglah kebijakan ABAC untuk mengizinkan operasi ketika tanda milik prinsipal cocok dengan tanda yang ada di sumber daya yang ingin diakses.

ABAC sangat berguna di lingkungan yang berkembang dengan cepat dan berguna di situasi saat manajemen kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tanda, berikan informasi tentang tanda di elemen kondisi dari kebijakan menggunakan kunci kondisi `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, atau `aws:TagKeys`.

Jika sebuah layanan mendukung ketiga kunci kondisi untuk setiap jenis sumber daya, nilainya adalah Ya untuk layanan tersebut. Jika suatu layanan mendukung ketiga kunci kondisi untuk hanya beberapa jenis sumber daya, nilainya adalah Parsial.

Untuk informasi selengkapnya tentang ABAC, lihat Tentukan izin dengan otorisasi ABAC dalam Panduan Pengguna IAM. Untuk melihat tutorial yang menguraikan langkah-langkah pengaturan ABAC, lihat Menggunakan kontrol akses berbasis atribut (ABAC) dalam Panduan Pengguna IAM.

Menggunakan kredensi sementara dengan ElastiCache

Mendukung kredensial sementara: Ya

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensi sementara. Untuk informasi tambahan, termasuk yang Layanan AWS bekerja dengan kredensi sementara, lihat Layanan AWS yang bekerja dengan IAM di Panduan Pengguna IAM.

Anda menggunakan kredensi sementara jika Anda masuk AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Misalnya, ketika Anda mengakses AWS menggunakan tautan masuk tunggal (SSO) perusahaan Anda, proses tersebut secara otomatis membuat kredensi sementara. Anda juga akan secara otomatis membuat kredensial sementara ketika Anda masuk ke konsol sebagai seorang pengguna lalu beralih peran. Untuk informasi selengkapnya tentang peralihan peran, lihat Beralih dari pengguna ke peran IAM (konsol) dalam Panduan Pengguna IAM.

Anda dapat membuat kredenal sementara secara manual menggunakan API AWS CLI atau AWS . Anda kemudian dapat menggunakan kredensi sementara tersebut untuk mengakses. AWS AWS merekomendasikan agar Anda secara dinamis menghasilkan kredensi sementara alih-alih menggunakan kunci akses jangka panjang. Untuk informasi lebih lanjut, lihat Kredensial keamanan sementara di IAM.

Izin principal lintas layanan untuk ElastiCache

Mendukung sesi akses maju (FAS): Ya

Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk

menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.

Peran layanan untuk ElastiCache

Mendukung peran layanan: Ya

Peran layanan adalah peran IAM yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

> **Warning**
>
> Mengubah izin untuk peran layanan dapat merusak ElastiCache fungsionalitas. Edit peran layanan hanya jika ElastiCache memberikan bimbingan untuk melakukannya.

Peran terkait layanan untuk ElastiCache

Mendukung peran terkait layanan: Ya

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Untuk detail tentang pembuatan atau manajemen peran terkait layanan, lihat Layanan AWS yang berfungsi dengan IAM. Cari layanan dalam tabel yang memiliki `Yes` di kolom Peran terkait layanan. Pilih tautan Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Contoh kebijakan berbasis identitas untuk Amazon ElastiCache

Secara default, pengguna dan peran tidak memiliki izin untuk membuat atau mengubah sumber daya ElastiCache. Mereka juga tidak dapat melakukan tugas dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS API. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM dengan menggunakan contoh dokumen kebijakan JSON ini, lihat Membuat kebijakan IAM (konsol) di Panduan Pengguna IAM.

Untuk detail tentang tindakan dan jenis sumber daya yang ditentukan oleh ElastiCache, termasuk format ARNs untuk setiap jenis sumber daya, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk Amazon ElastiCache di Referensi Otorisasi Layanan.

Topik

Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus ElastiCache sumber daya di akun Anda. Tindakan ini membuat Akun AWS Anda dikenai biaya. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit — Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Anda Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola AWS pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat Kebijakan yang dikelola AWS atau Kebijakan yang dikelola AWS untuk fungsi tugas dalam Panduan Pengguna IAM.
- Menerapkan izin dengan hak akses paling rendah – Ketika Anda menetapkan izin dengan kebijakan IAM, hanya berikan izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk mengajukan izin, lihat Kebijakan dan izin dalam IAM dalam Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan jika digunakan melalui yang spesifik Layanan AWS, seperti

AWS CloudFormation. Untuk informasi selengkapnya, lihat Elemen kebijakan JSON IAM: Kondisi dalam Panduan Pengguna IAM.

- Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional – IAM Access Analyzer memvalidasi kebijakan baru dan yang sudah ada sehingga kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM. IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, lihat Validasi kebijakan dengan IAM Access Analyzer dalam Panduan Pengguna IAM.
- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Anda, Akun AWS aktifkan MFA untuk keamanan tambahan. Untuk meminta MFA ketika operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat Amankan akses API dengan MFA dalam Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat Praktik terbaik keamanan di IAM dalam Panduan Pengguna IAM.

Menggunakan ElastiCache konsol

Untuk mengakses ElastiCache konsol Amazon, Anda harus memiliki set izin minimum. Izin ini harus memungkinkan Anda untuk membuat daftar dan melihat detail tentang ElastiCache sumber daya di Anda Akun AWS. Jika Anda membuat kebijakan berbasis identitas yang lebih ketat daripada izin minimum yang diperlukan, konsol tidak akan berfungsi sebagaimana mestinya untuk entitas (pengguna atau peran) dengan kebijakan tersebut.

Anda tidak perlu mengizinkan izin konsol minimum untuk pengguna yang melakukan panggilan hanya ke AWS CLI atau AWS API. Sebagai gantinya, izinkan akses hanya ke tindakan yang sesuai dengan operasi API yang coba mereka lakukan.

Untuk memastikan bahwa pengguna dan peran masih dapat menggunakan ElastiCache konsol, lampirkan juga kebijakan ElastiCache `ConsoleAccess` atau `ReadOnly` AWS terkelola ke entitas. Untuk informasi selengkapnya, lihat Menambah izin untuk pengguna dalam Panduan Pengguna IAM.

Mengizinkan pengguna melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini

mencakup izin untuk menyelesaikan tindakan ini di konsol atau menggunakan API atau secara terprogram. AWS CLI AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Memecahkan masalah ElastiCache identitas dan akses Amazon

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan ElastiCache dan IAM.

Topik

Saya tidak diotorisasi untuk melakukan tindakan di ElastiCache

Jika AWS Management Console memberitahu Anda bahwa Anda tidak berwenang untuk melakukan tindakan, maka Anda harus menghubungi administrator Anda untuk bantuan. Administrator Anda adalah orang yang memberikan nama pengguna dan kata sandi Anda.

Contoh kesalahan berikut terjadi ketika pengguna `mateojackson` mencoba menggunakan konsol untuk melihat detail tentang suatu sumber daya *`my-example-widget`* fiktif, tetapi tidak memiliki izin `elasticache:`*`GetWidget`* fiktif.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 elasticache:GetWidget on resource: my-example-widget
```

Dalam hal ini, Mateo meminta administratornya untuk memperbarui kebijakannya untuk mengizinkan dia mengakses sumber daya *`my-example-widget`* menggunakan tindakan `elasticache:`*`GetWidget`*.

Saya tidak berwenang untuk melakukan iam: PassRole

Jika Anda menerima kesalahan yang tidak diizinkan untuk melakukan `iam:PassRole` tindakan, kebijakan Anda harus diperbarui agar Anda dapat meneruskan peran ElastiCache.

Beberapa Layanan AWS memungkinkan Anda untuk meneruskan peran yang ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait layanan. Untuk melakukannya, Anda harus memiliki izin untuk meneruskan peran ke layanan.

Contoh kesalahan berikut terjadi ketika pengguna IAM bernama `marymajor` mencoba menggunakan konsol tersebut untuk melakukan tindakan di ElastiCache. Namun, tindakan tersebut memerlukan layanan untuk mendapatkan izin yang diberikan oleh peran layanan. Mary tidak memiliki izin untuk meneruskan peran tersebut pada layanan.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dalam kasus ini, kebijakan Mary harus diperbarui agar dia mendapatkan izin untuk melakukan tindakan `iam:PassRole` tersebut.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya ingin mengizinkan orang di luar AWS akun saya untuk mengakses ElastiCache sumber daya saya

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau orang-orang di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa saja yang dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis sumber daya atau daftar kontrol akses (ACLs), Anda dapat menggunakan kebijakan tersebut untuk memberi orang akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa referensi berikut:

- Untuk mempelajari apakah ElastiCache mendukung fitur-fitur ini, lihatBagaimana Amazon ElastiCache bekerja dengan IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda di seluruh sumber daya Akun AWS yang Anda miliki, lihat Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki di Panduan Pengguna IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda kepada pihak ketiga Akun AWS, lihat Menyediakan akses yang Akun AWS dimiliki oleh pihak ketiga dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat Menyediakan akses ke pengguna terautentikasi eksternal (federasi identitas) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara menggunakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM.

Kontrol akses

Anda dapat memiliki kredensi yang valid untuk mengautentikasi permintaan Anda, tetapi kecuali Anda memiliki izin, Anda tidak dapat membuat atau mengakses sumber daya. ElastiCache Misalnya, Anda harus memiliki izin untuk membuat ElastiCache klaster.

Bagian berikut menjelaskan cara mengelola izin untuk ElastiCache. Anda sebaiknya membaca gambaran umum terlebih dahulu.

- Ikhtisar mengelola izin akses ke sumber daya Anda ElastiCache
- Menggunakan kebijakan berbasis identitas (kebijakan IAM) untuk Amazon ElastiCache

Ikhtisar mengelola izin akses ke sumber daya Anda ElastiCache

Setiap AWS sumber daya dimiliki oleh AWS akun, dan izin untuk membuat atau mengakses sumber daya diatur oleh kebijakan izin. Administrator akun dapat melampirkan kebijakan izin pada identitas IAM (yaitu pengguna, grup, dan peran). Selain itu, Amazon ElastiCache juga mendukung melampirkan kebijakan izin ke sumber daya.

> **Note**
>
> Administrator akun (atau pengguna administrator) adalah pengguna dengan hak akses administrator. Untuk informasi selengkapnya, lihat Praktik Terbaik IAM dalam Panduan Pengguna IAM.

Untuk memberikan akses, tambahkan izin ke pengguna, grup, atau peran Anda:

- Pengguna dan grup di AWS IAM Identity Center:

  Buat rangkaian izin. Ikuti instruksi di Buat rangkaian izin di Panduan Pengguna AWS IAM Identity Center .
- Pengguna yang dikelola di IAM melalui penyedia identitas:

  Buat peran untuk federasi identitas. Ikuti instruksi dalam Buat peran untuk penyedia identitas pihak ketiga (federasi) dalam Panduan Pengguna IAM.
- Pengguna IAM:
  - Buat peran yang dapat diambil pengguna Anda. Ikuti instruksi dalam Buat peran untuk pengguna IAM dalam Panduan Pengguna IAM.
  - (Tidak disarankan) Lampirkan kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti petunjuk dalam Menambahkan izin ke pengguna (konsol) dalam Panduan Pengguna IAM.

Topik

- Menggunakan kebijakan berbasis identitas (kebijakan IAM) untuk Amazon ElastiCache
- Izin tingkat sumber daya
- Menggunakan kunci kondisi
- Menggunakan Peran Tertaut Layanan untuk Amazon ElastiCache
- ElastiCache Izin API: Referensi tindakan, sumber daya, dan kondisi

ElastiCache Sumber daya dan operasi Amazon

Untuk melihat daftar jenis sumber daya dan jenis ElastiCache sumber daya ARNs, lihat Sumber Daya yang Ditentukan oleh Amazon ElastiCache di Referensi Otorisasi Layanan. Untuk mempelajari tindakan mana yang dapat Anda tentukan ARN dari setiap sumber daya, lihat Tindakan yang Ditentukan oleh Amazon. ElastiCache

Memahami kepemilikan sumber daya

Pemilik sumber daya adalah AWS akun yang membuat sumber daya. Artinya, pemilik sumber daya adalah AWS akun entitas utama yang mengotentikasi permintaan yang membuat sumber daya. Entitas prinsipal dapat berupa akun root, pengguna IAM, atau peran IAM). Contoh berikut menggambarkan cara kerjanya:

- Misalkan Anda menggunakan kredensi akun root AWS akun Anda untuk membuat cluster cache. Dalam hal ini, AWS akun Anda adalah pemilik sumber daya. Pada ElastiCache, sumber dayanya adalah klaster cache.
- Misalkan Anda membuat pengguna IAM di AWS akun Anda dan memberikan izin untuk membuat cluster cache kepada pengguna tersebut. Dalam hal ini, pengguna tersebut dapat membuat klaster cache. Namun, AWS akun Anda, yang menjadi milik pengguna, memiliki sumber daya cluster cache.
- Misalkan Anda membuat peran IAM di AWS akun Anda dengan izin untuk membuat cluster cache. Dalam hal ini, siapa pun yang dapat mengambil peran tersebut akan dapat membuat klaster cache. AWS Akun Anda, tempat peran tersebut berada, memiliki sumber daya cluster cache.

Mengelola akses ke sumber daya

Kebijakan izin menjelaskan siapa yang memiliki akses ke suatu objek. Bagian berikut menjelaskan opsi yang tersedia untuk membuat kebijakan izin.

> **Note**
>
> Bagian ini membahas penggunaan IAM dalam konteks Amazon. ElastiCache Bagian ini tidak memberikan informasi yang mendetail tentang layanan IAM. Untuk dokumentasi IAM lengkap, lihat Apa yang Dimaksud dengan IAM? dalam Panduan Pengguna IAM. Untuk informasi tentang sintaksis dan deskripsi kebijakan IAM, lihat Referensi Kebijakan IAM AWS dalam Panduan Pengguna IAM.

Kebijakan yang terlampir pada identitas IAM disebut sebagai kebijakan berbasis identitas (kebijakan IAM). Kebijakan yang dilampirkan pada sumber daya disebut sebagai kebijakan berbasis-sumber daya.

Topik

- Kebijakan berbasis identitas (kebijakan IAM)
- Menentukan elemen kebijakan: Tindakan, efek, sumber daya, dan prinsipal
- Menentukan kondisi dalam kebijakan

Kebijakan berbasis identitas (kebijakan IAM)

Anda dapat melampirkan kebijakan ke identitas IAM Anda. Misalnya, Anda dapat melakukan hal berikut:

- Melampirkan kebijakan izin pada pengguna atau grup dalam akun Anda – Akun administrator dapat menggunakan kebijakan izin yang terkait dengan pengguna tertentu untuk memberikan izin. Dalam hal ini, izin adalah bagi pengguna untuk membuat ElastiCache sumber daya, seperti cluster cache, grup parameter, atau grup keamanan.
- Melampirkan kebijakan izin pada peran (memberikan izin lintas akun) – Anda dapat melampirkan kebijakan izin berbasis identitas ke peran IAM untuk memberikan izin lintas akun. Misalnya, administrator di Akun A dapat membuat peran untuk memberikan izin lintas akun ke AWS akun lain (misalnya, Akun B) atau AWS layanan sebagai berikut:
  1. Administrator akun A membuat peran IAM dan melampirkan kebijakan izin ke peran ini yang memberikan izin pada sumber daya di akun A.
  2. Administrator akun A melampirkan kebijakan kepercayaan ke peran yang mengidentifikasi Akun B sebagai prinsipal yang dapat mengambil peran tersebut.

3. Administrator Akun B kemudian dapat mendelegasikan izin untuk mengambil peran kepada setiap pengguna di Akun B. Melakukan hal ini memungkinkan pengguna di Akun B untuk membuat atau mengakses sumber daya di Akun A. Dalam beberapa kasus, Anda mungkin ingin memberikan izin AWS layanan untuk mengambil peran tersebut. Untuk mendukung pendekatan ini, prinsipal dalam kebijakan kepercayaan juga dapat merupakan prinsipal layanan AWS .

Untuk informasi selengkapnya tentang penggunaan IAM untuk mendelegasikan izin, lihat Manajemen Akses dalam Panduan Pengguna IAM.

Berikut ini adalah contoh kebijakan yang memungkinkan pengguna untuk melakukan `DescribeCacheClusters` tindakan untuk AWS akun Anda. ElastiCache juga mendukung identifikasi sumber daya tertentu menggunakan sumber daya ARNs untuk tindakan API. (Pendekatan ini juga disebut sebagai izin tingkat sumber daya).

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DescribeCacheClusters",
            "Effect": "Allow",
            "Action": [
                "elasticache:DescribeCacheClusters"
            ],
            "Resource": "resource-arn"
        }
    ]
}
```

Untuk informasi selengkapnya tentang menggunakan kebijakan berbasis identitas dengan ElastiCache, lihat. Menggunakan kebijakan berbasis identitas (kebijakan IAM) untuk Amazon ElastiCache Untuk informasi selengkapnya tentang pengguna, grup, peran, dan izin, lihat Identitas (Pengguna, Grup, dan Peran) dalam Panduan Pengguna IAM.

Menentukan elemen kebijakan: Tindakan, efek, sumber daya, dan prinsipal

Untuk setiap ElastiCache sumber daya Amazon (lihat ElastiCache Sumber daya dan operasi Amazon), layanan mendefinisikan sekumpulan operasi API (lihat Tindakan). Untuk memberikan izin untuk operasi API ini, ElastiCache tentukan serangkaian tindakan yang dapat Anda tentukan dalam kebijakan. Misalnya, untuk sumber daya ElastiCache cluster, tindakan berikut didefinisikan:`CreateCacheCluster`,`DeleteCacheCluster`, dan`DescribeCacheCluster`. Operasi API dapat memerlukan izin untuk lebih dari satu tindakan.

Berikut adalah elemen-elemen kebijakan yang paling dasar:

- Sumber daya – Dalam kebijakan, Anda menggunakan Amazon Resource Name (ARN) untuk mengidentifikasi sumber daya yang diatur kebijakan. Untuk informasi selengkapnya, lihat ElastiCache Sumber daya dan operasi Amazon.
- Tindakan – Anda menggunakan kata kunci tindakan untuk mengidentifikasi operasi sumber daya yang ingin Anda izinkan atau tolak. Misalnya, tergantung pada yang ditentukan`Effect`, `elasticache:CreateCacheCluster` izin mengizinkan atau menolak izin pengguna untuk melakukan operasi Amazon ElastiCache`CreateCacheCluster`.
- Efek – Anda menentukan efek ketika pengguna meminta tindakan tertentu—efek ini dapat berupa pemberian izin atau penolakan. Jika Anda tidak secara eksplisit memberikan akses ke (mengizinkan) sumber daya, akses akan ditolak secara implisit. Anda juga dapat secara eksplisit menolak akses ke sumber daya. Misalnya, Anda mungkin melakukannya untuk memastikan agar pengguna tidak dapat mengakses sumber daya, meskipun jika ada kebijakan berbeda yang memberikan akses.
- Prinsipal – Dalam kebijakan berbasis identitas (Kebijakan IAM), pengguna yang dilampiri kebijakan adalah prinsipal secara implisit. Untuk kebijakan berbasis sumber daya, Anda menentukan pengguna, akun, layanan, atau entitas lain yang diinginkan untuk menerima izin (berlaku hanya untuk kebijakan berbasis sumber daya).

Untuk mempelajari selengkapnya tentang sintaksis dan deskripsi kebijakan IAM, lihat Referensi Kebijakan IAM AWS dalam Panduan Pengguna IAM.

Untuk tabel yang menampilkan semua tindakan Amazon ElastiCache API, lihatElastiCache Izin API: Referensi tindakan, sumber daya, dan kondisi.

Menentukan kondisi dalam kebijakan

Ketika Anda memberikan izin, Anda dapat menggunakan bahasa kebijakan IAM untuk menentukan kondisi ketika kebijakan harus berlaku. Misalnya, Anda mungkin ingin kebijakan diterapkan hanya setelah tanggal tertentu. Untuk informasi selengkapnya tentang menentukan kondisi dalam bahasa kebijakan, lihat Kondisi dalam Panduan Pengguna IAM.

Untuk menyatakan kondisi, Anda menggunakan kunci kondisi standar. Untuk menggunakan tombol kondisi ElastiCache khusus, lihatMenggunakan kunci kondisi. Ada tombol kondisi AWS-wide yang dapat Anda gunakan sesuai kebutuhan. Untuk daftar lengkap tombol AWS-wide, lihat Kunci yang Tersedia untuk Ketentuan di Panduan Pengguna IAM.

AWS kebijakan terkelola untuk Amazon ElastiCache

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. AWS Kebijakan terkelola dirancang untuk memberikan izin bagi banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan semua pelanggan. AWS Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. Jika AWS memperbarui izin yang ditentukan dalam kebijakan AWS terkelola, pembaruan akan memengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan AWS terkelola saat baru Layanan AWS diluncurkan atau operasi API baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat Kebijakan terkelola AWS dalam Panduan Pengguna IAM.

AWS kebijakan terkelola: ElastiCacheServiceRolePolicy

Anda tidak dapat melampirkan ElastiCacheServiceRolePolicy ke entitas IAM Anda. Kebijakan ini dilampirkan pada peran terkait layanan yang memungkinkan ElastiCache untuk melakukan tindakan atas nama Anda.

Kebijakan ini memungkinkan ElastiCache untuk mengelola AWS sumber daya atas nama Anda sebagaimana diperlukan untuk mengelola cache Anda:

- `ec2`— Kelola sumber daya EC2 jaringan untuk dilampirkan ke node cache, termasuk titik akhir VPC (untuk cache tanpa server), Antarmuka Jaringan Elastis () (untuk cluster yang dirancang sendiriENIs), dan grup keamanan.
- `cloudwatch`— Memancarkan data metrik dari layanan ke CloudWatch.
- `outposts`— Izinkan pembuatan node cache di AWS Outposts.

Anda dapat menemukan ElastiCacheServiceRolePolicykebijakan di konsol IAM dan ElastiCacheServiceRolePolicydi Panduan Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AmazonElastiCacheFullAccess

Anda dapat melampirkan kebijakan `AmazonElastiCacheFullAccess` ke identitas IAM Anda.

Kebijakan ini memungkinkan prinsipal akses penuh untuk ElastiCache menggunakan Management Console: AWS

- `elasticache`— Akses semua APIs.
- `iam` – Membuat peran terkait layanan yang diperlukan untuk operasi layanan.
- `ec2`— Jelaskan EC2 sumber daya dependen yang diperlukan untuk pembuatan cache (VPC, subnet, grup keamanan) dan izinkan pembuatan titik akhir VPC (untuk cache tanpa server).
- `kms`— Izinkan penggunaan yang dikelola pelanggan CMKs untuk. encryption-at-rest
- `cloudwatch`— Izinkan akses ke metrik untuk menampilkan ElastiCache metrik di konsol.
- `application-autoscaling` – Mengizinkan akses untuk mendeskripsikan kebijakan Auto Scaling untuk cache.
- `logs` – Digunakan untuk mengisi log stream untuk fungsionalitas pengiriman log di konsol.
- `firehose` – Digunakan untuk mengisi pengiriman aliran untuk fungsionalitas pengiriman log di konsol.
- `s3` – Digunakan untuk mengisi bucket S3 untuk fungsionalitas pemulihan snapshot di konsol.
- `outposts`— Digunakan untuk mengisi AWS Outposts untuk pembuatan cache di konsol.
- `sns` – Digunakan untuk mengisi topik SNS untuk fungsionalitas notifikasi di konsol.

Anda dapat menemukan AmazonElastiCacheFullAccesskebijakan di konsol IAM dan AmazonElastiCacheFullAccessdi Panduan Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AmazonElastiCacheReadOnlyAccess

Anda dapat melampirkan kebijakan `AmazonElastiCacheReadOnlyAccess` ke identitas IAM Anda.

Kebijakan ini mengizinkan akses hanya-baca prinsipal untuk ElastiCache menggunakan Konsol Manajemen: AWS

- `elasticache`— Akses hanya-baca `Describe` APIs.

Anda dapat menemukan AmazonElastiCacheReadOnlyAccesskebijakan di konsol IAM dan AmazonElastiCacheReadOnlyAccessdi Panduan Referensi Kebijakan AWS Terkelola.

ElastiCache pembaruan kebijakan AWS terkelola

Lihat detail tentang pembaruan kebijakan AWS terkelola ElastiCache sejak layanan ini mulai melacak perubahan ini. Untuk peringatan otomatis tentang perubahan pada halaman ini, berlangganan umpan RSS di halaman Riwayat ElastiCache dokumen.

Perubahan	Deskripsi	Tanggal
AmazonElastiCacheFullAccess — Permbaruan ke kebijakan yang sudah ada	ElastiCache menambahkan izin baru untuk memungkinkan penskalaan vertikal untuk MemCached, untuk tindakan. `elasticache:ModifyCacheCluster`	Maret 27, 2025
AmazonElastiCacheFullAccess – Pembaruan ke kebijakan yang ada	ElastiCache menambahkan izin baru untuk memungkinkan pengelolaan cache tanpa server, dan untuk mengaktifkan penggunaan semua fitur layanan melalui konsol.	27 November 2023
ElastiCacheServiceRolePolicy – Pembaruan ke kebijakan yang ada	ElastiCache menambahkan izin baru untuk memungkinkan pengelolaan titik akhir VPC untuk sumber daya cache tanpa server.	27 November 2023
ElastiCache mulai melacak perubahan	ElastiCache mulai melacak perubahan untuk kebijakan AWS terkelolanya.	7 Februari 2020

Menggunakan kebijakan berbasis identitas (kebijakan IAM) untuk Amazon ElastiCache

Topik ini memberikan contoh kebijakan berbasis identitas di mana administrator akun dapat melampirkan kebijakan izin ke identitas IAM (yaitu, pengguna, grup, dan peran).

> **Important**
>
> Kami menyarankan Anda terlebih dahulu membaca topik yang menjelaskan konsep dasar dan opsi untuk mengelola akses ke ElastiCache sumber daya Amazon. Untuk informasi selengkapnya, lihat Ikhtisar mengelola izin akses ke sumber daya Anda ElastiCache .

Bagian dalam topik ini membahas hal berikut:

- AWS kebijakan terkelola untuk Amazon ElastiCache
- Contoh kebijakan yang dikelola pelanggan

Berikut ini menunjukkan contoh kebijakan izin saat menggunakan Redis OSS.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowClusterPermissions",
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateServerlessCache",
                "elasticache:CreateCacheCluster",
                "elasticache:DescribeServerlessCaches",
                "elasticache:DescribeReplicationGroups",
                "elasticache:DescribeCacheClusters",
                "elasticache:ModifyServerlessCache",
                "elasticache:ModifyReplicationGroup",
                "elasticache:ModifyCacheCluster"
            ],
            "Resource": "*"
        },
        {
```

```
            "Sid": "AllowUserToPassRole",
            "Effect": "Allow",
            "Action": [ "iam:PassRole" ],
            "Resource": "arn:aws:iam::123456789012:role/EC2-roles-for-cluster"
        }
    ]
}
```

Berikut ini menunjukkan contoh kebijakan izin saat menggunakan Memcached.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "AllowClusterPermissions",
        "Effect": "Allow",
        "Action": [
            "elasticache:CreateServerlessCache",
            "elasticache:CreateCacheCluster",
            "elasticache:DescribeServerlessCaches",
            "elasticache:DescribeCacheClusters",
            "elasticache:ModifyServerlessCache",
            "elasticache:ModifyCacheCluster"
        ],
        "Resource": "*"
    },
    {
        "Sid": "AllowUserToPassRole",
        "Effect": "Allow",
        "Action": [ "iam:PassRole" ],
        "Resource": "arn:aws:iam::123456789012:role/EC2-roles-for-cluster"
    }
    ]
}
```

Kebijakan tersebut memiliki dua pernyataan:

- Pernyataan pertama memberikan izin untuk ElastiCache tindakan Amazon (`elasticache:Create*`,,`elasticache:Describe*`) `elasticache:Modify*`

- Pernyataan kedua memberikan izin untuk tindakan IAM (`iam:PassRole`) pada nama peran IAM yang ditentukan pada akhir nilai `Resource`.

Kebijakan tidak menentukan elemen `Principal` karena dalam kebijakan berbasis identitas, Anda tidak menentukan prinsipal yang mendapatkan izin. Saat Anda melampirkan kebijakan kepada pengguna, pengguna tersebut menjadi prinsipal secara implisit. Saat Anda melampirkan kebijakan izin pada peran IAM, prinsipal yang diidentifikasi dalam kebijakan kepercayaan peran tersebut akan mendapatkan izin.

Untuk tabel yang menunjukkan semua tindakan Amazon ElastiCache API dan sumber daya yang diterapkan, lihatElastiCache Izin API: Referensi tindakan, sumber daya, dan kondisi.

Contoh kebijakan yang dikelola pelanggan

Jika Anda tidak menggunakan kebijakan default dan memilih untuk menggunakan kebijakan yang dikelola kustom, pastikan salah satu dari dua hal berikut. Anda harus memiliki izin untuk memanggil `iam:createServiceLinkedRole` (untuk informasi selengkapnya, lihat Contoh 4: Izinkan pengguna memanggil IAM API CreateServiceLinkedRole). Atau Anda seharusnya membuat peran ElastiCache terkait layanan.

Jika digabungkan dengan izin minimum yang diperlukan untuk menggunakan ElastiCache konsol Amazon, kebijakan contoh di bagian ini memberikan izin tambahan. Contoh-contohnya juga relevan dengan AWS SDKs dan AWS CLI.

Untuk petunjuk pengaturan pengguna dan grup IAM, lihat Membuat Pengguna dan Grup Administrator IAM Pertama Anda dalam Panduan Pengguna IAM.

> **Important**
>
> Selalu uji kebijakan IAM Anda secara menyeluruh sebelum menggunakannya dalam produksi. Beberapa ElastiCache tindakan yang tampak sederhana dapat memerlukan tindakan lain untuk mendukungnya saat Anda menggunakan ElastiCache konsol. Misalnya, `elasticache:CreateCacheCluster` memberikan izin untuk membuat klaster cache ElastiCache . Namun, untuk melakukan operasi ini, ElastiCache konsol menggunakan sejumlah `Describe` dan `List` tindakan untuk mengisi daftar konsol.

Contoh

Contoh 1: Izinkan pengguna akses hanya-baca ke sumber daya ElastiCache

Kebijakan berikut memberikan ElastiCache tindakan izin yang memungkinkan pengguna untuk mencantumkan sumber daya. Biasanya, Anda melampirkan jenis kebijakan izin ini ke grup pengelola.

JSON

```
{
   "Version": "2012-10-17",
   "Statement":[{
      "Sid": "ECReadOnly",
      "Effect":"Allow",
      "Action": [
          "elasticache:Describe*",
          "elasticache:List*"],
      "Resource":"*"
      }
   ]
}
```

Contoh 2: Izinkan pengguna untuk melakukan tugas administrator ElastiCache sistem umum

Tugas umum administrator sistem termasuk mengubah sumber daya. Administrator sistem mungkin juga ingin mendapatkan informasi tentang peristiwa ElastiCache . Kebijakan berikut memberikan izin pengguna untuk melakukan ElastiCache tindakan untuk tugas administrator sistem umum ini. Biasanya, Anda melampirkan jenis kebijakan izin ini ke grup administrator sistem.

JSON

```
{
```

```
   "Version": "2012-10-17",
   "Statement":[{
      "Sid": "ECAllowMutations",
      "Effect":"Allow",
      "Action":[
          "elasticache:Modify*",
          "elasticache:Describe*",
          "elasticache:ResetCacheParameterGroup"
      ],
      "Resource":"*"
      }
   ]
}
```

Contoh 3: Izinkan pengguna mengakses semua tindakan ElastiCache API

Kebijakan berikut memungkinkan pengguna untuk mengakses semua ElastiCache tindakan. Sebaiknya Anda memberikan jenis kebijakan izin ini hanya untuk pengguna administrator.

JSON

```
{
   "Version": "2012-10-17",
   "Statement":[{
      "Sid": "ECAllowAll",
      "Effect":"Allow",
      "Action":[
          "elasticache:*"
      ],
      "Resource":"*"
      }
   ]
}
```

Contoh 4: Izinkan pengguna memanggil IAM API CreateServiceLinkedRole

Kebijakan berikut mengizinkan pengguna untuk memanggil API `CreateServiceLinkedRole` IAM. Kami menyarankan Anda memberikan jenis kebijakan izin ini kepada pengguna yang memanggil operasi ElastiCache mutatif.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"CreateSLRAllows",
      "Effect":"Allow",
      "Action":[
        "iam:CreateServiceLinkedRole"
      ],
      "Resource":"*",
      "Condition":{
        "StringLike":{
          "iam:AWS ServiceName":"elasticache.amazonaws.com"
        }
      }
    }
  ]
}
```

Contoh 5: Mengizinkan pengguna untuk terhubung ke cache nirserver menggunakan autentikasi IAM

Kebijakan berikut mengizinkan setiap pengguna untuk terhubung ke cache nirserver menggunakan autentikasi IAM antara 01-04-2023 hingga 30-06-2023.

JSON

```
{
  "Version" : "2012-10-17",
  "Statement" :
  [
    {
      "Effect" : "Allow",
      "Action" : ["elasticache:Connect"],
      "Resource" : [
        "arn:aws:elasticache:us-east-1:123456789012:serverlesscache:*"
      ],
      "Condition": {
        "DateGreaterThan": {"aws:CurrentTime": "2023-04-01T00:00:00Z"},
        "DateLessThan": {"aws:CurrentTime": "2023-06-30T23:59:59Z"}
```

```
      }
    },
    {
      "Effect" : "Allow",
      "Action" : ["elasticache:Connect"],
      "Resource" : [
        "arn:aws:elasticache:us-east-1:123456789012:user:*"
      ]
    }
  ]
}
```

Izin tingkat sumber daya

Anda dapat membatasi cakupan izin dengan menentukan sumber daya dalam kebijakan IAM. Banyak tindakan ElastiCache API mendukung jenis sumber daya yang bervariasi tergantung pada perilaku tindakan. Setiap pernyataan kebijakan IAM memberikan izin untuk tindakan yang dilakukan pada sumber daya. Saat tindakan tersebut tidak dilakukan pada sumber daya yang disebutkan, atau saat Anda memberikan izin untuk melakukan tindakan pada semua sumber daya, maka nilai sumber daya dalam kebijakan tersebut adalah wildcard (*). Untuk banyak tindakan API, Anda dapat membatasi sumber daya yang dapat diubah oleh pengguna dengan menentukan Amazon Resource Name (ARN) sumber daya tersebut, atau pola ARN yang cocok dengan beberapa sumber daya. Untuk membatasi izin berdasarkan sumber daya, tentukan sumber daya berdasarkan ARN.

Untuk melihat daftar jenis sumber daya dan jenis ElastiCache sumber daya ARNs, lihat Sumber Daya yang Ditentukan oleh Amazon ElastiCache di Referensi Otorisasi Layanan. Untuk mempelajari tindakan mana yang dapat Anda tentukan ARN dari setiap sumber daya, lihat Tindakan yang Ditentukan oleh Amazon. ElastiCache

Contoh

- Contoh 1: Izinkan pengguna akses penuh ke jenis ElastiCache sumber daya tertentu
- Contoh 2: Menolak akses pengguna ke cache nirserver.

Contoh 1: Izinkan pengguna akses penuh ke jenis ElastiCache sumber daya tertentu

Kebijakan berikut secara eksplisit mengizinkan semua sumber daya dari jenis cache nirserver.

```
{
```

```
        "Sid": "Example1",
        "Effect": "Allow",
        "Action": "elasticache:*",
        "Resource": [
             "arn:aws:elasticache:us-east-1:account-id:serverlesscache:*"
        ]
}
```

Contoh 2: Menolak akses pengguna ke cache nirserver.

Contoh berikut secara eksplisit menolak akses ke cache nirserver tertentu.

```
{
        "Sid": "Example2",
        "Effect": "Deny",
        "Action": "elasticache:*",
        "Resource": [
            "arn:aws:elasticache:us-east-1:account-id:serverlesscache:name"
        ]
}
```

Menggunakan kunci kondisi

Anda dapat menentukan kondisi yang menentukan cara kebijakan IAM diberlakukan. Di ElastiCache, Anda dapat menggunakan `Condition` elemen kebijakan JSON untuk membandingkan kunci dalam konteks permintaan dengan nilai kunci yang Anda tentukan dalam kebijakan Anda. Untuk informasi selengkapnya, lihat Elemen kebijakan JSON IAM: Kondisi.

Untuk melihat daftar kunci ElastiCache kondisi, lihat Kunci Kondisi untuk Amazon ElastiCache di Referensi Otorisasi Layanan.

Untuk melihat daftar kunci kondisi global, lihat Kunci konteks kondisi global AWS.

Menggunakan ElastiCache Dengan Kunci Kondisi AWS Global

Saat menggunakan kunci kondisi AWS Global yang memerlukan ElastiCache Principal, gunakan `OR` kondisi dengan kedua Prinsip: dan. `elasticache.amazonaws.com` `ec.amazonaws.com`

> **Note**
>
> Jika Anda tidak menambahkan kedua Prinsip untuk ElastiCache, tindakan “Izinkan” atau “Tolak” yang dimaksudkan tidak akan diterapkan dengan benar untuk sumber daya apa pun yang tercantum dalam kebijakan Anda.

Contoh kebijakan dengan kunci kondisi `aws:CalledVia` global:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:*",
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringLike": {
          "aws:CalledVia": [
            "ec.amazonaws.com",
            "elasticache.amazonaws.com"
          ]
        }
      }
    }
  ]
}
```

Menentukan Kondisi: Menggunakan Kunci Kondisi

Untuk mengimplementasikan kontrol yang lebih spesifik, Anda menulis kebijakan izin IAM yang menentukan kondisi untuk mengontrol set parameter individual pada permintaan tertentu. Kemudian Anda menerapkan kebijakan ke pengguna, grup, atau peran IAM yang Anda buat menggunakan konsol IAM.

Untuk menerapkan kondisi tersebut, Anda menambahkan informasi kondisi pada pernyataan kebijakan IAM. Pada contoh berikut, Anda menentukan kondisi bahwa setiap klaster cache yang dirancang sendiri akan menjadi jenis simpul `cache.r5.large`.

> **Note**
>
> - Untuk membangun `Condition` elemen menggunakan kunci kondisi `String` tipe, gunakan operator kondisi tidak sensitif huruf besar/kecil `StringEqualsIgnoreCase` atau `StringNotEqualsIgnoreCase` untuk membandingkan kunci dengan nilai string.
> - ElastiCache memproses argumen masukan untuk `CacheNodeType` dan `CacheParameterGroupName` dengan cara yang tidak peka huruf besar/kecil. Untuk alasan ini, string mengkondisikan operator`StringEqualsIgnoreCase`, dan `StringNotEqualsIgnoreCase` harus digunakan dalam kebijakan izin yang mereferensikannya.

Berikut ini menunjukkan contoh kebijakan izin ini saat menggunakan Valkey atau Redis OSS.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster",
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster",
                "elasticache:CreateReplicationGroup"
```

```
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:cluster:*",
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
            "Condition": {
                "StringEquals": {
                    "elasticache:CacheNodeType": [
                        "cache.r5.large"
                    ]
                }
            }
        }
    ]
}
```

Berikut ini menunjukkan contoh kebijakan izin ini saat menggunakan Memcached.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:cluster:*"
            ],
            "Condition": {
```

```
                "StringEquals": {
                    "elasticache:CacheNodeType": [
                        "cache.r5.large"
                    ]
                }
            }
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Menandai sumber daya Anda ElastiCache .

Untuk informasi selengkapnya tentang penggunaan operator kondisi kebijakan, lihat ElastiCache Izin API: Referensi tindakan, sumber daya, dan kondisi.

Kebijakan Contoh: Menggunakan Kondisi untuk Kontrol Parameter Terperinci

Bagian ini menunjukkan contoh kebijakan untuk menerapkan kontrol akses berbutir halus pada parameter yang tercantum sebelumnya. ElastiCache

1. elasticache: MaximumDataStorage: Tentukan penyimpanan data maksimum cache tanpa server. Dengan kondisi yang disediakan, pelanggan tidak dapat membuat cache yang dapat menyimpan lebih dari jumlah data ditentukan.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowDependentResources",
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateServerlessCache"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:serverlesscachesnapshot:*",
                "arn:aws:elasticache:*:*:snapshot:*",
                "arn:aws:elasticache:*:*:usergroup:*"
            ]
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateServerlessCache"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:serverlesscache:*"
            ],
            "Condition": {
                "NumericLessThanEquals": {
                    "elasticache:MaximumDataStorage": "30"
                },
                "StringEquals": {
                    "elasticache:DataStorageUnit": "GB"
                }
            }
        }
    ]
}
```

2. ElastiCache: Maximum ECPUPer Second: Tentukan nilai ECPU maksimum per detik dari cache tanpa server. Dengan menggunakan kondisi yang disediakan, pelanggan tidak dapat membuat cache yang dapat mengeksekusi lebih dari jumlah tertentu ECPUs per detik.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowDependentResources",
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateServerlessCache"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:serverlesscachesnapshot:*",
                "arn:aws:elasticache:*:*:snapshot:*",
                "arn:aws:elasticache:*:*:usergroup:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateServerlessCache"
```

```
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:serverlesscache:*"
            ],
            "Condition": {
                "NumericLessThanEquals": {
                    "elasticache:MaximumECPUPerSecond": "100000"
                }
            }
        }
    ]
}
```

3. elasticache: CacheNodeType: Tentukan mana yang NodeType dapat dibuat pengguna. Dengan kondisi yang disediakan, pelanggan dapat menentukan nilai tunggal atau nilai rentang untuk jenis simpul.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
         {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster",
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster",
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:cluster:*",
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
```

```
            "Condition": {
                "StringEquals": {
                    "elasticache:CacheNodeType": [
                        "cache.t2.micro",
                        "cache.t2.medium"
                    ]
                }
            }
        }
    ]
}
```

4. elasticache: CacheNodeType: Dengan Memcached, tentukan mana yang NodeType dapat dibuat pengguna. Dengan kondisi yang disediakan, pelanggan dapat menentukan nilai tunggal atau nilai rentang untuk jenis simpul.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
         {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:cluster:*"
            ],
            "Condition": {
                "StringEquals": {
                    "elasticache:CacheNodeType": [
                        "cache.t2.micro",
```

```
                    "cache.t2.medium"
                ]
            }
        }
    ]
}
```

5. elasticache:NumNodeGroups: Buat grup replikasi dengan kurang dari 20 grup node.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
         {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
             "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
             "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
            "Condition": {
                "NumericLessThanEquals": {
                    "elasticache:NumNodeGroups": "20"
                }
            }
        }
    ]
}
```

6. elasticache:ReplicasPerNodeGroup: Tentukan replika per node antara 5 dan 10.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
         {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
            "Condition": {
                "NumericGreaterThanEquals": {
                    "elasticache:ReplicasPerNodeGroup": "5"
                },
                "NumericLessThanEquals": {
                    "elasticache:ReplicasPerNodeGroup": "10"
                }
            }
        }
    ]
}
```

7. elasticache:EngineVersion: Tentukan penggunaan engine versi 5.0.6.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
    {
           "Effect": "Allow",
           "Action": [
               "elasticache:CreateCacheCluster",
               "elasticache:CreateReplicationGroup"
           ],
           "Resource": [
               "arn:aws:elasticache:*:*:parametergroup:*",
               "arn:aws:elasticache:*:*:subnetgroup:*"
           ]
       },

       {
          "Effect": "Allow",
           "Action": [
               "elasticache:CreateCacheCluster",
               "elasticache:CreateReplicationGroup"
           ],
           "Resource": [
               "arn:aws:elasticache:*:*:cluster:*",
               "arn:aws:elasticache:*:*:replicationgroup:*"
           ],
           "Condition": {
               "StringEquals": {
                   "elasticache:EngineVersion": "5.0.6"
               }
           }
       }
   ]
}
```

8. elasticache:EngineVersion: Tentukan penggunaan mesin Memcached versi 1.6.6

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
     {
           "Effect": "Allow",
           "Action": [
               "elasticache:CreateCacheCluster"
           ],
           "Resource": [
```

```
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:cluster:*"
            ],
            "Condition": {
                "StringEquals": {
                    "elasticache:EngineVersion": "1.6.6"
                }
            }
        }
    ]
}
```

9. elasticache:EngineType: Tentukan hanya menggunakan mesin Valkey atau Redis OSS.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
         {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster",
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
```

```
                "elasticache:CreateCacheCluster",
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:cluster:*",
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
            "Condition": {
                "StringEquals": {
                    "elasticache:EngineType": "redis"
                }
            }
        }
    ]
}
```

10. elasticache:AtRestEncryptionEnabled: Tentukan bahwa grup replikasi hanya akan dibuat dengan enkripsi diaktifkan.

    JSON

```
{
    "Version": "2012-10-17",
    "Statement": [

         {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
```

```
            "Condition": {
                "Bool": {
                    "elasticache:AtRestEncryptionEnabled": "true"
                }
            }
        }
    ]
}
```

11. elastisakit: TransitEncryptionEnabled

    a. Setel kunci `elasticache:TransitEncryptionEnabled` kondisi `false` untuk CreateReplicationGroup tindakan untuk menentukan bahwa grup replikasi hanya dapat dibuat ketika TLS tidak digunakan:

    JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
            "Condition": {
                "Bool": {
                    "elasticache:TransitEncryptionEnabled": "false"
                }
            }
```

```
        }
    ]
}
```

Ketika kunci `elasticache:TransitEncryptionEnabled` kondisi disetel ke `false` dalam kebijakan untuk CreateReplicationGrouptindakan, `CreateReplicationGroup` permintaan hanya akan diizinkan jika TLS tidak digunakan (yaitu, jika permintaan tidak menyertakan parameter yang disetel ke `true` atau `TransitEncryptionEnabled` `TransitEncryptionMode` parameter yang disetel ke`required`.

b. Setel kunci `elasticache:TransitEncryptionEnabled` konditon ke `true` untuk CreateReplicationGrouptindakan untuk menentukan bahwa grup replikasi hanya dapat dibuat ketika TLS sedang digunakan:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
            "Condition": {
                "Bool": {
                    "elasticache:TransitEncryptionEnabled": "true"
                }
            }
```

```
        }
    ]
}
```

Jika kunci `elasticache:TransitEncryptionEnabled` kondisi disetel ke `true` dalam kebijakan untuk CreateReplicationGrouptindakan, `CreateReplicationGroup` permintaan hanya akan diizinkan jika permintaan menyertakan `TransitEncryptionEnabled` parameter yang disetel ke `true` dan `TransitEncryptionMode` parameter yang disetel ke`required`.

c. Atur `elasticache:TransitEncryptionEnabled` ke `true` untuk tindakan `ModifyReplicationGroup` guna menentukan bahwa grup replikasi hanya dapat diubah ketika TLS sedang digunakan:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "elasticache:ModifyReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
            "Condition": {
                "BoolIfExists": {
                    "elasticache:TransitEncryptionEnabled": "true"
                }
            }
        }
    ]
}
```

Jika kunci `elasticache:TransitEncryptionEnabled` kondisi disetel ke `true` dalam kebijakan untuk ModifyReplicationGrouptindakan, `ModifyReplicationGroup` permintaan hanya akan diizinkan jika permintaan menyertakan `TransitEncryptionMode` parameter yang disetel ke`required`. Parameter `TransitEncryptionEnabled` yang diatur ke `true` juga dapat disertakan, tetapi tidak diperlukan dalam kasus ini untuk mengaktifkan TLS.

12. elasticache:AutomaticFailoverEnabled: Tentukan bahwa grup replikasi hanya akan dibuat dengan failover otomatis diaktifkan.

    JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
         {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
            "Condition": {
                "Bool": {
                    "elasticache:AutomaticFailoverEnabled": "true"
                }
            }
        }
    ]
}
```

13. Elasticache:multiAZEnabled: Tentukan bahwa grup replikasi tidak dapat dibuat dengan multi-AZ dinonaktifkan.

    JSON

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
         {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster",
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },
        {
            "Effect": "Deny",
            "Action": [
                "elasticache:CreateCacheCluster",
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:cluster:*",
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
            "Condition": {
                "Bool": {
                    "elasticache:MultiAZEnabled": "false"
                }
            }
        }
    ]
}
```

14. elasticache:ClusterModeEnabled: Tentukan bahwa grup replikasi hanya dapat dibuat dengan mode cluster diaktifkan.

    JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
         {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateReplicationGroup"
            ],
```

```
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
            "Condition": {
                "Bool": {
                    "elasticache:ClusterModeEnabled": "true"
                }
            }
        }
    ]
}
```

15. elasticache:AuthTokenEnabled: Tentukan bahwa grup replikasi hanya dapat dibuat dengan token AUTH diaktifkan.

    JSON

```
{
    "Version": "2012-10-17",
    "Statement": [

         {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster",
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },
```

```
        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster",
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:cluster:*",
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
            "Condition": {
                "Bool": {
                    "elasticache:AuthTokenEnabled": "true"
                }
            }
        }
    ]
}
```

16. elasticache:SnapshotRetentionLimit: Tentukan jumlah hari (atau min/maks) untuk menyimpan snapshot. Kebijakan di bawah ini memberlakukan penyimpanan cadangan selama setidaknya 30 hari.

    JSON

```
{
    "Version": "2012-10-17",
    "Statement": [

         {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster",
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
```

```
                "elasticache:CreateCacheCluster",
                "elasticache:CreateReplicationGroup",
                "elasticache:CreateServerlessCache"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:cluster:*",
                "arn:aws:elasticache:*:*:replicationgroup:*",
                "arn:aws:elasticache:*:*:serverlesscache:*"
            ],
            "Condition": {
                "NumericGreaterThanEquals": {
                    "elasticache:SnapshotRetentionLimit": "30"
                }
            }
        }
    ]
}
```

17. elasticache:KmsKeyId: Tentukan penggunaan kunci AWS KMS yang dikelola pelanggan.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
        "Sid": "AllowDependentResources",
        "Effect": "Allow",
        "Action": [
            "elasticache:CreateServerlessCache"
        ],
        "Resource": [
            "arn:aws:elasticache:*:*:serverlesscachesnapshot:*",
            "arn:aws:elasticache:*:*:snapshot:*",
            "arn:aws:elasticache:*:*:usergroup:*"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "elasticache:CreateServerlessCache"
        ],
        "Resource": [
            "arn:aws:elasticache:*:*:serverlesscache:*"
```

```
        ],
        "Condition": {
            "StringEquals": {
                "elasticache:KmsKeyId": "my-key"
            }
        }
    }
  ]
}
```

18. elasticache: CacheParameterGroupName: Tentukan grup parameter non default dengan parameter spesifik dari organisasi di cluster Anda. Anda juga dapat menentukan pola penamaan untuk grup parameter Anda atau memblokir dan menghapus nama grup parameter tertentu. Berikut ini adalah contoh membatasi penggunaan hanya ""my-org-param-group.

    JSON

```
{
    "Version": "2012-10-17",
    "Statement": [

         {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster",
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster",
                "elasticache:CreateReplicationGroup"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:cluster:*",
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ],
```

```
            "Condition": {
                "StringEquals": {
                    "elasticache:CacheParameterGroupName": "my-org-param-group"
                }
            }
        }
    ]
}
```

19. elasticache: CacheParameterGroupName: Dengan Memcached, tentukan grup parameter non default dengan parameter spesifik dari organisasi di cluster Anda. Anda juga dapat menentukan pola penamaan untuk grup parameter Anda atau memblokir dan menghapus nama grup parameter tertentu. Berikut ini adalah contoh membatasi penggunaan hanya ""my-org-param-group.

    JSON

```
{
    "Version": "2012-10-17",
    "Statement": [

         {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:cluster:*"
            ],
            "Condition": {
                "StringEquals": {
                    "elasticache:CacheParameterGroupName": "my-org-param-group"
                }
```

```
            }
        }
    ]
}
```

20. elasticache: CreateCacheCluster: Menolak `CreateCacheCluster` tindakan jika tag permintaan `Project` hilang atau tidak sama dengan, atau. `Dev` `QA` `Prod`

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
          {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:parametergroup:*",
                "arn:aws:elasticache:*:*:subnetgroup:*",
                "arn:aws:elasticache:*:*:securitygroup:*",
                "arn:aws:elasticache:*:*:replicationgroup:*"
            ]
        },
        {
            "Effect": "Deny",
            "Action": [
                "elasticache:CreateCacheCluster"
            ],
            "Resource": [
                "arn:aws:elasticache:*:*:cluster:*"
            ],
            "Condition": {
                "Null": {
                    "aws:RequestTag/Project": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "elasticache:CreateCacheCluster",
                "elasticache:AddTagsToResource"
```

```
            ],
            "Resource": "arn:aws:elasticache:*:*:cluster:*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/Project": [
                        "Dev",
                        "Prod",
                        "QA"
                    ]
                }
            }
        }
    ]
}
```

21. elasticache:CacheNodeType: Mengizinkan `CreateCacheCluster` dengan `cacheNodeType` cache.r5.large atau cache.r6g.4xlarge dan tag. `Project=XYZ`

    JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
      "Effect": "Allow",
      "Action": [
        "elasticache:CreateCacheCluster",
        "elasticache:CreateReplicationGroup"
      ],
      "Resource": [
        "arn:aws:elasticache:*:*:parametergroup:*",
        "arn:aws:elasticache:*:*:subnetgroup:*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "elasticache:CreateCacheCluster"
      ],
      "Resource": [
        "arn:aws:elasticache:*:*:cluster:*"
      ],
      "Condition": {
        "StringEqualsIfExists": {
```

```
          "elasticache:CacheNodeType": [
            "cache.r5.large",
            "cache.r6g.4xlarge"
          ]
        },
        "StringEquals": {
          "aws:RequestTag/Project": "XYZ"
        }
      }
    }
  ]
}
```

22. elasticache:CacheNodeType: Mengizinkan `CreateCacheCluster` dengan `cacheNodeType` cache.r5.large atau cache.r6g.4xlarge dan tag. `Project=XYZ`

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
      "Effect": "Allow",
      "Action": [
        "elasticache:CreateCacheCluster"
      ],
      "Resource": [
        "arn:aws:elasticache:*:*:parametergroup:*",
        "arn:aws:elasticache:*:*:subnetgroup:*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "elasticache:CreateCacheCluster"
      ],
      "Resource": [
        "arn:aws:elasticache:*:*:cluster:*"
      ],
      "Condition": {
        "StringEqualsIfExists": {
          "elasticache:CacheNodeType": [
            "cache.r5.large",
            "cache.r6g.4xlarge"
```

```
          ]
        },
        "StringEquals": {
          "aws:RequestTag/Project": "XYZ"
        }
      }
    }
  ]
}
```

> **Note**
>
> Saat membuat kebijakan untuk memberlakukan tag dan kunci kondisi lainnya secara bersama, `IfExists` bersyarat mungkin diperlukan pada elemen kunci kondisi karena persyaratan kebijakan tambahan `elasticache:AddTagsToResource` untuk permintaan pembuatan dengan parameter `--tags`.

Menggunakan Peran Tertaut Layanan untuk Amazon ElastiCache

Amazon ElastiCache menggunakan peran AWS Identity and Access Management terkait layanan (IAM). Peran terkait layanan adalah jenis peran IAM unik yang ditautkan langsung ke AWS layanan, seperti Amazon. ElastiCache ElastiCache Peran terkait layanan Amazon telah ditentukan sebelumnya oleh Amazon. ElastiCache Peran tersebut menyertakan semua izin yang diperlukan layanan untuk memanggil layanan AWS atas nama klaster Anda.

Peran terkait layanan membuat pengaturan Amazon ElastiCache lebih mudah karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Peran sudah ada dalam AWS akun Anda tetapi ditautkan ke kasus ElastiCache penggunaan Amazon dan memiliki izin yang telah ditentukan sebelumnya. Hanya Amazon yang ElastiCache dapat mengambil peran ini, dan hanya peran ini yang dapat menggunakan kebijakan izin yang telah ditentukan sebelumnya. Anda dapat menghapus peran tersebut hanya setelah pertama kali menghapus sumber dayanya yang terkait. Ini melindungi ElastiCache sumber daya Amazon Anda karena Anda tidak dapat secara tidak sengaja menghapus izin yang diperlukan untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran terkait layanan, lihat Layanan AWS yang bisa digunakan dengan IAM dan carilah layanan yang memiliki opsi Ya di kolom Peran Terkait Layanan. Pilih Ya dengan sebuah tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Daftar Isi

Izin Peran Tertaut Layanan untuk Amazon ElastiCache

Izin untuk membuat peran terkait layanan

Untuk mengizinkan entitas IAM membuat peran terkait AWS ServiceRoleForElastiCache layanan

Tambahkan pernyataan kebijakan berikut ini ke izin untuk entitas IAM:

```
{
    "Effect": "Allow",
    "Action": [
        "iam:CreateServiceLinkedRole",
        "iam:PutRolePolicy"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/elasticache.amazonaws.com/AWS
 ServiceRoleForElastiCache*",
    "Condition": {"StringLike": {"iam:AWS ServiceName": "elasticache.amazonaws.com"}}
}
```

Untuk mengizinkan entitas IAM menghapus peran terkait AWS ServiceRoleForElastiCache layanan

Tambahkan pernyataan kebijakan berikut ini ke izin untuk entitas IAM:

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/elasticache.amazonaws.com/AWS
 ServiceRoleForElastiCache*",
    "Condition": {"StringLike": {"iam:AWS ServiceName": "elasticache.amazonaws.com"}}
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola untuk menyediakan akses penuh ke Amazon ElastiCache.

Membuat Peran Terkait Layanan (IAM)

Anda dapat membuat peran terkait layanan menggunakan konsol IAM, CLI, atau API.

Membuat Peran Terkait Layanan (Konsol IAM)

Anda dapat menggunakan konsol IAM untuk membuat peran terkait layanan.

Untuk membuat peran terkait layanan (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol IAM, pilih Peran. Kemudian pilih Buat peran baru.
3. Di bagian Pilih jenis entitas tepercaya, pilih Layanan AWS .
4. Di bawah Atau pilih layanan untuk melihat kasus penggunaannya, pilih ElastiCache.
5. Pilih Berikutnya: Izin.
6. Di bagian Nama kebijakan, perhatikan bahwa `ElastiCacheServiceRolePolicy` diperlukan untuk peran ini. Pilih Selanjutnya: Tag.
7. Perhatikan bahwa tag tidak didukung untuk peran Tertaut-Layanan. Pilih Selanjutnya: Tinjau.
8. (Opsional) Untuk Deskripsi peran, edit deskripsi untuk peran terkait layanan baru.

9. Tinjau peran, lalu pilih Buat peran.

Membuat Peran Terkait Layanan (CLI IAM)

Anda dapat menggunakan operasi IAM dari AWS Command Line Interface untuk membuat peran terkait layanan. Peran ini dapat mencakup kebijakan kepercayaan dan kebijakan terkait yang diperlukan oleh layanan untuk mengambil peran tersebut.

Untuk membuat peran terkait layanan (CLI)

Gunakan operasi berikut:

```
$ aws iam create-service-linked-role --aws-service-name elasticache.amazonaws.com
```

Membuat Peran Terkait Layanan (API IAM)

Anda dapat menggunakan API IAM untuk membuat peran terkait layanan. Peran ini dapat berisi kebijakan kepercayaan dan kebijakan terkait yang diperlukan oleh layanan untuk mengambil peran tersebut.

Untuk membuat peran terkait layanan (API)

Gunakan panggilan API CreateServiceLinkedRole. Dalam permintaan, sebutkan nama layanan `elasticache.amazonaws.com`.

Mengedit Deskripsi Peran Tertaut Layanan untuk Amazon ElastiCache

Amazon ElastiCache tidak mengizinkan Anda mengedit peran AWS ServiceRoleForElastiCache terkait layanan. Setelah membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin merujuk peran tersebut. Namun, Anda dapat menyunting Deskripsi peran menggunakan IAM.

Mengedit Deskripsi Peran Terkait Layanan (Konsol IAM)

Anda dapat menggunakan konsol IAM untuk mengedit deskripsi peran terkait layanan.

Untuk mengedit deskripsi peran terkait layanan (konsol)

1. Di panel navigasi konsol IAM, pilih Peran.
2. Pilih nama peran yang akan diubah.

3. Di ujung kanan Deskripsi peran, pilih Edit.
4. Masukkan deskripsi baru di kotak, lalu pilih Simpan.

Mengedit Deskripsi Peran Terkait Layanan (CLI IAM)

Anda dapat menggunakan operasi IAM dari AWS Command Line Interface untuk mengedit deskripsi peran terkait layanan.

Untuk mengubah deskripsi peran terkait layanan (CLI)

1. (Opsional) Untuk melihat deskripsi saat ini untuk peran, gunakan AWS CLI untuk operasi `get-role` IAM.

   Example

   ```
   $ aws iam get-role --role-name AWS ServiceRoleForElastiCache
   ```

   Gunakan nama peran, bukan ARN, untuk mengacu ke peran itu dengan operasi CLI. Misalnya, jika peran memiliki ARN berikut: `arn:aws:iam::123456789012:role/myrole`, peran akan dirujuk sebagai **`myrole`**.
2. Untuk memperbarui deskripsi peran terkait layanan, gunakan operasi AWS CLI untuk IAM. `update-role-description`

   Untuk Linux, macOS, atau Unix:

   ```
   $ aws iam update-role-description \
       --role-name AWS ServiceRoleForElastiCache \
       --description "new description"
   ```

   Untuk Windows:

   ```
   $ aws iam update-role-description ^
       --role-name AWS ServiceRoleForElastiCache ^
       --description "new description"
   ```

Mengedit Deskripsi Peran Terkait Layanan (API IAM)

Anda dapat menggunakan API IAM untuk mengedit deskripsi peran terkait layanan.

Untuk mengubah deskripsi peran terkait layanan (API)

1. (Opsional) Untuk melihat deskripsi peran saat ini, gunakan operasi API IAM GetRole.

   Example

   ```
   https://iam.amazonaws.com/
      ?Action=GetRole
      &RoleName=AWS ServiceRoleForElastiCache
      &Version=2010-05-08
      &AUTHPARAMS
   ```

2. Untuk memperbarui deskripsi peran, gunakan operasi API IAM UpdateRoleDescription.

   Example

   ```
   https://iam.amazonaws.com/
      ?Action=UpdateRoleDescription
      &RoleName=AWS ServiceRoleForElastiCache
      &Version=2010-05-08
      &Description="New description"
   ```

Menghapus Peran Tertaut Layanan untuk Amazon ElastiCache

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran terkait layanan, sebaiknya hapus peran tersebut. Dengan begitu, Anda tidak perlu lagi memantau atau memelihara entitas yang tidak digunakan. Namun, Anda harus membersihkan peran terkait layanan sebelum dapat menghapusnya.

Amazon ElastiCache tidak menghapus peran terkait layanan untuk Anda.

Membersihkan Peran Terkait Layanan

Sebelum Anda dapat menggunakan IAM untuk menghapus peran terkait layanan, pastikan terlebih dahulu bahwa peran tersebut tidak memiliki sumber daya (klaster atau grup replikasi) yang terkait dengannya.

Untuk memastikan peran terkait layanan memiliki sesi aktif di konsol IAM

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.

2. Di panel navigasi konsol IAM, pilih Peran. Kemudian pilih nama (bukan kotak centang) AWS ServiceRoleForElastiCache peran.
3. Di halaman Ringkasan untuk peran yang dipilih, pilih tab Penasihat Akses.
4. Di tab Penasihat Akses, tinjau aktivitas terbaru untuk peran terkait layanan tersebut.

Untuk menghapus ElastiCache sumber daya Amazon yang membutuhkan AWS ServiceRoleForElastiCache

- Untuk menghapus klaster, lihat referensi berikut:
  - Menggunakan AWS Management Console
  - Menggunakan AWS CLI untuk menghapus ElastiCache cluster
  - Menggunakan ElastiCache API
- Untuk menghapus grup replikasi, lihat referensi berikut:
  - Menghapus Grup Replikasi (Konsol)
  - Menghapus Grup Replikasi (AWS CLI)
  - Menghapus grup replikasi (API) ElastiCache

Menghapus Peran Terkait Layanan (Konsol IAM)

Anda dapat menggunakan konsol IAM untuk menghapus sebuah peran terkait layanan.

Untuk menghapus peran terkait layanan (konsol)

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol IAM, pilih Peran. Kemudian, pilih kotak centang di sebelah nama peran yang ingin dihapus, bukan nama atau baris itu sendiri.
3. Untuk Tindakan peran di bagian atas halaman, pilih Hapus peran.
4. Di kotak dialog konfirmasi, tinjau data layanan yang terakhir diakses, yang menunjukkan kapan masing-masing peran yang dipilih terakhir mengakses AWS layanan. Hal ini membantu Anda mengonfirmasi aktif tidaknya peran tersebut saat ini. Jika Anda ingin melanjutkan, pilih Ya, Hapus guna mengirimkan peran terkait layanan untuk penghapusan.
5. Perhatikan notifikasi konsol IAM untuk memantau progres penghapusan peran terkait layanan. Karena penghapusan peran terkait layanan IAM bersifat asinkron, setelah Anda mengirimkan peran tersebut untuk penghapusan, tugas penghapusan dapat berhasil atau gagal. Jika tugas

tersebut gagal, Anda dapat memilih Lihat detail atau Lihat Sumber Daya dari notifikasi untuk mempelajari alasan gagalnya penghapusan.

Menghapus Peran Terkait Layanan (CLI IAM)

Anda dapat menggunakan operasi IAM dari AWS Command Line Interface untuk menghapus peran terkait layanan.

Untuk menghapus peran terkait layanan (CLI)

1. Jika Anda tidak tahu nama peran terkait layanan yang ingin dihapus, masukkan perintah berikut. Perintah ini mencantumkan peran dan Nama Sumber Daya Amazon mereka (ARNs) di akun Anda.

   ```
   $ aws iam get-role --role-name role-name
   ```

   Gunakan nama peran, bukan ARN, untuk merujuk ke peran dengan operasi CLI. Misalnya, jika peran memiliki ARN `arn:aws:iam::123456789012:role/myrole`, peran akan dirujuk sebagai **`myrole`**.

2. Karena peran terkait layanan tidak dapat dihapus jika sedang digunakan atau memiliki sumber daya terkait, Anda harus mengirimkan permintaan penghapusan. Permintaan tersebut dapat ditolak jika kondisi ini tidak terpenuhi. Anda harus menangkap `deletion-task-id` dari tanggapan untuk memeriksa status tugas penghapusan. Masukkan perintah berikut untuk mengirimkan permintaan penghapusan peran terkait layanan.

   ```
   $ aws iam delete-service-linked-role --role-name role-name
   ```

3. Masukkan perintah berikut untuk memeriksa status tugas penghapusan.

   ```
   $ aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
   ```

   Status tugas penghapusan dapat berupa `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED`, atau `FAILED`. Jika penghapusan gagal, panggilan akan mengembalikan alasan kegagalan panggilan agar Anda dapat memecahkan masalah.

Menghapus Peran Terkait Layanan (API IAM)

Anda dapat menggunakan API IAM untuk menghapus peran terkait layanan.

Untuk menghapus peran terkait layanan (API)

1. Untuk mengirimkan permintaan penghapusan untuk peran terkait layanan, panggil DeleteServiceLinkedRole. Di permintaan tersebut, tentukan nama peran.

   Karena peran terkait layanan tidak dapat dihapus jika sedang digunakan atau memiliki sumber daya terkait, Anda harus mengirimkan permintaan penghapusan. Permintaan tersebut dapat ditolak jika kondisi ini tidak terpenuhi. Anda harus menangkap `DeletionTaskId` dari tanggapan untuk memeriksa status tugas penghapusan.
2. Untuk memeriksa status penghapusan, panggil GetServiceLinkedRoleDeletionStatus. Di permintaan tersebut, tentukan `DeletionTaskId`.

   Status tugas penghapusan dapat berupa `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED`, atau `FAILED`. Jika penghapusan gagal, panggilan akan mengembalikan alasan kegagalan panggilan agar Anda dapat memecahkan masalah.

ElastiCache Izin API: Referensi tindakan, sumber daya, dan kondisi

Saat Anda mengatur kontrol akses dan menulis kebijakan izin untuk dilampirkan ke kebijakan IAM (baik berbasis identitas atau berbasis sumber daya), gunakan tabel berikut sebagai referensi. Tabel mencantumkan setiap operasi Amazon ElastiCache API dan tindakan terkait yang dapat Anda berikan izin untuk melakukan tindakan. Anda menentukan tindakan dalam bidang `Action` kebijakan, dan Anda menentukan nilai sumber daya pada bidang `Resource` kebijakan. Kecuali dinyatakan sebaliknya, sumber daya wajib ditentukan. Beberapa bidang mencakup sumber daya wajib dan sumber daya opsional. Jika tidak terdapat ARN sumber daya, sumber daya dalam kebijakan adalah wildcard (*).

Anda dapat menggunakan tombol kondisi dalam ElastiCache kebijakan Anda untuk menyatakan kondisi. Untuk melihat daftar kunci kondisi ElastiCache khusus, bersama dengan tindakan dan jenis sumber daya yang diterapkan, lihatMenggunakan kunci kondisi. Untuk daftar lengkap kunci AWS-wide, lihat kunci konteks kondisi AWS global di Panduan Pengguna IAM.

> **Note**
>
> Untuk menentukan tindakan, gunakan awalan `elasticache:` diikuti dengan nama operasi API (misalnya, `elasticache:DescribeCacheClusters`).

Untuk melihat daftar ElastiCache tindakan, lihat Tindakan yang Ditentukan oleh Amazon ElastiCache di Referensi Otorisasi Layanan.

Validasi kepatuhan untuk Amazon ElastiCache

Auditor pihak ketiga menilai keamanan dan kepatuhan AWS layanan sebagai bagian dari beberapa program AWS kepatuhan, seperti SOC, PCI, FedRAMP, dan HIPAA.

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat Layanan AWS di Lingkup oleh Program Kepatuhan Layanan AWS dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat Program AWS Kepatuhan Program AWS .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat Mengunduh Laporan di AWS Artifact .

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- Kepatuhan dan Tata Kelola Keamanan – Panduan implementasi solusi ini membahas pertimbangan arsitektur serta memberikan langkah-langkah untuk menerapkan fitur keamanan dan kepatuhan.
- Referensi Layanan yang Memenuhi Syarat HIPAA — Daftar layanan yang memenuhi syarat HIPAA. Tidak semua memenuhi Layanan AWS syarat HIPAA.
- AWS Sumber Daya AWS — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- AWS Panduan Kepatuhan Pelanggan - Memahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan ini merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan untuk kontrol keamanan di berbagai kerangka kerja (termasuk Institut Standar dan Teknologi Nasional (NIST), Dewan Standar Keamanan Industri Kartu Pembayaran (PCI), dan Organisasi Internasional untuk Standardisasi (ISO)).
- Mengevaluasi Sumber Daya dengan Aturan dalam Panduan AWS Config Pengembang — AWS Config Layanan menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- AWS Security Hub— Ini Layanan AWS memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS. Security Hub menggunakan kontrol keamanan untuk sumber daya AWS Anda serta untuk memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat Referensi kontrol Security Hub.
- Amazon GuardDuty — Ini Layanan AWS mendeteksi potensi ancaman terhadap beban kerja Akun AWS, kontainer, dan data Anda dengan memantau lingkungan Anda untuk aktivitas yang mencurigakan dan berbahaya. GuardDuty dapat membantu Anda mengatasi berbagai persyaratan kepatuhan, seperti PCI DSS, dengan memenuhi persyaratan deteksi intrusi yang diamanatkan oleh kerangka kerja kepatuhan tertentu.
- AWS Audit ManagerIni Layanan AWS membantu Anda terus mengaudit AWS penggunaan Anda untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap peraturan dan standar industri.

- Ketika Amazon ElastiCache terdaftar sebagai berada dalam lingkup program kepatuhan, cakupan ini berlaku untuk semua mesin cache yang didukung oleh ElastiCache. Ini termasuk ElastiCache untuk Valkey, ElastiCache untuk Memcached, dan ElastiCache untuk Redis OSS.

Informasi selengkapnya

Untuk informasi umum tentang kepatuhan AWS Cloud, lihat berikut ini:

- Titik Akhir FIPS berdasarkan Layanan
- Pembaruan layanan di ElastiCache
- AWS Kepatuhan Cloud
- Model Tanggung Jawab Bersama
- AWS Program Kepatuhan PCI DSS

Ketahanan di Amazon ElastiCache

Infrastruktur AWS global dibangun di sekitar AWS Wilayah dan Zona Ketersediaan. AWS Wilayah menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang sangat redundan. Dengan Zona Ketersediaan, Anda dapat merancang dan mengoperasikan aplikasi dan basis data yang secara otomatis melakukan failover di antara Zona Ketersediaan tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur biasa yang terdiri dari satu atau beberapa pusat data.

Untuk informasi selengkapnya tentang AWS Wilayah dan Availability Zone, lihat Infrastruktur AWS Global.

Selain infrastruktur AWS global, Amazon ElastiCache menawarkan beberapa fitur untuk membantu mendukung ketahanan data dan kebutuhan cadangan Anda.

Topik

- Mitigasi Kegagalan

Mitigasi Kegagalan

Saat merencanakan ElastiCache implementasi Amazon Anda, Anda harus merencanakan sehingga kegagalan memiliki dampak minimal pada aplikasi dan data Anda. Topik pada bagian ini membahas pendekatan yang dapat Anda ambil untuk melindungi aplikasi dan data Anda dari kegagalan.

Topik

Mitigasi Kegagalan saat Menjalankan Memcached

Saat menjalankan mesin Memcached, Anda memiliki opsi berikut untuk memperkecil dampak kegagalan. Terdapat dua jenis kegagalan yang harus diatasi dalam rencana mitigasi kegagalan Anda: kegagalan simpul dan kegagalan Zona Ketersediaan.

Mitigasi Kegagalan Simpul

Cache nirserver secara otomatis memitigasi kegagalan simpul dengan arsitektur Multi-AZ yang direplikasi sehingga kegagalan simpul terlihat jelas untuk aplikasi Anda. Untuk mengurangi dampak kegagalan simpul di klaster yang dirancang sendiri, sebarkan data cache Anda ke lebih banyak simpul. Karena klaster yang dirancang sendiri tidak mendukung replikasi, kegagalan simpul akan selalu mengakibatkan hilangnya beberapa data dari klaster Anda.

Saat Anda membuat cluster Memcached, Anda dapat membuatnya dengan 1 hingga 60 node, atau lebih dengan permintaan khusus. Melakukan partisi data pada sejumlah besar simpul berarti lebih sedikit data yang hilang jika ada simpul yang gagal. Misalnya jika Anda membuat partisi data Anda pada 10 simpul, maka setiap simpul akan menyimpan sekitar 10% dari data cache Anda. Dalam hal ini, kegagalan simpul akan menghilangkan sekitar 10% dari cache Anda yang perlu diganti saat simpul pengganti dibuat dan disediakan. Jika data yang sama disimpan pada cache di 3 simpul yang besar, maka kegagalan satu simpul akan menghilangkan sekitar 33% dari data cache Anda.

Untuk informasi tentang menentukan jumlah simpul dalam sebuah klaster Memcached, lihat Membuat klaster Memcached (konsol).

Mitigasi Kegagalan Zona Ketersediaan

Cache nirserver secara otomatis memitigasi kegagalan simpul dengan arsitektur Multi-AZ yang direplikasi sehingga kegagalan AZ terlihat jelas untuk aplikasi Anda.

Untuk mengurangi dampak kegagalan Zona Ketersediaan di klaster yang dirancang sendiri, tempatkan simpul Anda di sebanyak mungkin Zona Ketersediaan. Jika terjadi kegagalan AZ, Anda akan kehilangan data yang di-cache di AZ itu, bukan data yang di-cache di yang lain. AZs

Mengapa perlu banyak simpul?

Jika wilayah saya hanya memiliki 3 Zona Ketersediaan, mengapa saya memerlukan lebih dari 3 simpul karena jika satu AZ gagal, maka saya akan kehilangan sekitar sepertiga data saya?

Ini adalah pertanyaan yang sangat bagus. Ingat bahwa kita mencoba mengurangi dua jenis kegagalan yang berbeda, yaitu kegagalan simpul dan Zona Ketersediaan. Anda benar, jika data Anda tersebar di beberapa Zona Ketersediaan dan salah satu zona gagal, maka Anda akan kehilangan hanya data cache di AZ itu, terlepas dari jumlah simpul yang Anda miliki. Namun, jika satu simpul gagal, memiliki lebih banyak simpul akan mengurangi proporsi data yang hilang.

Tidak ada "rumus ajaib" untuk menentukan berapa banyak simpul diperlukan untuk klaster Anda. Anda harus menimbang dampak kehilangan data vs kemungkinan kegagalan vs biaya, dan mendapatkan kesimpulan Anda sendiri.

Untuk informasi tentang menentukan jumlah simpul dalam sebuah klaster Memcached, lihat Membuat klaster Memcached (konsol).

Untuk informasi lain tentang wilayah dan Availability Zone, lihat Memilih wilayah dan zona ketersediaan untuk ElastiCache.

Mengurangi Kegagalan saat Menjalankan Valkey atau Redis OSS

Saat menjalankan mesin Valkey atau Redis OSS, Anda memiliki opsi berikut untuk meminimalkan dampak kegagalan node atau Availability Zone.

Mitigasi Kegagalan Simpul

Cache nirserver secara otomatis memitigasi kegagalan simpul dengan arsitektur Multi-AZ sehingga kegagalan simpul terlihat jelas untuk aplikasi Anda. Klaster yang dirancang sendiri harus dikonfigurasi dengan tepat untuk mengurangi kegagalan simpul individual.

Untuk mengurangi dampak kegagalan node Valkey atau Redis OSS pada cluster yang dirancang sendiri, Anda memiliki opsi berikut:

Topik

Mengurangi Kegagalan: Grup Replikasi Valkey atau Redis OSS

Grup replikasi Valkey atau Redis OSS terdiri dari satu node utama yang aplikasi Anda dapat membaca dari dan menulis ke, dan dari 1 hingga 5 node replika read-only. Setiap kali data ditulis ke simpul primer, data ini juga diperbarui secara asinkron ke simpul replika baca.

Saat salah satu replika baca gagal

1. ElastiCache mendeteksi replika baca yang gagal.
2. ElastiCache mengambil node yang gagal off line.
3. ElastiCache meluncurkan dan menyediakan node pengganti di AZ yang sama.
4. Simpul baru melakukan sinkronisasi dengan simpul primer.

Selama proses ini, aplikasi Anda dapat terus melakukan pembacaan dan penulisan menggunakan simpul lain.

Valkey atau Redis OSS Multi-AZ

Anda dapat mengaktifkan Multi-AZ pada grup replikasi Valkey atau Redis OSS Anda. Terlepas dari apakah Anda memiliki Multi-AZ yang aktif atau tidak, primer yang gagal akan terdeteksi dan diganti secara otomatis. Namun prosesnya akan berbeda tergantung apakah Multi-AZ aktif atau tidak.

Jika Multi-AZ aktif

1. ElastiCache mendeteksi kegagalan node primer.
2. ElastiCache mempromosikan simpul replika baca dengan lag replikasi paling sedikit ke simpul utama.
3. Replika lain melakukan sinkronisasi dengan simpul primer yang baru.
4. ElastiCache memutar replika baca di AZ primer yang gagal.
5. Simpul baru disinkronkan dengan primer yang baru dipromosikan.

Melakukan failover ke simpul replika umumnya lebih cepat daripada membuat dan menyediakan simpul primer baru. Ini berarti aplikasi Anda dapat melanjutkan kembali proses penulisan ke simpul primer Anda lebih cepat daripada ketika Multi-AZ tidak diaktifkan.

Untuk informasi selengkapnya, lihat Meminimalkan downtime ElastiCache dengan menggunakan Multi-AZ dengan Valkey dan Redis OSS.

Jika Multi-AZ tidak aktif

1. ElastiCache mendeteksi kegagalan primer.
2. ElastiCache mengambil offline utama.
3. ElastiCache membuat dan menyediakan simpul utama baru untuk menggantikan primer yang gagal.
4. ElastiCache menyinkronkan primer baru dengan salah satu replika yang ada.
5. Setelah sinkronisasi selesai, simpul baru berfungsi sebagai simpul primer klaster.

Selama langkah 1 sampai 4 proses ini, aplikasi Anda tidak dapat menulis ke simpul primer. Namun, aplikasi Anda dapat terus membaca dari simpul replika Anda.

Untuk perlindungan tambahan, sebaiknya Anda meluncurkan node di grup replikasi Anda di Availability Zones (AZs) yang berbeda. Jika Anda melakukan ini, kegagalan AZ hanya akan berdampak pada simpul di AZ tersebut dan bukan yang lain.

Untuk informasi selengkapnya, lihat Ketersediaan tinggi menggunakan grup replikasi.

Mitigasi Kegagalan Zona Ketersediaan

Cache nirserver secara otomatis memitigasi kegagalan simpul dengan arsitektur Multi-AZ yang direplikasi sehingga kegagalan AZ terlihat jelas untuk aplikasi Anda.

Untuk mengurangi dampak kegagalan Zona Ketersediaan di klaster yang dirancang sendiri, tempatkan simpul untuk setiap serpihan di sebanyak mungkin Zona Ketersediaan.

Sebanyak apa pun simpul yang Anda miliki di sebuah serpihan, jika semua simpul terletak di Zona Ketersediaan yang sama, kegagalan fatal pada AZ tersebut akan membuat semua data serpihan Anda hilang. Namun, jika Anda menemukan node Anda dalam beberapa AZs, kegagalan AZ apa pun mengakibatkan Anda hanya kehilangan node di AZ tersebut.

Setiap kali kehilangan simpul, Anda dapat mengalami penurunan performa karena operasi baca menjadi terbagi ke simpul yang lebih sedikit. Penurunan performa ini akan berlanjut hingga simpul tersebut diganti.

Untuk informasi tentang menentukan Availability Zones untuk Valkey atau Redis OSS node, lihat. Membuat cluster Valkey (mode cluster dinonaktifkan) (Konsol)

Untuk informasi selengkapnya tentang wilayah dan Zona Ketersediaan, lihat Memilih wilayah dan zona ketersediaan untuk ElastiCache.

Rekomendasi

Sebaiknya buat cache nirserver, bukan klaster yang dirancang sendiri, karena Anda secara otomatis mendapatkan toleransi kesalahan yang lebih baik tanpa konfigurasi tambahan. Saat membuat klaster yang dirancang sendiri, ada dua jenis kegagalan yang perlu direncanakan: kegagalan simpul individual dan kegagalan Zona Ketersediaan yang luas. Rencana mitigasi kegagalan terbaik akan mengatasi kedua jenis kegagalan tersebut.

Meminimalkan Dampak Kegagalan Simpul

Untuk meminimalkan dampak kegagalan node saat menggunakan Valkey atau Redis OSS, sebaiknya implementasi Anda menggunakan beberapa node di setiap shard dan mendistribusikan node di beberapa Availability Zone. Ini dilakukan secara otomatis untuk cache nirserver.

Untuk cluster yang dirancang sendiri di Valkey atau Redis OSS, kami menyarankan Anda mengaktifkan Multi-AZ pada grup replikasi Anda sehingga secara otomatis ElastiCache akan gagal ke replika jika node utama gagal.

Saat menjalankan Memcached dan membuat partisi data Anda di seluruh simpul, semakin banyak simpul yang Anda gunakan maka semakin kecil data yang hilang jika ada satu simpul yang gagal.

Meminimalkan Dampak Kegagalan Zona Ketersediaan

Untuk meminimalkan dampak kegagalan Zona Ketersediaan, sebaiknya Anda meluncurkan simpul Anda di sebanyak mungkin Zona Ketersediaan yang tersedia. Menyebarkan node Anda secara merata AZs akan meminimalkan dampak jika terjadi kegagalan AZ. Ini dilakukan secara otomatis untuk cache nirserver.

Tindakan pencegahan lain

Jika Anda menjalankan Valkey atau Redis OSS, maka selain hal di atas, kami sarankan Anda menjadwalkan pencadangan reguler cluster Anda. Cadangan (snapshot) membuat file .rdb yang

dapat Anda gunakan untuk memulihkan cache Anda jika terjadi kegagalan atau kerusakan. Lihat informasi yang lebih lengkap di Melakukan snapshot dan pemulihan.

Keamanan infrastruktur di AWS ElastiCache

Sebagai layanan terkelola, AWS ElastiCache dilindungi oleh prosedur keamanan jaringan AWS global yang dijelaskan di bagian Keamanan dan Kepatuhan di Pusat AWS Arsitektur.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses ElastiCache melalui jaringan. Klien harus mendukung Keamanan Lapisan Pengangkutan (TLS) 1.2 atau versi yang lebih baru. Kami merekomendasikan TLS 1.3 atau versi yang lebih baru. Klien juga harus mendukung cipher suite dengan perfect forward secrecy (PFS) seperti Ephemeral Diffie-Hellman (DHE) atau Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Sebagian besar sistem modern seperti Java 7 dan sistem yang lebih baru mendukung mode ini.

Selain itu, permintaan harus ditandatangani menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan prinsipal IAM. Atau Anda bisa menggunakan AWS Security Token Service (AWS STS) untuk membuat kredensial keamanan sementara guna menandatangani permintaan.

Pembaruan layanan di ElastiCache

ElastiCache secara otomatis memonitor armada cache, cluster, dan node Anda untuk menerapkan pembaruan layanan saat tersedia. Pembaruan layanan untuk cache nirserver diterapkan secara otomatis dan transparan. Untuk cluster yang dirancang sendiri, Anda menyiapkan jendela pemeliharaan yang telah ditentukan sehingga ElastiCache dapat menerapkan pembaruan ini. Namun, dalam beberapa kasus Anda mungkin menganggap cara ini terlalu kaku dan cenderung menghambat alur bisnis Anda.

Dengan pembaruan layanan, Anda mengontrol waktu dan jenis pembaruan yang diterapkan pada klaster yang dirancang sendiri. Anda juga dapat memantau kemajuan pembaruan ini ke ElastiCache cluster yang Anda pilih secara real time.

Topik

- Mengelola pembaruan layanan untuk cluster yang dirancang sendiri

Mengelola pembaruan layanan untuk cluster yang dirancang sendiri

ElastiCache pembaruan layanan untuk cluster yang dirancang sendiri dirilis secara teratur. Jika Anda memiliki satu atau beberapa cluster yang dirancang sendiri yang memenuhi syarat untuk pembaruan layanan tersebut, Anda menerima pemberitahuan melalui email, SNS, Personal Health Dashboard (PHD), dan CloudWatch acara Amazon saat pembaruan dirilis. Pembaruan juga ditampilkan di halaman Pembaruan Layanan di ElastiCache konsol. Dengan menggunakan dasbor ini, Anda dapat melihat semua pembaruan layanan dan statusnya untuk ElastiCache armada Anda. Pembaruan layanan untuk cache nirserver diterapkan secara transparan dan tidak dapat dikelola melalui Pembaruan Layanan.

Anda mengontrol waktu penerapan pembaruan sebelum pembaruan otomatis dimulai. Kami sangat menyarankan agar Anda menerapkan pembaruan jenis pembaruan keamanan sesegera mungkin untuk memastikan bahwa ElastiCache cluster Anda selalu up-to-date dengan patch keamanan saat ini.

Bagian berikut membahas opsi-opsi tersebut secara terperinci.

Menerapkan pembaruan layanan

Anda dapat memulai menerapkan pembaruan layanan untuk armada Anda sejak pembaruan berstatus tersedia. Pembaruan layanan bersifat kumulatif. Dengan kata lain, pembaruan apa pun yang belum diterapkan akan disertakan dalam pembaruan terbaru Anda.

Jika pembaruan layanan mengaktifkan pembaruan otomatis, Anda dapat memilih untuk mencatat tindakan apa pun saat tersedia. ElastiCache akan menjadwalkan untuk menerapkan pembaruan selama salah satu jendela pemeliharaan klaster Anda yang akan datang setelah tanggal mulai pembaruan Otomatis. Anda akan menerima notifikasi terkait untuk setiap tahap pembaruan.

> **Note**
>
> Anda dapat menerapkan hanya pembaruan layanan yang berstatus tersedia atau terjadwal.

Untuk informasi selengkapnya tentang meninjau dan menerapkan pembaruan khusus layanan apa pun ke ElastiCache kluster yang berlaku, lihat. Menerapkan pembaruan layanan menggunakan konsol

Jika pembaruan layanan baru tersedia untuk satu atau beberapa ElastiCache cluster, Anda dapat menggunakan ElastiCache konsol, API, atau AWS CLI untuk menerapkan pembaruan. Bagian berikut menjelaskan opsi yang dapat Anda gunakan untuk menerapkan pembaruan.

Menerapkan pembaruan layanan menggunakan konsol

Untuk melihat daftar pembaruan layanan yang tersedia, bersama informasi lainnya, buka halaman Pembaruan Layanan pada konsol.

1. Masuk ke AWS Management Console dan buka ElastiCache konsol Amazon di https://console.aws.amazon.com/elasticache/.
2. Pada panel navigasi, pilih Pembaruan Layanan.
3. Di bagian Pembaruan layanan, Anda dapat melihat hal berikut:

   - Nama pembaruan layanan: Nama unik pembaruan layanan
   - Jenis pembaruan: Jenis pembaruan layanan, yaitu pembaruan keamanan atau pembaruan mesin
   - Kepelikan pembaruan: Prioritas penerapan pembaruan:
     - kritis: Sebaiknya Anda menerapkan pembaruan ini segera (dalam waktu 14 hari atau kurang).
     - penting: Sebaiknya Anda menerapkan pembaruan ini secepatnya saat alur bisnis Anda memungkinkan (dalam 30 hari atau kurang).
     - sedang: Sebaiknya Anda menerapkan pembaruan ini secepatnya saat Anda bisa (dalam 60 hari atau kurang).
     - rendah: Sebaiknya Anda menerapkan pembaruan ini secepatnya saat Anda bisa (dalam 90 hari atau kurang).
   - Versi mesin: Jika jenis pembaruan adalah pembaruan mesin, yang diperbarui adalah versi mesin.
   - Tanggal Rilis: Waktu saat pembaruan dirilis dan tersedia untuk diterapkan pada klaster Anda.
   - Rekomendari Penerapan Sebelum Tanggal: ElastiCache tanggal pedoman untuk menerapkan pembaruan.
   - Status: Status pembaruan, yang merupakan salah satu dari berikut ini:
     - tersedia: Pembaruan tersedia untuk klaster yang diperlukan.
     - selesai: Pembaruan telah diterapkan.
     - dibatalkan: Pembaruan telah dibatalkan dan tidak diperlukan lagi.

- kedaluwarsa: Pembaruan tidak tersedia lagi untuk diterapkan.

4. Pilih pembaruan individual (bukan tombol di sebelah kirinya) untuk melihat detail pembaruan layanan.

   Di bagian Status pembaruan klaster, Anda dapat melihat daftar klaster yang berisi pembaruan yang belum diterapkan atau baru saja diterapkan. Untuk setiap klaster, Anda dapat melihat hal berikut:

   - Nama klaster: Nama dari klaster
   - Simpul diperbarui: Rasio simpul individual dalam klaster tertentu yang telah diperbarui atau tetap tersedia setelah pembaruan layanan tertentu.
   - Jenis Pembaruan: Jenis pembaruan layanan, yaitu pembaruan keamanan atau pembaruan mesin
   - Status: Status pembaruan layanan pada klaster, yang merupakan salah satu dari berikut ini:
     - tersedia: Pembaruan ini tersedia untuk klaster yang diperlukan.
     - sedang berlangsung: Pembaruan sedang diterapkan ke klaster ini.
     - dijadwalkan: Tanggal pembaruan telah dijadwalkan.
     - selesai: Pembaruan telah berhasil diterapkan. Klaster dengan status selesai akan ditampilkan selama 7 hari setelah selesai.

   Jika Anda memilih salah satu atau semua klaster dengan status tersedia atau dijadwalkan, lalu memilih Terapkan sekarang, pembaruan akan mulai diterapkan pada klaster tersebut.

Menerapkan pembaruan layanan menggunakan AWS CLI

Setelah Anda menerima notifikasi bahwa pembaruan layanan telah tersedia, Anda dapat memeriksa dan menerapkannya menggunakan AWS CLI:

- Untuk mendapatkan deskripsi pembaruan layanan yang tersedia, jalankan perintah berikut:

  ```
  aws elasticache describe-service-updates --service-update-status
  available
  ```

  Untuk informasi selengkapnya, lihat describe-service-updates.

- Untuk menerapkan pembaruan layanan pada daftar klaster, jalankan perintah berikut:

```
aws elasticache batch-apply-update-action --service-update
ServiceUpdateNameToApply=sample-service-update --cluster-names cluster-1
cluster2
```

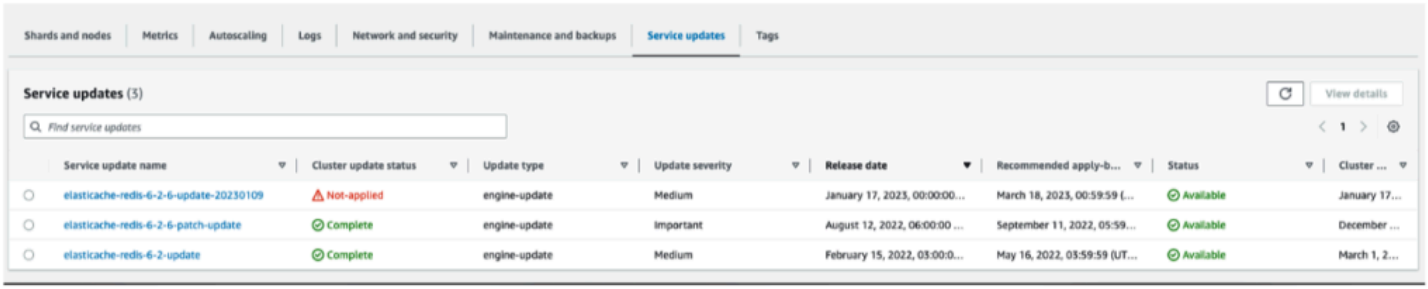
Untuk informasi selengkapnya, lihat batch-apply-update-action.

Memverifikasi bahwa Anda memiliki Pembaruan Layanan terbaru yang Diterapkan menggunakan AWS konsol

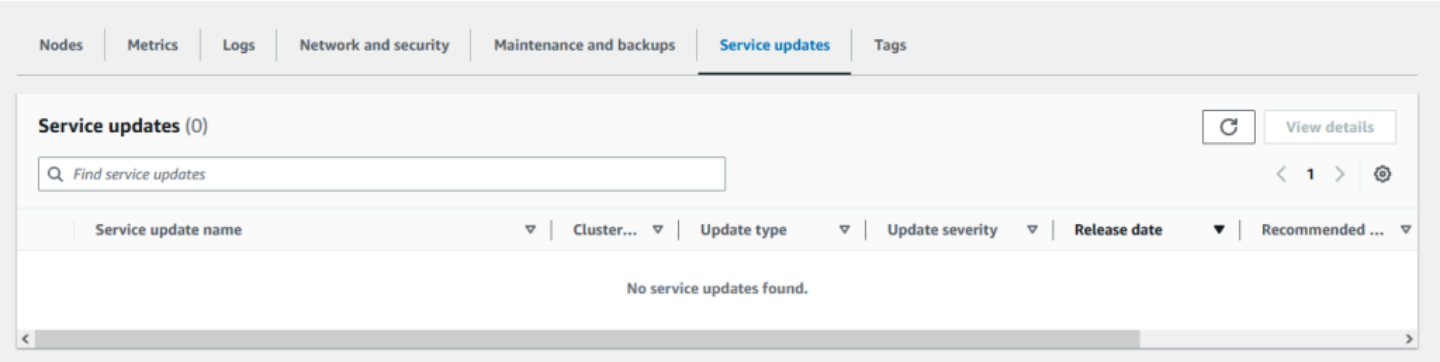
Anda dapat ElastiCache memverifikasi klaster Redis OSS Anda menjalankan pembaruan layanan terbaru dengan mengikuti langkah-langkah berikut:

1. Pilih klaster yang berlaku di halaman Redis OSS Clusters
2. Pilih Pembaruan layanan di panel navigasi untuk melihat pembaruan layanan yang berlaku untuk klaster tersebut, jika ada.

Jika konsol menampilkan daftar pembaruan layanan, Anda dapat memilih pembaruan layanan dan memilih Terapkan sekarang.



Jika konsol menampilkan “Tidak ada pembaruan layanan yang ditemukan”, itu berarti klaster ElastiCache untuk Redis OSS sudah memiliki pembaruan layanan terbaru yang diterapkan.



Menghentikan pembaruan layanan

Anda dapat menghentikan pembaruan pada klaster jika diperlukan. Misalnya, Anda mungkin ingin menghentikan pembaruan jika Anda mengalami lonjakan tak terduga pada klaster yang sedang diperbarui. Atau Anda sebaiknya menghentikan pembaruan yang memakan waktu terlalu lama dan mengganggu alur bisnis Anda pada jam sibuk.

Operasi Menghentikan akan segera menghentikan semua pembaruan pada klaster dan setiap simpul yang belum diperbarui. Operasi untuk simpul yang berstatus sedang berlangsung akan terus dilanjutkan hingga selesai. Namun, pembaruan akan dihentikan pada simpul lain di klaster yang sama yang berstatus pembaruan tersedia dan mengubah statusnya ke Menghentikan.

Saat alur kerja Menghentikan selesai, simpul yang berstatus Menghentikan akan berubah menjadi Dihentikan. Tergantung pada alur kerja pembaruan, simpul pada beberapa klaster mungkin tidak akan diperbarui sama sekali. Klaster lain mungkin akan menyertakan beberapa simpul yang sudah diperbarui dan simpul lain yang masih berstatus pembaruan tersedia.

Anda dapat kembali nanti untuk menyelesaikan proses pembaruan ketika alur bisnis Anda memungkinkan. Dalam kasus ini, pilih klaster yang sesuai yang ingin diselesaikan pembaruannya, lalu pilih Terapkan Sekarang. Untuk informasi selengkapnya, lihat Menerapkan pembaruan layanan.

Menggunakan konsol

Anda dapat mengganggu pembaruan layanan menggunakan ElastiCache konsol. Contoh berikut menunjukkan cara melakukannya:

- Setelah pembaruan layanan berlangsung pada kluster yang dipilih, ElastiCache konsol menampilkan tab Lihat/Berhenti Pembaruan di bagian atas dasbor. ElastiCache
- Untuk menghentikan pembaruan, pilih Hentikan Pembaruan.
- Saat Anda menghentikan pembaruan, pilih klaster dan periksa statusnya. Status klaster akan berubah menjadi Menghentikan dan pada akhirnya berstatus Dihentikan.

Menggunakan AWS CLI

Anda dapat menghentikan pembaruan layanan menggunakan AWS CLI. Contoh kode berikut ini menunjukkan cara untuk melakukannya.

Untuk grup replikasi, lakukan hal berikut:

```
aws elasticache batch-stop-update-action --service-update-name sample-service-update --replication-group-ids my-replication-group-1 my-replication-group-2
```

Untuk klaster cache, lakukan hal berikut:

```
aws elasticache batch-stop-update-action --service-update-name sample-service-update --cache-cluster-ids my-cache-cluster-1 my-cache-cluster-2
```

Lihat informasi yang lebih lengkap di BatchStopUpdateAction.

Kerentanan dan Eksposur Umum (CVE): Kerentanan keamanan yang ditangani di ElastiCache

Kerentanan dan Eksposur Umum (CVE) adalah daftar entri untuk kerentanan keamanan siber yang diketahui publik. Setiap entri adalah tautan yang berisi nomor identifikasi, deskripsi, dan setidaknya satu referensi publik. Anda dapat menemukan di halaman ini daftar kerentanan keamanan yang telah ditangani. ElastiCache

Kami menyarankan Anda untuk selalu meningkatkan ke versi ElastiCache Valkey, Redis OSS atau ElastiCache Memcached terbaru agar terlindungi dari kerentanan yang diketahui. Saat mengoperasikan Cache ElastiCache Tanpa Server, perbaikan CVE secara otomatis diterapkan ke cache Anda. Saat mengoperasikan cluster yang dirancang sendiri dengan Valkey atau Redis OSS, ElastiCache paparkan komponen PATCH. Misalnya, ketika menggunakan ElastiCache untuk Redis OSS versi 6.2.6, versi utama adalah 6, versi minor adalah 2, dan versi patch adalah 6. Versi PATCH adalah untuk perbaikan bug yang kompatibel ke belakang, perbaikan keamanan, dan perubahan non-fungsional.

Anda dapat menggunakan tabel berikut untuk memverifikasi apakah versi tertentu dari ElastiCache Valkey dan Redis OSS memiliki perbaikan untuk kerentanan keamanan tertentu. Jika klaster ElastiCache Valkey atau Redis OSS Anda menjalankan versi tanpa perbaikan keamanan, lihat tabel di bawah ini dan ambil tindakan. Anda dapat meningkatkan ke versi ElastiCache Valkey atau Redis OSS yang lebih baru yang berisi perbaikan, atau jika Anda menggunakan versi yang berisi perbaikan, pastikan Anda memiliki pembaruan layanan terbaru yang diterapkan dengan merujuk ke. Mengelola pembaruan layanan untuk cluster yang dirancang sendiri Untuk informasi selengkapnya tentang versi ElastiCache mesin yang didukung dan cara memutakhirkan, lihatVersi mesin dan peningkatan di ElastiCache.

Note

- Jika CVE ditangani dalam ElastiCache versi, itu berarti CVE juga dibahas dalam versi yang lebih baru. Jadi misalnya jika kerentanan ditangani ElastiCache untuk Redis OSS Versi 6.0.5, ini terus berlanjut untuk Versi 6.2.6, 7.0.7, dan 7.1.
- Tanda bintang (*) dalam tabel berikut menunjukkan bahwa Anda harus memiliki pembaruan layanan terbaru yang diterapkan ElastiCache untuk Redis OSS Cluster yang menjalankan versi ElastiCache untuk Redis OSS yang ditentukan untuk mengatasi kerentanan keamanan. Untuk informasi selengkapnya tentang cara memverifikasi bahwa Anda telah menerapkan pembaruan layanan terbaru untuk versi Redis OSS ElastiCache untuk Redis OSS yang dijalankan klaster Anda, lihat. Mengelola pembaruan layanan untuk cluster yang dirancang sendiri

ElastiCache untuk versi Redis OSS	CVEs Dialamatkan
Redis OSS 6.0.5	CVE-2022-24735 *, CVE-2022-24736 *
Redis OSS 6.2.6	CVE-2022-24834 *, CVE-2022-35977 *, CVE-2022-36021 *, CVE-2022-24735, CVE-2022-24736, CVE-2023-22458, CVE-2023-25155 CVE-2023-28856 , CVE-2023-45145
Redis OSS 7.0.7	CVE-2023-41056 *, CVE-2022-24834* , CVE-2022-35977, CVE-2022-36021, CVE-2022-24735, CVE-2022-24736
Redis OSS 7.1.0	CVE-2023-41056, CVE-2022-24834, CVE-2022-35977, CVE-2022-36021, CVE-2022-24735 , CVE-2022-24736

Pencatatan dan pemantauan di Amazon ElastiCache

Untuk mengelola cache Anda, penting bagi Anda untuk mengetahui kinerja cache Anda. ElastiCache menghasilkan metrik yang dipublikasikan ke Amazon CloudWatch Logs untuk memantau kinerja cache Anda. Selain itu, ElastiCache menghasilkan peristiwa ketika perubahan signifikan terjadi pada sumber daya cache Anda (misalnya cache baru dibuat, atau cache dihapus).

Topik

- Mencatat panggilan ElastiCache API Amazon dengan AWS CloudTrail

Metrik dan acara tanpa server untuk Valkey dan Redis OSS

ElastiCache menawarkan berbagai metrik dan peristiwa untuk pemantauan saat bekerja dengan cache tanpa server. Ini termasuk CloudWatch metrik, metrik tingkat perintah, dan log peristiwa yang dapat dicerna melalui Amazon. EventBridge

Topik

Metrik cache nirserver

`AWS/ElastiCache`Namespace menyertakan CloudWatch metrik berikut untuk cache tanpa server Valkey atau Redis OSS Anda.

Kode metrik untuk Valkey atau Redis OSS

Metrik	Deskripsi	Unit
BytesUsedForCache	Jumlah total byte yang digunakan oleh data yang disimpan dalam cache Anda.	Byte
ElastiCacheProcessingUnits	Jumlah total ElastiCacheProcessingUnits (ECPUs) yang dikonsumsi oleh permintaan yang dijalankan di cache Anda	Hitungan
SuccessfulReadRequestLatency	Latensi permintaan baca yang berhasil.	Mikrodetik
SuccessfulWriteRequestLatency	Latensi permintaan tulis yang berhasil	Mikrodetik
TotalCmdsCount	Jumlah total semua perintah yang dijalankan pada cache Anda	Hitungan
CacheHitRate	Menunjukkan laju keberhasilan cache Anda. Ini dihitung menggunakan statistik `cache_hits` dan `cache_misses` dengan cara berikut: `cache_hits /(cache_hits + cache_misses)` .	Persen
CacheHits	Jumlah pencarian kunci hanya-baca yang berhasil di cache.	Hitungan
CurrConnections	Jumlah koneksi klien ke cache Anda.	Hitungan

Metrik	Deskripsi	Unit
ThrottledCmds	Jumlah permintaan yang dibatasi oleh ElastiCache karena beban kerja menskalakan lebih cepat daripada yang dapat diskalakan. ElastiCache	Hitungan
NewConnections	Jumlah seluruh koneksi yang telah diterima oleh server selama periode ini.	Hitungan
CurrItems	Jumlah item dalam cache.	Hitungan
CurrVolatileItems	Jumlah item dalam cache dengan TTL.	Hitungan
NetworkBytesIn	Total byte yang ditransfer ke cache	Byte
NetworkBytesOut	Total byte yang ditransfer ke luar dari cache	Byte
Evictions	Hitungan kunci yang dikosongkan oleh cache	Hitungan
IamAuthenticationExpirations	Jumlah total koneksi Valkey atau Redis OSS yang diautentikasi IAM yang kedaluwarsa. Anda dapat menemukan informasi selengkapnya tentang Autentikasi dengan IAM dalam panduan pengguna.	Hitungan

Metrik	Deskripsi	Unit
IamAuthenticationThrottling	Jumlah total permintaan Valkey atau Redis OSS AUTH atau HELLO yang diautenti kasi IAM yang dibatasi. Anda dapat menemukan informasi selengkapnya tentang Autentikasi dengan IAM dalam panduan pengguna.	Hitungan
KeyAuthorizationFailures	Jumlah seluruh percobaan pengguna yang gagal untuk mengakses kunci akses karena tidak mempunyai izin akses. Sebaiknya atur peringatan untuk hal ini guna mendeteksi percobaan akses yang tidak sah.	Hitungan
AuthenticationFailures	Jumlah total upaya yang gagal untuk mengautentikasi ke Valkey atau Redis OSS menggunakan perintah AUTH. Sebaiknya atur peringatan untuk hal ini guna mendeteks i percobaan akses yang tidak sah.	Hitungan

Metrik	Deskripsi	Unit
CommandAuthorizationFailures	Jumlah seluruh percobaan pengguna yang gagal untuk menjalankan perintah karena tidak memiliki izin untuk memanggil perintah itu. Sebaiknya atur peringatan untuk hal ini guna mendeteksi percobaan akses yang tidak sah.	Hitungan

Metrik tingkat perintah

ElastiCache juga memancarkan metrik tingkat perintah berikut. Untuk setiap jenis perintah, ElastiCache memancarkan jumlah total perintah dan jumlah yang ECPUs dikonsumsi oleh jenis perintah tersebut.

Metrik	Deskripsi	Unit
EvalBasedCmds	Jumlah perintah get yang telah diterima oleh cache.	Hitungan
EvalBasedCmdsECPUs	ECPUs dikonsumsi oleh perintah berbasis evaluasi.	Hitungan
GeoSpatialBasedCmds	Jumlah seluruh perintah untuk perintah berbasis geospasial. Ini berasal dari statistik Commandstats Valkey atau Redis OSS. Ini berasal dengan menjumlahkan semua perintah jenis geo: geoadd, geodist, geohash, geopos, georadius, dan georadius bymember.	Hitungan

Metrik	Deskripsi	Unit
GeoSpatialBasedCmdsECPUs	ECPUs dikonsumsi oleh perintah berbasis geospasial.	Hitungan
GetTypeCmds	Jumlah seluruh perintah jenis hanya-baca. Ini berasal dari statistik commandstats Valkey atau Redis OSS dengan menjumlahkan semua perintah tipe read-only (get, hget, scard, lrange, dan sebagainy a.)	Hitungan
GetTypeCmdsECPUs	ECPUs dikonsumsi oleh perintah baca.	Hitungan
HashBasedCmds	Jumlah seluruh perintah yang berbasis hash. Ini berasal dari statistik commandstats Valkey atau Redis OSS dengan menjumlahkan semua perintah yang bertindak atas satu atau lebih hash (hget, hkeys, hvals, hdel, dan sebagainya).	Hitungan
HashBasedCmdsECPUs	ECPUs dikonsumsi oleh perintah berbasis hash.	Hitungan
HyperLogLogBasedCmds	Jumlah total perintah HyperLogLog berbasis. Ini berasal dari statistik commandstats Valkey atau Redis OSS dengan menjumlahkan semua jenis perintah pf (pfadd, pfcount, pfmerge, dan sebagainya.).	Hitungan

Metrik	Deskripsi	Unit
HyperLogLogBasedCmdsECPUs	ECPUs dikonsumsi oleh perintah HyperLogLog berbasis.	Hitungan
JsonBasedCmds	Jumlah total perintah JSON, termasuk perintah baca dan tulis. Ini berasal dari statistik commandstats Valkey atau Redis OSS dengan menjumlahkan semua perintah JSON yang bertindak atas kunci JSON.	Hitungan
JsonBasedCmdsECPUs	ECPUs dikonsumsi oleh semua perintah JSON, termasuk perintah baca dan tulis.	Hitungan
JsonBasedGetCmds	Jumlah seluruh perintah JSON hanya-baca. Ini berasal dari statistik commandstats Valkey atau Redis OSS dengan menjumlahkan semua perintah baca JSON yang bertindak atas kunci JSON.	Hitungan
JsonBasedGetCmdsECPUs	ECPUs dikonsumsi oleh perintah read-only JSON.	Hitungan
JsonBasedSetCmds	Jumlah seluruh perintah JSON jenis tulis. Ini berasal dari statistik commandstats Valkey atau Redis OSS dengan menjumlahkan semua perintah tulis JSON yang bertindak atas kunci JSON.	Hitungan

Metrik	Deskripsi	Unit
JsonBasedSetCmdsECPUs	ECPUs dikonsumsi oleh perintah tulis JSON.	Hitungan
KeyBasedCmds	Jumlah seluruh perintah yang berbasis kunci. Ini berasal dari statistik commandstats Valkey atau Redis OSS dengan menjumlahkan semua perintah yang bertindak atas satu atau lebih kunci di beberapa struktur data (del, expire, rename, dan sebagainya.).	Hitungan
KeyBasedCmdsECPUs	ECPUs dikonsumsi oleh perintah berbasis kunci.	Hitungan
ListBasedCmds	Jumlah seluruh perintah yang berbasis daftar. Ini berasal dari statistik commandst ats Valkey atau Redis OSS dengan menjumlahkan semua perintah yang bekerja pada satu atau lebih daftar (lindex, lrange, lpush, ltrim, dan sebagainya).	Hitungan
ListBasedCmdsECPUs	ECPUs dikonsumsi oleh perintah berbasis daftar.	Hitungan

Metrik	Deskripsi	Unit
NonKeyTypeCmds	Jumlah seluruh perintah yang tidak berbasis kunci. Ini berasal dari statistik commandstats Valkey atau Redis OSS dengan menjumlahkan semua perintah yang tidak bertindak atas kunci, misalnya, acl, dbsize atau info.	Hitungan
NonKeyTypeCmdsECPUs	ECPUs dikonsumsi oleh non-key-based perintah.	Hitungan
PubSubBasedCmds	Jumlah total perintah untuk pub/sub fungsionalitas. Ini berasal dari statistik statistik Valkey atau Redis OSS dengan menjumlahkan semua perintah yang digunakan untuk pub/sub fungsionalitas: psubscribe, publish, pubsub, punsubscribe, ssubscrib e, sunsubscribe, spublish, subscribe, dan unsubscribe.	Hitungan
PubSubBasedCmdsECPUs	ECPUs dikonsumsi oleh perintah pub/sub-based.	Hitungan

Metrik	Deskripsi	Unit
SetBasedCmds	Jumlah seluruh perintah yang berbasis set. Ini berasal dari statistik commandstats Valkey atau Redis OSS dengan menjumlahkan semua perintah yang bekerja pada satu atau lebih set (scard, sdiff, sadd, sunion, dan sebagainya).	Hitungan
SetBasedCmdsECPUs	ECPUs dikonsumsi oleh perintah berbasis set.	Hitungan
SetTypeCmds	Jumlah seluruh perintah jenis tulis. Ini berasal dari statistik commandstats Valkey atau Redis OSS dengan menjumlahkan semua jenis perintah mutatif yang beroperasi pada data (set, hset, sadd, lpop, dan sebagainya.)	Hitungan
SetTypeCmdsECPUs	ECPUs dikonsumsi oleh perintah tulis.	Hitungan
SortedSetBasedCmds	Jumlah seluruh perintah yang berbasis sorted set. Ini berasal dari statistik commandst ats Valkey atau Redis OSS dengan menjumlahkan semua perintah yang bertindak atas satu atau lebih set yang diurutkan (zcount, zrange, zrank, zadd, dan sebagainya).	Hitungan

Metrik	Deskripsi	Unit
SortedSetBasedCmdsECPUs	ECPUs dikonsumsi oleh perintah berbasis diurutkan.	Hitungan
StringBasedCmds	Jumlah seluruh perintah yang berbasis string. Ini berasal dari statistik Commandstats Valkey atau Redis OSS dengan menjumlahkan semua perintah yang bekerja pada satu atau lebih string (strlen, setex, setrange, dan sebagainya).	Hitungan
StringBasedCmdsECPUs	ECPUs dikonsumsi oleh perintah berbasis string.	Hitungan
StreamBasedCmds	Jumlah seluruh perintah yang berbasis aliran. Ini berasal dari statistik commandstats Valkey atau Redis OSS dengan menjumlahkan semua perintah yang bertindak atas satu atau lebih tipe data stream (xrange, xlen, xadd, xdel, dan sebagainya).	Hitungan
StreamBasedCmdsECPUs	ECPUs dikonsumsi oleh perintah berbasis streaming.	Hitungan

Peristiwa cache nirserver

ElastiCache mencatat peristiwa yang berhubungan dengan cache tanpa server Anda. Informasi ini mencakup tanggal dan waktu peristiwa, nama sumber dan jenis sumber peristiwa, serta deskripsi dari peristiwa tersebut. Anda dapat dengan mudah mengambil peristiwa dari log menggunakan ElastiCache konsol, AWS CLI perintah deskripsi-peristiwa, atau tindakan API. ElastiCache `DescribeEvents`

Anda dapat memilih untuk memantau, menelan, mengubah, dan menindaklanjuti ElastiCache acara menggunakan Amazon EventBridge. Pelajari lebih lanjut di panduan EventBridge memulai Amazon.

Melihat ElastiCache acara (Konsol)

Untuk melihat acara menggunakan ElastiCache konsol:

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/
2. Untuk melihat daftar semua peristiwa yang tersedia, pada panel navigasi, pilih Peristiwa.
3. Pada layar Peristiwa setiap baris dari daftar merepresentasikan satu peristiwa dan menampilkan sumber peristiwa, jenis peristiwa, waktu GMT peristiwa, serta deskripsi dari peristiwa itu. Dengan Filter, Anda dapat menentukan apakah Anda ingin melihat semua peristiwa, atau hanya peristiwa dari jenis tertentu dalam daftar peristiwa.

Melihat ElastiCache acara (AWS CLI)

Untuk menghasilkan daftar ElastiCache peristiwa menggunakan AWS CLI, gunakan perintah deskripsi-peristiwa. Anda dapat menggunakan parameter opsional untuk mengontrol jenis peristiwa yang tercantum, jangka waktu peristiwa yang dicantumkan, jumlah maksimum peristiwa yang akan dicantumkan, dan lainnya.

Kode berikut mencantumkan hingga 40 peristiwa cache nirserver.

```
aws elasticache describe-events --source-type serverless-cache --max-items 40
```

Kode berikut mencantumkan semua peristiwa untuk cache nirserver selama 24 jam terakhir (1440 menit).

```
aws elasticache describe-events --source-type serverless-cache --duration 1440
```

Peristiwa Nirserver

Bagian ini mendokumentasikan berbagai jenis peristiwa yang mungkin Anda terima untuk cache nirserver.

Peristiwa Pembuatan Cache Nirserver

Jenis-Detail	Deskripsi	Unit	Sumber	Pesan
Cache dibuat	Cache arn	pembuatan	cache-nirserver	Cache <cache-name> dibuat dan siap digunakan.
Cache dibuat	Cache arn Snapshot arn	pembuatan	cache-nirserver	Cache <cache-name> dibuat dan data dipulihkan dari snapshot. Cache Anda siap digunakan.
Pembuatan cache gagal	Cache arn	kegagalan	cache-nirserver	Gagal membuat cache <cache-name>. Alamat IP yang bebas tidak cukup untuk membuat titik akhir VPC.
Pembuatan cache gagal	Cache arn	kegagalan	cache-nirserver	Gagal membuat cache <cache-name>. Subnet tidak valid yang disediakan dalam permintaan.
Pembuatan cache gagal	Cache arn	kegagalan	cache-nirserver	Gagal membuat cache <cache-name>. Batas kuota tercapai untuk membuat titik akhir VPC.

Jenis-Detail	Deskripsi	Unit	Sumber	Pesan
Pembuatan cache gagal	Cache arn	kegagalan	cache-nirserver	Gagal membuat cache <cache-name>. Anda tidak memiliki izin untuk membuat titik akhir VPC.
Pembuatan cache gagal	Cache arn	kegagalan	cache-nirserver	Gagal membuat cache <cache-name>. Pengguna dengan versi Valkey atau Redis OSS yang tidak kompatibel hadir di grup pengguna < >. user-group-name
Pembuatan cache gagal	Cache arn Cache snapshot arn	kegagalan	cache-nirserver	Gagal membuat cache <cache-name>. Grup pengguna yang disediakan < user-group-name > tidak ada.

Jenis-Detail	Deskripsi	Unit	Sumber	Pesan
Pembuatan cache gagal	Cache arn	kegagalan	cache-nirserver	Gagal membuat cache <cache-name>. Pemulihan data dari snapshot gagal karena <reason>. Alasan kegagalan: • gagal mengambil file dari S3. • md5 yang diharapkan tidak cocok dengan md5 aktual. • file RDB yang disediakan memiliki versi yang tidak didukung.

Acara Pembaruan Cache Tanpa Server (Valkey atau Redis OSS)

Jenis-Detail	Daftar sumber daya	Kategori	Sumber	Pesan
Cache diperbarui	Cache arn	perubahan konfigurasi	cache-nirserver	SecurityGroups diperbarui untuk cache<cache-name>.

Jenis-Detail	Daftar sumber daya	Kategori	Sumber	Pesan
Cache diperbarui	Cache arn	perubahan konfigurasi	cache-nirserver	Tag diperbaru i untuk cache <cache-name>.
Pembaruan cache gagal	Cache arn	perubahan konfigurasi	cache-nirserver	Pembaruan cache <cache-name> gagal. Pengguna dengan versi Valkey atau Redis OSS yang tidak kompatibe l hadir di grup pengguna < >. user-group-name
Pembaruan cache gagal	Cache arn	perubahan konfigurasi	cache-nirserver	Pembaruan cache <cache-name>gagal. SecurityGroups pembaruan gagal.
Pembaruan cache gagal	Cache arn	perubahan konfigurasi	cache-nirserver	Pembaruan cache <cache-name>gagal. SecurityGroups pembaruan gagal karena izin yang tidak mencukupi.

Jenis-Detail	Daftar sumber daya	Kategori	Sumber	Pesan
Pembaruan cache gagal	Cache arn	perubahan konfigurasi	cache-nirserver	Pembaruan cache <cache-name>gagal. SecurityGroups pembaruan gagal karena SecurityGroups tidak valid.

Acara Penghapusan Cache Tanpa Server (Valkey atau Redis OSS)

Jenis-Detail	Daftar sumber daya	Kategori	Sumber	Pesan
Cache dihapus	Cache arn	penghapusan	cache-nirserver	Cache <cache-name> telah dihapus.

Acara Batas Penggunaan Cache Tanpa Server (Valkey atau Redis OSS)

Jenis-Detail	Deskripsi	Unit	Sumber	Pesan
Cache diperbarui	Cache arn	perubahan konfigurasi	cache-nirserver	Batas diperbarui untuk cache <cache-name>.
Batas cache semakin dekat	Cache arn	pemberitahuan	cache-nirserver	Slot <X> menggunakan lebih <Y> % dari batas per slot 32 GB. Misalnya, Slot 10 menggunakan

Jenis-Detail	Deskripsi	Unit	Sumber	Pesan
				lebih 90% dari batas per slot 32 GB.
Pembaruan cache gagal	Cache arn	kegagalan	cache-nirserver	Pembaruan batas ke cache <cache-name> gagal karena cache telah dihapus.
Pembaruan cache gagal	Cache arn	kegagalan	cache-nirserver	Pembaruan batas ke cache <cache-name> gagal karena konfigurasi tidak valid.
Pembaruan cache gagal	Cache arn	kegagalan	cache-nirserver	Pembaruan batas ke cache <cache-name> gagal karena data cache saat ini melebihi batas baru. Harap bersihkan beberapa data sebelum menerapkan batas.

Acara Snapshot Cache Tanpa Server (Valkey atau Redis OSS)

Jenis-Detail	Daftar-sumber daya	Kategori	Sumber	Pesan
Snapshot dibuat	Cache arn Snapshot arn	pembuatan	serverless-cache-snapshot	Snapshot <snapshot-name> dibuat untuk cache <cache-name>.
Pembuatan snapshot gagal	Cache arn Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal membuat snapshot untuk cache <cache-name>. Pembuatan snapshot <snapshot-name> gagal dengan Kunci yang Dikelola Pelanggan<key-id> <reason>. Pesan alasan kegagalan: • karena Kunci yang Dikelola Pelanggan dinonaktifkan • karena Kunci yang Dikelola Pelanggan tidak dapat ditemukan • karena waktu permintaan habis

Jenis-Detail	Daftar-sumber daya	Kategori	Sumber	Pesan
Pembuatan snapshot gagal	Cache arn Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal membuat snapshot untuk cache <cache-name>. Pembuatan snapshot <snapshot-name> gagal karena <reason>. Alasan default: • karena kesalahan internal
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspor snapshot untuk cache <cache-name>. Tidak dapat mengekspor snapshot ke bucket %s karena ElastiCache tidak memiliki izin ke bucket.

Jenis-Detail	Daftar-sumber daya	Kategori	Sumber	Pesan
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspor snapshot ke bucket '%s' karena sudah ada objek dengan nama yang sama di bucket.
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspor snapshot ke bucket '%s' karena Id akun pemilik bucket telah berubah.

Jenis-Detail	Daftar-sumber daya	Kategori	Sumber	Pesan
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspor snapshot ke bucket '%s' karena bucket S3 tidak dapat diakses.
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspor snapshot ke bucket '%s' karena bucket tidak dapat diakses.

Jenis-Detail	Daftar-sumber daya	Kategori	Sumber	Pesan
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspor snapshot ke bucket '%s' karena bucket tidak tersedia.
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspo r snapshot ke bucket '%s' dengan snapshot sumber Kunci Terkelola Pelanggan %s <reason>.

Jenis-Detail	Daftar-sumber daya	Kategori	Sumber	Pesan
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspor snapshot ke bucket '%s'.
Salinan snapshot gagal	Snapshot arn-1 Snapshot arn-2	kegagalan	serverless-cache-snapshot	Gagal menyalin snapshot <snapshot-name>. Tidak dapat menyalin snapshot '%s' ke snapshot '%s' dengan snapshot sumber Kunci Terkelola Pelanggan <key-id> <reason-name>.

Jenis-Detail	Daftar-sumber daya	Kategori	Sumber	Pesan
Salinan snapshot gagal	Snapshot arn-1 Snapshot arn-2	kegagalan	serverless-cache-snapshot	Gagal menyalin snapshot <snapshot-name>. Tidak dapat menyalin snapshot '%s' ke snapshot '%s' dengan snapshot target Kunci Terkelola Pelanggan '%s' '%s'.

Metrik dan acara cluster yang dirancang sendiri untuk Valkey dan Redis OSS

ElastiCache menawarkan berbagai metrik dan acara untuk memantau cluster yang dirancang sendiri saat bekerja dengan Valkey dan Redis OSS. Ini termasuk metrik tingkat host, metrik tingkat perintah, dan log peristiwa yang tersedia melalui Amazon Simple Notification Service (SNS) AWS CLI dan Amazon Simple Notification Service (SNS).

Topik

Metrik untuk klaster yang dirancang sendiri

Saat Anda mendesain sendiri cluster, ElastiCache memancarkan metrik di setiap level node, termasuk metrik tingkat host dan metrik cache.

Untuk informasi selengkapnya tentang metrik tingkat host, lihat Metrik Tingkat Host.

Untuk informasi selengkapnya tentang metrik tingkat simpul, lihat Metrik untuk Valkey dan Redis OSS.

Acara untuk cluster yang dirancang sendiri (Valkey dan Redis OSS)

ElastiCache mencatat peristiwa yang berhubungan dengan cache yang dirancang sendiri. Saat bekerja dengan cluster yang dirancang sendiri, Anda dapat melihat peristiwa klaster di ElastiCache konsol, menggunakan AWS CLI, atau menggunakan Amazon Simple Notification Service (SNS). Acara cluster yang dirancang sendiri tidak dipublikasikan ke Amazon EventBridge.

Informasi peristiwa klaster yang dirancang sendiri menyertakan tanggal dan waktu peristiwa, nama sumber dan jenis sumber peristiwa, serta deskripsi peristiwa. Anda dapat dengan mudah mengambil peristiwa dari log menggunakan ElastiCache konsol, AWS CLI perintah deskripsi-peristiwa, atau tindakan API. ElastiCache DescribeEvents

Melihat ElastiCache acara (Konsol)

Prosedur berikut menampilkan peristiwa menggunakan ElastiCache konsol.

Untuk melihat peristiwa menggunakan ElastiCache konsol

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/
2. Untuk melihat daftar semua peristiwa yang tersedia, pada panel navigasi, pilih Peristiwa.
3. Pada layar Peristiwa setiap baris dari daftar merepresentasikan satu peristiwa dan menampilkan sumber peristiwa, jenis peristiwa, waktu GMT peristiwa, serta deskripsi dari peristiwa itu. Dengan Filter, Anda dapat menentukan apakah Anda ingin melihat semua peristiwa, atau hanya peristiwa dari jenis tertentu dalam daftar peristiwa.

Melihat ElastiCache acara (AWS CLI)

Untuk menghasilkan daftar ElastiCache peristiwa menggunakan AWS CLI, gunakan perintah deskripsi-peristiwa. Anda dapat menggunakan parameter opsional untuk mengontrol jenis peristiwa yang tercantum, jangka waktu peristiwa yang dicantumkan, jumlah maksimum peristiwa yang akan dicantumkan, dan lainnya.

Kode berikut mencantumkan hingga 40 peristiwa klaster yang dirancang sendiri.

```
aws elasticache describe-events --source-type cache-cluster --max-items 40
```

Kode berikut mencantumkan semua peristiwa untuk cache yang dirancang sendiri selama 24 jam terakhir (1440 menit).

```
aws elasticache describe-events --source-type cache-cluster --duration 1440
```

Peristiwa klaster yang dirancang sendiri

Bagian ini berisi daftar peristiwa yang dapat Anda terima untuk klaster yang dirancang sendiri.

ElastiCache Peristiwa berikut memicu notifikasi Amazon SNS. Untuk informasi tentang detail peristiwa, lihat Melihat ElastiCache acara.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:AddCacheNodeComplete	`ElastiCache:AddCacheNodeComplete : cache-cluster`	Simpul cache telah ditambahkan ke klaster cache dan siap untuk digunakan.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache: AddCacheNodeFailed karena alamat IP gratis yang tidak mencukupi	`ElastiCache:AddCacheNodeFailed : cluster-name`	Simpul cache tidak dapat ditambahkan karena alamat IP yang tersedia tidak cukup.
ElastiCache:CacheClusterParametersChanged	`ElastiCache:CacheClusterParametersChanged : cluster-name`	Satu atau beberapa parameter klaster cache telah berubah.
ElastiCache:CacheClusterProvisioningComplete	`ElastiCache:CacheClusterProvisioningComplete cluster-name-0001-005`	Penyediaan klaster cache selesai, dan simpul cache dalam klaster cache siap untuk digunakan.
ElastiCache: CacheClusterProvisioningFailed karena status jaringan yang tidak kompatibel	`ElastiCache:CacheClusterProvisioningFailed : cluster-name`	Percobaan dilakukan untuk meluncurkan klaster cache baru ke cloud privat virtual (VPC) yang tidak ada.
ElastiCache:CacheClusterScalingComplete	`CacheClusterScalingComplete : cluster-name`	Penskalaan untuk klaster cache berhasil diselesaikan.
ElastiCache:CacheClusterScalingFailed	ElastiCache:CacheClusterScalingFailed : *cluster-name*	Operasi peningkatan skala pada klaster cache gagal.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CacheClusterSecurityGroupModified	`ElastiCache:CacheClusterSecurityGroupModified : cluster-name`	Salah satu peristiwa berikut telah terjadi: • Daftar grup keamanan cache yang diizinkan untuk klaster cache yang telah diubah. • Satu atau lebih grup EC2 keamanan baru telah diotorisasi pada salah satu grup keamanan cache yang terkait dengan cluster cache. • Satu atau lebih grup EC2 keamanan telah dicabut dari salah satu grup keamanan cache yang terkait dengan cluster cache.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CacheNodeReplaceStarted	`ElastiCache:CacheNodeReplaceStarted : cluster-name`	ElastiCache telah mendeteksi bahwa host yang menjalankan node cache terdegradasi atau tidak dapat dijangkau dan telah mulai mengganti node cache. **Note** Entri DNS untuk simpul cache yang diganti tidak berubah. Pada kebanyakan kasus, Anda tidak perlu menyegarkan daftar server untuk klien Anda ketika peristiwa ini terjadi. Namun, beberapa pustaka klien cache mungkin berhenti menggunakan node cache bahkan ElastiCache setelah mengganti node cache; dalam hal ini, aplikasi harus menyegarkan daftar server ketika peristiwa ini terjadi.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CacheNodeReplaceComplete	`ElastiCache:CacheNodeReplaceComplete : cluster-name`	ElastiCache telah mendeteksi bahwa host yang menjalankan node cache terdegradasi atau tidak dapat dijangkau dan telah selesai mengganti node cache. **Note** Entri DNS untuk simpul cache yang diganti tidak berubah. Pada kebanyakan kasus, Anda tidak perlu menyegarkan daftar server untuk klien Anda ketika peristiwa ini terjadi. Namun, beberapa pustaka klien cache mungkin berhenti menggunakan node cache bahkan ElastiCache setelah mengganti node cache; dalam hal ini, aplikasi harus menyegarkan daftar server ketika peristiwa ini terjadi.
ElastiCache:CacheNodesRebooted	`ElastiCache:CacheNodesRebooted : cluster-name`	Satu atau beberapa simpul cache telah di-boot ulang. Pesan (Memcached): `"Cache node %s shutdown"` Kemudian pesan kedua: `"Cache node %s restarted"`

Nama Peristiwa	Pesan	Deskripsi
ElastiCache: CertificateRenewalComplete (Valkey atau Redis OSS saja)	`ElastiCache:CertificateRenewalComplete`	Sertifikat Amazon CA berhasil diperbarui.
ElastiCache:CreateReplicationGroupComplete	`ElastiCache:CreateReplicationGroupComplete : cluster-name`	Grup replikasi berhasil dibuat.
ElastiCache:DeleteCacheClusterComplete	`ElastiCache:DeleteCacheClusterComplete : cluster-name`	Penghapusan klaster cache dan semua simpul cache terkait telah selesai.
ElastiCache: FailoverComplete (Valkey atau Redis OSS saja)	`ElastiCache:FailoverComplete : mycluster`	Failover ke simpul replika telah berhasil.
ElastiCache:ReplicationGroupIncreaseReplicaCountFinished	`ElastiCache:ReplicationGroupIncreaseReplicaCountFinished : cluster-name-0001-005`	Jumlah replika dalam klaster telah meningkat.
ElastiCache:ReplicationGroupIncreaseReplicaCountStarted	`ElastiCache:ReplicationGroupIncreaseReplicaCountStarted : cluster-name-0003-004`	Proses penambahan replika ke klaster Anda telah dimulai.
ElastiCache:NodeReplacementCanceled	`ElastiCache:NodeReplacementCanceled : cluster-name`	Sebuah simpul di klaster Anda yang dijadwalkan akan diganti tidak lagi dijadwalkan akan diganti.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:NodeReplacementRescheduled	`ElastiCache:NodeReplacementRescheduled : cluster-name`	Sebuah simpul di klaster Anda yang sebelumnya dijadwalkan akan diganti telah dijadwalkan ulang untuk diganti selama periode baru yang dijelaskan dalam notifikasi. Untuk informasi tentang tindakan yang dapat Anda ambil, lihat Mengganti node (Valkey dan Redis OSS).
ElastiCache:NodeReplacementScheduled	`ElastiCache:NodeReplacementScheduled : cluster-name`	Sebuah simpul di klaster Anda dijadwalkan akan diganti selama periode yang dijelaskan dalam notifikasi. Untuk informasi tentang tindakan yang dapat Anda ambil, lihat Mengganti node (Valkey dan Redis OSS).
ElastiCache:RemoveCacheNodeComplete	`ElastiCache:RemoveCacheNodeComplete : cluster-name`	Sebuah simpul cache telah dihapus dari klaster cache.
ElastiCache:ReplicationGroupScalingComplete	`ElastiCache:ReplicationGroupScalingComplete : cluster-name`	Operasi peningkatan skala pada grup replikasi berhasil diselesaikan.
ElastiCache:ReplicationGroupScalingFailed	`"Failed applying modification to cache node type to %s."`	Operasi peningkatan skala pada grup replikasi gagal.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:ServiceUpdateAvailableForNode	`"Service update is available for cache node %s."`	Pembaruan mandiri tersedia untuk simpul.
ElastiCache: SnapshotComplete (Valkey atau Redis OSS saja)	`ElastiCache:SnapshotComplete :` *`cluster-name`*	Sebuah snapshot cache telah berhasil diselesaikan.
ElastiCache: SnapshotFailed (Valkey atau Redis OSS saja)	`SnapshotFailed :` *`cluster-name`*	Sebuah snapshot cache telah gagal. Lihat peristiwa klaster cache untuk penyebab yang lebih terperinci. Jika Anda mendeskripsikan snapshot, lihat `DescribeSnapshots`, statusnya adalah `failed`.

Metrik dan peristiwa untuk Memcached

Bagian ini menjelaskan metrik dan peristiwa yang dapat Anda pantau saat bekerja dengan cache yang dirancang sendiri dan tanpa server Memcached.

Topik

Metrik cache tanpa server memcache

Bagian ini menjelaskan metrik dan peristiwa yang dapat Anda pantau saat bekerja dengan cache tanpa server Memcached.

`AWS/ElastiCache`Namespace menyertakan CloudWatch metrik berikut untuk cache tanpa server Memcached Anda.

Metrik	Deskripsi	Unit
BytesUsedForCache	Jumlah total byte yang digunakan oleh data yang disimpan dalam cache Anda.	Byte
ElastiCacheProcessingUnits	Jumlah total ElastiCacheProcessingUnits (ECPUs) yang dikonsumsi oleh permintaan yang dijalankan di cache Anda	Hitungan
SuccessfulReadRequestLatency	Latensi permintaan baca yang berhasil.	Mikrodetik
SuccessfulWriteRequestLatency	Latensi permintaan tulis yang berhasil	Mikrodetik
TotalCmdsCount	Jumlah total semua perintah yang dijalankan pada cache Anda	Hitungan

Metrik	Deskripsi	Unit
CurrConnections	Jumlah koneksi klien ke cache Anda.	Hitungan
ThrottledCmds	Jumlah permintaan yang dibatasi oleh ElastiCache karena beban kerja menskalakan lebih cepat daripada skala yang dapat diskalakan. ElastiCache	Hitungan
NewConnections	Jumlah seluruh koneksi yang telah diterima oleh server selama periode ini.	Hitungan
CurrItems	Jumlah item dalam cache.	Hitungan
NetworkBytesIn	Total byte yang ditransfer ke cache	Byte
NetworkBytesOut	Total byte yang ditransfer ke luar dari cache	Byte
Evictions	Hitungan kunci yang dikosongkan oleh cache	Hitungan
Diklaim kembali	Jumlah kunci yang sudah tidak berlaku menurut cache	Hitungan

Metrik tingkat perintah

ElastiCache juga memancarkan metrik tingkat perintah Memcached berikut

Metrik	Deskripsi	Unit
CmdGet	Jumlah perintah get yang telah diterima oleh cache.	Hitungan

Metrik	Deskripsi	Unit
CmdSet	Jumlah perintah set yang telah diterima oleh cache.	Hitungan
CmdTouch	Jumlah perintah touch yang telah diterima oleh cache.	Hitungan
GetHits	Jumlah permintaan get yang telah diterima oleh cache di mana kunci yang diminta ditemukan.	Hitungan
GetMisses	Jumlah permintaan get yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Hitungan
IncrHits	Jumlah permintaan increment yang telah diterima oleh cache di mana kunci yang diminta ditemukan.	Hitungan
IncrMisses	Jumlah permintaan increment yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Hitungan
DecrHits	Jumlah permintaan decrement yang telah diterima oleh cache di mana kunci yang diminta ditemukan.	Hitungan
DecrMisses	Jumlah permintaan decrement yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Hitungan

Metrik	Deskripsi	Unit
DeleteHits	Jumlah permintaan delete yang telah diterima oleh cache di mana kunci yang diminta ditemukan.	Hitungan
DeleteMisses	Jumlah permintaan delete yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Hitungan
TouchHits	Jumlah kunci yang telah di-touch dan diberi waktu kedaluwarsa yang baru.	Hitungan
TouchMisses	Jumlah kunci yang telah di-touch, tetapi tidak ditemukan.	Hitungan
CasHits	Jumlah permintaan cas yang telah diterima oleh cache di mana kunci yang diminta ditemukan dan nilai cas cocok.	Hitungan
CasMisses	Jumlah permintaan cas yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Hitungan
CasBadval	Jumlah permintaan cas yang telah diterima oleh cache di mana nilai cas tidak cocok dengan nilai cas yang tersimpan.	Hitungan
CmdFlush	Jumlah perintah flush yang telah diterima oleh cache.	Hitungan

Peristiwa cache tanpa server memcache

ElastiCache mencatat peristiwa yang berhubungan dengan cache tanpa server Anda. Informasi ini mencakup tanggal dan waktu peristiwa, nama sumber dan jenis sumber peristiwa, serta deskripsi dari peristiwa tersebut. Anda dapat dengan mudah mengambil peristiwa dari log menggunakan ElastiCache konsol, AWS CLI perintah deskripsi-peristiwa, atau tindakan API. ElastiCache `DescribeEvents`

Anda dapat memilih untuk memantau, menelan, mengubah, dan menindaklanjuti ElastiCache acara menggunakan Amazon EventBridge. Pelajari lebih lanjut di panduan EventBridge memulai Amazon.

Melihat ElastiCache acara (Konsol)

Untuk melihat acara menggunakan ElastiCache konsol:

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/
2. Untuk melihat daftar semua peristiwa yang tersedia, pada panel navigasi, pilih Peristiwa.
3. Pada layar Peristiwa setiap baris dari daftar merepresentasikan satu peristiwa dan menampilkan sumber peristiwa, jenis peristiwa, waktu GMT peristiwa, serta deskripsi dari peristiwa itu. Dengan Filter, Anda dapat menentukan apakah Anda ingin melihat semua peristiwa, atau hanya peristiwa dari jenis tertentu dalam daftar peristiwa.

Melihat ElastiCache acara (AWS CLI)

Untuk menghasilkan daftar ElastiCache peristiwa menggunakan AWS CLI, gunakan perintah deskripsi-peristiwa. Anda dapat menggunakan parameter opsional untuk mengontrol jenis peristiwa yang tercantum, jangka waktu peristiwa yang dicantumkan, jumlah maksimum peristiwa yang akan dicantumkan, dan lainnya.

Kode berikut mencantumkan hingga 40 peristiwa cache nirserver.

```
aws elasticache describe-events --source-type serverless-cache --max-items 40
```

Kode berikut mencantumkan semua peristiwa untuk cache nirserver selama 24 jam terakhir (1440 menit).

```
aws elasticache describe-events --source-type serverless-cache --duration 1440
```

Peristiwa Nirserver

Bagian ini mendokumentasikan berbagai jenis peristiwa yang mungkin Anda terima untuk cache nirserver.

Peristiwa Pembuatan Cache Nirserver

Jenis-Detail	Deskripsi	Unit	Sumber	Pesan
Cache dibuat	Cache arn	pembuatan	cache-nirserver	Cache <cache-name> dibuat dan siap digunakan.
Pembuatan cache gagal	Cache arn	kegagalan	cache-nirserver	Gagal membuat cache <cache-name>. Alamat IP yang bebas tidak cukup untuk membuat titik akhir VPC.
Pembuatan cache gagal	Cache arn	kegagalan	cache-nirserver	Gagal membuat cache <cache-name>. Subnet tidak valid yang disediakan dalam permintaan.
Pembuatan cache gagal	Cache arn	kegagalan	cache-nirserver	Gagal membuat cache <cache-name>. Batas kuota tercapai untuk membuat titik akhir VPC.
Pembuatan cache gagal	Cache arn	kegagalan	cache-nirserver	Gagal membuat cache <cache-na

Jenis-Detail	Deskripsi	Unit	Sumber	Pesan
				me>. Anda tidak memiliki izin untuk membuat titik akhir VPC.

Acara Pembaruan Cache Tanpa Server (Memcached)

Jenis-Detail	Daftar sumber daya	Kategori	Sumber	Pesan
Cache diperbarui	Cache arn	perubahan konfigurasi	cache-nirserver	SecurityGroups diperbarui untuk cache<cache-name>.
Cache diperbarui	Cache arn	perubahan konfigurasi	cache-nirserver	Tag diperbaru i untuk cache <cache-name>.
Pembaruan cache gagal	Cache arn	perubahan konfigurasi	cache-nirserver	Pembaruan ke cache <cache-name>gagal. SecurityGroups pembaruan gagal.
Pembaruan cache gagal	Cache arn	perubahan konfigurasi	cache-nirserver	Pembaruan ke cache <cache-name>gagal. SecurityGroups pembaruan gagal karena izin yang tidak mencukupi.

Jenis-Detail	Daftar sumber daya	Kategori	Sumber	Pesan
Pembaruan cache gagal	Cache arn	perubahan konfigurasi	cache-nirserver	Pembaruan ke cache <cache-name>gagal. SecurityGroups pembaruan gagal karena SecurityGroups tidak valid.

Acara Penghapusan Cache Tanpa Server (Memcached)

Jenis-Detail	Daftar sumber daya	Kategori	Sumber	Pesan
Cache dihapus	Cache arn	penghapusan	cache-nirserver	Cache <cache-name> telah dihapus.

Peristiwa Batas Penggunaan Cache Tanpa Server (Memcached)

Jenis-Detail	Deskripsi	Unit	Sumber	Pesan
Cache diperbarui	Cache arn	perubahan konfigurasi	cache-nirserver	Batas yang diperbarui untuk cache <cache-name>.
Pembaruan cache gagal	Cache arn	kegagalan	cache-nirserver	Pembaruan batas ke cache <cache-name> gagal karena

Jenis-Detail	Deskripsi	Unit	Sumber	Pesan
				cache telah dihapus.
Pembaruan cache gagal	Cache arn	kegagalan	cache-nirserver	Pembaruan batas ke cache <cache-name> gagal karena konfigurasi tidak valid.

Acara Snapshot Cache Tanpa Server (Memcached)

Jenis-Detail	Daftar-sumber daya	Kategori	Sumber	Pesan
Snapshot dibuat	Cache arn Snapshot arn	pembuatan	serverless-cache-snapshot	Snapshot <snapshot-name> dibuat untuk cache <cache-name>.
Pembuatan snapshot gagal	Cache arn Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal membuat snapshot untuk cache <cache-name>. Pembuatan snapshot <snapshot-name> gagal dengan Kunci yang Dikelola Pelanggan<key-id> <reason>. Pesan alasan kegagalan:

Jenis-Detail	Daftar-sumber daya	Kategori	Sumber	Pesan
				• karena Kunci yang Dikelola Pelanggan dinonaktifkan • karena Kunci yang Dikelola Pelanggan tidak dapat ditemukan • karena waktu permintaan habis
Pembuatan snapshot gagal	Cache arn Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal membuat snapshot untuk cache <cache-name>. Pembuatan snapshot <snapshot-name> gagal karena <reason>. Alasan default: • karena kesalahan internal

Jenis-Detail	Daftar-sumber daya	Kategori	Sumber	Pesan
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspo r snapshot ke bucket %s karena ElastiCac he tidak memiliki izin ke bucket.
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspor snapshot ke bucket '%s' karena sudah ada objek dengan nama yang sama di bucket.

Jenis-Detail	Daftar-sumber daya	Kategori	Sumber	Pesan
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspor snapshot ke bucket '%s' karena Id akun pemilik bucket telah berubah.
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspor snapshot ke bucket '%s' karena bucket S3 tidak dapat diakses.

Jenis-Detail	Daftar-sumber daya	Kategori	Sumber	Pesan
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspor snapshot ke bucket '%s' karena bucket tidak dapat diakses.
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspor snapshot ke bucket '%s' karena bucket tidak tersedia.

Jenis-Detail	Daftar-sumber daya	Kategori	Sumber	Pesan
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspo r snapshot ke bucket '%s' dengan snapshot sumber Kunci Terkelola Pelanggan %s <reason>.
Ekspor snapshot gagal	Snapshot arn	kegagalan	serverless-cache-snapshot	Gagal mengekspo r snapshot untuk cache <cache-name>. Tidak dapat mengekspor snapshot ke bucket '%s'.

Jenis-Detail	Daftar-sumber daya	Kategori	Sumber	Pesan
Salinan snapshot gagal	Snapshot arn-1 Snapshot arn-2	kegagalan	serverless-cache-snapshot	Gagal menyalin snapshot <snapshot-name>. Tidak dapat menyalin snapshot '%s' ke snapshot '%s' dengan snapshot sumber Kunci Terkelola Pelanggan <key-id> <reason-name>.
Salinan snapshot gagal	Snapshot arn-1 Snapshot arn-2	kegagalan	serverless-cache-snapshot	Gagal menyalin snapshot <snapshot-name>. Tidak dapat menyalin snapshot '%s' ke snapshot '%s' dengan snapshot target Kunci Terkelola Pelanggan '%s' '%s'.

Mencatat panggilan ElastiCache API Amazon dengan AWS CloudTrail

Amazon ElastiCache terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di Amazon ElastiCache. CloudTrail menangkap semua panggilan API untuk Amazon ElastiCache sebagai peristiwa, termasuk panggilan

dari ElastiCache konsol Amazon dan dari panggilan kode ke operasi Amazon ElastiCache API. Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara secara berkelanjutan ke bucket Amazon S3, termasuk acara untuk Amazon. ElastiCache Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa terbaru di CloudTrail konsol dalam Riwayat acara. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat ke Amazon ElastiCache, alamat IP dari mana permintaan itu dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan.

Untuk mempelajari selengkapnya CloudTrail, lihat Panduan AWS CloudTrail Pengguna.

ElastiCache Informasi Amazon di CloudTrail

CloudTrail diaktifkan di AWS akun Anda saat Anda membuat akun. Ketika aktivitas terjadi di Amazon ElastiCache, aktivitas tersebut direkam dalam suatu CloudTrail peristiwa bersama dengan peristiwa AWS layanan lainnya dalam riwayat Acara. Anda dapat melihat, mencari, dan mengunduh acara terbaru di AWS akun Anda. Untuk informasi selengkapnya, lihat Melihat Acara dengan Riwayat CloudTrail Acara.

Untuk catatan peristiwa yang sedang berlangsung di AWS akun Anda, termasuk acara untuk Amazon ElastiCache, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Secara default, ketika Anda membuat jejak di konsol, jejak ini diterapkan ke semua Wilayah. Trail mencatat peristiwa dari semua wilayah di AWS partisi dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi AWS layanan lain untuk menganalisis lebih lanjut dan menindaklanjuti data peristiwa yang dikumpulkan dalam CloudTrail log. Untuk informasi selengkapnya, lihat berikut:

- Gambaran Umum untuk Membuat Jejak
- CloudTrail Layanan dan Integrasi yang Didukung
- Mengkonfigurasi Notifikasi Amazon SNS untuk CloudTrail
- Menerima File CloudTrail Log dari Beberapa Wilayah dan Menerima File CloudTrail Log dari Beberapa Akun

Semua ElastiCache tindakan Amazon dicatat oleh CloudTrail dan didokumentasikan dalam Referensi ElastiCache API. Misalnya, panggilan ke`CreateCacheCluster`, `DescribeCacheCluster` dan `ModifyCacheCluster` tindakan menghasilkan entri dalam file CloudTrail log.

Setiap entri peristiwa atau log berisi informasi tentang entitas yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan hal berikut ini:

- Apakah permintaan tersebut dibuat dengan kredensial root atau pengguna IAM.
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna terfederasi.
- Apakah permintaan itu dibuat oleh AWS layanan lain.

Untuk informasi lain, lihat Elemen userIdentity CloudTrail .

Memahami entri file ElastiCache log Amazon

Trail adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai file log ke bucket Amazon S3 yang Anda tentukan. CloudTrail file log berisi satu atau lebih entri log. Peristiwa mewakili permintaan tunggal dari sumber manapun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, sehingga file tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan `CreateCacheCluster` tindakan.

```
{
    "eventVersion":"1.01",
    "userIdentity":{
        "type":"IAMUser",
        "principalId":"EXAMPLEEXAMPLEEXAMPLE",
        "arn":"arn:aws:iam::123456789012:user/elasticache-allow",
        "accountId":"123456789012",
        "accessKeyId":"AKIAIOSFODNN7EXAMPLE",
        "userName":"elasticache-allow"
    },
    "eventTime":"2014-12-01T22:00:35Z",
    "eventSource":"elasticache.amazonaws.com",
    "eventName":"CreateCacheCluster",
    "awsRegion":"us-west-2",
    "sourceIPAddress":"192.0.2.01",
    "userAgent":"AWS CLI/ElastiCache 1.10 API 2014-12-01",
    "requestParameters":{
        "numCacheNodes":2,
        "cacheClusterId":"test-memcached",
        "engine":"memcached",
        "aZMode":"cross-az",
        "cacheNodeType":"cache.m1.small",
```

```
    },
    "responseElements":{
        "engine":"memcached",
        "clientDownloadLandingPage":"https://console.aws.amazon.com/elasticache/
home#client-download:",
        "cacheParameterGroup":{
            "cacheParameterGroupName":"default.memcached1.4",
            "cacheNodeIdsToReboot":{
            },
            "parameterApplyStatus":"in-sync"
        },
        "preferredAvailabilityZone":"Multiple",
        "numCacheNodes":2,
        "cacheNodeType":"cache.m1.small",

        "cacheClusterStatus":"creating",
        "autoMinorVersionUpgrade":true,
        "preferredMaintenanceWindow":"thu:05:00-thu:06:00",
        "cacheClusterId":"test-memcached",
        "engineVersion":"1.4.14",
        "cacheSecurityGroups":[
            {
                "status":"active",
                "cacheSecurityGroupName":"default"
            }
        ],
        "pendingModifiedValues":{
        }
    },
    "requestID":"104f30b3-3548-11e4-b7b8-6d79ffe84edd",
    "eventID":"92762127-7a68-42ce-8787-927d2174cde1"
}
```

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan `DescribeCacheCluster` tindakan. Perhatikan bahwa untuk semua panggilan Amazon ElastiCache `Describe*` Describe (), `ResponseElements` bagian tersebut dihapus dan muncul sebagai`null`.

```
{
    "eventVersion":"1.01",
    "userIdentity":{
        "type":"IAMUser",
        "principalId":"EXAMPLEEXAMPLEEXAMPLE",
```

```
        "arn":"arn:aws:iam::123456789012:user/elasticache-allow",
        "accountId":"123456789012",
        "accessKeyId":"AKIAIOSFODNN7EXAMPLE",
        "userName":"elasticache-allow"
    },
    "eventTime":"2014-12-01T22:01:00Z",
    "eventSource":"elasticache.amazonaws.com",
    "eventName":"DescribeCacheClusters",
    "awsRegion":"us-west-2",
    "sourceIPAddress":"192.0.2.01",
    "userAgent":"AWS CLI/ElastiCache 1.10 API 2014-12-01",
    "requestParameters":{
        "showCacheNodeInfo":false,
        "maxRecords":100
    },
    "responseElements":null,
    "requestID":"1f0b5031-3548-11e4-9376-c1d979ba565a",
    "eventID":"a58572a8-e81b-4100-8e00-1797ed19d172"
}
```

Contoh berikut menunjukkan entri CloudTrail log yang merekam `ModifyCacheCluster` tindakan.

```
{
    "eventVersion":"1.01",
    "userIdentity":{
        "type":"IAMUser",
        "principalId":"EXAMPLEEXAMPLEEXAMPLE",
        "arn":"arn:aws:iam::123456789012:user/elasticache-allow",
        "accountId":"123456789012",
        "accessKeyId":"AKIAIOSFODNN7EXAMPLE",
        "userName":"elasticache-allow"
    },
    "eventTime":"2014-12-01T22:32:21Z",
    "eventSource":"elasticache.amazonaws.com",
    "eventName":"ModifyCacheCluster",
    "awsRegion":"us-west-2",
    "sourceIPAddress":"192.0.2.01",
    "userAgent":"AWS CLI/ElastiCache 1.10 API 2014-12-01",
    "requestParameters":{
        "applyImmediately":true,
        "numCacheNodes":3,
        "cacheClusterId":"test-memcached"
    },
```

```
    "responseElements":{
        "engine":"memcached",
        "clientDownloadLandingPage":"https://console.aws.amazon.com/elasticache/
home#client-download:",
        "cacheParameterGroup":{
            "cacheParameterGroupName":"default.memcached1.4",
            "cacheNodeIdsToReboot":{
            },
            "parameterApplyStatus":"in-sync"
        },
        "cacheClusterCreateTime":"Dec 1, 2014 10:16:06 PM",
        "preferredAvailabilityZone":"Multiple",
        "numCacheNodes":2,
        "cacheNodeType":"cache.m1.small",
        "cacheClusterStatus":"modifying",
        "autoMinorVersionUpgrade":true,
        "preferredMaintenanceWindow":"thu:05:00-thu:06:00",
        "cacheClusterId":"test-memcached",
        "engineVersion":"1.4.14",
        "cacheSecurityGroups":[
            {
                "status":"active",
                "cacheSecurityGroupName":"default"
            }
        ],
        "configurationEndpoint":{
            "address":"test-memcached.example.cfg.use1prod.cache.amazonaws.com",
            "port":11211
        },
        "pendingModifiedValues":{
            "numCacheNodes":3
        }
    },
    "requestID":"807f4bc3-354c-11e4-9376-c1d979ba565a",
    "eventID":"e9163565-376f-4223-96e9-9f50528da645"
}
```

Pemantauan acara Amazon SNS ElastiCache

Saat peristiwa penting terjadi untuk klaster, ElastiCache kirimkan pemberitahuan ke topik Amazon SNS tertentu. Contohnya meliputi kegagalan menambahkan simpul, keberhasilan menambahkan simpul, perubahan grup keamanan, dan lainnya. Dengan memantau peristiwa penting, Anda dapat

mengetahui status klaster Anda saat ini dan dapat mengambil tindakan korektif sesuai peristiwa tersebut.

Topik

Mengelola ElastiCache notifikasi Amazon SNS

Anda dapat mengonfigurasi ElastiCache untuk mengirim notifikasi untuk peristiwa kluster penting menggunakan Amazon Simple Notification Service (Amazon SNS). Dalam contoh ini, Anda akan mengonfigurasi klaster dengan Amazon Resource Name (ARN) dari topik Amazon SNS untuk menerima notifikasi.

> **Note**
>
> - Topik ini mengasumsikan bahwa Anda telah mendaftar ke Amazon SNS dan telah mengatur serta berlangganan topik Amazon SNS. Untuk informasi selengkapnya tentang cara melakukannya, lihat Panduan Developer Amazon Simple Notification Service.
> - Secara default, `API modify-replication-group` mempengaruhi semua grup di Wilayah dan bukan hanya grup yang ditentukan saat ini. Jika Anda ingin mengonfigurasi satu grup tertentu di Wilayah secara berbeda dari grup lain, Anda dapat menggunakan `--notification-topic-arn` opsi untuk membuat topik terpisah untuk grup tersebut.

Menambahkan topik Amazon SNS

Bagian berikut menunjukkan cara menambahkan topik Amazon SNS menggunakan AWS Konsol, API AWS CLI, atau ElastiCache Konsol.

Menambahkan topik Amazon SNS (Konsol)

Prosedur berikut menunjukkan cara menambahkan topik Amazon SNS untuk klaster. Saat menggunakan Valkey atau Redis OSS untuk menambahkan topik Amazon SNS untuk grup replikasi di langkah 2, alih-alih memilih cluster, pilih grup replikasi. Kemudian ikuti langkah-langkah yang tersisa yang sama.

> **Note**
>
> Proses ini juga dapat digunakan untuk mengubah topik Amazon SNS.

Untuk menambahkan atau mengubah topik Amazon SNS untuk klaster (Konsol)

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Di Klaster, pilih klaster yang ingin Anda tambahkan atau ubah ARN topik Amazon SNS-nya.
3. Pilih Ubah.
4. Di Ubah Klaster di bagian Topik untuk Notifikasi SNS, pilih topik SNS yang ingin Anda tambahkan, atau pilih Input ARN manual dan ketik ARN topik Amazon SNS.
5. Pilih Ubah.

Menambahkan topik Amazon SNS (AWS CLI)

Untuk menambah atau memodifikasi topik Amazon SNS untuk klaster, gunakan perintah. AWS CLI `modify-cache-cluster`

Contoh kode berikut menambahkan ARN topik Amazon SNS ke my-cluster.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-cache-cluster \
    --cache-cluster-id my-cluster \
    --notification-topic-arn arn:aws:sns:us-
west-2:123456789xxx:ElastiCacheNotifications
```

Untuk Windows:

```
aws elasticache modify-cache-cluster ^
    --cache-cluster-id my-cluster ^
    --notification-topic-arn arn:aws:sns:us-west-2:123456789xx:ElastiCacheNotifications
```

Untuk informasi selengkapnya, lihat modify-cache-cluster.

Menambahkan topik Amazon SNS (API) ElastiCache

Untuk menambah atau mengubah topik Amazon SNS untuk klaster, panggil tindakan `ModifyCacheCluster` dengan parameter berikut:

- `CacheClusterId=my-cluster`
- `TopicArn=arn%3Aaws%3Asns%3Aus-west-2%3A565419523791%3AElastiCacheNotifications`

Example

```
https://elasticache.amazon.com/
    ?Action=ModifyCacheCluster
    &ApplyImmediately=false
    &CacheClusterId=my-cluster
    &NotificationTopicArn=arn%3Aaws%3Asns%3Aus-
west-2%3A565419523791%3AElastiCacheNotifications
    &Version=2014-12-01
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Date=20141201T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20141201T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Untuk informasi selengkapnya, lihat ModifyCacheCluster.

Mengaktifkan dan menonaktifkan notifikasi Amazon SNS

Anda dapat mengaktifkan atau menonaktifkan notifikasi untuk klaster. Prosedur berikut menunjukkan cara menonaktifkan notifikasi Amazon SNS.

Mengaktifkan dan menonaktifkan notifikasi Amazon SNS (Konsol)

Untuk menonaktifkan notifikasi Amazon SNS menggunakan AWS Management Console

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.

2. Untuk melihat daftar klaster Anda yang menjalankan Memcached, pada panel navigasi, pilih Memcached.

   Untuk melihat daftar cluster Anda yang menjalankan Valkey atau Redis OSS, di panel navigasi pilih Valkey atau Redis OSS.
3. Pilih kotak di sebelah kiri klaster yang ingin diubah notifikasinya.
4. Pilih Ubah.
5. Di Ubah Klaster di bagian Topik untuk Notifikasi SNS, pilih Nonaktifkan Notifikasi.
6. Pilih Ubah.

Mengaktifkan dan menonaktifkan notifikasi Amazon SNS (AWS CLI)

Untuk menonaktifkan notifikasi Amazon SNS, gunakan perintah `modify-cache-cluster` dengan parameter berikut:

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-cache-cluster \
    --cache-cluster-id my-cluster \
    --notification-topic-status inactive
```

Untuk Windows:

```
aws elasticache modify-cache-cluster ^
    --cache-cluster-id my-cluster ^
    --notification-topic-status inactive
```

> **Note**
>
> Ketika cluster cache milik grup replikasi, Anda harus menggunakan `modify-replication-group` perintah CLI untuk mengaktifkan atau menonaktifkan pemberitahuan SNS.

Mengaktifkan dan menonaktifkan notifikasi Amazon SNS (API) ElastiCache

Untuk menonaktifkan notifikasi Amazon SNS, panggil tindakan `ModifyCacheCluster` dengan parameter berikut:

- `CacheClusterId=my-cluster`
- `NotificationTopicStatus=inactive`

Panggilan ini menghasilkan output seperti yang berikut ini:

Example

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=ModifyCacheCluster
    &ApplyImmediately=false
    &CacheClusterId=my-cluster
    &NotificationTopicStatus=inactive
    &Version=2014-12-01
    &SignatureVersion=4
    &SignatureMethod=HmacSHA256
    &Timestamp=20141201T220302Z
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Date=20141201T220302Z
    &X-Amz-SignedHeaders=Host
    &X-Amz-Expires=20141201T220302Z
    &X-Amz-Credential=<credential>
    &X-Amz-Signature=<signature>
```

Melihat ElastiCache acara

ElastiCache mencatat peristiwa yang berhubungan dengan instance klaster, grup keamanan, dan grup parameter Anda. Informasi ini mencakup tanggal dan waktu peristiwa, nama sumber dan jenis sumber peristiwa, serta deskripsi dari peristiwa tersebut. Anda dapat dengan mudah mengambil peristiwa dari log menggunakan ElastiCache konsol, AWS CLI `describe-events` perintah, atau tindakan ElastiCache `DescribeEvents` API.

Prosedur berikut menunjukkan cara melihat semua ElastiCache acara selama 24 jam terakhir (1440 menit).

Melihat ElastiCache acara (Konsol)

Prosedur berikut menampilkan peristiwa menggunakan ElastiCache konsol.

Untuk melihat peristiwa menggunakan ElastiCache konsol

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Untuk melihat daftar semua peristiwa yang tersedia, pada panel navigasi, pilih Peristiwa.

   Di layar Acara, setiap baris daftar mewakili satu peristiwa dan menampilkan sumber acara, jenis acara (cache-cluster, cache-parameter-group cache-security-group, atau cache-subnet-group), waktu GMT acara, dan deskripsi acara.

   Dengan Filter, Anda dapat menentukan apakah Anda ingin melihat semua peristiwa, atau hanya peristiwa dari jenis tertentu dalam daftar peristiwa.

Melihat ElastiCache acara (AWS CLI)

Untuk menghasilkan daftar ElastiCache acara menggunakan AWS CLI, gunakan perintah`describe-events`. Anda dapat menggunakan parameter opsional untuk mengontrol jenis peristiwa yang dicantumkan, kerangka waktu peristiwa yang tercantum, jumlah maksimum peristiwa untuk dicantumkan, dan banyak lagi.

Kode berikut menampilkan daftar hingga 40 peristiwa klaster cache.

```
aws elasticache describe-events --source-type cache-cluster --max-items 40
```

Kode berikut menampilkan daftar semua peristiwa selama 24 jam terakhir (1.440 menit).

```
aws elasticache describe-events --source-type cache-cluster --duration 1440
```

Output dari perintah `describe-events` terlihat seperti berikut ini.

```
aws elasticache describe-events --source-type cache-cluster --max-items 40
{
    "Events": [
        {
            "SourceIdentifier": "my-mem-cluster",
            "SourceType": "cache-cluster",
            "Message": "Finished modifying number of nodes from 1 to 3",
            "Date": "2020-06-09T02:01:21.772Z"
        },
        {
            "SourceIdentifier": "my-mem-cluster",
            "SourceType": "cache-cluster",
            "Message": "Added cache node 0002 in availability zone us-west-2a",
            "Date": "2020-06-09T02:01:21.716Z"
        },
        {
            "SourceIdentifier": "my-mem-cluster",
            "SourceType": "cache-cluster",
            "Message": "Added cache node 0003 in availability zone us-west-2a",
            "Date": "2020-06-09T02:01:21.706Z"
        },
        {
            "SourceIdentifier": "my-mem-cluster",
            "SourceType": "cache-cluster",
            "Message": "Increasing number of requested nodes",
            "Date": "2020-06-09T01:58:34.178Z"
        },
        {
            "SourceIdentifier": "mycluster-0003-004",
            "SourceType": "cache-cluster",
            "Message": "Added cache node 0001 in availability zone us-west-2c",
            "Date": "2020-06-09T01:51:14.120Z"
        },
        {
            "SourceIdentifier": "mycluster-0003-004",
            "SourceType": "cache-cluster",
            "Message": "This cache cluster does not support persistence (ex:
 'appendonly').  Please use a different instance type to enable persistence.",
            "Date": "2020-06-09T01:51:14.095Z"
```

```
        },
        {
            "SourceIdentifier": "mycluster-0003-004",
            "SourceType": "cache-cluster",
            "Message": "Cache cluster created",
            "Date": "2020-06-09T01:51:14.094Z"
        },
        {
            "SourceIdentifier": "mycluster-0001-005",
            "SourceType": "cache-cluster",
            "Message": "Added cache node 0001 in availability zone us-west-2b",
            "Date": "2020-06-09T01:42:55.603Z"
        },
        {
            "SourceIdentifier": "mycluster-0001-005",
            "SourceType": "cache-cluster",
            "Message": "This cache cluster does not support persistence (ex:
 'appendonly').  Please use a different instance type to enable persistence.",
            "Date": "2020-06-09T01:42:55.576Z"
        },
        {
            "SourceIdentifier": "mycluster-0001-005",
            "SourceType": "cache-cluster",
            "Message": "Cache cluster created",
            "Date": "2020-06-09T01:42:55.574Z"
        },
        {
            "SourceIdentifier": "mycluster-0001-004",
            "SourceType": "cache-cluster",
            "Message": "Added cache node 0001 in availability zone us-west-2b",
            "Date": "2020-06-09T01:28:40.798Z"
        },
        {
            "SourceIdentifier": "mycluster-0001-004",
            "SourceType": "cache-cluster",
            "Message": "This cache cluster does not support persistence (ex:
 'appendonly').  Please use a different instance type to enable persistence.",
            "Date": "2020-06-09T01:28:40.775Z"
        },
        {
            "SourceIdentifier": "mycluster-0001-004",
            "SourceType": "cache-cluster",
            "Message": "Cache cluster created",
            "Date": "2020-06-09T01:28:40.773Z"
```

```
        }
    ]
}
```

Untuk informasi selengkapnya, seperti parameter yang tersedia dan nilai parameter yang diizinkan, lihat `describe-events`.

Melihat ElastiCache acara (ElastiCache API)

Untuk membuat daftar ElastiCache peristiwa menggunakan ElastiCache API, gunakan `DescribeEvents` tindakan. Anda dapat menggunakan parameter opsional untuk mengontrol jenis peristiwa yang tercantum, jangka waktu peristiwa yang dicantumkan, jumlah maksimum peristiwa yang akan dicantumkan, dan lainnya.

Kode berikut menampilkan daftar 40 peristiwa klaster cache terbaru.

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DescribeEvents
   &MaxRecords=40
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &SourceType=cache-cluster
   &Timestamp=20150202T192317Z
   &Version=2015-02-02
   &X-Amz-Credential=<credential>
```

Kode berikut menampilkan daftar peristiwa klaster cache selama 24 jam terakhir (1.440 menit).

```
https://elasticache.us-west-2.amazonaws.com/
   ?Action=DescribeEvents
   &Duration=1440
   &SignatureVersion=4
   &SignatureMethod=HmacSHA256
   &SourceType=cache-cluster
   &Timestamp=20150202T192317Z
   &Version=2015-02-02
   &X-Amz-Credential=<credential>
```

Perintah di atas akan menghasilkan output yang serupa dengan berikut ini.

```
<DescribeEventsResponse xmlns="http://elasticache.amazonaws.com/doc/2015-02-02/">
    <DescribeEventsResult>
```

```
        <Events>
            <Event>
                <Message>Cache cluster created</Message>
                <SourceType>cache-cluster</SourceType>
                <Date>2015-02-02T18:22:18.202Z</Date>
                <SourceIdentifier>mem01</SourceIdentifier>
            </Event>

 (...output omitted...)

        </Events>
    </DescribeEventsResult>
    <ResponseMetadata>
        <RequestId>e21c81b4-b9cd-11e3-8a16-7978bb24ffdf</RequestId>
    </ResponseMetadata>
</DescribeEventsResponse>
```

Untuk informasi selengkapnya, seperti parameter yang tersedia dan nilai parameter yang diizinkan, lihat `DescribeEvents`.

Notifikasi Peristiwa dan Amazon SNS

ElastiCache dapat mempublikasikan pesan menggunakan Amazon Simple Notification Service (SNS) ketika peristiwa penting terjadi pada cluster cache. Fitur ini dapat digunakan untuk menyegarkan daftar server pada mesin klien yang terhubung ke setiap titik akhir simpul cache pada klaster cache.

> **Note**
>
> Untuk informasi selengkapnya tentang Amazon Simple Notification Service (SNS), termasuk informasi tentang harga dan tautan ke dokumentasi Amazon SNS, lihat Halaman produk Amazon SNS.

Notifikasi dipublikasikan ke topik Amazon SNS tertentu. Berikut ini adalah persyaratan untuk notifikasi:

- Hanya satu topik yang dapat dikonfigurasi untuk ElastiCache pemberitahuan.
- AWS Akun yang memiliki topik Amazon SNS harus merupakan akun yang sama yang memiliki cluster cache tempat notifikasi diaktifkan.
- Topik Amazon SNS yang Anda publikasikan tidak dapat dienkripsi.

> **Note**
>
> Topik Amazon SNS yang terenkripsi (diam) dapat dilampirkan ke klaster. Namun, status topik dari ElastiCache konsol akan ditampilkan sebagai tidak aktif, yang secara efektif memisahkan topik dari cluster saat ElastiCache mendorong pesan ke topik.

- Topik Amazon SNS harus berada di Wilayah yang sama dengan cluster. ElastiCache

ElastiCache Acara

ElastiCache Peristiwa berikut memicu notifikasi Amazon SNS. Untuk informasi tentang detail peristiwa, lihat Melihat ElastiCache acara.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:AddCacheNodeComplete	`ElastiCache:AddCacheNodeComplete : cache-cluster`	Simpul cache telah ditambahkan ke klaster cache dan siap untuk digunakan.
ElastiCache: AddCacheNodeFailed karena alamat IP gratis yang tidak mencukupi	`ElastiCache:AddCacheNodeFailed : cluster-name`	Simpul cache tidak dapat ditambahkan karena alamat IP yang tersedia tidak cukup.
ElastiCache:CacheClusterParametersChanged	`ElastiCache:CacheClusterParametersChanged : cluster-name`	Satu atau beberapa parameter klaster cache telah berubah.
ElastiCache:CacheClusterProvisioningComplete	`ElastiCache:CacheClusterProvisioningComplete cluster-name-0001-005`	Penyediaan klaster cache selesai, dan simpul cache dalam klaster cache siap untuk digunakan.
ElastiCache: CacheClusterProvisioningFailed karena status jaringan yang tidak kompatibel	`ElastiCache:CacheClusterProvisioningFailed : cluster-name`	Percobaan dilakukan untuk meluncurkan klaster cache baru ke cloud privat virtual (VPC) yang tidak ada.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CacheClusterScalingComplete	`CacheClusterScalingComplete : cluster-name`	Penskalaan untuk klaster cache berhasil diselesaikan.
ElastiCache:CacheClusterScalingFailed	ElastiCache:CacheClusterScalingFailed : *cluster-name*	Operasi peningkatan skala pada klaster cache gagal.
ElastiCache:CacheClusterSecurityGroupModified	`ElastiCache:CacheClusterSecurityGroupModified : cluster-name`	Salah satu peristiwa berikut telah terjadi: • Daftar grup keamanan cache yang diizinkan untuk klaster cache yang telah diubah. • Satu atau lebih grup EC2 keamanan baru telah diotorisasi pada salah satu grup keamanan cache yang terkait dengan cluster cache. • Satu atau lebih grup EC2 keamanan telah dicabut dari salah satu grup keamanan cache yang terkait dengan cluster cache.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CacheNodeReplaceStarted	`ElastiCache:CacheNodeReplaceStarted : cluster-name`	ElastiCache telah mendeteksi bahwa host yang menjalankan node cache terdegradasi atau tidak dapat dijangkau dan telah mulai mengganti node cache. **Note** Entri DNS untuk simpul cache yang diganti tidak berubah. Pada kebanyakan kasus, Anda tidak perlu menyegarkan daftar server untuk klien Anda ketika peristiwa ini terjadi. Namun, beberapa pustaka klien cache mungkin berhenti menggunakan node cache bahkan ElastiCache setelah mengganti node cache; dalam hal ini, aplikasi harus menyegarkan daftar server ketika peristiwa ini terjadi.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CacheNodeReplaceComplete	`ElastiCache:CacheNodeReplaceComplete : cluster-name`	ElastiCache telah mendeteksi bahwa host yang menjalankan node cache terdegradasi atau tidak dapat dijangkau dan telah selesai mengganti node cache. **Note** Entri DNS untuk simpul cache yang diganti tidak berubah. Pada kebanyakan kasus, Anda tidak perlu menyegarkan daftar server untuk klien Anda ketika peristiwa ini terjadi. Namun, beberapa pustaka klien cache mungkin berhenti menggunakan node cache bahkan ElastiCache setelah mengganti node cache; dalam hal ini, aplikasi harus menyegarkan daftar server ketika peristiwa ini terjadi.
ElastiCache:CacheNodesRebooted	`ElastiCache:CacheNodesRebooted : cluster-name`	Satu atau beberapa simpul cache telah di-boot ulang. Pesan (Memcached): `"Cache node %s shutdown"` Kemudian pesan kedua: `"Cache node %s restarted"`

Nama Peristiwa	Pesan	Deskripsi
ElastiCache: CertificateRenewalComplete (Valkey atau Redis OSS saja)	`ElastiCache:CertificateRenewalComplete`	Sertifikat Amazon CA berhasil diperbarui.
ElastiCache:CreateReplicationGroupComplete	`ElastiCache:CreateReplicationGroupComplete : cluster-name`	Grup replikasi berhasil dibuat.
ElastiCache:DeleteCacheClusterComplete	`ElastiCache:DeleteCacheClusterComplete : cluster-name`	Penghapusan klaster cache dan semua simpul cache terkait telah selesai.
ElastiCache: FailoverComplete (Valkey atau Redis OSS saja)	`ElastiCache:FailoverComplete : mycluster`	Failover ke simpul replika telah berhasil.
ElastiCache:ReplicationGroupIncreaseReplicaCountFinished	`ElastiCache:ReplicationGroupIncreaseReplicaCountFinished : cluster-name-0001-005`	Jumlah replika dalam klaster telah meningkat.
ElastiCache:ReplicationGroupIncreaseReplicaCountStarted	`ElastiCache:ReplicationGroupIncreaseReplicaCountStarted : cluster-name-0003-004`	Proses penambahan replika ke klaster Anda telah dimulai.
ElastiCache:NodeReplacementCanceled	`ElastiCache:NodeReplacementCanceled : cluster-name`	Sebuah simpul di klaster Anda yang dijadwalkan akan diganti tidak lagi dijadwalkan akan diganti.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:NodeReplacementRescheduled	`ElastiCache:NodeReplacementRescheduled : cluster-name`	Sebuah simpul di klaster Anda yang sebelumnya dijadwalkan akan diganti telah dijadwalkan ulang untuk diganti selama periode baru yang dijelaskan dalam notifikasi. Untuk informasi tentang tindakan yang dapat Anda ambil, lihat Mengganti node (Valkey dan Redis OSS).
ElastiCache:NodeReplacementScheduled	`ElastiCache:NodeReplacementScheduled : cluster-name`	Sebuah simpul di klaster Anda dijadwalkan akan diganti selama periode yang dijelaskan dalam notifikasi. Untuk informasi tentang tindakan yang dapat Anda ambil, lihat Mengganti node (Valkey dan Redis OSS).
ElastiCache:RemoveCacheNodeComplete	`ElastiCache:RemoveCacheNodeComplete : cluster-name`	Sebuah simpul cache telah dihapus dari klaster cache.
ElastiCache:ReplicationGroupScalingComplete	`ElastiCache:ReplicationGroupScalingComplete : cluster-name`	Operasi peningkatan skala pada grup replikasi berhasil diselesaikan.
ElastiCache:ReplicationGroupScalingFailed	`"Failed applying modification to cache node type to %s."`	Operasi peningkatan skala pada grup replikasi gagal.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:ServiceUpdateAvailableForNode	`"Service update is available for cache node %s."`	Pembaruan mandiri tersedia untuk simpul.
ElastiCache: SnapshotComplete (Valkey atau Redis OSS saja)	`ElastiCache:SnapshotComplete :` *`cluster-name`*	Sebuah snapshot cache telah berhasil diselesaikan.
ElastiCache: SnapshotFailed (Valkey atau Redis OSS saja)	`SnapshotFailed :` *`cluster-name`*	Sebuah snapshot cache telah gagal. Lihat peristiwa klaster cache untuk penyebab yang lebih terperinci. Jika Anda mendeskripsikan snapshot, lihat `DescribeSnapshots`, statusnya adalah `failed`.

Topik terkait

- Melihat ElastiCache acara

Pengiriman log

> **Note**
>
> Log Lambat didukung untuk Valkey 7.x ke atas, dan cluster cache Redis OSS dan grup replikasi menggunakan engine versi 6.0 dan seterusnya.
> Engine Log didukung untuk Valkey 7.x dan di atasnya, dan cluster cache Redis OSS dan grup replikasi menggunakan engine versi 6.2 dan seterusnya.

Pengiriman log memungkinkan Anda melakukan streaming SLOWLOG atau Engine Log ke salah satu dari dua tujuan:

- Amazon Data Firehose
- CloudWatch Log Amazon

Anda mengaktifkan dan mengonfigurasi pengiriman log saat membuat atau memodifikasi klaster menggunakan ElastiCache APIs. Setiap entri log akan dikirimkan ke destinasi yang ditentukan dalam salah satu dari dua format berikut: JSON atau TEXT.

Sejumlah entri log lambat yang tetap diambil dari mesin secara berkala. Tergantung pada nilai yang ditentukan untuk parameter mesin `slowlog-max-len`, entri log lambat tambahan mungkin tidak dikirimkan ke destinasi.

Anda dapat memilih untuk mengubah konfigurasi pengiriman atau menonaktifkan pengiriman log kapan saja menggunakan AWS konsol atau salah satu modifikasi APIs, salah satu modify-cache-clusteratau modify-replication-group.

Anda harus menetapkan parameter `apply-immediately` untuk semua perubahan pengiriman log.

> **Note**
>
> Biaya Amazon CloudWatch Logs berlaku saat pengiriman log diaktifkan, bahkan saat log dikirim langsung ke Amazon Data Firehose. Untuk informasi selengkapnya, lihat bagian Log Penjual di CloudWatch Harga Amazon.

Isi dari entri log lambat

Log Lambat berisi informasi berikut:

- CacheClusterId— ID dari cluster cache
- CacheNodeId— ID dari node cache
- Id – Pengidentifikasi progresif unik untuk setiap entri log lambat
- Timestamp – Stempel waktu Unix menunjukkan saat perintah yang dicatat ke log diproses
- Duration – Jumlah waktu yang diperlukan untuk eksekusinya, dalam mikrodetik
- Command – Perintah yang digunakan oleh klien. Misalnya, `set foo bar` di `foo` mana kuncinya dan `bar` nilainya. ElastiCache menggantikan nama kunci dan nilai yang sebenarnya dengan `(2 more arguments)` untuk menghindari mengekspos data sensitif.
- ClientAddress— Alamat IP klien dan port

- ClientName— Nama klien jika diatur melalui `CLIENT SETNAME` perintah

Isi entri log mesin

Log ElastiCache Mesin berisi informasi berikut:

- CacheClusterId— ID dari cluster cache
- CacheNodeId— ID dari node cache
- Level log — LogLevel bisa salah satu dari yang berikut:`VERBOSE("-")`,`NOTICE("*")`,`WARNING("#")`.
- Time - Waktu UTC dari pesan yang dicatat. Waktu berada dalam format berikut: `"DD MMM YYYY hh:mm:ss.ms UTC"`
- Role – Peran simpul asal log dipancarkan. Ini bisa menjadi salah satu dari yang berikut: “M” untuk Primer, “S” untuk replika, “C” untuk proses anak penulis yang sedang dikerjakan RDB/AOF atau “X” untuk sentinel.
- Pesan — Pesan log mesin.

Izin untuk mengonfigurasi pencatatan log

Anda harus menyertakan izin IAM berikut dalam kebijakan user/role IAM Anda:

- `logs:CreateLogDelivery`
- `logs:UpdateLogDelivery`
- `logs:DeleteLogDelivery`
- `logs:GetLogDelivery`
- `logs:ListLogDeliveries`

Untuk informasi selengkapnya, lihat Gambaran umum manajemen akses: Izin dan kebijakan.

Spesifikasi format log dan jenis log

Log lambat

Log lambat mendukung JSON dan TEXT

Contoh berikut menunjukkan format JSON:

```
{
  "CacheClusterId": "logslowxxxxmsxj",
  "CacheNodeId": "0001",
  "Id": 296,
  "Timestamp": 1605631822,
  "Duration (us)": 0,
  "Command": "GET ... (1 more arguments)",
  "ClientAddress": "192.168.12.104:55452",
  "ClientName": "logslowxxxxmsxj##"
}
```

Contoh berikut menunjukkan format TEXT:

```
logslowxxxxmsxj,0001,1605631822,30,GET ... (1 more
 arguments),192.168.12.104:55452,logslowxxxxmsxj##
```

Log mesin

Log mesin mendukung JSON dan TEXT

Contoh berikut menunjukkan format JSON:

```
{
  "CacheClusterId": "xxxxxxxxxzy-engine-log-test",
  "CacheNodeId": "0001",
  "LogLevel": "VERBOSE",
  "Role": "M",
  "Time": "12 Nov 2020 01:28:57.994 UTC",
  "Message": "Replica is waiting for next BGSAVE before synchronizing with the primary.
 Check back later"
}
```

Contoh berikut menunjukkan format TEXT:

```
xxxxxxxxxxxzy-engine-log-test/0001:M 29 Oct 2020 20:12:20.499 UTC * A slow-running Lua
 script detected that is still in execution after 10000 milliseconds.
```

ElastiCache tujuan pencatatan

Bagian ini menjelaskan tujuan pencatatan yang dapat Anda pilih untuk ElastiCache log Anda. Setiap bagian menyediakan panduan untuk mengonfigurasi pencatatan log untuk jenis tujuan dan

informasi tentang semua perilaku yang spesifik untuk jenis destinasi. Setelah mengonfigurasi tujuan pencatatan, Anda dapat memberikan spesifikasinya ke konfigurasi ElastiCache logging untuk mulai masuk ke sana.

Topik

- CloudWatch Log Amazon
- Amazon Data Firehose

CloudWatch Log Amazon

- Anda menentukan grup CloudWatch log Log tempat log akan dikirimkan.
- Log dari beberapa cluster Valkey atau Redis OSS dan grup replikasi dapat dikirim ke grup log yang sama.
- Log stream baru akan dibuat untuk setiap simpul dalam klaster cache atau grup replikasi dan log akan dikirimkan ke masing-masing log stream itu. Nama log stream akan menggunakan format berikut: `elasticache/${engine-name}/${cache-cluster-id}/${cache-node-id}/${log-type}`

Izin untuk mempublikasikan log ke CloudWatch Log

Anda harus memiliki pengaturan izin berikut untuk mengonfigurasi ElastiCache untuk mengirim log ke grup CloudWatch log Log:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:UpdateLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries"
            ],
            "Resource": [
```

```
                "*"
            ],
            "Effect": "Allow",
            "Sid": "ElastiCacheLogging"
        },
        {
            "Sid": "ElastiCacheLoggingCWL",
            "Action": [
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Log yang dikirim ke CloudWatch Log.

Amazon Data Firehose

- Anda menentukan aliran pengiriman Firehose tempat log akan dikirimkan.
- Log dari beberapa cluster Valkey atau Redis OSS dan grup replikasi dapat dikirim ke aliran pengiriman yang sama.
- Log dari setiap simpul dalam klaster cache atau grup replikasi akan dikirimkan ke aliran pengiriman yang sama. Anda dapat membedakan pesan log dari simpul cache yang berbeda berdasarkan `cache-cluster-id` dan `cache-node-id` yang disertakan di tiap pesan log.
- Pengiriman log ke Firehose saat ini tidak tersedia di Wilayah Asia Pasifik (Osaka).

Izin untuk mempublikasikan log ke Firehose

Anda harus memiliki izin berikut untuk mengonfigurasi untuk mengirim log ElastiCache ke aliran pengiriman Amazon Kinesis Data Firehose.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:UpdateLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "ElastiCacheLogging"
        },
        {
            "Sid": "ElastiCacheLoggingFHSLR",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Sid": "ElastiCacheLoggingFH",
            "Action": [
                "firehose:TagDeliveryStream"
            ],
            "Resource": "Amazon Kinesis Data Firehose delivery stream ARN",
            "Effect": "Allow"
        }
    ]
}
```

Menentukan pengiriman log menggunakan Konsol

Menggunakan AWS Management Console Anda dapat membuat cluster Valkey atau Redis OSS (mode cluster dinonaktifkan) dengan mengikuti langkah-langkah di Membuat cluster Valkey (mode cluster dinonaktifkan) (Konsol) atau membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) menggunakan langkah-langkah di. Membuat cluster Valkey atau Redis OSS (mode cluster diaktifkan) (Konsol) Dalam kedua kasus, Anda mengonfigurasi pengiriman log dengan melakukan hal berikut;

1. Di bawah Pengaturan lanjutan, pilih Log dan kemudian periksa Log lambat atau log Mesin.
2. Di bagian Format log, pilih Teks atau JSON.
3. Di bawah Jenis Tujuan, pilih CloudWatch Log atau Kinesis Firehose.
4. Di bagian Tujuan log, pilih Buat baru dan masukkan nama bucket Amazon S3, nama grup CloudWatchLogs log, atau nama aliran Firehose Data Kinesis, atau pilih Pilih yang ada, lalu pilih nama grup Log atau CloudWatch nama aliran Firehose Data Kinesis,

Saat mengubah klaster:

Anda dapat memilih untuk enable/disable mencatat pengiriman atau mengubah jenis tujuan, format, atau tujuan:

1. Masuk ke Konsol dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Dari panel navigasi, pilih cluster Valkey atau cluster Redis OSS.
3. Dari daftar klaster, pilih klaster yang ingin diubah. Pilih Nama klaster, bukan kotak centang di sampingnya.
4. Pada halaman Nama klaster, pilih tab Log.
5. Untuk enable/disable memperlambat log, pilih Aktifkan log lambat atau Nonaktifkan log lambat.
6. Untuk log enable/disable mesin, pilih Aktifkan log mesin atau Nonaktifkan log mesin.
7. Untuk mengubah konfigurasi Anda, pilih Ubah log lambat atau Ubah log mesin:
   - Di bawah Jenis Tujuan, pilih CloudWatch Log atau Kinesis Firehose.
   - Di bawah Tujuan log, pilih Buat baru dan masukkan nama grup CloudWatchLogs log Anda atau nama aliran Kinesis Data Firehose Anda. Atau pilih Pilih yang ada dan kemudian pilih nama grup CloudWatchLogs log Anda atau nama aliran Kinesis Data Firehose Anda.

Menentukan pengiriman log menggunakan AWS CLI

Log Lambat

Buat grup replikasi dengan pengiriman log lambat ke CloudWatch Log.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
    --replication-group-id test-slow-log \
    --replication-group-description test-slow-log \
    --engine redis \
    --cache-node-type cache.r5.large \
    --num-cache-clusters 2 \
    --log-delivery-configurations '{
        "LogType":"slow-log",
        "DestinationType":"cloudwatch-logs",
        "DestinationDetails":{
          "CloudWatchLogsDetails":{
            "LogGroup":"my-log-group"
          }
        },
        "LogFormat":"json"
      }'
```

Untuk Windows:

```
aws elasticache create-replication-group ^
    --replication-group-id test-slow-log ^
    --replication-group-description test-slow-log ^
    --engine redis ^
    --cache-node-type cache.r5.large ^
    --num-cache-clusters 2 ^
    --log-delivery-configurations '{
        "LogType":"slow-log",
        "DestinationType":"cloudwatch-logs",
        "DestinationDetails":{
          "CloudWatchLogsDetails":{
            "LogGroup":"my-log-group"
          }
        },
        "LogFormat":"json"
      }'
```

Ubah grup replikasi untuk mengirimkan log lambat ke CloudWatch Log

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
    --replication-group-id test-slow-log \
    --apply-immediately \
    --log-delivery-configurations '
    {
      "LogType":"slow-log",
      "DestinationType":"cloudwatch-logs",
      "DestinationDetails":{
        "CloudWatchLogsDetails":{

          "LogGroup":"my-log-group"
        }
      },
      "LogFormat":"json"
    }'
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
    --replication-group-id test-slow-log ^
    --apply-immediately ^
    --log-delivery-configurations '
    {
      "LogType":"slow-log",
      "DestinationType":"cloudwatch-logs",
      "DestinationDetails":{
        "CloudWatchLogsDetails":{
          "LogGroup":"my-log-group"
        }
      },
      "LogFormat":"json"
    }'
```

Ubah grup replikasi untuk menonaktifkan pengiriman log lambat

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
    --replication-group-id test-slow-log \
```

```
    --apply-immediately \
    --log-delivery-configurations '
    {
      "LogType":"slow-log",
      "Enabled":false
    }'
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
    --replication-group-id test-slow-log ^
    --apply-immediately ^
    --log-delivery-configurations '
    {
      "LogType":"slow-log",
      "Enabled":false
    }'
```

Log Mesin

Buat grup replikasi dengan pengiriman log mesin ke CloudWatch Log.

Untuk Linux, macOS, atau Unix:

```
aws elasticache create-replication-group \
    --replication-group-id test-slow-log \
    --replication-group-description test-slow-log \
    --engine redis \
    --cache-node-type cache.r5.large \
    --num-cache-clusters 2 \
    --log-delivery-configurations '{
        "LogType":"engine-log",
        "DestinationType":"cloudwatch-logs",
        "DestinationDetails":{
          "CloudWatchLogsDetails":{
            "LogGroup":"my-log-group"
          }
        },
        "LogFormat":"json"
      }'
```

Untuk Windows:

```
aws elasticache create-replication-group ^
    --replication-group-id test-slow-log ^
    --replication-group-description test-slow-log ^
    --engine redis ^
    --cache-node-type cache.r5.large ^
    --num-cache-clusters 2 ^
    --log-delivery-configurations '{
        "LogType":"engine-log",
        "DestinationType":"cloudwatch-logs",
        "DestinationDetails":{
          "CloudWatchLogsDetails":{
            "LogGroup":"my-log-group"
          }
        },
        "LogFormat":"json"
      }'
```

Ubah grup replikasi untuk mengirimkan log engine ke Firehose

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
    --replication-group-id test-slow-log \
    --apply-immediately \
    --log-delivery-configurations '
    {
      "LogType":"engine-log",
      "DestinationType":"kinesis-firehose",
      "DestinationDetails":{
      "KinesisFirehoseDetails":{
         "DeliveryStream":"test"
       }
     },
      "LogFormat":"json"
    }'
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
    --replication-group-id test-slow-log ^
    --apply-immediately ^
    --log-delivery-configurations '
```

```
    {
      "LogType":"engine-log",
      "DestinationType":"kinesis-firehose",
      "DestinationDetails":{
      "KinesisFirehoseDetails":{
         "DeliveryStream":"test"
       }
      },
      "LogFormat":"json"
    }'
```

Ubah grup replikasi untuk beralih ke format mesin

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
    --replication-group-id test-slow-log \
    --apply-immediately \
    --log-delivery-configurations '
    {
       "LogType":"engine-log",
       "LogFormat":"json"
    }'
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
    --replication-group-id test-slow-log ^
    --apply-immediately ^
    --log-delivery-configurations '
    {
       "LogType":"engine-log",
       "LogFormat":"json"
    }'
```

Ubah grup replikasi untuk menonaktifkan pengiriman log mesin

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-replication-group \
    --replication-group-id test-slow-log \
    --apply-immediately \
```

```
    --log-delivery-configurations '
    {
      "LogType":"engine-log",
      "Enabled":false
    }'
```

Untuk Windows:

```
aws elasticache modify-replication-group ^
    --replication-group-id test-slow-log ^
    --apply-immediately ^
    --log-delivery-configurations '
    {
      "LogType":"engine-log",
      "Enabled":false
    }'
```

Pemantauan penggunaan dengan CloudWatch Metrik

ElastiCache menyediakan metrik yang memungkinkan Anda memantau cluster Anda. Anda dapat mengakses metrik ini melalui CloudWatch. Untuk informasi lebih lanjut tentang CloudWatch, lihat CloudWatch dokumentasi.

ElastiCache menyediakan metrik tingkat host (misalnya, penggunaan CPU) dan metrik yang khusus untuk perangkat lunak mesin cache (misalnya, cache mendapat dan kehilangan cache). Metrik ini diukur dan dipublikasikan untuk setiap simpul Cache dalam interval waktu 60 detik.

> **Important**
>
> Anda harus mempertimbangkan untuk menyetel CloudWatch alarm pada metrik kunci tertentu, sehingga Anda akan diberi tahu jika kinerja cluster cache Anda mulai menurun. Untuk informasi selengkapnya, lihat Metrik Apa yang Harus Saya Pantau? dalam panduan ini.

Topik

- Metrik Apa yang Harus Saya Pantau?
- Memilih Statistik dan Periode Metrik
- Pemantauan Metrik CloudWatch Kluster dan Node

Metrik Tingkat Host

Ruang nama `AWS/ElastiCache` mencakup metrik tingkat host berikut untuk simpul cache individual. Metrik ini diukur dan dipublikasikan untuk setiap simpul Cache dalam interval waktu 60 detik.

Lihat Juga

- Metrik untuk Valkey dan Redis OSS

Metrik	Deskripsi	Unit
`CPUUtilization`	Persentase pemanfaatan CPU untuk keseluruhan host. Karena Valkey dan Redis OSS adalah single-threaded, kami sarankan Anda memantau `EngineCPUUtilization` metrik untuk node dengan 4 atau lebih v. CPUs	Persen
`CPUCreditBalance`	Jumlah kredit CPU perolehan yang dikumpulkan oleh instans sejak diluncurkan atau dimulai. Untuk Standar T2, CPUCredit Saldo juga mencakup jumlah kredit peluncuran yang telah diperoleh. Kredit diakumulasi ke saldo kredit setelah diperoleh, dan dihapus dari saldo kredit saat digunakan. Saldo kredit memiliki batas maksimum, yang ditentukan oleh ukuran instans. Setelah batas tercapai, setiap kredit yang baru diperoleh akan dibuang. Untuk T2 Standar, kredit peluncuran tidak termasuk dalam penghitungan batas.	Kredit (Menit vCPU)

Metrik	Deskripsi	Unit
	Kredit dalam CPUCredit Saldo tersedia untuk dibelanjakan untuk melampaui pemanfaatan CPU dasarnya. Metrik kredit CPU tersedia hanya dalam frekuensi lima menit. Metrik ini tidak tersedia untuk instans performa yang dapat melonjak T2.	
`CPUCreditUsage`	Jumlah kredit CPU yang digunakan oleh instans untuk pemanfaatan CPU. Satu kredit CPU sama dengan satu vCPU yang berjalan pada pemanfaatan 100% selama satu menit atau kombinasi yang setara dari CPUs v, pemanfaatan, dan waktu (misalnya, satu vCPU berjalan pada pemanfaatan 50% selama dua menit atau dua CPUs v berjalan pada pemanfaatan 25% selama dua menit). Metrik kredit CPU tersedia hanya dalam frekuensi lima menit. Jika Anda menentukan periode lebih dari lima menit, gunakan statistik Sum, bukan statistik Average. Metrik ini tidak tersedia untuk instans performa yang dapat melonjak T2.	Kredit (Menit vCPU)
`FreeableMemory`	Jumlah memori kosong yang tersedia di host. Angka ini berasal dari RAM, buffer, dan cache yang dilaporkan oleh OS sebagai memori yang dapat dibebaskan.	Byte
`NetworkBytesIn`	Jumlah byte yang telah dibaca oleh host dari jaringan.	Byte

Metrik	Deskripsi	Unit
NetworkBytesOut	Jumlah byte yang dikirimkan ke semua antarmuka jaringan oleh instans.	Byte
NetworkPacketsIn	Jumlah paket yang diterima di semua antarmuka jaringan oleh instans. Metrik ini mengidentifikasi volume lalu lintas masuk dari segi jumlah paket pada satu instans tunggal.	Hitungan
NetworkPacketsOut	Jumlah paket yang dikirimkan di semua antarmuka jaringan oleh instans. Metrik ini mengidentifikasi volume lalu lintas keluar dari segi jumlah paket pada satu instans tunggal.	Hitungan
NetworkBandwidthInAllowanceExceeded	Jumlah paket antri atau dijatuhkan karena kumpulan bandwidth yang masuk melebihi maksimum untuk instans.	Hitungan
NetworkConntrackAllowanceExceeded	Jumlah paket turun karena pelacakan koneksi melebihi maksimum untuk instans dan koneksi baru tidak dapat dibuat. Hal ini dapat mengakibatkan hilangnya paket untuk lalu lintas ke atau dari instans.	Hitungan
NetworkBandwidthOutAllowanceExceeded	Jumlah paket antre atau dijatuhkan karena bandwidth agregat yang keluar melebihi maksimum untuk instans.	Hitungan
NetworkPacketsPerSecondAllowanceExceeded	Jumlah paket yang diantrekan atau dijatuhkan karena paket per detik dua arah melebihi maksimum untuk instans.	Hitungan
NetworkMaxBytesIn	Semburan maksimum per detik byte yang diterima dalam setiap menit.	Byte
NetworkMaxBytesOut	Semburan maksimum per detik byte yang ditransmisikan dalam setiap menit.	Byte

Metrik	Deskripsi	Unit
NetworkMaxPacketsIn	Semburan maksimum per detik menerima paket dalam setiap menit.	Hitungan
NetworkMaxPacketsOut	Semburan maksimum per detik paket yang ditransmisikan dalam setiap menit.	Hitungan
SwapUsage	Jumlah swap yang digunakan pada host.	Byte

Metrik untuk Valkey dan Redis OSS

Amazon ElastiCacheNamespace mencakup metrik Valkey dan Redis OSS berikut. Metrik ini sama saat menggunakan mesin Valkey.

Dengan pengecualianReplicationLag,, EngineCPUUtilizationSuccessfulWriteRequestLatency, danSuccessfulReadRequestLatency, metrik ini berasal dari info perintah. Setiap metrik dihitung di tingkat simpul cache.

Untuk dokumentasi lengkap info perintah, lihat http://valkey. io/commands/info.

Lihat Juga

- Metrik Tingkat Host

Metrik	Deskripsi	Unit
ActiveDefragHits	Jumlah realokasi nilai per menit yang dilakukan oleh proses defragmentasi aktif. Ini berasal dari active_defrag_hits statistik di INFO.	Bilangan
AuthenticationFailures	Jumlah total upaya yang gagal untuk mengautentikasi ke Valkey atau Redis OSS menggunakan perintah AUTH. Anda dapat menemukan informasi selengkapnya tentang setiap kegagalan autentikasi menggunakan perintah ACL LOG. Sebaiknya atur peringatan	Hitungan

Metrik	Deskripsi	Unit
	untuk hal ini guna mendeteksi percobaan akses yang tidak sah.	
BytesUsedForCache	Jumlah total byte yang dialokasikan oleh Valkey atau Redis OSS untuk semua tujuan, termasuk dataset, buffer, dan sebagainya.	Byte
	Dimension: Tier=Memory untuk cluster Valkey atau Redis OSS menggunakanTingkatan data di ElastiCache: Jumlah total byte yang digunakan untuk cache oleh memori. Ini adalah nilai used_memory statistik di INFO.	Byte
	Dimension: Tier=SSD untuk cluster Valkey atau Redis OSS menggunakanTingkatan data di ElastiCache: Jumlah total byte yang digunakan untuk cache oleh SSD.	Byte
BytesReadFromDisk	Jumlah total byte yang dibaca dari disk per menit. Didukung hanya untuk klaster yang menggunakan Tingkatan data di ElastiCache.	Byte
BytesWrittenToDisk	Total jumlah byte yang ditulis ke disk per menit. Didukung hanya untuk klaster yang menggunakan Tingkatan data di ElastiCache.	Byte
CacheHits	Jumlah pencarian kunci hanya-baca yang berhasil di kamus utama. Ini berasal dari keyspace_hits statistik di INFO.	Hitungan
CacheMisses	Jumlah pencarian kunci hanya-baca yang tidak berhasil di kamus utama. Ini berasal dari keyspace_misses statistik di INFO.	Hitungan

Metrik	Deskripsi	Unit
`CommandAuthorizationFailures`	Jumlah seluruh percobaan pengguna yang gagal untuk menjalankan perintah karena tidak memiliki izin untuk memanggil perintah itu. Anda dapat menemukan informasi selengkapnya tentang setiap kegagalan autentikasi menggunakan perintah ACL LOG. Sebaiknya atur peringatan untuk hal ini guna mendeteksi percobaan akses yang tidak sah.	Hitungan
`CacheHitRate`	Menunjukkan efisiensi penggunaan instance Valkey atau Redis OSS. Jika rasio cache lebih rendah dari sekitar 0,8, hal ini berarti bahwa sejumlah besar kunci telah dikosongkan, kedaluwarsa, atau tidak ada. Rasio ini dihitung menggunakan statistik `cache_hits` dan `cache_misses` dengan cara berikut: `cache_hits /(cache_hits + cache_misses)` .	Persen
`ChannelAuthorizationFailures`	Jumlah seluruh percobaan pengguna yang gagal untuk mengakses saluran akses karena tidak mempunyai izin akses. Anda dapat menemukan informasi selengkapnya tentang setiap kegagalan autentikasi menggunakan perintah ACL LOG. Sebaiknya atur peringatan untuk metrik ini guna mendeteksi percobaan akses yang tidak sah.	Hitungan
`CurrConnections`	Jumlah koneksi klien, tidak termasuk koneksi dari replika baca. ElastiCache menggunakan 4 hingga 6 koneksi untuk memantau cluster dalam setiap kasus. Ini berasal dari `connected_clients` statistik di INFO.	Hitungan

Metrik	Deskripsi	Unit
CurrItems	Jumlah item dalam cache. Ini berasal dari keyspace statistik, menjumlahkan semua kunci di seluruh ruang kunci.	Hitungan
	Dimension: Tier=Memory untuk klaster menggunakan Tingkatan data di ElastiCache. Jumlah item dalam memori.	Hitungan
	Dimension: Tier=SSD (solid state drive) untuk klaster yang menggunakan Tingkatan data di ElastiCache. Jumlah item dalam SSD.	Hitungan
CurrVolatileItems	Jumlah total kunci di semua basis data yang memiliki set ttl. Ini berasal dari expires statistik, menjumlahkan semua kunci dengan set ttl di seluruh ruang kunci.	Hitungan
DatabaseCapacityUsagePercentage	Persentase kapasitas data total untuk klaster yang sedang digunakan. Pada instance Data Tiered, metrik dihitung sebagai(used_memory - mem_not_counted_for_evict + SSD used) / (maxmemory + SSD total capacity), di mana used_memory dan maxmemory diambil dari INFO. Dalam semua kasus lain, metrik dihitung menggunakanused_memory/maxmemory.	Persen

Metrik	Deskripsi	Unit

`DatabaseCapacityUsageCountedForEvictPercentage`	Persentase kapasitas data total untuk klaster yang sedang digunakan, tidak termasuk memori yang digunakan untuk overhead dan COB. Metrik ini dihitung sebagai:
	
	`used_memory - mem_not_counted_for_evict/maxmemory`
	
	Pada instans Bertingkatan Data, metrik dihitung sebagai berikut:
	
	`(used_memory + SSD used) / (maxmemory + SSD total capacity)`
	
	Dimana `used_memory` dan `maxmemory` diambil dari Info
		Persen

`DatabaseMemoryUsagePercentage`	Persentase memori untuk klaster yang sedang digunakan. Ini dihitung menggunakan `used_memory/maxmemory` dari INFO.	Persen
`DatabaseMemoryUsageCountedForEvictPercentage`	Persentase memori untuk klaster yang sedang digunakan, tidak termasuk memori yang digunakan untuk overhead dan COB. Ini dihitung menggunakan `used_memory-mem_not_counted_for_evict/maxmemory` dari INFO.	Persen

Metrik	Deskripsi	Unit
`DB0AverageTTL`	Mengekspos `avg_ttl` DBO dari `keyspace` statistik perintah INFO. Replika tidak menghenti kan masa berlaku kunci. Sebaliknya, replika menunggu simpul primer untuk menghenti kan masa berlaku kunci. Ketika simpul primer menghentikan masa berlaku kunci (atau mengosongkannya karena LRU), simpul tersebut mensintesis perintah `DEL`, yang ditransmisikan ke semua replika. Oleh karena itu, DB0AverageTTL bernilai 0 untuk simpul replika, karena replika tidak membuat kunci kedaluwarsa sehingga tidak melacak TTL.	Milidetik

Metrik	Deskripsi	Unit
`EngineCPUUtilization`	Menyediakan pemanfaatan CPU dari benang mesin Valkey atau Redis OSS. Karena Valkey dan Redis OSS adalah single-threaded, Anda dapat menggunakan metrik ini untuk menganalisis beban proses itu sendiri. `EngineCPUUtilization` Metrik memberikan visibilitas proses yang lebih tepat. Anda dapat menggunakannya dalam hubungannya dengan metrik `CPUUtilization` . `CPUUtilization` mengungkapkan pemanfaatan CPU untuk instans server secara keseluruhan, termasuk sistem operasi lain dan proses manajemen. Untuk tipe node yang lebih besar dengan empat v CPUs atau lebih, gunakan `EngineCPUUtilization` metrik untuk memantau dan menetapkan ambang batas untuk penskalaan. **Note** Pada ElastiCache host, proses latar belakang memantau host untuk memberikan pengalaman database terkelola. Proses latar belakang ini dapat menimbulkan beban kerja CPU yang cukup besar. Ini tidak signifikan pada host yang lebih besar dengan lebih dari dua vCPUs. Tapi itu dapat mempengaruhi host yang lebih kecil dengan 2v CPUs atau lebih sedikit. Jika Anda hanya memantau `EngineCPUUtilization` metrik, Anda tidak akan menyadari situasi di mana host kelebihan beban dengan penggunaan CPU tinggi dari Valkey atau Redis	Persen

Metrik	Deskripsi	Unit
	OSS dan penggunaan CPU yang tinggi dari proses pemantauan latar belakang. Oleh karena itu, kami merekomendasikan pemantauan `CPUUtilization` metrik untuk host dengan dua v CPUs atau kurang.	
`Evictions`	Jumlah kunci yang telah dikosongkan karena batas `maxmemory` . Ini berasal dari `evicted_keys` statistik di INFO.	Hitungan
`GlobalDatastoreReplicationLag`	Ini adalah lag antara simpul primer dari Wilayah sekunder dan simpul primer dari Wilayah primer. Untuk mode cluster yang diaktifkan Valkey atau Redis OSS, lag menunjukkan penundaan maksimum di antara pecahan.	Detik
`IamAuthenticationExpirations`	Jumlah total koneksi Valkey atau Redis OSS yang diautentikasi IAM yang kedaluwarsa. Anda dapat menemukan informasi selengkapnya tentang Autentikasi dengan IAM dalam panduan pengguna.	Hitungan
`IamAuthenticationThrottling`	Jumlah total permintaan Valkey atau Redis OSS AUTH atau HELLO yang diautentikasi IAM yang dibatasi. Anda dapat menemukan informasi selengkapnya tentang Autentikasi dengan IAM dalam panduan pengguna.	Hitungan
`IsMaster`	Menunjukkan apakah simpul tersebut adalah simpul primer dari serpihan/klaster saat ini. Metrik ini dapat bernilai 0 (bukan primer) atau 1 (primer).	Hitungan

Metrik	Deskripsi	Unit
KeyAuthorizationFailures	Jumlah seluruh percobaan pengguna yang gagal untuk mengakses kunci akses karena tidak mempunyai izin akses. Anda dapat menemukan informasi selengkapnya tentang setiap kegagalan autentikasi menggunakan perintah ACL LOG. Sebaiknya atur peringatan untuk hal ini guna mendeteksi percobaan akses yang tidak sah.	Hitungan
KeysTracked	Jumlah kunci yang dilacak oleh Valkey atau Redis OSS key tracking sebagai persentase. tracking-table-max-keys Pelacakan kunci digunakan untuk membantu caching sisi klien dan memberitahukan klien jika kunci diubah.	Hitungan
MemoryFragmentationRatio	Menunjukkan efisiensi dalam alokasi memori mesin Valkey atau Redis OSS. Ambang batas tertentu menandakan perilaku yang berbeda. Nilai yang disarankan adalah memiliki fragmentasi di atas 1,0. Ini dihitung mem_fragmentation_ratio statistic dari INFO.	Bilangan

Metrik	Deskripsi	Unit
`NewConnections`	Jumlah seluruh koneksi yang telah diterima oleh server selama periode ini. Ini berasal dari `total_connections_received` statistik di INFO. **Note** Jika Anda menggunakan ElastiCache Redis OSS versi 5 atau lebih rendah, antara dua dan empat koneksi yang dilaporkan oleh metrik ini digunakan oleh ElastiCache untuk memantau cluster. Namun, ketika menggunakan ElastiCache untuk Redis OSS versi 6 atau lebih tinggi, koneksi yang digunakan oleh ElastiCache untuk memantau cluster tidak termasuk dalam metrik ini.	Hitungan
`NumItemsReadFromDisk`	Jumlah total item yang diambil dari disk per menit. Didukung hanya untuk klaster yang menggunakan Tingkatan data di ElastiCache.	Hitungan
`NumItemsWrittenToDisk`	Jumlah total item yang ditulis ke disk per menit. Didukung hanya untuk klaster yang menggunakan Tingkatan data di ElastiCache.	Hitungan
`MasterLinkHealthStatus`	Status ini memiliki dua nilai: 0 atau 1. Nilai 0 menunjukkan bahwa data di node ElastiCache utama tidak sinkron dengan Valkey atau Redis OSS aktif. EC2 Nilai 1 menunjukkan bahwa data sudah sinkron. Untuk menyelesaikan migrasi, gunakan operasi CompleteMigrationAPI.	Boolean

Metrik	Deskripsi	Unit
`Reclaimed`	Jumlah seluruh peristiwa kedaluwarsa kunci. Ini berasal dari `expired_keys` statistik di INFO.	Hitungan
`ReplicationBytes`	Untuk simpul dalam konfigurasi yang direplikasi, `ReplicationBytes` melaporkan jumlah byte yang dikirimkan oleh primer ke semua replikanya. Metrik ini merepresentasikan beban tulis pada grup replikasi. Ini berasal dari `master_repl_offset` statistik di INFO.	Byte
`ReplicationLag`	Metrik ini hanya berlaku untuk simpul yang berjalan sebagai replika baca. Hal ini menunjukkan seberapa jauh ketinggalan, dalam detik, suatu replika dalam menerapkan perubahan dari simpul primer. Untuk Valkey 7.2 dan seterusnya, dan Redis OSS 5.0.6 dan seterusnya, lag dapat diukur dalam milidetik.	Detik
`SaveInProgress`	Metrik biner ini menghasilkan 1 jika penyimpanan latar belakang (bercabang atau tak bercabang) sedang berlangsung, dan 0 jika sebaliknya. Proses simpan di latar belakang biasanya digunakan selama snapshot dan sinkronisasi. Operasi ini dapat menyebabkan performa menurun. Menggunakan metrik `SaveInProgress`, Anda dapat mendiagnosis apakah performa yang menurun ini disebabkan oleh proses penyimpanan di latar belakang. Ini berasal dari `rdb_bgsave_in_progress` statistik di INFO.	Boolean

Metrik	Deskripsi	Unit
`TrafficManagementActive`	Menunjukkan apakah ElastiCache untuk Redis OSS secara aktif mengelola lalu lintas dengan menyesuaikan lalu lintas yang dialokasikan ke perintah masuk, pemantauan atau replikasi. Lalu lintas dikelola ketika lebih banyak perintah dikirim ke node daripada yang dapat diproses oleh Valkey atau Redis OSS dan digunakan untuk menjaga stabilitas dan pengoperasian mesin yang optimal. Setiap titik data 1 dapat menunjukkan bahwa simpul diskalakan untuk beban kerja yang disediakan. **Note** Jika metrik ini tetap aktif, evaluasi klaster untuk memutuskan apakah penaikan skala atau penskalaan keluar diperlukan. Metrik terkait termasuk `NetworkBandwidthOutAllowanceExceeded` dan `EngineCPUUtilization` .	Boolean
`SuccessfulWriteRequestLatency`	Latensi permintaan tulis yang berhasil. Statistik yang valid: Rata-rata, Jumlah, Min, Maks, Jumlah Sampel, setiap persentil antara p0 dan p100. Jumlah sampel hanya mencakup perintah yang berhasil dieksekusi.	Mikrodetik
`SuccessfulReadRequestLatency`	Latensi permintaan baca yang berhasil. Statistik yang valid: Rata-rata, Jumlah, Min, Maks, Jumlah Sampel, setiap persentil antara p0 dan p100. Jumlah sampel hanya mencakup perintah yang berhasil dieksekusi.	Mikrodetik

Metrik	Deskripsi	Unit
`ErrorCount`	Jumlah total perintah yang gagal selama periode waktu yang ditentukan. Statistik yang valid: Rata-rata, Jumlah, Min, Maks	Hitungan

CPUUtilization Ketersediaan mesin

AWS Wilayah yang tercantum berikut ini tersedia di semua jenis node yang didukung.

Wilayah	Nama wilayah
`us-east-2`	AS Timur (Ohio)
`us-east-1`	AS Timur (Virginia Utara)
`us-west-1`	AS Barat (California Utara)
`us-west-2`	US West (Oregon)
`ap-northeast-1`	Asia Pacific (Tokyo)
`ap-northeast-2`	Asia Pasifik (Seoul)
`ap-northeast-3`	Asia Pasifik (Osaka)
`ap-east-1`	Asia Pasifik (Hong Kong)
`ap-south-1`	Asia Pasifik (Mumbai)
`ap-southeast-1`	Asia Pasifik (Singapura)
`ap-southeast-2`	Asia Pasifik (Sydney)
`ap-southeast-3`	Asia Pasifik (Jakarta)
`ca-central-1`	Kanada (Pusat)

Wilayah	Nama wilayah
cn-north-1	Tiongkok (Beijing)
cn-northwest-2	China (Ningxia)
me-south-1	Timur Tengah (Bahrain)
eu-central-1	Eropa (Frankfurt)
eu-west-1	Eropa (Irlandia)
eu-west-2	Eropa (London)
eu-west-3	Eropa (Paris)
eu-south-1	Eropa (Milan)
af-south-1	Afrika (Cape Town)
eu-north-1	Eropa (Stockholm)
sa-east-1	Amerika Selatan (Sao Paulo)
us-gov-west-1	AWS GovCloud (AS-Barat)
us-gov-east-1	AWS GovCloud (AS-Timur)

Berikut ini adalah kumpulan jenis perintah tertentu, yang berasal dari info commandstats. Bagian commandstats menyediakan statistik berdasarkan jenis perintah, termasuk jumlah panggilan, jumlah waktu CPU yang dikonsumsi oleh perintah ini, dan CPU rata-rata yang dikonsumsi per eksekusi perintah. Untuk setiap jenis perintah, baris berikut ditambahkan: `cmdstat_XXX: calls=XXX,usec=XXX,usec_per_call=XXX`.

Metrik latensi yang tercantum berikut dihitung menggunakan statistik commandstats dari INFO. Metrik dihitung dengan cara berikut: `delta(usec)/delta(calls)`. `delta` dihitung sebagai perbedaan dalam satu menit. Latensi didefinisikan sebagai waktu CPU yang dibutuhkan ElastiCache untuk memproses perintah. Perhatikan bahwa untuk klaster yang menggunakan tingkatan data, waktu yang dibutuhkan untuk mengambil item dari SSD tidak termasuk dalam pengukuran ini.

Untuk daftar lengkap perintah yang tersedia, lihat perintah dalam dokumentasi Valkey.

Metrik	Deskripsi	Unit
`ClusterBasedCmds`	Jumlah seluruh perintah yang berbasis klaster. Ini berasal dari `commandstats` statistik dengan menjumlahkan semua perintah yang bertindak atas cluster (`cluster slot`,`cluster info`, dan seterusnya).	Hitungan
`ClusterBasedCmdsLatency`	Latensi perintah berbasis klaster.	Mikrodetik
`EvalBasedCmds`	Jumlah seluruh perintah untuk perintah berbasis eval. Ini berasal dari `commandstats` statistik dengan menjumlahkaneval,evalsha.	Hitungan
`EvalBasedCmdsLatency`	Latensi perintah berbasis eval.	Mikrodetik
`GeoSpatialBasedCmds`	Jumlah seluruh perintah untuk perintah berbasis geospasial. Ini berasal dari `commandstats` statistik. Ini diperoleh dengan menjumlahkan semua perintah jenis geo: geoadd, geodist, geohash, geopos, georadius, dan georadiusbymember.	Hitungan
`GeoSpatialBasedCmdsLatency`	Latensi perintah berbasis geospasial.	Mikrodetik
`GetTypeCmds`	Jumlah seluruh perintah jenis read-only. Ini berasal dari `commandstats` statistik dengan menjumlahkan semua perintah read-only tipe (get,, hgetscard,lrange, dan sebagainya.)	Hitungan
`GetTypeCmdsLatency`	Latensi perintah baca.	Mikrodetik
`HashBasedCmds`	Jumlah seluruh perintah yang berbasis hash. Ini berasal dari `commandstats` statistik dengan menjumlahkan semua perintah	Hitungan

Metrik	Deskripsi	Unit
	yang bertindak atas satu atau lebih hash (hget,,hkeys, hvalshdel, dan seterusnya).	
HashBasedCmdsLatency	Latensi perintah berbasis hash.	Mikrodetik
HyperLogLogBasedCmds	Jumlah seluruh perintah berbasis HyperLogLog . Ini berasal dari commandstats statistik dengan menjumlahkan semua pf jenis perintah (pfadd,, pfcountpfmerge, dan sebagainya.).	Hitungan
HyperLogLogBasedCmdsLatency	Latensi perintah HyperLogLog berbasis.	Mikrodetik
JsonBasedCmds	Jumlah total perintah JSON, termasuk perintah baca dan tulis. Ini berasal dari commandstats statistik dengan menjumlahkan semua perintah JSON yang bertindak atas kunci JSON.	Hitungan
JsonBasedCmdsLatency	Latensi semua perintah JSON, termasuk perintah baca dan tulis.	Mikrodetik
JsonBasedGetCmds	Jumlah seluruh perintah JSON hanya-baca. Ini berasal dari commandstats statistik dengan menjumlahkan semua perintah baca JSON yang bertindak atas kunci JSON.	Hitungan
JsonBasedGetCmdsLatency	Latensi perintah hanya-baca JSON.	Mikrodetik
JsonBasedSetCmds	Jumlah seluruh perintah JSON jenis tulis. Ini berasal dari commandstats statistik dengan menjumlahkan semua perintah tulis JSON yang bertindak atas kunci JSON.	Hitungan
JsonBasedSetCmdsLatency	Latensi perintah tulis JSON.	Mikrodetik

Metrik	Deskripsi	Unit
KeyBasedCmds	Jumlah seluruh perintah yang berbasis kunci. Ini berasal dari commandstats statistik dengan menjumlahkan semua perintah yang bertindak atas satu atau lebih kunci di beberapa struktur data (del,expire,rename, dan seterusnya.).	Hitungan
KeyBasedCmdsLatency	Latensi perintah berbasis kunci.	Mikrodetik
ListBasedCmds	Jumlah seluruh perintah yang berbasis daftar. Ini berasal dari commandstats statistik dengan menjumlahkan semua perintah yang bertindak atas satu atau lebih daftar (lindex,lrange,lpush,ltrim, dan seterusnya).	Hitungan
ListBasedCmdsLatency	Latensi perintah berbasis daftar.	Mikrodetik
NonKeyTypeCmds	Jumlah seluruh perintah yang tidak berbasis kunci. Ini berasal dari commandstats statistik dengan menjumlahkan semua perintah yang tidak bertindak atas kunci, misalnyaacl, dbsize atauinfo.	Hitungan
NonKeyTypeCmdsLatency	Latensi non-key-based perintah.	Mikrodetik
PubSubBasedCmds	Jumlah total perintah untuk pub/sub fungsionalitas. Ini berasal dari commandstats statistik dengan menjumlahkan semua perintah yang digunakan untuk pub/sub fungsionalitas:psubscribe,publish,pubsub,punsubscribe,ssubscribe,sunsubscribe, spublishsubscribe, danunsubscribe.	Hitungan

Metrik	Deskripsi	Unit
PubSubBasedCmdsLatency	Latensi perintah berbasis pub/sub.	Mikrodetik
SetBasedCmds	Jumlah seluruh perintah yang berbasis set. Ini berasal dari commandstats statistik dengan menjumlahkan semua perintah yang bertindak atas satu atau lebih set (scard,sdiff,sadd,sunion, dan seterusnya).	Hitungan
SetBasedCmdsLatency	Latensi perintah berbasis set.	Mikrodetik
SetTypeCmds	Jumlah seluruh perintah jenis write. Ini berasal dari commandstats statistik dengan menjumlahkan semua mutative jenis perintah yang beroperasi pada data (set,,hset, saddlpop, dan sebagainya.)	Hitungan
SetTypeCmdsLatency	Latensi perintah tulis.	Mikrodetik
SortedSetBasedCmds	Jumlah seluruh perintah yang berbasis sorted set. Ini berasal dari commandstats statistik dengan menjumlahkan semua perintah yang bertindak atas satu atau lebih set yang diurutkan (zcount,,zrange, zrankzadd, dan seterusnya).	Hitungan
SortedSetBasedCmdsLatency	Latensi perintah berbasis urutan.	Mikrodetik
StringBasedCmds	Jumlah seluruh perintah yang berbasis string. Ini berasal dari commandstats statistik dengan menjumlahkan semua perintah yang bertindak atas satu atau lebih string (strlen,, setexsetrange, dan seterusnya).	Hitungan

Metrik	Deskripsi	Unit
StringBasedCmdsLatency	Latensi perintah berbasis string.	Mikrodetik
StreamBasedCmds	Jumlah seluruh perintah yang berbasis aliran. Ini berasal dari commandstats statistik dengan menjumlahkan semua perintah yang bertindak atas satu atau lebih jenis data aliran (xrange,,xlen, xaddxdel, dan sebagainya).	Hitungan
StreamBasedCmdsLatency	Latensi perintah berbasis aliran.	Mikrodetik

Metrik untuk Memcached

Ruang nama AWS/ElastiCache mencakup metrik Redis berikut.

ElastiCache Namespace AWS/mencakup metrik berikut yang diturunkan dari perintah statistik Memcached. Setiap metrik dihitung di tingkat simpul cache.

Lihat juga

- Metrik Tingkat Host

Metrik	Deskripsi	Unit
BytesReadIntoMemcached	Jumlah byte yang telah dibaca dari jaringan oleh simpul cache.	Byte
BytesUsedForCacheItems	Jumlah byte yang digunakan untuk menyimpan item cache.	Byte
BytesWrittenOutFromMemcached	Jumlah byte yang telah ditulis ke jaringan oleh simpul cache.	Byte

Metrik	Deskripsi	Unit
CasBadval	Jumlah permintaan CAS (check and set) yang telah diterima oleh cache di mana nilai CAS tidak cocok dengan nilai CAS yang tersimpan.	Hitungan
CasHits	Jumlah permintaan CAS yang telah diterima oleh cache di mana kunci yang diminta ditemukan dan nilai CAS cocok.	Hitungan
CasMisses	Jumlah permintaan CAS yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Hitungan
CmdFlush	Jumlah perintah flush yang telah diterima oleh cache.	Hitungan
CmdGet	Jumlah perintah get yang telah diterima oleh cache.	Hitungan
CmdSet	Jumlah perintah set yang telah diterima oleh cache.	Hitungan
CurrConnections	Hitungan jumlah koneksi yang terhubung ke cache dalam sekejap. ElastiCache menggunakan 2 hingga 3 koneksi untuk memantau cluster. Selain yang disebutkan di atas, Memcached membuat sejumlah koneksi internal yang setara dengan dua kali jumlah thread yang digunakan untuk jenis simpul. Hitungan thread untuk berbagai jenis simpul dapat dilihat dalam `Nodetype Specific Parameters` dari Grup Parameter yang berlaku. Koneksi total adalah jumlah dari koneksi klien, koneksi untuk pemantauan, dan koneksi internal yang disebutkan di atas.	Hitungan

Metrik	Deskripsi	Unit
CurrItems	Hitungan jumlah item yang saat ini disimpan dalam cache.	Hitungan
DecrHits	Jumlah permintaan decrement yang telah diterima oleh cache di mana kunci yang diminta ditemukan.	Hitungan
DecrMisses	Jumlah permintaan decrement yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Hitungan
DeleteHits	Jumlah permintaan delete yang telah diterima oleh cache di mana kunci yang diminta ditemukan.	Hitungan
DeleteMisses	Jumlah permintaan delete yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Hitungan
Evictions	Jumlah item belum kedaluwarsa yang dikosongkan oleh cache untuk memberikan ruang bagi penulisan baru.	Hitungan
GetHits	Jumlah permintaan get yang telah diterima oleh cache di mana kunci yang diminta ditemukan.	Hitungan
GetMisses	Jumlah permintaan get yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Hitungan
IncrHits	Jumlah permintaan increment yang telah diterima oleh cache di mana kunci yang diminta ditemukan.	Hitungan
IncrMisses	Jumlah permintaan increment yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Hitungan

Metrik	Deskripsi	Unit
Reclaimed	Jumlah item kedaluwarsa yang dikosongkan oleh cache untuk memberikan ruang bagi penulisan baru.	Hitungan

Untuk Memcached 1.4.14, tersedia metrik tambahan berikut.

Metrik	Deskripsi	Unit
BytesUsedForHash	Jumlah byte yang saat ini digunakan oleh tabel hash.	Byte
CmdConfigGet	Jumlah kumulatif permintaan config get.	Hitungan
CmdConfigSet	Jumlah kumulatif permintaan config set.	Hitungan
CmdTouch	Jumlah kumulatif permintaan touch.	Hitungan
CurrConfig	Jumlah konfigurasi tersimpan saat ini.	Hitungan
EvictedUnfetched	Jumlah item valid yang dikosongkan dari cache least recently used cache (LRU) yang tidak pernah di-touch setelah ditetapkan.	Hitungan
ExpiredUnfetched	Jumlah item kedaluwarsa yang diklaim ulang dari LRU yang tidak pernah di-touch setelah ditetapkan.	Hitungan
SlabsMoved	Jumlah keseluruhan slab page yang telah dipindahkan.	Hitungan
TouchHits	Jumlah kunci yang telah di-touch dan diberi waktu kedaluwarsa yang baru.	Hitungan
TouchMisses	Jumlah item yang telah di-touch, tetapi tidak ditemukan.	Hitungan

ElastiCache Namespace AWS/mencakup metrik tingkat cache terhitung berikut.

Metrik	Deskripsi	Unit
`NewConnections`	Jumlah koneksi baru yang telah diterima oleh cache. Hal ini berasal dari statistik total_connections Memcached dengan mencatat perubahan pada total_connections selama suatu periode waktu. Ini akan selalu setidaknya 1, karena koneksi disediakan untuk a ElastiCache.	Hitungan
`NewItems`	Jumlah item baru yang telah disimpan oleh cache. Hal ini berasal dari statistik total_items Memcached dengan mencatat perubahan pada total_items selama suatu periode waktu.	Hitungan
`UnusedMemory`	Jumlah memori yang tidak digunakan oleh data. Hal ini berasal dari statistik limit_max bytes dan bytes Memcached dengan mengurangkan bytes dari limit_maxbytes. Karena Memcached overhead menggunakan memori selain yang digunakan oleh data, tidak UnusedMemory boleh dianggap sebagai jumlah memori yang tersedia untuk data tambahan. Anda mungkin mengalami pengosongan meskipun Anda masih memiliki memori yang tidak terpakai. Untuk informasi yang lebih mendetail, lihat Memcached item memory usage.	Byte

Metrik Apa yang Harus Saya Pantau?

CloudWatch Metrik berikut menawarkan wawasan yang baik tentang ElastiCache kinerja. Dalam kebanyakan kasus, kami menyarankan Anda menyetel CloudWatch alarm untuk metrik ini sehingga Anda dapat mengambil tindakan korektif sebelum masalah kinerja terjadi.

Metrik yang Perlu Dipantau

- CPUUtilization
- Mesin CPUUtilization
- SwapUsage (Valkey dan Redis OSS)
- Evictions
- CurrConnections
- Memori (Valkey dan Redis OSS)
- Jaringan
- Latensi
- Replikasi
- Manajemen Lalu Lintas (Valkey dan Redis OSS)

CPUUtilization

Ini adalah metrik tingkat host yang dilaporkan sebagai persentase. Untuk informasi selengkapnya, lihat Metrik Tingkat Host.

Valkey dan Redis OSS

Untuk tipe node yang lebih kecil dengan 2v CPUs atau kurang, gunakan `CPUUtilization` metrik untuk memantau beban kerja Anda.

Secara umum, sebaiknya atur ambang batas Anda sebesar 90% dari CPU yang tersedia. Karena Valkey dan Redis OSS keduanya single-threaded, nilai ambang sebenarnya harus dihitung sebagai sebagian kecil dari total kapasitas node. Sebagai contoh, misalkan Anda menggunakan jenis simpul yang memiliki dua inti. Dalam hal ini, ambang batas untuk CPUUtilization adalah 90/2, atau 45%.

Anda akan perlu menentukan ambang batas Anda sendiri, berdasarkan jumlah inti pada simpul cache yang Anda gunakan. Jika Anda melampaui ambang batas ini, dan beban kerja utama Anda

berasal dari permintaan baca, skalakan keluar klaster cache Anda dengan menambahkan replika baca. Jika beban kerja utama dari permintaan tulis, bergantung pada konfigurasi klaster Anda, sebaiknya Anda:

- Cluster Valkey atau Redis OSS (mode cluster dinonaktifkan): tingkatkan dengan menggunakan jenis instance cache yang lebih besar.
- Cluster Valkey atau Redis OSS (mode cluster enabled): tambahkan lebih banyak pecahan untuk mendistribusikan beban kerja tulis di lebih banyak node primer.

> **Tip**
>
> Alih-alih menggunakan metrik Host-LevelCPUUtilization, pengguna Valkey dan Redis OSS mungkin dapat menggunakan metrikEngineCPUUtilization, yang melaporkan persentase penggunaan pada inti mesin Valkey atau Redis OSS. Untuk melihat apakah metrik ini tersedia di node Anda dan untuk informasi selengkapnya, lihat Metrik untuk Valkey dan Redis OSS.

Untuk tipe node yang lebih besar dengan 4v CPUs atau lebih, Anda mungkin ingin menggunakan `EngineCPUUtilization` metrik, yang melaporkan persentase penggunaan pada inti mesin Valkey atau Redis OSS. Untuk melihat apakah metrik ini tersedia di node Anda dan untuk informasi selengkapnya, lihat Metrik untuk Redis OSS.

Memcache

Karena Memcached bersifat multi-thread, metrik ini dapat mencapai 90%. Jika Anda melebihi ambang batas ini, skala cluster cache Anda dengan menggunakan jenis node cache yang lebih besar atau skala dengan menambahkan lebih banyak node cache.

Mesin CPUUtilization

Untuk tipe node yang lebih besar dengan 4v CPUs atau lebih, Anda mungkin ingin menggunakan `EngineCPUUtilization` metrik, yang melaporkan persentase penggunaan pada inti mesin Redis OSS. Untuk melihat apakah metrik ini tersedia di node Anda dan untuk informasi selengkapnya, lihat Metrik untuk Valkey dan Redis OSS.

Untuk informasi selengkapnya, lihat CPUsbagian di Memantau praktik terbaik dengan Amazon ElastiCache untuk Redis OSS menggunakan Amazon. CloudWatch

SwapUsage (Valkey dan Redis OSS)

Ini adalah metrik tingkat host yang dilaporkan dalam byte. Untuk informasi selengkapnya, lihat Metrik Tingkat Host.

`FreeableMemory` CloudWatch Metrik yang mendekati 0 (yaitu, di bawah 100MB) atau `SwapUsage` metrik lebih besar dari `FreeableMemory` metrik menunjukkan node berada di bawah tekanan memori. Jika tidak, lihat topik berikut:

- Memastikan Anda memiliki cukup memori untuk membuat snapshot Valkey atau Redis OSS
- Mengelola memori cadangan untuk Valkey dan Redis OSS

Evictions

Ini adalah metrik mesin cache. Sebaiknya tentukan ambang batas alarm Anda sendiri untuk metrik ini berdasarkan kebutuhan aplikasi Anda.

Jika Anda menggunakan Memcached dan melebihi ambang batas yang Anda pilih, tingkatkan skala cluster Anda dengan menggunakan tipe node yang lebih besar atau skala dengan menambahkan lebih banyak node.

CurrConnections

Ini adalah metrik mesin cache. Sebaiknya tentukan ambang batas alarm Anda sendiri untuk metrik ini berdasarkan kebutuhan aplikasi Anda.

Peningkatan jumlah CurrConnectionsmungkin menunjukkan masalah dengan aplikasi Anda; Anda perlu menyelidiki perilaku aplikasi untuk mengatasi masalah ini.

Untuk informasi selengkapnya, lihat bagian Koneksi di Memantau praktik terbaik dengan Amazon ElastiCache untuk Redis OSS menggunakan Amazon. CloudWatch

Memori (Valkey dan Redis OSS)

Memori adalah aspek inti dari Valkey dan Redis OSS. Memahami pemanfaatan memori dari klaster Anda diperlukan untuk menghindari kehilangan data dan mengakomodasi pertumbuhan set data Anda pada masa mendatang. Statistik tentang pemanfaatan memori node tersedia di bagian memori dari perintah INFO.

Untuk informasi selengkapnya, lihat bagian Memori di Memantau praktik terbaik dengan Amazon ElastiCache untuk Redis OSS menggunakan Amazon. CloudWatch

Jaringan

Salah satu faktor penentu untuk kapasitas bandwidth jaringan dari klaster Anda adalah jenis simpul yang telah Anda pilih. Untuk informasi selengkapnya tentang kapasitas jaringan node Anda, lihat ElastiCache harga Amazon.

Untuk informasi selengkapnya, lihat bagian Jaringan di Memantau praktik terbaik dengan Amazon ElastiCache untuk Redis OSS menggunakan Amazon. CloudWatch

Latensi

Mengukur waktu respons untuk instance ElastiCache for Valkey dapat didekati dengan berbagai cara tergantung pada tingkat granularitas yang diperlukan. Tahapan kunci yang berkontribusi pada keseluruhan waktu respons sisi server ElastiCache untuk Valkey adalah pra-pemrosesan perintah, eksekusi perintah, dan pasca-pemrosesan perintah.

Metrik latensi khusus perintah yang berasal dari perintah Valkey INFO seperti GetTypeCmdsLatency dan fokus SetTypeCmdsLatency metrik secara khusus pada mengeksekusi logika perintah inti untuk perintah Valkey. Metrik ini akan sangat membantu jika kasus penggunaan Anda adalah untuk menentukan waktu eksekusi perintah atau latensi agregat per struktur data.

Metrik latensi `SuccessfulWriteRequestLatency` dan `SuccessfulReadRequestLatency` ukur total waktu yang ElastiCache dibutuhkan mesin Valkey untuk menanggapi permintaan.

> **Note**
>
> Nilai yang meningkat untuk `SuccessfulWriteRequestLatency` dan `SuccessfulReadRequestLatency` metrik dapat terjadi saat menggunakan pipelining Valkey dengan CLIENT REPLY diaktifkan pada klien Valkey. Valkey pipelining adalah teknik untuk meningkatkan kinerja dengan mengeluarkan beberapa perintah sekaligus, tanpa menunggu respons terhadap setiap perintah individu. Untuk menghindari nilai yang meningkat, kami sarankan untuk mengonfigurasi klien Valkey Anda ke perintah pipeline dengan CLIENT REPLY OFF.

Untuk informasi selengkapnya, lihat bagian Latensi di Memantau praktik terbaik dengan Amazon ElastiCache menggunakan Amazon CloudWatch.

Replikasi

Volume data yang direplikasi akan terlihat melalui metrik `ReplicationBytes`. Metrik ini tidak memberikan wawasan tentang kondisi replikasi, meskipun merepresentasikan beban tulis pada grup replikasi. Untuk tujuan ini, Anda dapat menggunakan metrik `ReplicationLag`.

Untuk informasi selengkapnya, lihat bagian Replikasi di Memantau praktik terbaik dengan Amazon ElastiCache untuk Redis OSS menggunakan Amazon. CloudWatch

Manajemen Lalu Lintas (Valkey dan Redis OSS)

ElastiCache untuk Redis OSS secara otomatis mengelola lalu lintas terhadap node ketika lebih banyak perintah yang masuk dikirim ke node daripada yang dapat diproses oleh Valkey atau Redis OSS. Hal ini dilakukan untuk menjaga operasi dan stabilitas mesin yang optimal.

Ketika lalu lintas dikelola secara aktif pada simpul, metrik `TrafficManagementActive` akan memancarkan titik data 1. Hal ini menunjukkan bahwa simpul mungkin kurang diskalakan untuk beban kerja yang disediakan. Jika metrik ini tetap 1 untuk jangka waktu yang lama, evaluasi klaster untuk memutuskan apakah penaikan skala atau penskalaan ke luar diperlukan.

Untuk informasi selengkapnya, lihat metrik `TrafficManagementActive` di halaman Metrik.

Memilih Statistik dan Periode Metrik

Meskipun CloudWatch akan memungkinkan Anda untuk memilih statistik dan periode untuk setiap metrik, tidak semua kombinasi akan berguna. Misalnya, statistik Rata-rata, Minimum, dan Maksimum untuk CPUUtilization berguna, tetapi statistik Jumlah tidak.

Semua ElastiCache sampel diterbitkan untuk durasi 60 detik untuk setiap node cache individu. Untuk periode 60 detik, metrik simpul cache hanya akan berisi satu sampel tunggal.

Untuk informasi selengkapnya tentang cara mengambil metrik untuk simpul cache, lihat Pemantauan Metrik CloudWatch Kluster dan Node.

Pemantauan Metrik CloudWatch Kluster dan Node

ElastiCache dan CloudWatch terintegrasi sehingga Anda dapat mengumpulkan berbagai metrik. Anda dapat memantau metrik ini menggunakan CloudWatch.

> **Note**
>
> Contoh berikut memerlukan alat baris CloudWatch perintah. Untuk informasi selengkapnya tentang CloudWatch dan untuk mengunduh alat pengembang, lihat halaman CloudWatch produk.

Prosedur berikut menunjukkan cara menggunakan CloudWatch untuk mengumpulkan statistik ruang penyimpanan untuk cluster cache selama satu jam terakhir.

> **Note**
>
> Nilai `StartTime` dan `EndTime` yang diberikan pada contoh di bawah ini adalah untuk tujuan ilustrasi. Anda harus mengganti nilai waktu mulai dan akhir yang sesuai untuk simpul cache Anda.

Untuk informasi tentang ElastiCache batasan, lihat Batas AWS Layanan untuk ElastiCache.

Pemantauan Metrik CloudWatch Kluster dan Node (Konsol)

Untuk mengumpulkan statistik pemanfaatan CPU untuk klaster cache

1. Masuk ke AWS Management Console dan buka ElastiCache konsol di https://console.aws.amazon.com/elasticache/.
2. Pilih simpul cache yang metriknya ingin Anda lihat.

   > **Note**
   >
   > Memilih lebih dari 20 simpul akan menonaktifkan tampilan metrik pada konsol.

   a. Pada halaman Cache Clusters dari AWS Management Console, klik nama satu atau beberapa cluster cache.

      Halaman detail untuk klaster cache akan muncul.

   b. Klik tab Simpul di bagian atas jendela.

   c. Di tab Simpul pada jendela detail, pilih simpul cache yang ingin Anda lihat metriknya.

      Daftar CloudWatch Metrik yang tersedia muncul di bagian bawah jendela konsol.

   d. Klik metrik Pemanfaatan CPU.

      CloudWatch Konsol akan terbuka, menampilkan metrik yang Anda pilih. Anda dapat menggunakan kotak daftar drop-down Statistik dan Periode, serta tab Rentang waktu untuk mengubah metrik yang ditampilkan.

Memantau Metrik CloudWatch Cluster dan Node menggunakan CLI CloudWatch

Untuk mengumpulkan statistik pemanfaatan CPU untuk klaster cache

- Untuk Linux, macOS, atau Unix:

```
aws cloudwatch get-metric-statistics \
    --namespace AWS/ElastiCache \
    --metric-name CPUUtilization \
    --dimensions='[{"Name":"CacheClusterId","Value":"test"},
{"Name":"CacheNodeId","Value":"0001"}]' \
    --statistics=Average \
    --start-time 2018-07-05T00:00:00 \
    --end-time 2018-07-06T00:00:00 \
    --period=3600
```

Untuk Windows:

```
aws cloudwatch get-metric-statistics ^
    --namespace AWS/ElastiCache ^
    --metric-name CPUUtilization ^
    --dimensions='[{"Name":"CacheClusterId","Value":"test"},
{"Name":"CacheNodeId","Value":"0001"}]' ^
    --statistics=Average ^
    --start-time 2018-07-05T00:00:00 ^
    --end-time 2018-07-06T00:00:00 ^
    --period=3600
```

Memantau Metrik CloudWatch Kluster dan Node menggunakan API CloudWatch

Untuk mengumpulkan statistik pemanfaatan CPU untuk klaster cache

- Panggil CloudWatch API `GetMetricStatistics` dengan parameter berikut (perhatikan bahwa waktu mulai dan akhir ditampilkan sebagai contoh saja; Anda perlu mengganti waktu mulai dan akhir yang sesuai):
  - `Statistics.member.1=Average`
  - `Namespace=AWS/ElastiCache`
  - `StartTime=2013-07-05T00:00:00`
  - `EndTime=2013-07-06T00:00:00`
  - `Period=60`
  - `MeasureName=CPUUtilization`
  - `Dimensions=CacheClusterId=mycachecluster,CacheNodeId=0002`

  Example

```
http://monitoring.amazonaws.com/
    ?Action=GetMetricStatistics
    &SignatureVersion=4
    &Version=2014-12-01
    &StartTime=2018-07-05T00:00:00
    &EndTime=2018-07-06T23:59:00
```

```
    &Period=3600
    &Statistics.member.1=Average
    &Dimensions.member.1="CacheClusterId=mycachecluster"
    &Dimensions.member.2="CacheNodeId=0002"
    &Namespace=&AWS;/ElastiCache
    &MeasureName=CPUUtilization
    &Timestamp=2018-07-07T17%3A48%3A21.746Z
    &AWS;AccessKeyId=<&AWS; Access Key ID>
    &Signature=<Signature>
```

Kuota untuk ElastiCache

AWS Akun Anda memiliki kuota default, sebelumnya disebut sebagai batas, untuk setiap layanan. AWS Kecuali dinyatakan lain, setiap kuota bersifat khusus per Wilayah. Anda dapat meminta peningkatan untuk beberapa kuota dan kuota lainnya tidak dapat ditingkatkan.

Untuk melihat kuota ElastiCache, buka konsol Service Quotas. Pada panel navigasi, pilih Layanan AWS dan pilih ElastiCache.

Untuk meminta peningkatan kuota, lihat Meminta Peningkatan Kuota dalam Panduan Pengguna Service Quotas. Jika kuota belum tersedia dalam Service Quotas, gunakan formulir penambahan batas.

AWS Akun Anda memiliki kuota berikut yang terkait ElastiCache dengan.

Nama	Nilai default	Deskripsi	Nama Metrik
Simpul per Wilayah	300	Jumlah maksimum node di semua cluster di Region. Kuota ini berlaku untuk node non-reserved Anda dalam Region yang diberikan. Anda dapat memiliki hingga 300 node non-reserved di Region yang sama.	NodesPerRegion
Simpul per klaster (Memcached)	60	Jumlah maksimum node dalam cluster Memcached individu.	Memcached NodesPerCluster
Node per cluster (mode cluster diaktifkan)	90	Jumlah maksimum node dalam Redis OSS individu atau cluster Valkey.	NodesPerCluster

Nama	Nilai default	Deskripsi	Nama Metrik
Grup parameter per Wilayah	300	Jumlah maksimum grup parameter yang dapat Anda buat di Wilayah.	ParameterGroup
Grup subnet per Wilayah	300	Jumlah maksimum grup subnet yang dapat Anda buat di Wilayah.	SubnetGroup
Subnet per grup subnet	20	Jumlah maksimum subnet yang dapat Anda tentukan untuk grup subnet.	SubnetsPerSubnetGroup
Cache tanpa server per Wilayah	40	Jumlah maksimum cache tanpa server di Wilayah.	ServerlessCache
Snapshot nirserver per hari per cache	24	Jumlah maksimum snapshot tanpa server yang dapat Anda ambil per hari untuk setiap cache tanpa server.	SnapshotsPerDayPerCache
Pengguna per Wilayah	2000	Jumlah total maksimum pengguna yang dapat Anda buat di Wilayah.	Pengguna
Grup pengguna per Wilayah	200	Jumlah maksimum grup pengguna yang dapat Anda buat di Wilayah.	UserGroup

Nama	Nilai default	Deskripsi	Nama Metrik
Pengguna per grup pengguna	100	Jumlah maksimum pengguna yang dapat Anda tentukan untuk grup pengguna.	UsersPerUserGroup

Referensi

Topik di bagian ini mencakup bekerja dengan Amazon ElastiCache API dan ElastiCache bagian dari AWS CLI. Juga disertakan pada bagian ini adalah pesan kesalahan dan notifikasi layanan yang umum.

- Menggunakan ElastiCache API
- ElastiCache Referensi API
- ElastiCache bagian dari AWS CLI Referensi
- Pesan ElastiCache kesalahan Amazon
- Notifikasi

Menggunakan ElastiCache API

Bagian ini memberikan deskripsi berorientasi tugas tentang cara menggunakan dan mengimplementasikan operasi. ElastiCache Untuk penjelasan lengkap tentang operasi ini, lihat Referensi Amazon ElastiCache API.

Topik

- Menggunakan API kueri
- Pustaka yang tersedia
- Memecahkan masalah aplikasi

Menggunakan API kueri

Parameter kueri

Permintaan berbasis Kueri HTTP adalah permintaan HTTP yang menggunakan HTTP kata kerja GET atau POST dan parameter Kueri yang bernama `Action`.

Setiap permintaan Kueri harus menyertakan beberapa parameter umum untuk menangani autentikasi dan sejumlah tindakan pilihan.

Beberapa operasi mengambil daftar parameter. Daftar ini ditentukan menggunakan notasi `param.`*n*. Nilai *n* adalah bilangan bulat mulai dari 1.

Autentikasi permintaan Kueri

Anda hanya dapat mengirim permintaan Kueri melalui HTTPS, dan Anda harus menyertakan tanda tangan di setiap permintaan Kueri. Bagian ini menjelaskan cara membuat tanda tangan. Metode yang dijelaskan dalam prosedur berikut ini dikenal sebagai tanda tangan versi 4.

Berikut adalah langkah dasar yang digunakan untuk mengautentikasi permintaan ke AWS. Ini mengasumsikan Anda terdaftar AWS dan memiliki ID Kunci Akses dan Kunci Akses Rahasia.

Proses autentikasi kueri

1. Pengirim membuat permintaan ke. AWS
2. Pengirim menghitung tanda tangan permintaan, Keyed-Hashing untuk Kode Autentikasi Pesan Berbasis Hash (HMAC) dengan fungsi hash SHA-1, seperti yang didefinisikan dalam bagian berikutnya dari topik ini.
3. Pengirim permintaan mengirimkan data permintaan, tanda tangan, dan ID Kunci Akses (pengenal kunci dari Kunci Akses Rahasia yang digunakan) ke. AWS
4. AWS menggunakan ID Kunci Akses untuk mencari Kunci Akses Rahasia.
5. AWS menghasilkan tanda tangan dari data permintaan dan Kunci Akses Rahasia menggunakan algoritma yang sama yang digunakan untuk menghitung tanda tangan dalam permintaan.
6. Jika tanda tangan cocok, maka permintaan tersebut dianggap autentik. Jika perbandingan gagal, permintaan dibatalkan, dan AWS akan memberikan respons kesalahan.

> **Note**
>
> Jika permintaan berisi parameter `Timestamp`, tanda tangan yang dihitung untuk permintaan akan berakhir masa berlakunya dalam 15 menit mengikuti nilainya.
> Jika permintaan berisi parameter `Expires`, tanda tangan berakhir pada waktu yang ditentukan oleh parameter `Expires`.

Untuk menghitung tanda tangan permintaan

1. Buat string kueri kanonikalisasi yang akan Anda butuhkan nanti dalam prosedur ini:

a. Urutkan komponen string kueri UTF-8 berdasarkan nama parameter dengan pengurutan byte alami. Parameter dapat berasal dari URI GET atau dari badan POST (ketika Content-Type adalah x-www-form-urlencoded application/).

b. URL mengodekan nama parameter dan nilainya sesuai dengan aturan berikut:

   i. Jangan melakukan enkode URL karakter tanpa fungsi khusus apa pun yang ditentukan RFC 3986. Karakter tanpa fungsi khusus ini adalah A-Z, a-z, 0-9, tanda hubung (-), garis bawah (_), titik (.), dan tanda gelombang (~).

   ii. Gunakan enkode persen pada semua karakter lain dengan %XY, di mana X dan Y adalah karakter hex 0-9 dan huruf besar A-F.

   iii. Gunakan enkode persen pada karakter UTF-8 yang diperluas dalam bentuk %XY%ZA...

   iv. Gunakan enkode persen pada karakter spasi sebagai %20 (dan bukan +, seperti skema pengenkodean yang umum dilakukan).

c. Pisahkan nama parameter yang dikodekan dari nilai yang dikodekan dengan tanda sama dengan (=) (karakter ASCII 61), meskipun jika nilai parameter itu kosong.

d. Pisahkan pasangan nama-nilai dengan tanda ampersand (&) (kode ASCII 38).

2. Buat string untuk ditandatangani sesuai dengan tata bahasa semu berikut ("\n" merepresentasikan baris baru ASCII).

```
StringToSign = HTTPVerb + "\n" +
ValueOfHostHeaderInLowercase + "\n" +
HTTPRequestURI + "\n" +
CanonicalizedQueryString <from the preceding step>
```

   Komponen HTTPRequest URI adalah komponen jalur absolut HTTP dari URI hingga, tetapi tidak termasuk, string kueri. Jika HTTPRequest URI kosong, gunakan garis miring (/).

3. Hitung HMAC yang sesuai dengan RFC 2104 dengan string yang baru saja Anda buat, Kunci Akses Rahasia Anda sebagai kunci, dan SHA256 atau sebagai algoritma hash. SHA1

   Untuk informasi lebih lanjut, lihat https://www.ietf. org/rfc/rfc2104.txt.

4. Konversikan nilai yang dihasilkan ke base64.

5. Sertakan nilai sebagai nilai dari parameter `Signature` dalam permintaan.

Misalnya, berikut adalah permintaan sampel (baris baru ditambahkan untuk memperjelas).

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=DescribeCacheClusters
    &CacheClusterIdentifier=myCacheCluster
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &Version=2014-12-01
```

Untuk string kueri sebelumnya, Anda akan menghitung tanda tangan HMAC atas string berikut.

```
GET\n
    elasticache.amazonaws.com\n
    Action=DescribeCacheClusters
    &CacheClusterIdentifier=myCacheCluster
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &Version=2014-12-01
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Credential=AKIADQKE4SARGYLE%2F20140523%2Fus-west-2%2Felasticache
%2Faws4_request
    &X-Amz-Date=20141201T223649Z
    &X-Amz-SignedHeaders=content-type%3Bhost%3Buser-agent%3Bx-amz-content-sha256%3Bx-
amz-date
        content-type:
        host:elasticache.us-west-2.amazonaws.com
        user-agent:CacheServicesAPICommand_Client
    x-amz-content-sha256:
    x-amz-date:
```

Hasilnya adalah permintaan yang ditandatangani berikut.

```
https://elasticache.us-west-2.amazonaws.com/
    ?Action=DescribeCacheClusters
    &CacheClusterIdentifier=myCacheCluster
    &SignatureMethod=HmacSHA256
    &SignatureVersion=4
    &Version=2014-12-01
    &X-Amz-Algorithm=&AWS;4-HMAC-SHA256
    &X-Amz-Credential=AKIADQKE4SARGYLE/20141201/us-west-2/elasticache/aws4_request
    &X-Amz-Date=20141201T223649Z
    &X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date
    &X-Amz-Signature=2877960fced9040b41b4feaca835fd5cfeb9264f768e6a0236c9143f915ffa56
```

Untuk informasi mendetail tentang proses penandatanganan dan penghitungan tanda tangan permintaan, lihat topik Proses Penandatanganan Tanda Tangan Versi 4 dan subtopiknya.

Pustaka yang tersedia

AWS menyediakan kit pengembangan perangkat lunak (SDKs) untuk pengembang perangkat lunak yang lebih suka membangun aplikasi menggunakan bahasa khusus APIs daripada Query API. Ini SDKs menyediakan fungsi dasar (tidak termasuk dalam APIs), seperti otentikasi permintaan, percobaan ulang permintaan, dan penanganan kesalahan sehingga lebih mudah untuk memulai. SDKs dan sumber daya tambahan tersedia untuk bahasa pemrograman berikut:

- Java
- Windows dan .NET
- PHP
- Python
- Ruby

Untuk informasi tentang bahasa lain, lihat Contoh Kode & Pustaka.

Memecahkan masalah aplikasi

ElastiCache menyediakan kesalahan spesifik dan deskriptif untuk membantu Anda memecahkan masalah saat berinteraksi dengan API. ElastiCache

Mengambil kesalahan

Biasanya, Anda ingin aplikasi Anda memeriksa apakah permintaan menimbulkan kesalahan sebelum Anda menghabiskan waktu untuk memproses hasil. Cara termudah untuk mengetahui apakah terjadi kesalahan adalah dengan mencari `Error` node dalam respons dari ElastiCache API.

XPath sintaks menyediakan cara sederhana untuk mencari keberadaan `Error` node, serta cara mudah untuk mengambil kode kesalahan dan pesan. Cuplikan kode berikut menggunakan Perl dan XPath modul XML:: untuk menentukan apakah terjadi kesalahan selama permintaan. Jika terjadi kesalahan, kode akan mencetak pesan dan kode kesalahan pertama dalam tanggapannya.

```
use XML::XPath;
```

```
my $xp = XML::XPath->new(xml =>$response);
if ( $xp->find("//Error") )
{print "There was an error processing your request:\n", " Error code: ",
$xp->findvalue("//Error[1]/Code"), "\n", " ",
$xp->findvalue("//Error[1]/Message"), "\n\n"; }
```

Tips penyelesaian masalah

Dianjurkan proses berikut untuk mendiagnosis dan menyelesaikan masalah dengan API ElastiCache .

- Verifikasi ElastiCache bahwa berjalan dengan benar.

  Untuk melakukan ini, cukup buka jendela browser dan kirimkan permintaan kueri ke ElastiCache layanan (seperti https://elasticache.amazonaws.com). A MissingAuthenticationTokenException atau 500 Internal Server Error mengonfirmasi bahwa layanan tersedia dan menanggapi permintaan.

- Periksa struktur permintaan Anda.

  Setiap ElastiCache operasi memiliki halaman referensi di Referensi ElastiCache API. Periksa ulang bahwa Anda menggunakan parameter dengan benar. Untuk mengetahui kemungkinan kesalahan, lihat contoh permintaan atau skenario pengguna untuk melihat apakah contoh tersebut melakukan operasi serupa.

- Periksa forum.

  ElastiCache memiliki forum diskusi di mana Anda dapat mencari solusi untuk masalah yang dialami orang lain di sepanjang jalan. Untuk melihat forum, kunjungi

  https://forums.aws.amazon.com/ .

Menyiapkan antarmuka baris ElastiCache perintah

Bagian ini menjelaskan prasyarat untuk menjalankan alat baris perintah, tempat mendapatkannya, cara mengatur lingkungannya, dan contoh umum penggunaan alat ini.

Ikuti instruksi dalam topik ini hanya jika Anda mau ElastiCache. AWS CLI

> **Important**
>
> Amazon ElastiCache Command Line Interface (CLI) tidak mendukung ElastiCache perbaikan apa pun setelah versi API 2014-09-30. Untuk menggunakan ElastiCache fungsionalitas yang lebih baru dari baris perintah, gunakan antarmuka baris AWS perintah.

Topik

Prasyarat

Dokumen ini mengasumsikan bahwa Anda dapat bekerja di lingkungan Linux/UNIX atau Windows. Alat baris ElastiCache perintah Amazon juga berfungsi pada Mac OS X, yang merupakan lingkungan berbasis Unix; Namun, tidak ada instruksi Mac OS X tertentu yang disertakan dalam panduan ini.

Sebagai konvensi, semua teks baris perintah diawali dengan prompt baris perintah **`PROMPT>`** generik. Prompt baris perintah yang sebenarnya pada mesin Anda mungkin berbeda. Kami juga menggunakan **`$`** untuk menunjukkan perintah Linux/UNIX tertentu dan **`C:\>`** untuk perintah khusus Windows. Contoh output yang dihasilkan dari perintah ditampilkan tepat setelahnya tanpa awalan apa pun.

Lingkungan runtime Java

Alat baris perintah yang digunakan dalam panduan ini memerlukan Java versi 5 atau yang lebih baru. Dapat menggunakan penginstalan JRE atau JDK. Untuk melihat dan mengunduh JREs berbagai platform, termasuk Linux/UNIX dan Windows, lihat Unduhan Java SE.

Mengatur variabel Java Home

Alat baris perintah bergantung pada variabel lingkungan (`JAVA_HOME`) untuk menemukan Java Runtime. Variabel lingkungan ini harus ditetapkan ke jalur lengkap dari direktori yang berisi

subdirektori bernama `bin` yang berisi executable `java` (di Linux dan UNIX) atau executable `java.exe` (di Windows).

Untuk mengatur variabel Java Home

1. Atur variabel Java Home.
   - Di Linux dan UNIX, masukkan perintah berikut:

     ```
     $ export JAVA_HOME=<PATH>
     ```

   - Di Windows, masukkan perintah berikut:

     ```
     C:\> set JAVA_HOME=<PATH>
     ```

2. Konfirmasikan pengaturan jalur dengan menjalankan **`$JAVA_HOME/bin/java -version`** dan memeriksa output-nya.
   - Di Linux/UNIX, Anda akan melihat output seperti yang berikut ini:

     ```
     $ $JAVA_HOME/bin/java -version
     java version "1.6.0_23"
     Java(TM) SE Runtime Environment (build 1.6.0_23-b05)
     Java HotSpot(TM) Client VM (build 19.0-b09, mixed mode, sharing)
     ```

   - Di Windows, Anda akan melihat output seperti yang berikut ini:

     ```
     C:\> %JAVA_HOME%\bin\java -version
     java version "1.6.0_23"
     Java(TM) SE Runtime Environment (build 1.6.0_23-b05)
     Java HotSpot(TM) Client VM (build 19.0-b09, mixed mode, sharing)
     ```

Mendapatkan alat baris perintah

Alat baris perintah tersedia sebagai file ZIP pada ElastiCache situs web Alat Developer. Alat-alat ini ditulis dalam Java, dan termasuk skrip shell untuk Windows 2000/XP/Vista/Windows 7, Linux/UNIX,

dan Mac OSX. File ZIP bersifat mandiri dan tidak memerlukan penginstalan; cukup unduh file zip lalu ekstrak ke direktori di mesin lokal Anda.

Menyiapkan alat

Alat baris perintah bergantung pada variabel lingkungan (AWS_ELASTICACHE_HOME) untuk menemukan pustaka pendukung. Anda perlu mengatur variabel lingkungan ini sebelum dapat menggunakan alat tersebut. Tetapkan ke jalur direktori tempat Anda mengekstrak file alat baris perintah. Direktori ini bernama ElastiCacheCli -A.B.NNNN (A, B dan n adalah versi/nomor rilis), dan berisi subdirektori bernama bin dan lib.

Untuk mengatur variabel lingkungan AWS_ELASTICACHE_HOME

- Buka jendela baris perintah dan masukkan salah satu perintah berikut untuk mengatur variabel lingkungan AWS_ELASTICACHE_HOME.
  - Di Linux dan UNIX, masukkan perintah berikut:

```
$ export &AWS;_ELASTICACHE_HOME=<path-to-tools>
```

  - Di Windows, masukkan perintah berikut:

```
C:\> set &AWS;_ELASTICACHE_HOME=<path-to-tools>
```

Agar alat ini lebih mudah digunakan, sebaiknya tambahkan direktori BIN alat ini ke PATH sistem Anda. Sisa dari panduan ini akan mengasumsikan bahwa direktori BIN sudah ada dalam jalur sistem Anda.

Untuk menambahkan direktori BIN alat ini ke jalur sistem Anda

- Masukkan perintah berikut untuk menambahkan direktori BIN alat ini ke PATH sistem Anda.
  - Di Linux dan UNIX, masukkan perintah berikut:

```
$ export PATH=$PATH:$&AWS;_ELASTICACHE_HOME/bin
```

  - Di Windows, masukkan perintah berikut:

```
C:\> set PATH=%PATH%;%&AWS;_ELASTICACHE_HOME%\bin
```

> **Note**
>
> Variabel lingkungan Windows direset ketika Anda menutup jendela perintah. Anda sebaiknya menetapkannya secara permanen. Lihat dokumentasi untuk versi Windows Anda untuk informasi selengkapnya.

> **Note**
>
> Jalur yang berisi spasi harus di-wrapping dengan tanda kutip ganda, misalnya: "C:\Program Files\Java"

Memberikan kredensial untuk alat

Alat baris perintah memerlukan Kunci AWS Akses dan Kunci Akses Rahasia yang disediakan dengan AWS akun Anda. Anda bisa mendapatkannya menggunakan baris perintah atau dari file kredensial yang terletak di sistem lokal Anda.

Penerapan menyertakan file template $ {AWS_ELASTICACHE_HOME}/credential-file-path.template yang perlu Anda edit dengan informasi Anda. Berikut adalah konten file templat:

```
AWS AccessKeyId=<Write your AWS access ID>
AWS SecretKey=<Write your AWS secret key>
```

> **Important**
>
> Di UNIX, batasi izin untuk pemilik file kredensial:
>
> ```
> $ chmod 600 <the file created above>
> ```

Dengan pengaturan file kredensial, Anda harus mengatur variabel lingkungan AWS_CREDENTIAL_FILE sehingga alat dapat menemukan informasi Anda. ElastiCache

Untuk mengatur variabel lingkungan AWS_CREDENTIAL_FILE

1. Atur variabel lingkungan:

   - Di Linux dan UNIX, perbarui variabel menggunakan perintah berikut:

     ```
     $ export &AWS;_CREDENTIAL_FILE=<the file created above>
     ```

   - Di Windows, tetapkan variabel menggunakan perintah berikut:

     ```
     C:\> set &AWS;_CREDENTIAL_FILE=<the file created above>
     ```

2. Periksa bahwa pengaturan Anda bekerja dengan baik, jalankan perintah berikut:

   ```
   elasticache --help
   ```

   Anda harus melihat halaman penggunaan untuk semua perintah ElastiCache.

Variabel lingkungan

Variabel lingkungan dapat berguna untuk membuat skrip, mengonfigurasi nilai default, atau mengganti variabel untuk sementara.

Selain variabel lingkungan AWS_CREDENTIAL_FILE, sebagian besar alat API yang disertakan dengan Antarmuka Baris ElastiCache Perintah mendukung variabel berikut:

- EC2_REGION - AWS Wilayah yang akan digunakan.
- AWS_ELASTICACHE_URL – URL yang digunakan untuk panggilan layanan. Tidak diperlukan untuk menentukan titik akhir regional yang berbeda jika EC2 _REGION ditentukan atau parameter --region dilewatkan.

Contoh berikut menunjukkan cara menyetel variabel lingkungan EC2 _REGION untuk mengonfigurasi wilayah yang digunakan oleh alat API:

Linux, OS X, atau Unix

```
$ export EC2_REGION=us-west-1
```

Windows

```
$ set EC2_REGION=us-west-1
```

Pesan ElastiCache kesalahan Amazon

Pesan kesalahan berikut dikembalikan oleh Amazon ElastiCache. Anda mungkin menerima pesan kesalahan lain yang dikembalikan oleh ElastiCache, AWS layanan lain, atau oleh Valkey, Memcached, atau Redis OSS. Untuk deskripsi pesan kesalahan dari sumber selain ElastiCache, lihat dokumentasi dari sumber yang menghasilkan pesan kesalahan.

- Cluster node quota exceeded
- Customer's node quota exceeded
- Manual snapshot quota exceeded
- Insufficient cache cluster capacity

Pesan Kesalahan: Kuota simpul klaster terlampaui. Setiap klaster dapat memiliki paling banyak %n simpul di wilayah ini.

Penyebab: Anda mencoba membuat atau mengubah sebuah klaster yang mengakibatkan klaster akan memiliki lebih dari %n simpul.

Solusi: Ubah permintaan Anda sehingga klaster tidak memiliki lebih dari %n simpul. Atau, jika Anda membutuhkan lebih dari %n node, buat permintaan Anda menggunakan formulir permintaan Amazon ElastiCache Node.

Untuk informasi selengkapnya, lihat ElastiCache Batas Amazon di Referensi Umum Amazon Web Services.

Pesan Kesalahan: Kuota simpul pelanggan terlampaui. Anda dapat memiliki maksimal %n simpul di wilayah ini Atau, Anda telah mencapai kuota Anda sebanyak %s simpul di wilayah ini.

Penyebab: Anda mencoba membuat atau mengubah sebuah klaster yang mengakibatkan akun Anda memiliki lebih dari %n simpul di seluruh klaster di wilayah ini.

Solusi: Ubah permintaan Anda sehingga jumlah simpul di wilayah di semua klaster untuk akun ini tidak melebihi %n. Atau, jika Anda membutuhkan lebih dari %n node, buat permintaan Anda menggunakan formulir permintaan Amazon ElastiCache Node.

Untuk informasi selengkapnya, lihat ElastiCache Batas Amazon di Referensi Umum Amazon Web Services.

Pesan Kesalahan: Jumlah maksimum snapshot manual untuk klaster ini yang diambil dalam waktu 24 jam telah tercapai atau Jumlah maksimum snapshot manual untuk simpul ini yang diambil dalam waktu 24 jam telah mencapai kuota %n

Penyebab: Anda mencoba mengambil snapshot manual dari sebuah klaster ketika Anda telah mengambil jumlah maksimum snapshot manual yang diizinkan dalam periode 24 jam.

Solusi: Tunggu 24 jam untuk mencoba kembali snapshot manual dari klaster tersebut. Atau, jika Anda perlu mengambil snapshot manual sekarang, ambil snapshot dari simpul lain yang memiliki data yang sama, seperti simpul yang berbeda dalam sebuah klaster.

Pesan Kesalahan: InsufficientCacheClusterCapacity

Penyebab: AWS saat ini tidak memiliki kapasitas Sesuai Permintaan yang tersedia untuk melayani permintaan Anda.

Solusi:

- Tunggu beberapa menit, lalu kirim permintaan Anda lagi; kapasitas sering kali dapat berubah.
- Kirim permintaan baru dengan pengurangan jumlah simpul atau serpihan (grup simpul). Misalnya, jika Anda membuat satu permintaan untuk meluncurkan 15 instans, cobalah membuat 3 permintaan untuk 5 instans, bukan 15 permintaan untuk 1 simpul.
- Jika Anda meluncurkan klaster, kirimkan permintaan baru tanpa menentukan Zona Ketersediaan.
- Jika Anda meluncurkan sebuah klaster, kirimkan permintaan baru menggunakan jenis simpul yang berbeda (skalanya dapat dinaikkan di tahap berikutnya). Lihat informasi yang lebih lengkap di Penskalaan ElastiCache.

Notifikasi

Topik ini mencakup ElastiCache pemberitahuan yang mungkin menarik bagi Anda. Notifikasi adalah situasi atau peristiwa yang, dalam kebanyakan kasus, bersifat sementara, sehingga hanya

berlangsung sampai solusi ditemukan dan diimplementasikan. Notifikasi umumnya memiliki tanggal mulai dan tanggal penyelesaian, yang setelahnya, notifikasi akan menjadi tidak relevan lagi. Setiap notifikasi mungkin atau mungkin tidak relevan bagi Anda. Kami menganjurkan pedoman implementasi yang, jika diikuti, akan meningkatkan performa klaster Anda.

Pemberitahuan tidak mengumumkan ElastiCache fitur atau fungsionalitas baru atau yang lebih baik.

ElastiCache Pemberitahuan umum

Saat ini tidak ada ElastiCache pemberitahuan luar biasa yang tidak spesifik untuk mesin.

ElastiCache untuk pemberitahuan Memcached

ElastiCache Pemberitahuan berikut khusus untuk mesin Memcached.

ElastiCache untuk pemberitahuan khusus Memcached

- Peringatan: Perayap LRU Memcached menyebabkan kesalahan segmentasi

Peringatan: Perayap LRU Memcached menyebabkan kesalahan segmentasi

⚠ Tanggal Peringatan: 28 Februari 2017
Dalam keadaan tertentu, klaster Anda mungkin menunjukkan ketidakstabilan dengan kesalahan segmentasi pada Perayap LRU Memcached. Hal ini adalah masalah dalam mesin Memcached yang sudah ada selama beberapa waktu. Masalah ini terlihat jelas dalam Memcached 1.4.33 ketika Perayap LRU diaktifkan secara default.
Jika Anda mengalami masalah ini, sebaiknya nonaktifkan Perayap LRU sampai ada perbaikan. Untuk melakukannya, gunakan `lru_crawler disable` pada baris perintah atau ubah nilai parameter `lru_crawler` (lebih disarankan).
Tanggal Penyelesaian:
Penyelesaian:

ElastiCache untuk pemberitahuan khusus Redis OSS

Saat ini tidak ada yang luar biasa ElastiCache untuk pemberitahuan Redis OSS.

ElastiCache Sejarah dokumentasi

- Versi API: 2015-02-02
- Pembaruan dokumentasi terbaru: 27 November 2023

Tabel berikut menjelaskan perubahan penting dalam setiap rilis Panduan ElastiCache Pengguna setelah Maret 2018. Untuk notifikasi tentang pembaruan-pembaruan dokumentasi ini, Anda dapat berlangganan ke sebuah umpan RSS.

ElastiCache Update Terbaru

Perubahan	Deskripsi	Tanggal
Support ElastiCache untuk penskalaan vertikal sesuai permintaan Memcached dan penskalaan horizontal otomatis	ElastiCache dengan Memcached sekarang mendukung penskalaan vertikal sesuai permintaan dan penskalaan horizontal otomatis. Ini termasuk pembaruan kebijakan untuk tindakan `elasticache:ModifyCacheCluster`	April 10, 2025
Support untuk ElastiCache dengan Valkey	ElastiCache sekarang mendukung Valkey. Valkey 7.2.6 kompatibel dengan Redis OSS 7.2 Untuk informasi lebih lanjut, lihat Valkey.	Oktober 8, 2024
Ukuran node cadangan fleksibel	ElastiCache sekarang mendukung Ukuran node cadangan fleksibel. Untuk informasi selengkapnya, lihat ElastiCache Harga Amazon.	Oktober 1, 2024

Support ElastiCache untuk Memcached 1.6.22	ElastiCache untuk Memcached sekarang mendukung Memcached 1.6.22. Ini mencakup perbaikan bug kumulatif dari versi 1.6.18. Untuk informasi selengkapnya, lihat Memcached Versi 1.6.22.	10 Januari 2024
ElastiCache untuk Redis OSS menambahkan dukungan untuk ukuran node C7gn tambahan	ElastiCache untuk Redis OSS menambahkan dukungan untuk ukuran node C7gn tambahan.	10 Januari 2024

ElastiCache untuk Redis OSS sekarang mendukung pembuatan cache tanpa server	Anda sekarang dapat membuat cache nirserver , yang menyederhanakan manajemen cache dan langsung melakukan penskalaan untuk mendukung aplikasi yang paling menuntut. Untuk informasi selengkap nya, lihat Memilih di antara opsi deployment. Sebagai bagian dari fitur ini, izin baru ditambahkan ke `ElastiCacheServiceRolePolicy` dan `AmazonElastiCacheFullAccess` untuk memungkinkan pengaitan cache nirserver dengan titik akhir VPC terkelola. Selain itu, izin ditambahkan untuk mendukung pengalama n konsol yang direvisi menggunakan kebijakan `AmazonElastiCacheFullAccess` .	27 November 2023

ElastiCache untuk Memcached sekarang mendukung pembuatan cache tanpa server	Anda sekarang dapat membuat cache nirserver , yang menyederhanakan manajemen cache dan langsung melakukan penskalaan untuk mendukung aplikasi yang paling menuntut. Untuk informasi selengkap nya, lihat Memilih di antara opsi deployment. Sebagai bagian dari fitur ini, izin baru ditambahkan ke `ElastiCacheServiceRolePolicy` dan `AmazonElastiCacheFullAccess` untuk memungkinkan pengaitan cache nirserver dengan titik akhir VPC terkelola. Selain itu, izin ditambahkan untuk mendukung pengalaman konsol yang direvisi menggunakan kebijakan `AmazonElastiCacheFullAccess` .	27 November 2023
ElastiCache untuk Redis OSS sekarang mendukung memodifikasi mode cluster	Anda sekarang dapat memigrasikan klaster dari Mode Klaster Dinonaktifkan (CMD) ke Mode Klaster Diaktifkan (CME). Untuk informasi selengkapnya, lihat Mengubah mode klaster.	11 Mei 2023

ElastiCache untuk Redis OSS sekarang mendukung modifikasi pengaturan enkripsi dalam transit	Anda sekarang dapat mengubah konfigurasi TLS klaster Redis OSS Anda tanpa perlu membangun kembali atau menyediakan kembali cluster atau memengaruhi ketersediaan aplikasi. Untuk informasi selengkapnya, lihat Mengaktifkan enkripsi bergerak pada klaster yang sudah ada.	28 Desember 2022
ElastiCache untuk Redis OSS sekarang mendukung otentikasi pengguna menggunakan IAM	Otentikasi IAM memungkinkan Anda untuk mengautentikasi koneksi ke Redis ElastiCache OSS menggunakan identitas IAM. AWS Hal ini memungkinkan Anda memperkuat model keamanan Anda dan menyederhanakan banyak tugas keamanan administratif. Untuk informasi selengkapnya, lihat Autentikasi dengan IAM.	16 November 2022
ElastiCache untuk Redis OSS sekarang mendukung Redis OSS 7	Rilis ini membawa beberapa fitur baru ke Amazon ElastiCache untuk Redis OSS: fungsi Redis OSS, perbaikan ACL dan Sharded Pub/Sub. Untuk informasi lebih lanjut, lihat ElastiCache untuk Redis OSS versi 7.0.	8 November 2022

ElastiCache untuk Redis OSS sekarang mendukung IPV6	ElastiCache mendukung Protokol Internet versi 4 dan 6 (IPv4 dan IPv6), memungkinkan Anda untuk mengkonfigurasi cluster Anda untuk hanya menerima IPv4 koneksi, hanya IPv6 koneksi atau keduanya IPv4 dan IPv6 koneksi (dual-stack). IPv6 didukung untuk beban kerja menggunakan mesin Redis OSS versi 6.2 dan seterusnya pada semua instance yang dibangun di atas sistem Nitro. Tidak ada biaya tambahan untuk mengakses ElastiCache lebih IPv6. Untuk informasi selengkapnya, lihat Memilih jenis jaringan.	7 November 2022

ElastiCache untuk Memcached sekarang mendukung IPV6	ElastiCache mendukung Protokol Internet versi 4 dan 6 (IPv4 dan IPv6), memungkinkan Anda untuk mengkonfigurasi cluster Anda untuk hanya menerima IPv4 koneksi, hanya IPv6 koneksi atau keduanya IPv4 dan IPv6 koneksi (dual-stack). IPv6 didukung untuk beban kerja menggunakan mesin Memcached versi 1.6.6 dan seterusnya pada semua instance yang dibangun di sistem Nitro. Tidak ada biaya tambahan untuk mengakses ElastiCache lebih IPv6. Untuk informasi selengkapnya, lihat Memilih jenis jaringan.	7 November 2022
ElastiCache untuk Memcached sekarang mendukung enkripsi dalam transit	Enkripsi bergerak adalah fitur opsional yang memungkin kan Anda meningkatkan keamanan data Anda pada titik yang paling rentan – saat data dalam keadaan bergerak dari satu lokasi ke lokasi lain. Ini didukung pada versi Memcached 1.6.12 dan yang lebih baru. Untuk informasi selengkapnya, lihat enkripsi ElastiCache dalam transit (TLS).	26 Mei 2022

ElastiCache untuk Redis OSS sekarang mendukung format Notasi JavaScript Objek (JSON) asli	Format asli JavaScript Object Notation (JSON) adalah cara sederhana dan tanpa skema untuk menyandikan kumpulan data kompleks di dalam cluster Redis OSS. Anda dapat menyimpan dan mengakses data secara native menggunakan format JavaScript Object Notation (JSON) di dalam cluster Redis OSS dan memperbarui data JSON yang disimpan dalam cluster tersebut, tanpa perlu mengelola kode khusus untuk membuat serial dan deseriali sasinya. Untuk informasi selengkapnya, lihat Memulai dengan JSON.	25 Mei 2022
ElastiCache sekarang mendukung PrivateLink	AWS PrivateLink memungkin kan Anda mengakses operasi ElastiCache API secara pribadi tanpa gateway internet, perangkat NAT, koneksi VPN, atau koneksi Direct AWS Connect. Untuk informasi selengkapnya, lihat Amazon ElastiCache API dan antarmuka VPC endpoint (AWS PrivateLi nk) untuk Redis OSS atau Amazon API ElastiCache dan antarmuka VPC endpoint () untuk Memcached.AWS PrivateLink	24 Januari 2022

ElastiCache untuk Redis OSS sekarang mendukung Redis OSS 6.2 dan Data Tiering	Amazon ElastiCache untuk Redis OSS memperken alkan versi berikutnya dari mesin Redis OSS yang didukung oleh Amazon. ElastiCache ElastiCache untuk Redis OSS 6.2 mencakup peningkatan kinerja untuk cluster berkemampuan TLS menggunakan tipe node x86 dengan 8 v CPUs atau lebih atau tipe node Graviton2 dengan 4 v atau lebih. CPUs ElastiCache untuk Redis OSS juga memperkenalkan data tiering. Anda dapat menggunakan tingkatan data sebagai cara berbiaya lebih rendah untuk menskalakan klaster Anda hingga ratusan terabyte kapasitas. Untuk informasi selengkapnya, lihat ElastiCache untuk Redis OSS versi 6.2 (ditingkatkan) dan Tingkat data.	23 November 2021

Dukungan untuk Auto Scaling	ElastiCache untuk Redis OSS sekarang mendukung Auto Scaling. ElastiCache untuk Redis OSS auto scaling adalah kemampuan untuk menambah atau mengurangi pecahan atau replika yang diinginkan dalam layanan Redis OSS Anda ElastiCache secara otomatis. ElastiCache memanfaatkan layanan Application Auto Scaling untuk menyediakan fungsionalitas ini. Untuk informasi selengkapnya, lihat Auto Scaling ElastiCache untuk klaster Redis OSS.	19 Agustus 2021
Support untuk pengiriman log Redis OSS Slow	ElastiCache sekarang memungkinkan Anda melakukan streaming Redis OSS SLOWLOG ke salah satu dari dua tujuan: Amazon Data Firehose atau Amazon Logs. CloudWatch Untuk informasi selengkapnya, lihat Pengiriman log.	22 April 2021

Dukungan untuk pemberian tag sumber daya dan kunci kondisi	ElastiCache sekarang mendukung penandaan untuk membantu Anda mengelola cluster dan sumber daya lainnya ElastiCache . Untuk informasi selengkapnya, lihat Menandai ElastiCache sumber daya Anda. ElastiCache juga memperkenalkan dukungan untuk kunci kondisi. Anda dapat menentukan kondisi yang menentukan penerapan kebijakan IAM. Untuk informasi selengkapnya, lihat Menggunakan kunci kondisi.	7 April 2021
Dukungan untuk pemberian tag sumber daya dan kunci kondisi	ElastiCache sekarang mendukung penandaan untuk membantu Anda mengelola cluster dan sumber daya lainnya ElastiCache . Untuk informasi selengkapnya, lihat Menandai ElastiCache sumber daya Anda. ElastiCache juga memperkenalkan dukungan untuk kunci kondisi. Anda dapat menentukan kondisi yang menentukan penerapan kebijakan IAM. Untuk informasi selengkapnya, lihat Menggunakan kunci kondisi.	7 April 2021

ElastiCache sekarang tersedia di AWS Outposts	AWS Outposts membawa AWS layanan asli, infrastruktur, dan model operasi ke hampir semua pusat data, ruang co-lokasi, atau fasilitas lokal. Anda dapat menerapkan ElastiCache di Outposts untuk menyiapkan, mengoperasikan, dan menggunakan cache lokal, seperti yang Anda lakukan di cloud. Untuk informasi selengkapnya, lihat Menggunakan Outposts.	8 Oktober 2020
ElastiCache sekarang mendukung Redis OSS 6	Amazon ElastiCache untuk Redis OSS memperkenalkan versi berikutnya dari mesin Redis OSS yang didukung oleh Amazon. ElastiCache Versi ini mencakup autentikasi pengguna dengan kontrol akses berbasis peran, dukungan tanpa versi, caching sisi klien, dan peningkatan operasional yang signifikan. Untuk informasi selengkapnya, lihat ElastiCache untuk Redis OSS Versi 6.0 (Ditingkatkan).	7 Oktober 2020

ElastiCache sekarang mendukung Local Zones	Zona Lokal adalah perpanjangan dari AWS Wilayah yang secara geografis dekat dengan pengguna Anda. Anda dapat memperluas virtual private cloud (VPC) dari AWS Region induk ke Local Zones dengan membuat subnet baru dan menetapkannya ke Local Zone. Untuk informasi selengkapnya, lihat Menggunakan Zona Lokal.	25 September 2020
ElastiCache untuk Redis OSS sekarang mendukung penskalaan lingkungan Redis OSS Cluster Anda hingga 500 node atau 500 pecahan	Mode Redis OSS Cluster memungkinkan konfigurasi yang dapat Anda gunakan untuk mempartisi data Anda di beberapa pecahan dan menawarkan skalabilitas, kinerja, dan ketersediaan yang lebih baik. Fitur ini tersedia di Amazon ElastiCache untuk Redis OSS versi 5.0.6 dan seterusnya di semua AWS Wilayah dan untuk semua yang ada dan yang baru ElastiCache untuk lingkungan Redis OSS Cluster. Untuk informasi selengkapnya, lihat Redis OSS Nodes and Shards.	13 Agustus 2020

ElastiCache sekarang mendukung izin tingkat sumber daya	Sekarang Anda dapat membatasi cakupan izin pengguna dengan menentukan ElastiCache sumber daya dalam kebijakan AWS Identity and Access Management (IAM). Untuk informasi selengkapnya, lihat Izin di tingkat sumber daya.	12 Agustus 2020
ElastiCache untuk Redis OSS menambahkan metrik Amazon tambahan CloudWatch	ElastiCache untuk Redis OSS sekarang mendukung CloudWatch metrik baru, termasuk dan. `PubSubCmds` `HyperLogLogBasedCmds` Untuk daftar lengkapnya, lihat Metrik untuk Redis OSS.	10 Juni 2020
ElastiCache sekarang mendukung pembaruan otomatis cluster ElastiCache	Amazon ElastiCache sekarang mendukung pembaruan otomatis ElastiCache cluster setelah “direkomendasikan berlaku berdasarkan tanggal” pembaruan layanan telah berlalu. ElastiCache akan menggunakan jendela pemeliharaan Anda untuk menjadwalkan pembaruan otomatis cluster yang berlaku. Untuk informasi selengkapnya, lihat Pembaruan mandiri.	13 Mei 2020

ElastiCache untuk Redis OSS sekarang mendukung Global Datastore untuk Redis OSS	Fitur Global Datastore for Redis OSS menawarkan replikasi yang dikelola sepenuhnya, cepat, andal, dan aman di seluruh Wilayah. AWS Dengan menggunakan fitur ini, Anda dapat membuat klaster replika baca lintas wilayah ElastiCache untuk Redis OSS guna mengaktifkan pembacaan latensi rendah dan pemulihan bencana di seluruh Wilayah. AWS Anda dapat membuat, mengubah, dan menjelaskan penyimpanan data global. Anda juga dapat menambah atau menghapus AWS Wilayah dari datastore global Anda dan mempromosikan AWS Wilayah sebagai yang utama dalam datastore global. Untuk informasi selengkapnya, lihat Replikasi Lintas AWS Wilayah Menggunakan Datastore Global.	16 Maret 2020
ElastiCache untuk Redis OSS sekarang mendukung Redis OSS versi 5.0.6	Untuk informasi selengkapnya, lihat ElastiCache untuk Redis OSS Versi 5.0.6 (Ditingkatkan).	18 Desember 2019

Amazon ElastiCache sekarang mendukung node cache T3-Standar	Anda sekarang dapat meluncurkan node cache T3-Standard burstable serba guna generasi berikutnya di Amazon. ElastiCache Instans EC2 T3-Standard Amazon memberikan tingkat kinerja CPU dasar dengan kemampuan untuk meningkatkan penggunaan CPU kapan saja hingga kredit yang masih harus dibayar habis. Untuk informasi selengkapnya, lihat Jenis Simpul yang Didukung.	12 November 2019

Amazon ElastiCache sekarang mendukung modifikasi token AUTH pada yang sudah ada ElastiCache untuk server Redis OSS	ElastiCache untuk Redis OSS 5.0.6 sekarang memungkinkan Anda untuk memodifikasi token otentikasi dengan mengatur dan memutar token baru. Anda sekarang dapat mengubah token aktif saat sedang digunakan. Anda juga dapat menambahkan token baru ke klaster yang sudah ada yang mengaktifkan enkripsi bergerak yang sebelumnya dipersiapkan tanpa token autentikasi. Ini adalah proses dua langkah di mana Anda dapat menetapkan dan merotasi token tanpa mengganggu permintaan klien. Fitur ini saat ini tidak didukung pada AWS CloudFormation. Untuk informasi selengkapnya, lihat Mengautentikasi Pengguna dengan Perintah AUTH Redis OSS.	30 Oktober 2019
Amazon ElastiCache sekarang mendukung migrasi data online dari Redis OSS di Amazon EC2	Anda sekarang dapat menggunakan Migrasi Online untuk memigrasikan data Anda dari Redis OSS yang dihosting sendiri di Amazon ke Amazon EC2 . ElastiCache Untuk informasi selengkapnya, lihat Migrasi Online ke ElastiCache.	28 Oktober 2019

ElastiCache untuk Redis OSS memperkenalkan penskalaan vertikal online untuk mode Redis OSS Cluster.	Anda sekarang dapat meningkatkan atau menurunkan skala Redis OSS Cluster Anda sesuai permintaan. ElastiCache untuk Redis OSS mengubah ukuran cluster Anda dengan mengubah jenis node, sementara cluster terus online dan melayani permintaan masuk. Untuk informasi selengkapnya, lihat Penskalaan Vertikal Online dengan Mengubah Jenis Simpul.	20 Agustus 2019
ElastiCache untuk Redis OSS sekarang memungkinkan pengguna untuk menggunakan endpoint pembaca tunggal untuk Amazon Anda ElastiCache untuk Redis OSS cluster.	Fitur ini memungkinkan Anda mengarahkan semua lalu lintas baca ke cluster Redis OSS Anda ElastiCache melalui satu titik akhir tingkat cluster untuk memanfaatkan penyeimbangan beban dan ketersediaan yang lebih tinggi. Untuk informasi selengkapnya, lihat Menemukan Titik Akhir Koneksi.	13 Juni 2019

ElastiCache untuk Redis OSS sekarang memungkinkan pengguna untuk menerapkan pembaruan layanan pada jadwal mereka sendiri	Dengan fitur ini, Anda dapat memilih untuk menerapkan pembaruan layanan yang tersedia pada saat yang Anda pilih dan tidak hanya selama periode pemeliharaan. Ini akan meminimalkan gangguan layanan, terutama selama arus bisnis puncak, dan membantu memastikan Anda tetap patuh jika klaster Anda berada dalam program kepatuhan yang ElastiCache didukung. Untuk informasi selengkapnya, lihat Pembaruan Layanan Mandiri di Amazon ElastiCache dan validasi Kepatuhan untuk Amazon. ElastiCache	4 Juni 2019
ElastiCache Penawaran Instans Cadangan Standar: Sebagian di Muka, Semua di Muka dan Tidak Ada di Muka.	Instans Cadangan memberi Anda fleksibilitas untuk memesan ElastiCache instans Amazon untuk jangka waktu satu atau tiga tahun berdasarkan jenis dan AWS Wilayah instans. Untuk informasi selengkapnya, lihat Mengelola Biaya dengan Simpul Terpesan.	18 Januari 2019
ElastiCache untuk dukungan Redis OSS hingga 250 node per cluster Redis OSS	Batas node atau shard dapat ditingkatkan hingga maksimum 250 per ElastiCache untuk cluster Redis OSS. Untuk informasi selengkapnya, lihat Serpihan.	19 November 2018

ElastiCache untuk dukungan Redis OSS untuk autofailover dan backup dan restore pada semua node T2	ElastiCache untuk Redis OSS memperkenalkan dukungan untuk autofailover, membuat snapshot, dan backup dan restore pada semua node T2. Untuk informasi selengkapnya, lihat ElastiCache Redis OSS Backup and Restore and Snapshot.	19 November 2018
ElastiCache untuk dukungan Redis OSS untuk node M5 dan R5	ElastiCache untuk Redis OSS sekarang mendukung node M5 dan R5, tipe instans tujuan umum dan dioptimalkan memori berdasarkan Sistem Nitro. AWS Untuk informasi selengkapnya, lihat Jenis Simpul yang Didukung.	23 Oktober 2018
Dukungan untuk perubahan jumlah replika baca secara dinamis	ElastiCache untuk Redis OSS telah menambahkan dukungan untuk menambahkan dan menghapus replika baca dari cluster mana pun tanpa downtime cluster. Untuk informasi selengkapnya tentang perubahan ini dan perubahan lainnya dalam rilis ini, lihat Mengubah Jumlah Replika di Panduan ElastiCache Pengguna Redis OSS. Lihat juga DecreaseReplicaCountdan IncreaseReplicaCountdi Referensi ElastiCache API.	17 September 2018

Sertifikasi kepatuhan FedRAMP	ElastiCache untuk Redis OSS sekarang disertifikasi untuk kepatuhan FedRAMP. Untuk informasi selengkapnya, lihat Validasi kepatuhan untuk Amazon ElastiCache.	30 Agustus 2018
Peningkatan mesin Valkey atau Redis OSS (mode cluster diaktifkan)	Amazon ElastiCache untuk Redis OSS telah menambahkan dukungan untuk meningkatkan versi mesin Valkey atau Redis OSS (mode cluster diaktifkan). Untuk informasi selengkapnya, lihat Meningkatkan Versi Mesin.	20 Agustus 2018
Sertifikasi kepatuhan PCI DSS	ElastiCache untuk Redis OSS sekarang disertifikasi untuk kepatuhan PCI DSS. Untuk informasi selengkapnya, lihat Validasi kepatuhan untuk Amazon ElastiCache.	5 Juli 2018
Support ElastiCache untuk Redis OSS 4.0.10	ElastiCache untuk Redis OSS sekarang mendukung Redis OSS 4.0.10, termasuk enkripsi dan pengubahan ukuran cluster online dalam satu versi. Untuk informasi selengkapnya, lihat ElastiCache untuk Redis OSS Versi 4.0.10 (Ditingkatkan).	14 Juni 2018

Perubahan struktur Panduan Pengguna	Panduan ElastiCache Pengguna tunggal sekarang direstrukturisasi sehingga ada panduan pengguna terpisah untuk Redis OSS (untuk Panduan Pengguna Redis OSS) dan ElastiCache untuk Memcached (untuk Panduan Pengguna Memcached). ElastiCache Struktur dokumentasi di bagian AWS CLI Command Reference : elasticache dan Referensi Amazon ElastiCache API tetap tidak berubah.	20 April 2018
Support untuk CPUUtilization metrik Engine	ElastiCache untuk Redis OSS menambahkan metrik baru`EngineCPUUtilizati on` , yang melaporkan persentase kapasitas CPU Anda yang saat ini sedang digunakan. Untuk informasi selengkapnya, lihat Metrik untuk Redis OSS.	9 April 2018

Tabel berikut menjelaskan perubahan penting pada Panduan ElastiCache Pengguna sebelum Maret 2018.

Perubahan	Deskripsi	Tanggal Diubah
Dukungan untuk Wilayah Asia Pasifik (Osaka-lokal).	ElastiCache Menambahkan dukungan untuk Wilayah Asia Pasifik (Osaka-lokal). Wilayah Asia Pasifik (Osaka) saat ini mendukung Zona Ketersediaan tunggal dan hanya berdasarkan	12 Februari 2018

Perubahan	Deskripsi	Tanggal Diubah
	undangan saja. Untuk informasi selengkapnya, lihat berikut ini: • Wilayah yang didukung • Jenis simpul cache yang didukung	
Dukungan untuk Eropa (Paris).	ElastiCache menambahkan dukungan untuk Wilayah UE (Paris). Untuk informasi selengkapnya, lihat berikut ini: • Wilayah yang didukung • Jenis simpul cache yang didukung	18 Desember 2017
Dukungan untuk Wilayah Tiongkok (Ningxia)	Amazon ElastiCache menambahkan dukungan untuk Wilayah Tiongkok (Ningxia). Untuk informasi selengkapnya, lihat berikut ini: • Wilayah yang didukung • Jenis simpul cache yang didukung	11 Desember 2017
Dukungan untuk Peran Terkait Layanan	Rilis dukungan ElastiCache tambahan untuk Service Linked Roles (SLR) ini. Untuk informasi selengkapnya, lihat berikut ini: • Menggunakan Peran Tertaut Layanan untuk Amazon ElastiCache • Siapkan izin Anda (hanya ElastiCache pengguna baru)	7 Desember 2017

Perubahan	Deskripsi	Tanggal Diubah
Dukungan untuk jenis simpul R4	Rilis ini ElastiCache menambahkan jenis node R4 dukungan di semua AWS Wilayah yang didukung oleh ElastiCache. Anda dapat membeli jenis simpul R4 sebagai item Sesuai Permintaan atau sebagai Simpul Cache Terpesan. Untuk informasi selengkapnya, lihat berikut ini: • Jenis simpul cache yang didukung • Parameter khusus jenis simpul Memcached • Parameter spesifik tipe node Redis OSS	20 November 2017
ElastiCache untuk Redis OSS 3.2.10 dan dukungan untuk resharding online	Amazon ElastiCache untuk Redis OSS menambahkan dukungan ElastiCache untuk Redis OSS 3.2.10. ElastiCache untuk Redis OSS juga memperkenalkan pengubahan ukuran cluster online untuk menambah atau menghapus pecahan dari cluster sambil terus melayani permintaan yang masuk. I/O Untuk informasi selengkapnya, lihat berikut ini: • Perubahan ukuran klaster online • Resharding online untuk Valkey atau Redis OSS (mode cluster diaktifkan)	9 November 2017
Kelayakan HIPAA	ElastiCache untuk Redis OSS versi 3.2.6 sekarang disertifikasi untuk kelayakan HIPAA saat enkripsi diaktifkan di cluster Anda. Untuk informasi selengkapnya, lihat berikut ini: • Validasi kepatuhan untuk Amazon ElastiCache • Keamanan data di Amazon ElastiCache	2 November 2017

Perubahan	Deskripsi	Tanggal Diubah
ElastiCache untuk Redis OSS 3.2.6 dan dukungan untuk enkripsi	ElastiCache menambahkan dukungan ElastiCache untuk Redis OSS 3.2.6, yang mencakup dua fitur enkripsi: • Enkripsi bergerak mengenkripsi data Anda setiap kali data bergerak dari satu tempat ke tempat lain, misalnya antara beberapa simpul di klaster atau antara klaster dan aplikasi Anda. • Enkripsi diam mengenkripsi data pada disk Anda selama operasi sinkronisasi dan pencadangan. Untuk informasi selengkapnya, lihat berikut ini: • Keamanan data di Amazon ElastiCache • Mesin dan versi yang didukung	25 Oktober 2017
Topik pola koneksi	ElastiCache dokumentasi menambahkan topik yang mencakup berbagai pola untuk mengakses ElastiCache cluster di VPC Amazon. Untuk informasi lain, lihat Pola Akses untuk Mengakses ElastiCache Cache di VPC Amazon dalam Panduan Pengguna ElastiCache .	24 April 2017
Dukungan untuk Memcached 1.4.34	ElastiCache menambahkan dukungan Memcached versi 1.4.34, yang menggabungkan sejumlah perbaikan ke versi Memcached sebelumnya. Untuk informasi lebih lanjut, lihat Memcached 1.4.34 Catatan Rilis di Memcached on. GitHub	10 April 2017

Perubahan	Deskripsi	Tanggal Diubah
Dukungan untuk menguji Failover Otomatis	ElastiCache menambahkan dukungan untuk menguji Failover Otomatis pada cluster Redis OSS yang mendukung replikasi. Untuk informasi selengkapnya, lihat berikut ini: • Menguji failover otomatis di Panduan Pengguna ElastiCache . • TestFailover di Referensi API ElastiCache . • test-failover dalam Referensi AWS CLI .	4 April 2017
Pemulihan Redis OSS yang ditingkatkan	ElastiCache menambahkan cadangan dan pemulihan Redis OSS yang disempurnakan dengan mengubah ukuran cluster. Fitur ini mendukung pemulihan cadangan ke klaster dengan jumlah serpihan yang berbeda daripada klaster yang digunakan untuk membuat cadangan. (Untuk API dan CLI, fitur ini dapat memulihka n jumlah grup simpul yang berbeda, bukannya jumlah serpihan yang berbeda.) Pembaruan ini juga mendukung konfigurasi slot Redis OSS yang berbeda. Untuk informasi selengkapnya, lihat Melakukan pemulihan dari cadangan ke dalam cache baru.	15 Maret 2017

Perubahan	Deskripsi	Tanggal Diubah
Parameter manajemen memori Redis OSS baru	ElastiCache menambahkan parameter Redis OSS baru `reserved-memory-percent` , yang membuat pengelolaan memori cadangan Anda lebih mudah. Parameter ini tersedia di semua versi ElastiCache untuk Redis OSS. Untuk informasi selengkapnya, lihat berikut ini: • Mengelola memori cadangan untuk Valkey dan Redis OSS • Parameter baru untuk Redis OSS 3.2.4	15 Maret 2017
Dukungan untuk Memcached 1.4.33	ElastiCache menambahkan dukungan untuk Memcached versi 1.4.33. Untuk informasi selengkapnya, lihat berikut ini: • ElastiCache versi 1.4.33 untuk Memcached • Parameter yang ditambahkan di Memcached 1.4.33	20 Desember 2016
Dukungan untuk Wilayah EU West (London)	ElastiCache menambahkan dukungan untuk Wilayah UE (London). Hanya jenis simpul T2 dan M4 yang didukung saat ini. Untuk informasi selengkapnya, lihat berikut ini: • Wilayah yang didukung • Jenis simpul cache yang didukung	13 Desember 2016
Dukungan untuk Wilayah Kanada (Montreal)	ElastiCache menambahkan dukungan untuk Wilayah Kanada (Montreal). Hanya tipe simpul M4 dan T2 yang saat ini didukung di Wilayah ini AWS . Untuk informasi selengkapnya, lihat berikut ini: • Wilayah yang didukung • Jenis simpul cache yang didukung	8 Desember 2016

Perubahan	Deskripsi	Tanggal Diubah
Dukungan untuk jenis simpul M4 dan R3	ElastiCache menambahkan dukungan untuk tipe node R3 dan M4 di Wilayah Amerika Selatan (São Paulo) dan tipe node M4 di Wilayah China (Beijing) . Untuk informasi selengkapnya, lihat berikut ini: • Wilayah yang didukung • Jenis simpul cache yang didukung	1 November 2016
Dukungan Wilayah AS Timur 2 (Ohio)	ElastiCache menambahkan dukungan untuk Wilayah Timur AS (Ohio) (us-east-2) dengan tipe node M4, T2, dan R3. Untuk informasi selengkap nya, lihat berikut ini: • Wilayah yang didukung • Jenis simpul cache yang didukung	17 Oktober 2016

Perubahan	Deskripsi	Tanggal Diubah
Support untuk Redis OSS Cluster	ElastiCache menambahkan dukungan untuk Redis OSS Cluster (ditingkatkan). Pelanggan yang menggunakan Redis OSS Cluster, dapat mempartisi data mereka hingga 15 pecahan (grup node). Setiap serpihan mendukung replikasi maksimum hingga 5 replika baca per serpihan. Waktu failover otomatis Redis OSS Cluster sekitar seperempat selama versi sebelumnya. Rilis ini mencakup konsol manajemen yang dirancang ulang yang menggunakan terminolo gi yang mengikuti perkembangan penggunaan industri. Untuk informasi selengkapnya, lihat berikut ini: • Membandingkan Memcached dan Redis OSS • ElastiCache komponen dan fitur — perhatika n bagian pada Simpul, Serpihan, Klaster, dan Replikasi. • ElastiCache terminologi	12 Oktober 2016
Dukungan jenis simpul M4	ElastiCache menambahkan dukungan untuk keluarga tipe node M4 di sebagian besar AWS Wilayah yang didukung olehElastiCache. Anda dapat membeli jenis simpul M4 sebagai item Sesuai Permintaan atau sebagai Simpul Cache Terpesan. Untuk informasi selengkapnya, lihat berikut ini: • Jenis simpul cache yang didukung • Parameter khusus jenis simpul Memcached • Parameter spesifik tipe node Redis OSS	3 Agustus 2016

Perubahan	Deskripsi	Tanggal Diubah
Dukungan Wilayah Mumbai	ElastiCache Menambahkan dukungan untuk Wilayah Asia Pasifik (Mumbai). Untuk informasi selengkapnya, lihat berikut ini: • Jenis simpul cache yang didukung • Parameter khusus jenis simpul Memcached • Parameter spesifik tipe node Redis OSS	27 Juni 2016
Ekspor snapshot	ElastiCache menambahkan kemampuan untuk mengekspor snapshot Redis OSS sehingga Anda dapat mengaksesnya dari luar. ElastiCache Untuk informasi selengkapnya, lihat berikut ini: • Mengekspor cadangandi Panduan ElastiCache Pengguna Amazon • CopySnapshotdi Referensi ElastiCache API Amazon	26 Mei 2016
Menaikkan skala jenis simpul	ElastiCache menambahkan kemampuan untuk meningkatkan tipe node Redis OSS Anda. Untuk informasi selengkapnya, lihat Penskalaan ElastiCache.	24 Maret 2016
Peningkatan mesin yang mudah	ElastiCache menambahkan kemampuan untuk dengan mudah meningkatkan mesin cache Redis OSS Anda. Untuk informasi selengkapnya, lihat Manajemen Versi untuk ElastiCache.	22 Maret 2016
Dukungan untuk jenis simpul R3	ElastiCache menambahkan dukungan untuk tipe node R3 di Wilayah China (Beijing) dan Wilayah Amerika Selatan (São Paulo). Untuk informasi selengkapnya, lihat Jenis simpul cache yang didukung.	16 Maret 2016

Perubahan	Deskripsi	Tanggal Diubah
Mengakses ElastiCache menggunakan fungsi Lambda	Menambahkan tutorial tentang mengonfigurasi fungsi Lambda untuk ElastiCache mengakses di VPC Amazon. Untuk informasi selengkapnya, lihat ElastiCache Tutorial dan video lainnya.	12 Februari 2016
Support untuk Redis OSS 2.8.24	ElastiCache menambahkan dukungan untuk Redis OSS versi 2.8.24 dengan perbaikan ditambahk an sejak Redis OSS 2.8.23. Perbaikan mencakup perbaikan bug dan dukungan untuk membuat log untuk alamat akses memori yang buruk. Untuk informasi selengkapnya, lihat berikut ini: • ElastiCache versi 2.8.24 untuk Redis OSS (ditingkatkan) • Catatan Rilis Redis OSS 2.8	20 Januari 2016
Dukungan untuk Wilayah Asia Pasifik (Seoul)	ElastiCache menambahkan dukungan untuk Wilayah Asia Pasifik (Seoul) (ap-northeast-2) dengan tipe node t2, m3, dan r3.	6 Januari 2016
Perubahan ElastiCache konsol Amazon.	Karena versi Redis OSS yang lebih baru memberikan pengalaman pengguna yang lebih baik dan lebih stabil, Redis OSS versi 2.6.13, 2.8.6, dan 2.8.19 tidak lagi terdaftar di Konsol Manajemen. ElastiCache Untuk opsi lainnya dan informasi selengkapnya, lihat Mesin dan versi yang didukung.	15 Desember 2015

Perubahan	Deskripsi	Tanggal Diubah
Support untuk Redis OSS 2.8.23.	ElastiCache menambahkan dukungan untuk Redis OSS versi 2.8.23 dengan perbaikan ditambahkan sejak Redis OSS 2.8.22. Perbaikan ini mencakup perbaikan bug dan dukungan untuk parameter baru `close-on-slave-write` yang, jika diaktifkan, memutuskan klien yang mencoba menulis ke replika hanya-baca. Untuk informasi selengkapnya, lihat ElastiCache versi 2.8.23 untuk Redis OSS (ditingkatkan).	13 November 2015

Perubahan	Deskripsi	Tanggal Diubah
Support untuk Redis OSS 2.8.22.	ElastiCache menambahkan dukungan untuk Redis OSS versi 2.8.22 dengan ElastiCache tambahan tambahan dan perbaikan sejak versi 2.8.21. Peningkatan meliputi: • Pelaksanaan proses menyimpan forkless yang memungkinkan penyimpanan yang berhasil saat memori tersedia rendah yang dapat menyebabk an penyimpanan dengan forked gagal. • CloudWatch Metrik tambahan — SaveInPro gressdan ReplicationBytes. • Untuk mengaktifkan sinkronisasi sebagian, parameter Redis OSS `repl-backlog-size` sekarang berlaku untuk semua cluster. Untuk daftar lengkap perubahan dan informasi selengkapnya, lihat ElastiCache versi 2.8.22 untuk Redis OSS (ditingkatkan). Rilis dokumentasi ini mencakup reorganisasi dokumentasi dan penghapusan dokumentasi antarmuka baris ElastiCache perintah (CLI). Untuk penggunaan baris perintah, lihat AWS Baris Perintah untuk ElastiCache.	28 September 2015

Perubahan	Deskripsi	Tanggal Diubah
Dukungan untuk Memcached 1.4.28.	ElastiCache menambahkan dukungan untuk Memcached versi 1.4.24 dan perbaikan Memcached sejak versi 1.4.14. Rilis ini menambahkan dukungan untuk manajemen cache least recently used (LRU) sebagai tugas di latar belakang, pilihan dari jenkins atau murmur3 sebagai algoritma hashing Anda, perintah baru, dan perbaikan bug lainnya. Untuk informasi selengkapnya, lihat Memcached release notes .	27 Agustus 2015
Dukungan untuk Penemuan Otomatis Memcached menggunakan PHP 5.6	Rilis Amazon ini ElastiCache menambahkan dukungan untuk klien Memcached Auto Discovery untuk PHP versi 5.6. Untuk informasi selengkap nya, lihat Mengompilasi kode sumber untuk klien ElastiCache cluster untuk PHP.	29 Juli 2015
Support untuk Redis OSS 2.8.21	ElastiCache menambahkan dukungan untuk Redis OSS versi 2.8.21 dan perbaikan Redis OSS sejak versi 2.8.19. Rilis Redis OSS ini mencakup beberapa perbaikan bug. Untuk informasi lebih lanjut, lihat catatan rilis Redis OSS 2.8.	29 Juli 2015
Topik baru: Mengakses ElastiCache dari luar AWS	Menambahkan topik baru tentang cara mengakses ElastiCache sumber daya dari luar AWS. Untuk informasi selengkapnya, lihat Mengakses ElastiCache sumber daya dari luar AWS.	9 Juli 2015

Perubahan	Deskripsi	Tanggal Diubah
Pesan penggantian simpul ditambahkan	ElastiCache menambahkan tiga pesan yang berkaitan dengan penggantian node terjadwal ,ElastiCache:NodeReplacementScheduled,ElastiCache:NodeReplacementRescheduled, danElastiCache:NodeReplacementCanceled. Untuk informasi dan tindakan selengkapnya yang dapat Anda lakukan saat node dijadwalkan untuk diganti, ElastiCache lihatNotifikasi Peristiwa dan Amazon SNS.	11 Juni 2015

Perubahan	Deskripsi	Tanggal Diubah
Support untuk Redis OSS v 2.8.19.	ElastiCache menambahkan dukungan untuk Redis OSS versi 2.8.19 dan perbaikan Redis OSS sejak versi 2.8.6. Dukungan ini mencakup dukungan untuk: • Struktur HyperLogLog data, dengan perintah Redis OSS PFADD, PFCOUNT, dan PFMERGE. • Kueri rentang leksikografis dengan perintah baru ZRANGEBYLEX, ZLEXCOUNT, dan ZREMRANGEBYLEX. • Memperkenalkan sejumlah perbaikan bug, yaitu mencegah simpul primer mengirim data yang sudah usang ke simpul replika dengan menggagalkan SYNC primer ketika proses anak untuk simpan di latar belakang (bgsave) berhenti tiba-tiba. Untuk informasi lebih lanjut tentang HyperLogLog, lihat struktur data baru Redis OSS: the. HyperLogLog Untuk informasi lebih lanjut tentang PFADD, PFCOUNT, dan PFMERGE, lihat Dokumentasi Redis OSS dan klik. HyperLogLog	11 Maret 2015
Dukungan untuk tag alokasi biaya	ElastiCache menambahkan dukungan untuk tag alokasi biaya. Untuk informasi selengkapnya, lihat Memantau biaya dengan tag alokasi biaya.	9 Februari 2015
Support untuk AWS GovCloud Wilayah (AS-Barat)	ElastiCache menambahkan dukungan untuk Wilayah AWS GovCloud (AS-Barat) (us-gov-west-1).	29 Januari 2015

Perubahan	Deskripsi	Tanggal Diubah
Dukungan untuk Wilayah Eropa (Frankfurt)	ElastiCache menambahkan dukungan untuk Wilayah Eropa (Frankfurt) (eu-central-1).	19 Januari 2015
Dukungan multi-AZ untuk grup replikasi Redis OSS	ElastiCache menambahkan dukungan untuk Multi-AZ dari node utama ke replika baca dalam grup replikasi Redis OSS. ElastiCache memantau kesehatan kelompok replikasi. Jika primer gagal, ElastiCache secara otomatis mempromosikan replika ke primer, kemudian menggantikan replika. Untuk informasi selengkapnya, lihat Meminimalkan downtime ElastiCache dengan menggunakan Multi-AZ dengan Valkey dan Redis OSS.	24 Oktober 2014
AWS CloudTrail pencatatan panggilan API didukung	ElastiCache menambahkan dukungan untuk menggunakan AWS CloudTrail untuk mencatat semua panggilan ElastiCache API. Untuk informasi selengkapnya, lihat Mencatat panggilan ElastiCache API Amazon dengan AWS CloudTrail.	15 September 2014
Ukuran instans baru didukung	ElastiCache menambahkan dukungan untuk instance Tujuan Umum (T2) tambahan. Untuk informasi selengkapnya, lihat Mengkonfigurasi parameter mesin menggunakan grup ElastiCache parameter.	11 September 2014
Penempatan simpul yang fleksibel didukung untuk Memcached	ElastiCache menambahkan dukungan untuk membuat node Memcached di beberapa Availability Zone.	23 Juli 2014

Perubahan	Deskripsi	Tanggal Diubah
Ukuran instans baru didukung	ElastiCache menambahkan dukungan untuk instans Tujuan Umum (M3) tambahan dan instans Memory Optimized (R3). Untuk informasi selengkapnya, lihat Mengkonfigurasi parameter mesin menggunakan grup ElastiCache parameter.	1 Juli 2014
Penemuan otomatis PHP	Ditambahkan dukungan untuk penemuan otomatis PHP versi 5.5.	13 Mei 2014
Backup dan restore untuk Redis OSS cluster	Dalam rilis ini, ElastiCache memungkinkan pelanggan untuk membuat snapshot dari cluster Redis OSS mereka, dan membuat cluster baru menggunakan snapshot ini. Cadangan adalah salinan cluster pada saat tertentu dalam waktu, dan terdiri dari metadata cluster dan semua data dalam cache Redis OSS. Cadangan disimpan di Amazon S3, dan pelanggan dapat memulihkan data dari snapshot ke klaster baru setiap saat. Untuk informasi selengkapnya, lihat Melakukan snapshot dan pemulihan.	24 April 2014
Redis OSS 2.8.6	ElastiCache mendukung Redis OSS 2.8.6, selain Redis OSS 2.6.13. Dengan Redis OSS 2.8.6, pelanggan dapat meningkatkan ketahanan dan toleransi kesalahan replika baca, dengan dukungan untuk sinkronisasi ulang sebagian, dan jumlah minimum replika baca yang ditentukan pengguna yang harus tersedia setiap saat. Redis OSS 2.8.6 juga menawarkan dukungan penuh untuk publish-and-subscribe, di mana klien dapat diberitahu tentang peristiwa yang terjadi di server.	13 Maret 2014

Perubahan	Deskripsi	Tanggal Diubah
Mesin cache Redis OSS	ElastiCache menawarkan perangkat lunak mesin cache Redis OSS, selain Memcached. Pelanggan yang saat ini menggunakan Redis OSS dapat "menyemai" cluster cache ElastiCache Redis OSS baru dengan data mereka yang ada dari file snapshot Redis OSS, memudahkan migrasi ke lingkungan yang dikelola. ElastiCache Untuk mendukung kemampuan replikasi Redis OSS, ElastiCache API sekarang mendukung grup replikasi. Pelanggan dapat membuat grup replikasi dengan node cache Redis OSS primer, dan menambahkan satu atau lebih node replika baca yang secara otomatis tetap disinkronkan dengan data cache di node utama. Aplikasi sarat operasi baca dapat dilepaskan ke replika baca, sehingga mengurangi beban pada simpul primer. Replika baca juga dapat menjaga terhadap kehilangan data dalam hal peristiwa kegagalan simpul cache primer.	3 September 2013
Dukungan Amazon Virtual Private Cloud (VPC) default	Dalam rilis ElastiCache ini, terintegrasi penuh dengan Amazon Virtual Private Cloud (VPC). Untuk pelanggan baru, klaster cache dibuat dalam Amazon VPC secara default. Untuk informasi selengkapnya, lihat Amazon VPCs dan ElastiCache keamanan.	8 Januari 2013
PHP mendukung penemuan otomatis simpul cache	Rilis awal penemuan otomatis simpul cache menyediakan dukungan untuk program Java. Dalam rilis ini, ElastiCache membawa dukungan penemuan otomatis node cache ke PHP.	2 Januari 2013

Perubahan	Deskripsi	Tanggal Diubah
Dukungan untuk Amazon Virtual Private Cloud (VPC)	Dalam rilis ini, ElastiCache cluster dapat diluncurkan di Amazon Virtual Private Cloud (VPC). Secara default, klaster cache pelanggan baru dibuat di Amazon VPC secara otomatis; pelanggan yang sudah ada dapat bermigrasi ke Amazon VPC sesuai kebutuhannya sendiri. Untuk informasi selengkapnya, lihat Amazon VPCs dan ElastiCache keamanan.	20 Desember 2012
Penemuan otomatis simpul cache dan versi mesin cache baru	ElastiCache menyediakan cache node auto discovery—kemampuan untuk program klien untuk secara otomatis menentukan semua node cache dalam sebuah cluster, dan untuk memulai dan memelihara koneksi ke semua node ini. Rilis ini juga menawarkan versi mesin cache baru: Memcached versi 1.4.14. Mesin cache baru ini menyediakan kemampuan penyeimbangan slab yang ditingkatkan, peningkatan performa dan skalabilitas yang signifikan, serta beberapa perbaikan bug. Ada beberapa parameter baru cache yang dapat dikonfigurasi. Untuk informasi selengkapnya, lihat Mengkonfigurasi parameter mesin menggunakan grup ElastiCache parameter.	28 November 2012
Jenis baru simpul cache	Rilis ini menyediakan empat jenis simpul cache tambahan.	13 November 2012
Simpul cache terpesan	Rilis ini menambahkan dukungan untuk simpul cache terpesan. q	5 April 2012
Panduan baru	Ini adalah rilis pertama Panduan ElastiCache Pengguna Amazon.	22 Agustus 2011

AWS Glosarium

Untuk AWS terminologi terbaru, lihat AWS glosarium di Referensi.Glosarium AWS

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.