



ElastiCache untuk Panduan Pengguna Memcached

Amazon ElastiCache



Versi API 2015-02-02

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon ElastiCache: ElastiCache untuk Panduan Pengguna Memcached

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan dalam hubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di kalangan pelanggan, atau dengan cara apa pun yang menghina atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan kekayaan masing-masing pemiliknya, yang mungkin berafiliasi, terkait dengan, atau disponsori oleh Amazon, atau mungkin tidak.

Table of Contents

Apa itu ElastiCache untuk Memcached?	1
Caching nirserver	1
Klaster yang dirancang sendiri	2
Cara kerjanya	2
Mesin cache dan caching	2
Kasus penggunaan	8
Penyimpanan Data Dalam Memori	8
Testimoni Pelanggan ElastiCache	10
ElastiCache for Memcached	10
Alat untuk mengelola implementasi Anda	12
Menggunakan AWS Management Console	12
Menggunakan AWS CLI	12
Menggunakan SDK AWS	12
Menggunakan API ElastiCache	12
Lihat juga	13
Memilih antara opsi deployment	13
Membandingkan cache yang dirancang sendiri dari Memcached dan Redis	14
Mulai menggunakan ElastiCache for Redis	20
Pengaturan	20
Mendaftar untuk Akun AWS	20
Membuat pengguna administratif	21
Memberikan akses terprogram	22
Menyiapkan izin	24
Menyiapkan EC2	25
Berikan akses jaringan	25
Buat cache	26
Buat cache nirserver	26
Membaca dan menulis data	27
(Opsional) Menghapus	32
Langkah Berikutnya	33
Tutorial: Mengonfigurasi fungsi Lambda untuk Mengakses Amazon ElastiCache di Amazon VPC	34
Langkah 1: Buat cache ElastiCache	34
Langkah 2: Buat fungsi Lambda	36

Langkah 3: Uji fungsi Lambda	40
Tutorial dan video	42
Video	43
Memilih wilayah dan zona ketersediaan	49
Pertimbangan terkait Zona Ketersediaan	50
Wilayah dan titik akhir yang didukung	52
Menempatkan simpul Anda	57
Menggunakan Local Zones	57
Mengaktifkan zona lokal	58
Menggunakan Outposts	58
Menggunakan Outposts dengan konsol Memcached	59
Menggunakan Outposts dengan CLI AWS	60
Merancang kluster ElastiCache Anda sendiri	62
ElastiCache untuk komponen dan fitur Memcached	63
Simpul	63
Kluster	64
AWS Wilayah dan zona ketersediaan	65
Titik akhir	66
Grup parameter	66
Keamanan	67
Grup subnet	67
Peristiwa	67
Mengelola kluster	68
Memilih jenis jaringan	70
Penemuan Otomatis	72
Menyiapkan kluster	115
Membuat kluster	122
Melihat detail kluster	125
Mengubah kluster	130
Melakukan boot ulang kluster	134
Menambahkan simpul ke kluster	136
Menghapus simpul dari kluster	142
Membatalkan operasi penambahan atau penghapusan simpul yang tertunda	148
Menghapus kluster	149
Mengakses kluster Anda	152
Menemukan titik akhir koneksi	158

Mengelola simpul	166
Melihat Status ElastiCache Node	166
Menghubungkan ke simpul	172
Tipe simpul yang didukung	175
Mengganti simpul	185
Simpul terpesan	187
Memigrasikan simpul generasi sebelumnya	199
Bekerja dengan ElastiCache	201
Versi mesin dan peningkatannya	201
Versi Memcached yang didukung	202
Versi mesin dan pemutakhiran	206
Cara meningkatkan versi mesin	207
Praktik terbaik dan strategi caching	209
Praktik terbaik dengan klien Memcached	209
Perintah Memcached yang didukung	217
Strategi Pembuatan Cache	218
Mengelola klaster yang dirancang sendiri	224
Mengelola pemeliharaan	224
Mengonfigurasi parameter mesin menggunakan grup parameter	226
Penskalaan ElastiCache untuk Memcached	269
Penskalaan ElastiCache untuk Memcached	269
Menetapkan batas penskalaan untuk mengelola biaya	269
Pra-penskalaan dengan Tanpa Server ElastiCache	269
Mengatur batas penskalaan menggunakan konsol dan AWS CLI	271
Penskalaan ElastiCache untuk cluster yang dirancang sendiri Memcached	272
Menandai sumber daya ElastiCache Anda	276
Memantau biaya dengan tanda	283
Mengelola tanda menggunakan AWS CLI	285
Mengelola tanda menggunakan API ElastiCache	288
Lensa Amazon ElastiCache Well-Architected	291
Pilar Keunggulan Operasional	292
Pilar Keamanan	301
Pilar Keandalan	307
Pilar Efisiensi Kinerja	313
Pilar Optimisasi Biaya	324
Memecahkan masalah	330

Masalah koneksi	330
Kesalahan klien Redis	331
Memecahkan masalah latensi tinggi di Tanpa Server ElastiCache	332
Memecahkan masalah pembatasan di Tanpa Server ElastiCache	333
Topik-Topik Terkait	334
Langkah pemecahan masalah tambahan	334
Grup keamanan	335
ACL jaringan	335
Tabel rute	337
Resolusi DNS	337
Mengidentifikasi masalah dengan diagnostik sisi server	338
Validasi konektivitas jaringan	344
Batas terkait jaringan	346
Penggunaan CPU	347
Koneksi yang dihentikan dari sisi server	351
Pemecahan masalah sisi klien untuk instans Amazon EC2	352
Membedah waktu yang dibutuhkan untuk menyelesaikan satu permintaan tunggal	353
Keamanan	356
Perlindungan data	357
Keamanan data di Amazon ElastiCache	357
Privasi lalu lintas kerja internet	367
Amazon VPC dan keamanan ElastiCache	367
API Amazon ElastiCache dan titik akhir VPC antarmuka (AWS PrivatLink)	391
Subnet dan grup subnet	394
Pengelolaan Identitas dan Akses	402
Audiens	402
Mengautentikasi dengan identitas	403
Mengelola kebijakan menggunakan akses	407
Cara Amazon ElastiCache bekerja dengan IAM	409
Contoh kebijakan berbasis identitas	417
Pemecahan Masalah	420
Kontrol akses	422
Gambaran umum pengelolaan akses	423
Validasi kepatuhan	455
Informasi selengkapnya	457
Ketahanan	457

Mitigasi Kegagalan	457
Keamanan infrastruktur	460
Pembaruan layanan	460
Mengelola pembaruan layanan	460
Pencatatan dan pemantauan	467
Metrik dan peristiwa nirserver	467
Metrik nirserver	467
Peristiwa nirserver	471
Metrik dan peristiwa yang dirancang sendiri	475
Metrik klaster yang dirancang sendiri	476
Peristiwa klaster yang dirancang sendiri	476
Pemantauan penggunaan	484
Pemantauan peristiwa Amazon SNS	498
Logging panggilan API Amazon ElastiCache dengan AWS CloudTrail	514
Informasi Amazon ElastiCache di CloudTrail	515
Memahami entri file log Amazon ElastiCache	516
Kuota	520
Referensi	521
Menggunakan API ElastiCache	521
Menggunakan API kueri	521
Pustaka yang tersedia	525
Memecahkan masalah aplikasi	525
Siapkan AWS CLI untuk ElastiCache	526
Prasyarat	527
Mendapatkan alat baris perintah	528
Menyiapkan alat	529
Memberikan kredensial untuk alat	530
Variabel lingkungan	531
Pesan kesalahan	532
Notifikasi	533
Notifikasi ElastiCache Umum	533
Notifikasi ElastiCache for Memcached	534
.....	535
AWS Glosarium	552
.....	dliii

Apa itu Amazon ElastiCache untuk Memcached?

Selamat datang di Amazon ElastiCache untuk Panduan Pengguna Memcached. 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 berkinerja tinggi, dapat diskalakan, dan hemat biaya. Layanan ini juga membantu menghilangkan kompleksitas yang terkait deployment dan pengelolaan 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 Redis dan Memcached. Gunakan panduan untuk mesin yang sesuai. Jika Anda tidak yakin mesin yang ingin digunakan, lihat [Membandingkan cache yang dirancang sendiri dari Memcached dan Redis](#) pada panduan ini.

Caching nirserver

ElastiCache untuk Memcached menawarkan caching tanpa server, yang menyederhanakan penambahan dan pengoperasian cache berbasis Memcached untuk aplikasi Anda. ElastiCache untuk Memcached 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 dan komputasi yang digunakan oleh aplikasi Anda, dan secara otomatis menskalakan kapasitas untuk memenuhi kebutuhan aplikasi Anda. ElastiCache menawarkan pengalaman endpoint sederhana bagi pengembang, dengan mengabstraksi infrastruktur dan perangkat lunak cache yang mendasarinya. ElastiCache mengelola penyediaan perangkat keras, pemantauan, penggantian node, dan penambalan perangkat lunak secara otomatis dan transparan, sehingga Anda dapat memfokuskan pengembangan aplikasi, daripada mengoperasikan cache.

ElastiCache untuk Memcached Serverless kompatibel dengan Memcached 1.6 dan di atasnya.

Merancang sendiri ElastiCache untuk cluster Memcached

Jika Anda memerlukan kontrol berbutir halus atas cluster ElastiCache untuk Memcached Anda, Anda dapat memilih untuk mendesain cluster Memcached 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 adalah layanan yang dikelola sepenuhnya, secara otomatis mengelola penyediaan perangkat keras, pemantauan, penggantian node, dan penambalan perangkat lunak untuk cluster Anda.

Merancang sendiri ElastiCache untuk cluster Memcached menawarkan fleksibilitas dan kontrol yang lebih besar atas cluster Anda. Misalnya, Anda dapat memilih untuk mengoperasikan klaster dengan ketersediaan Single-AZ atau ketersediaan cross-AZ tergantung kebutuhan Anda. 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 klaster Memcached Anda.

Saat mendesain sendiri ElastiCache untuk Memcached, Anda dapat memilih untuk menjalankan Memcached 1.4 ke atas.

Cara kerjanya

Di sini Anda dapat menemukan ikhtisar komponen utama dari penerapan ElastiCache untuk Memcached.

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 untuk Memcached menawarkan dua opsi penerapan: Cluster tanpa server dan yang dirancang sendiri. Lihat [Memilih antara opsi deployment](#)

Note

Amazon ElastiCache bekerja dengan mesin Redis dan Memcached. Gunakan panduan untuk mesin yang sesuai. Jika Anda tidak yakin mesin yang ingin digunakan, lihat [Membandingkan cache yang dirancang sendiri dari Memcached dan Redis](#) pada panduan ini.

Topik

- [Cara ElastiCache kerja Memcached](#)
- [Dimensi harga](#)

Cara ElastiCache kerja Memcached

ElastiCache untuk Memcached Tanpa Server

ElastiCache untuk Memcached Serverless 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 Memcached Anda untuk mulai mengakses cache Anda.

Note

ElastiCache untuk Memcached Serverless hanya kompatibel dengan klien Memcached yang mendukung TLS.

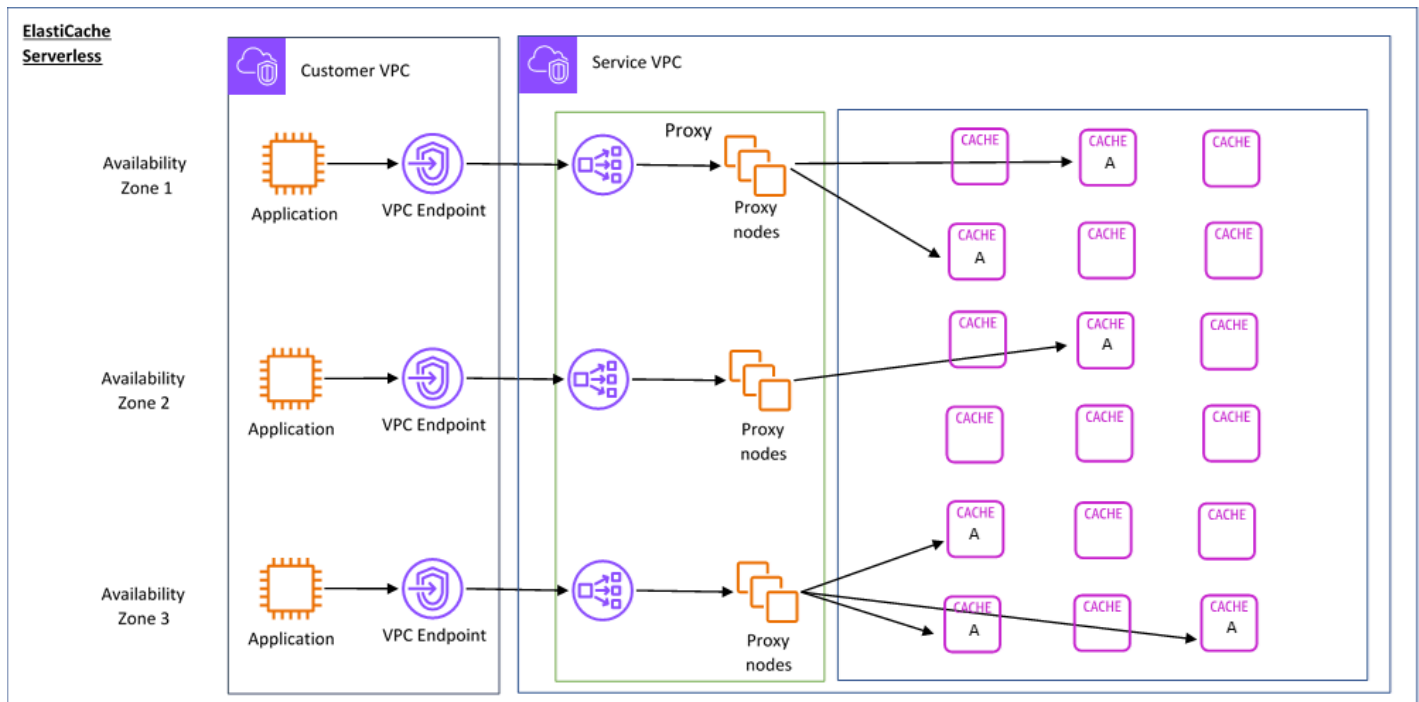
Manfaat Utama

- 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. Secara otomatis memantau simpul cache yang mendasarinya dan menggantikannya 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 Memcached baru tersedia, ElastiCache akan mengirimkan pemberitahuan.

- Keamanan: Nirserver selalu mengenkripsi data bergerak dan saat tidak aktif. 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 proxy membantu mengurangi konfigurasi klien yang kompleks, karena klien tidak perlu menemukan kembali topologi kluster jika terjadi perubahan pada kluster 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 proxy oleh penyeimbang beban jaringan. Ketika aplikasi Anda mengeksekusi perintah cache, simpul proxy yang terhubung ke aplikasi Anda mengeksekusi permintaan di simpul cache di cache Anda. Lapisan proxy mengabstraksi topologi

kluster 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.

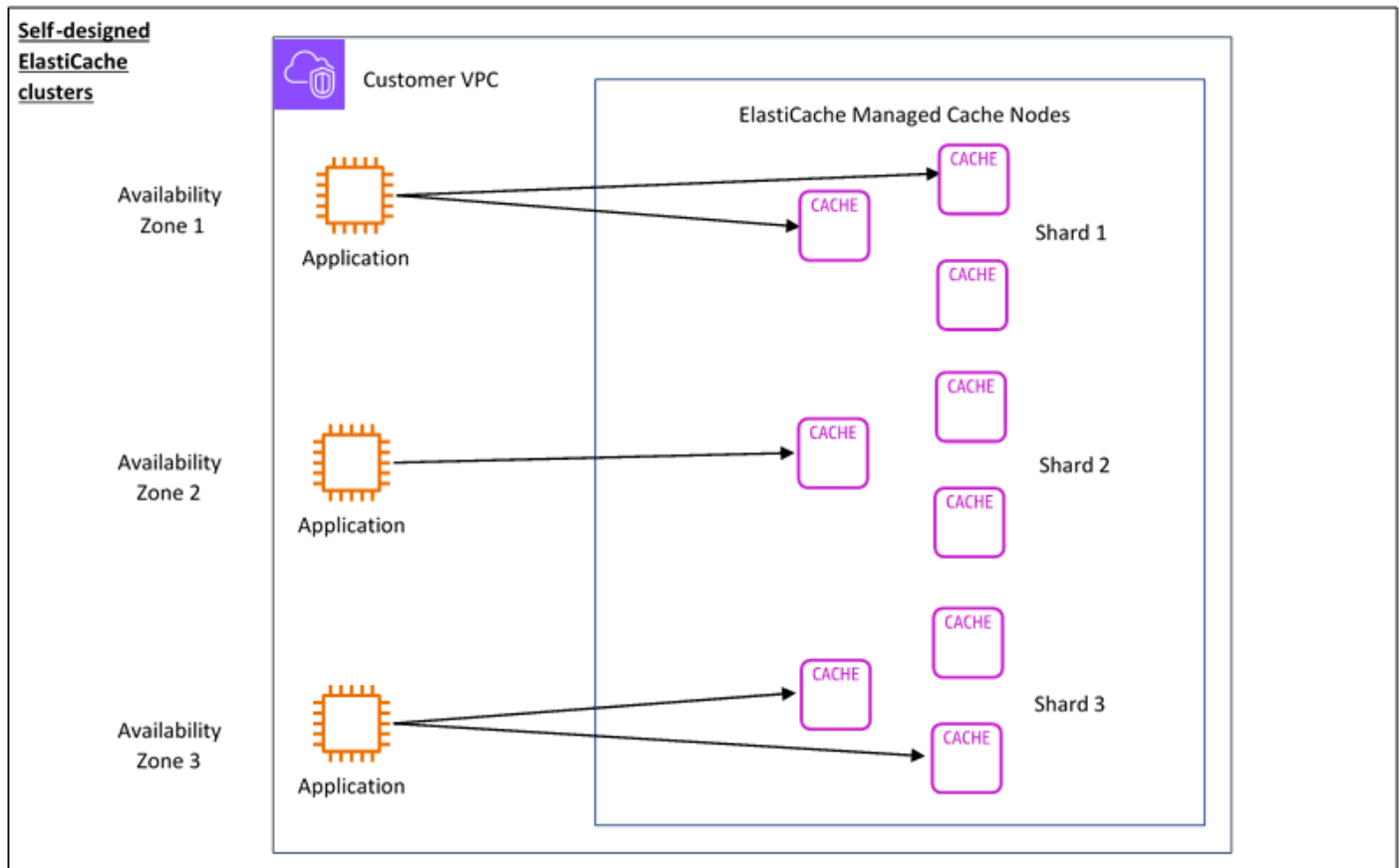
Cluster yang dirancang sendiri ElastiCache

Anda dapat memilih untuk mendesain ElastiCache cluster Anda sendiri dengan memilih keluarga node cache, ukuran, dan jumlah node untuk cluster Anda. Merancang kluster Anda sendiri membuat Anda dapat mengontrol konfigurasi dan penskalaan kluster dengan lebih baik.

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. Anda juga dapat mendesain kluster Anda dengan simpul di beberapa AZ untuk mencapai ketersediaan tinggi.
- **Kontrol terperinci:** Saat mendesain kluster sendiri, Anda memiliki kontrol yang lebih besar dalam menyempurnakan pengaturan cache Anda. Misalnya, Anda dapat menggunakan [Mengonfigurasi parameter mesin menggunakan grup parameter](#) untuk mengonfigurasi mesin cache.
- **Menskalakan secara vertikal dan horizontal:** Anda dapat memilih untuk menskalakan kluster Anda secara manual dengan menambah atau mengurangi ukuran simpul cache saat diperlukan. Anda juga dapat menskalakan secara horizontal dengan menambahkan simpul.

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 sendiri ElastiCache untuk cluster Memcached, 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 Pemrosesan (ECPU)

Anda membayar permintaan yang dijalankan aplikasi Anda di ElastiCache Tanpa Server di Unit ElastiCache Pemrosesan (ECPU), 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 mengonsumsi 3,2 ECPU.
- Perintah yang beroperasi pada beberapa item akan menggunakan lebih banyak ECPU secara proporsional. Misalnya, jika aplikasi Anda melakukan multiget pada 3 item, aplikasi akan menggunakan 3 ECPU.
- Perintah yang beroperasi pada lebih banyak item dan mentransfer lebih banyak data akan menggunakan ECPU berdasarkan dimensi yang lebih tinggi. 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 data yang ditransfer hanya berukuran 2 KB, perintah akan mengonsumsi 3 ECPU.

ElastiCache Tanpa server memancarkan metrik baru yang disebut yang membantu Anda memahami ECPU `ElastiCacheProcessingUnits` yang dikonsumsi oleh beban kerja Anda.

Jam simpul

Anda dapat memilih untuk mendesain kluster cache Anda sendiri dengan memilih keluarga simpul EC2, ukuran, jumlah simpul, dan penempatan di seluruh Zona Ketersediaan. Saat mendesain sendiri kluster Anda, Anda membayar per jam untuk setiap simpul cache.

Kasus Penggunaan ElastiCache Umum dan Cara ElastiCache Dapat Membantu

Baik ketika menyajikan berita terkini atau katalog produk maupun menjual tiket sebuah 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 data dari cache dalam memori. Pemuatan yang terasa lebih lambat cenderung membuat pelanggan beralih ke bisnis lain.

Contoh berikut mengilustrasikan beberapa cara penggunaan ElastiCache untuk meningkatkan performa aplikasi Anda secara keseluruhan.

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 lebih mahal daripada menemukan kunci dalam cache pasangan nilai kunci. Beberapa kueri basis data khususnya mahal 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 tentang IAM, lihat hal berikut:

- [Strategi Caching](#) dalam Panduan Pengguna ElastiCache for Memcached

Testimoni Pelanggan ElastiCache

Untuk mempelajari bagaimana bisnis seperti Airbnb, PBS, Esri, dan lainnya menggunakan Amazon ElastiCache untuk mengembangkan bisnis mereka dengan pengalaman pelanggan yang disempurnakan, lihat [Cara Orang Lain Menggunakan Amazon ElastiCache](#).

Anda juga dapat menonton [Video tutorial](#) ini untuk kasus penggunaan pelanggan ElastiCache lainnya.

ElastiCache for Memcached

Sebaiknya mulai dengan membaca bagian berikut, dan rujuk bagian tersebut saat diperlukan:

- Sorotan dan harga layanan – [Halaman detail produk](#) memberikan gambaran produk ElastiCache secara umum, sorotan layanan, dan harga.
- Video ElastiCache – Bagian [Video ElastiCache](#) memiliki video yang memperkenalkan Amazon ElastiCache for Memcached, yang meliputi kasus penggunaan umum, serta demo cara menggunakan ElastiCache for Memcached untuk mengurangi latensi dan meningkatkan throughput aplikasi Anda.
- Memulai – Bagian [Mulai menggunakan Amazon ElastiCache for Redis](#) memiliki contoh yang memandu Anda melalui proses membuat klaster cache, memberikan otorisasi akses pada klaster cache, menyambung ke simpul cache, dan menghapus klaster cache.
- Performa sesuai skala – [Performa sesuai skala dengan Amazon ElastiCache](#) laporan resmi yang membahas strategi caching yang mengaktifkan aplikasi Anda bekerja dengan baik pada skalanya.

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

- [Dokumentasi AWS Command Line Interface](#)

Bagian ini memberikan informasi tentang mengunduh AWS CLI, membuat AWS CLI bekerja pada sistem Anda, dan memberikan kredensial AWS Anda.

- [Dokumentasi AWS CLI untuk ElastiCache](#)

Dokumen terpisah ini mencakup semua hal mengenai AWS CLI untuk perintah ElastiCache, termasuk sintaks dan contoh.

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

- [Alat untuk Amazon Web Services](#)

Amazon Web Services menyediakan sejumlah kit pengembangan perangkat lunak (SDK) dengan dukungan untuk ElastiCache for Memcached. Anda dapat membuat kode untuk ElastiCache menggunakan Java, .NET, PHP, Ruby, dan bahasa lainnya. SDK ini dapat secara signifikan menyederhanakan pengembangan aplikasi Anda dengan memformat permintaan Anda untuk ElastiCache, melakukan parsing tanggapan, serta memberikan logika coba lagi dan penanganan kesalahan.

- [Menggunakan API ElastiCache](#)

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

- [Referensi API Amazon ElastiCache](#)

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

Alat untuk mengelola implementasi Anda

Setelah Anda memberikan akses instans Amazon EC2 ke kluster ElastiCache Anda, ada empat cara agar Anda dapat mengelola kluster ElastiCache Anda: AWS Management Console, AWS CLI untuk ElastiCache, SDK AWS untuk ElastiCache, dan API ElastiCache.

Menggunakan AWS Management Console

AWS Management Console adalah cara termudah untuk mengelola Amazon ElastiCache for Memcached. Konsol memungkinkan Anda membuat kluster cache, menambah dan menghapus simpul cache, dan melakukan tugas administratif lain tanpa harus menulis kode apapun. Konsol tersebut juga menyediakan grafik performa simpul cache dari CloudWatch, yang menampilkan aktivitas mesin cache, memori dan pemanfaatan CPU, serta metrik lainnya. Untuk informasi lain, lihat topik spesifik di Panduan Pengguna ini.

Menggunakan AWS CLI

Anda juga dapat menggunakan AWS Command Line Interface (AWS CLI) untuk ElastiCache. AWS CLI memudahkan untuk melakukan operasi satu per satu, seperti memulai atau menghentikan kluster cache Anda. Anda juga dapat menginvokasi perintah AWS CLI untuk ElastiCache dari bahasa skrip pilihan Anda, memungkinkan Anda mengotomatiskan tugas berulang. Untuk informasi lebih lanjut tentang AWS CLI, lihat Panduan Pengguna dan [Referensi Perintah AWS CLI](#).

Menggunakan SDK AWS

Jika Anda ingin mengakses ElastiCache dari aplikasi, Anda dapat menggunakan salah satu kit pengembangan perangkat lunak (SDK) AWS. SDK membungkus panggilan API ElastiCache, dan mengisolasi aplikasi Anda dari perincian tingkat rendah dari API ElastiCache. Anda menyediakan kredensial Anda, dan pustaka SDK menangani autentikasi dan penandatanganan permintaan. Untuk informasi lebih lanjut tentang penggunaan SDK AWS, lihat [Alat untuk Amazon Web Services](#).

Menggunakan API ElastiCache

Anda juga dapat menulis kode aplikasi secara langsung terhadap API layanan web ElastiCache. Saat menggunakan API, Anda harus menulis kode yang diperlukan untuk membangun dan melakukan autentikasi terhadap permintaan HTTP Anda, melakukan parsing hasil dari ElastiCache, dan menangani kesalahan apa pun. Untuk informasi selengkapnya tentang API, lihat [Menggunakan API ElastiCache](#).

Lihat juga

Untuk informasi lebih mendetail tentang cara mengelola deployment Amazon ElastiCache for Memcached Anda, lihat hal berikut:

- [Bekerja dengan ElastiCache](#)
- [Privasi lalu lintas kerja internet](#)
- [Logging dan pemantauan di Amazon ElastiCache](#)

Memilih antara opsi deployment

Amazon ElastiCache memiliki dua opsi deployment:

- Caching nirserver
- Merancang klaster Anda sendiri

Caching nirserver

Amazon ElastiCache Nirserver menyederhanakan pembuatan cache dan menskalakan secara instan untuk mendukung aplikasi pelanggan yang paling menuntut. Dengan ElastiCache Nirserver, Anda dapat membuat cache yang sangat tersedia dan dapat diskalakan dalam waktu kurang dari satu menit, menghilangkan kebutuhan untuk penyediaan, perencanaan, dan pengelolaan kapasitas klaster cache. ElastiCache Nirserver secara otomatis menyimpan data secara redundan di tiga Zona Ketersediaan dan menyediakan 99,99% ketersediaan [Perjanjian Tingkat Layanan \(SLA\)](#). ElastiCache menyediakan replikasi data otomatis di seluruh AZ, sehingga menghapus kebutuhan untuk mengelola replika dan perangkat lunak khusus secara manual agar tetap sinkron.

Klaster yang dirancang sendiri

Jika Anda memerlukan kontrol terperinci atas klaster ElastiCache for Memcached, Anda dapat memilih untuk merancang klaster Memcached Anda sendiri dengan ElastiCache. ElastiCache memungkinkan Anda mengoperasikan klaster berbasis simpul, dengan memilih tipe simpul, jumlah simpul, dan penempatan simpul di seluruh Zona Ketersediaan AWS untuk klaster Anda. Karena ElastiCache adalah layanan yang terkelola sepenuhnya, ElastiCache secara otomatis mengelola penyediaan perangkat keras, pemantauan, penggantian simpul, dan penambalan perangkat lunak untuk klaster Anda.

Memilih antara opsi deployment

Pilih caching Nirserver jika:

- Anda membuat cache baru untuk beban kerja baru atau tidak dikenal
- Anda memiliki lalu lintas aplikasi yang tidak dapat diprediksi
- Anda ingin cara termudah untuk memulai dengan cache

Pilih untuk merancang klaster ElastiCache Anda sendiri jika:

- Anda sudah menjalankan ElastiCache Nirserver dan menginginkan kontrol yang lebih terperinci atas jenis simpul yang menjalankan Memcached, jumlah simpul, dan penempatan simpul.
- Anda tidak mengharapkan lalu lintas aplikasi Anda berfluktuasi banyak atau Anda dapat secara akurat memprediksi puncak dan palung lalu lintas aplikasi Anda.
- Anda dapat memperkirakan persyaratan kapasitas Anda untuk mengontrol biaya.

Topik terkait:

- [Merancang dan mengelola klaster ElastiCache Anda sendiri untuk implementasi Memcached](#)

Membandingkan cache yang dirancang sendiri dari Memcached dan Redis

Amazon ElastiCache mendukung mesin cache Memcached dan Redis. 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 klaster cache atau grup replikasi, Anda dapat memutakhirkan versi mesin ke versi lebih baru, tetapi Anda tidak dapat menurunkan ke versi mesin yang lebih lama. Jika Anda ingin menggunakan versi mesin yang lama, Anda harus menghapus klaster cache atau grup replikasi yang sudah ada dan membuatnya lagi dengan versi mesin yang lama tersebut.

Secara umum, mesin yang ada terlihat serupa. Setiap mesin merupakan 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 menyimpan obyek ke dalam cache.

Pilih Redis dengan versi ElastiCache for Redis jika hal berikut berlaku untuk Anda:

- ElastiCache for Redis versi 7.0 (Ditingkatkan)

Anda ingin menggunakan [Redis Functions](#), [Sharded Pub/Sub](#), atau [Redis ACL improvements](#). Untuk informasi selengkapnya, lihat [Redis Versi 7.0 \(Ditingkatkan\)](#).

- ElastiCache for Redis versi 6.2 (Ditingkatkan)

Anda ingin kemampuan untuk mengatur tingkatan data antara memori dan SSD menggunakan tipe node r6gd. Untuk informasi selengkapnya, lihat [Tingkatan data](#).

- ElastiCache for Redis versi 6.0 (Ditingkatkan)

Anda ingin mengautentikasi pengguna dengan kontrol akses berbasis peran.

Untuk informasi selengkapnya, lihat [Redis Versi 6.0 \(Ditingkatkan\)](#).

- ElastiCache for Redis versi 5.0.0 (Ditingkatkan)

Anda ingin menggunakan [Redis streams](#), yakni struktur data log yang memungkinkan produsen untuk menambahkan item baru secara real-time dan juga memungkinkan konsumen untuk mengonsumsi pesan baik dalam mode blok atau non-blok.

Untuk informasi selengkapnya, lihat [Redis Versi 5.0.0 \(Ditingkatkan\)](#).

- ElastiCache for Redis versi 4.0.10 (Ditingkatkan)

Mendukung enkripsi serta secara dinamis menambahkan atau menghapus serpihan dari kluster Redis (mode kluster aktif).

Untuk informasi selengkapnya, lihat [Redis Versi 4.0.10 \(Ditingkatkan\)](#).

Versi berikut tidak digunakan lagi, telah mencapai atau segera mencapai akhir masa pakainya.

- ElastiCache for Redis versi 3.2.10 (Ditingkatkan)

Mendukung kemampuan untuk secara dinamis menambahkan atau menghapus serpihan dari kluster Redis (mode kluster aktif).

 Important

Saat ini ElastiCache for Redis 3.2.10 tidak mendukung enkripsi.

Untuk informasi selengkapnya, lihat berikut ini:

- [Redis versi 3.2.10 \(Ditingkatkan\)](#)
- Praktik terbaik resharding online untuk Redis, Untuk informasi selengkapnya, lihat referensi berikut:
 - [Praktik Terbaik: Resharding Online](#)
 - [Resharding dan Penyeimbangan Ulang Serpihan Secara Online untuk Redis \(Mode Kluster Aktif\)](#)
- Untuk informasi selengkapnya tentang penskalaan kluster Redis, lihat [Penskalaan](#).
- ElastiCache for Redis versi 3.2.6 (Ditingkatkan)

Jika Anda membutuhkan fungsionalitas dari versi Redis sebelumnya ditambah fitur berikut, pilih ElastiCache for Redis 3.2.6:

- Enkripsi saat dalam pengiriman. Untuk informasi selengkapnya, lihat [Enkripsi Saat Dalam Pengiriman Amazon ElastiCache for Redis](#).
- Enkripsi saat dalam penyimpanan. Untuk informasi selengkapnya, lihat [Enkripsi Saat Transit Amazon ElastiCache for Redis](#).
- ElastiCache for Redis (Mode kluster aktif) versi 3.2.4

Jika Anda membutuhkan fungsionalitas Redis 2.8.x ditambah fitur berikut, pilih Redis 3.2.4 (mode berkluster):

- Anda perlu membuat partisi data Anda di dua hingga 500 grup simpul (khusus mode berkluster).
- Anda membutuhkan pengindeksan geospasial (mode berkluster atau mode tanpa kluster).
- Anda tidak perlu mendukung beberapa basis data.
- ~~ElastiCache for Redis (mode tanpa kluster) 2.8.x dan 3.2.4 (Ditingkatkan)~~

Jika hal berikut berlaku untuk Anda, pilih Redis 2.8.x atau Redis 3.2.4 (mode tanpa klaster):

- 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 perlu mereplikasi data Anda dari primer ke satu atau beberapa replika baca untuk aplikasi dengan operasi baca intensif.
- Anda perlu melakukan failover otomatis jika simpul primer Anda gagal.
- Anda perlu mempublikasikan dan berlangganan kemampuan (pub/sub) — untuk memberitahu klien peristiwa di server.
- Anda memerlukan kemampuan pencadangan dan pemulihan.
- Anda perlu mendukung beberapa basis data.

Ringkasan perbandingan Memcached, Redis (mode kluster nonaktif), dan Redis (mode kluster aktif)

	Memcached	Redis (mode kluster nonaktif)	Redis (mode kluster aktif)
Engine versions+	1.4.5 and later	4.0.10 and later	4.0.10 and later
Data types	Simple	2.8.x - Complex * Complex	3.2.x and later - Complex
Data partitioning	Yes	No	Yes
Cluster is modifiable	Yes	Yes	3.2.10 and later - Limited
Online resharding	No	No	3.2.10 and later
Encryption	in-transit 1.6.12 and later	4.0.10 and later	4.0.10 and later
Data tiering	No	6.2 and later	6.2 and later
Sertifikasi kepatuhan			
Compliance Certification			
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	Yes	No	No
Node type upgrade	No	Yes	Yes
Engine upgrading	Yes	Yes	Yes

	Memcached	Redis (mode kluster nonaktif)	Redis (mode kluster aktif)
High availability (replication)	No	Yes	Yes
Automatic failover	No	Optional	Required
Pub/Sub capabilities	No	Yes	Yes
Sorted sets	No	Yes	Yes
Backup and restore	No	Yes	Yes
Geospatial indexing	No	4.0.10 and later	Yes

Catatan:

string, objects (like databases)

* string, sets, sorted sets, lists, hashes, bitmaps, hyperloglog

string, sets, sorted sets, lists, hashes, bitmaps, hyperloglog, geospatial indexes

+ Excludes versions which are deprecated, have reached or soon to reach end of life.

Setelah Anda memilih mesin untuk kluster Anda, sebaiknya gunakan versi terbaru mesin tersebut. Untuk informasi selengkapnya, lihat [Versi ElastiCache for Memcached yang Didukung](#) atau [Versi ElastiCache for Redis yang Didukung](#).

Mulai menggunakan Amazon ElastiCache for Redis

Topik di bagian ini memandu Anda melalui proses pembuatan, pemberian akses, koneksi ke, dan terakhir penghapusan cache nirsumber Memcached menggunakan konsol ElastiCache.

Topik

- [Pengaturan](#)
- [Langkah 1: Buat Cache](#)
- [Langkah 2: Baca dan tulis data ke cache](#)
- [Langkah 3: \(Opsional\) Hapus](#)
- [Langkah Berikutnya](#)
- [Tutorial: Mengonfigurasi fungsi Lambda untuk Mengakses Amazon ElastiCache di Amazon VPC](#)
- [Tutorial dan video ElastiCache](#)

Pengaturan

Untuk mengatur ElastiCache:

Topik

- [Mendaftar untuk Akun AWS](#)
- [Membuat pengguna administratif](#)
- [Memberikan akses terprogram](#)
- [Siapkan izin Anda \(hanya ElastiCache pengguna baru\)](#)
- [Menyiapkan EC2](#)
- [Berikan akses jaringan dari grup keamanan Amazon VPC ke cache Anda](#)

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/signup>.

2. Ikuti petunjuk secara online.

Anda akan diminta untuk menerima panggilan telepon dan memasukkan kode verifikasi pada keypad telepon sebagai bagian dari prosedur pendaftaran.

Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWS dibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya dalam akun. Sebagai praktik terbaik keamanan, [tetapkan akses administratif ke pengguna administratif](#), dan hanya gunakan pengguna root untuk melakukan [tugas yang memerlukan akses pengguna root](#).

AWS mengirim Anda email konfirmasi setelah proses pendaftaran selesai. Anda dapat melihat aktivitas akun saat ini dan mengelola akun dengan mengunjungi <https://aws.amazon.com/> dan memilih Akun Saya.

Membuat pengguna 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 Anda Pengguna root akun AWS

1. Masuk ke [AWS Management Console](#) sebagai pemilik akun dengan memilih pengguna Root dan masukkan alamat Akun AWS email Anda. Di halaman berikutnya, masukkan kata sandi Anda.

Untuk bantuan masuk menggunakan pengguna root, lihat [Masuk sebagai pengguna root](#) dalam Panduan Pengguna AWS Sign-In .

2. Aktifkan 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.

Membuat pengguna 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 sebuah pengguna administratif.

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 administratif

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

Memberikan akses terprogram

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 terprogram, pilih salah satu opsi berikut.

Pengguna mana yang membutuhkan akses terprogram?	Untuk	Oleh
Identitas tenaga kerja (Pengguna yang dikelola di Pusat Identitas IAM)	Gunakan kredensial sementara untuk menandatangani permintaan terprogram ke AWS CLI, AWS SDK, atau API. AWS	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan. <ul style="list-style-type: none"> • Untuk AWS CLI, lihat Mengkonfigurasi yang akan AWS CLI digunakan AWS IAM Identity Center dalam Panduan AWS Command Line Interface Pengguna. • Untuk AWS SDK, alat, dan AWS API, lihat otentikasi Pusat Identitas IAM di

Pengguna mana yang membutuhkan akses terprogram?	Untuk	Oleh
		Panduan Referensi AWS SDK dan Alat.
IAM	Gunakan kredensial sementara untuk menandatangani permintaan terprogram ke AWS CLI, AWS SDK, atau API. AWS	Mengikuti petunjuk dalam Menggunakan kredensial sementara dengan AWS sumber daya di Panduan Pengguna IAM.
IAM	(Tidak direkomendasikan) Gunakan kredensial jangka panjang untuk menandatangani permintaan terprogram ke AWS CLI, AWS SDK, atau API. AWS	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan. <ul style="list-style-type: none"> • Untuk mengetahui AWS CLI, lihat Mengautentikasi menggunakan kredensial pengguna IAM di Panduan Pengguna. AWS Command Line Interface • Untuk AWS SDK dan alat bantu, lihat Mengautentikasi menggunakan kredensial jangka panjang di Panduan Referensi AWS SDK dan Alat. • Untuk AWS API, 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, tambahkan izin ke pengguna, grup, atau peran Anda:

- Pengguna dan grup di AWS IAM Identity Center:

Buat rangkaian izin. Ikuti petunjuk dalam [Buat set izin](#) dalam Panduan Pengguna AWS IAM Identity Center .

- Pengguna yang dikelola di IAM melalui penyedia identitas:

Buat peran untuk federasi identitas. Ikuti petunjuk dalam [Membuat peran untuk penyedia identitas pihak ketiga \(federasi\)](#) di Panduan Pengguna IAM.

- Pengguna IAM:

- Buat peran yang dapat diambil pengguna Anda. Ikuti petunjuk dalam [Membuat peran untuk pengguna IAM](#) dalam Panduan Pengguna IAM.
- (Tidak disarankan) Pasang kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti petunjuk di [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 tertaut 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 \(IAM\)](#)
- [Kebijakan terkelola AWS untuk Amazon ElastiCache](#)
- [Menggunakan Peran Tertaut Layanan untuk Amazon ElastiCache](#)

Menyiapkan EC2

Anda perlu menyiapkan instans EC2 dari mana Anda akan terhubung ke cache Anda.

- Jika Anda belum memiliki instans EC2, pelajari cara menyiapkan instans EC2 di sini: [Memulai EC2](#).
- Instans EC2 Anda harus berada di 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 bahwa instans EC2 Anda ada di VPC default dan memiliki grup keamanan default.

Berikan akses jaringan dari grup keamanan Amazon VPC ke cache Anda

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

1. Masuk ke AWS Command Line Interface dan buka konsol [Amazon 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 Inbound, dan kemudian:
 - a. Pilih Edit.
 - b. Pilih Tambahkan aturan.
 - c. Pada kolom Jenis, pilih Aturan TCP khusus.
 - d. Di kotak rentang Port, ketik 11211.
 - e. Pada kotak Sumber, pilih Dari mana pun yang memiliki rentang port (0.0.0.0/0) sehingga setiap instans Amazon EC2 yang Anda luncurkan di dalam Amazon VPC Anda dapat terhubung ke cache Anda.
 - f. Jika Anda menggunakan ElastiCache tanpa server, tambahkan aturan lain dengan memilih Tambahkan aturan.
 - g. Pada kolom Jenis, pilih Aturan TCP Khusus.
 - h. Di kotak rentang Port, ketik 11212.

- i. Pada kotak Sumber, pilih Dari mana pun yang memiliki rentang port (0.0.0.0/0) sehingga setiap instans Amazon EC2 yang Anda luncurkan di dalam Amazon VPC Anda dapat terhubung ke cache Anda.
- j. Pilih Simpan

Langkah 1: Buat Cache

Cache yang akan Anda luncurkan akan aktif, dan tidak berjalan di sandbox. Anda akan dikenai biaya penggunaan ElastiCache standar untuk cache hingga cache ini dihapus. Jumlah biayanya cukup kecil (biasanya kurang dari satu dolar) jika Anda menyelesaikan latihan yang dijelaskan di sini dalam satu sesi dan menghapus cache tersebut ketika Anda sudah selesai. Untuk informasi selengkapnya tentang biaya penggunaan ElastiCache, lihat [Amazon ElastiCache](#).

Buat cache nirserver

AWS Management Console

Untuk membuat cache baru menggunakan konsol ElastiCache:

1. Masuk ke AWS Management Console dan buka konsol ElastiCache 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 menggunakan AWS CLI

Contoh AWS CLI 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 selesai membuat cache, gunakan perintah `describe-serverless-caches`.

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 [Langkah 2: Baca dan tulis data ke cache](#).

Langkah 2: Baca dan tulis data ke cache

Bagian ini mengasumsikan bahwa Anda telah membuat instans Amazon EC2 dan dapat menghubungkan ke instans tersebut. Untuk petunjuk cara melakukannya, lihat [Panduan Memulai Amazon EC2](#).

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

Konfigurasi

Sebelum Anda mulai, pastikan Anda memiliki port yang tepat yang tersedia untuk diakses.

Port utama: 11211

Port yang dioptimalkan baca: 11212

Cache Memcached tanpa server mengiklankan dua port dengan nama host yang sama. Port utama memungkinkan penulisan dan pembacaan dengan jaminan konsistensi yang sama seperti OSS Memcached. Port yang dioptimalkan untuk dibaca memungkinkan penulisan dan tambahan latensi rendah yang akhirnya konsisten.

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 kolom "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 [Enkripsi bergerak ElastiCache \(TLS\)](#)

Menghubungkan menggunakan klien Java Memcached

Untuk informasi tentang cara menghubungkan menggunakan klien Java Memcached, lihat [Enkripsi bergerak ElastiCache \(TLS\)](#)

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"
```

Menghubungkan menggunakan klien NodeJS/TS Memcached (memcache Electrode-IO)

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/memcache> dan <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 [Langkah 3: \(Opsional\) Hapus](#).

Langkah 3: (Opsional) Hapus

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 konsol Amazon ElastiCache di <https://console.aws.amazon.com/elasticache/>.
2. Pada dasbor konsol ElastiCache, pilih mesin yang dijalankan cache yang ingin dihapus. 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 dalam satu waktu dari konsol ElastiCache. 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 kluster.
6. Jika Anda memilih Hapus, status cache berubah menjadi menghapus.

Setelah cache Anda masuk ke status PENGHAPUSAN, Anda tidak lagi dikenakan biaya untuk itu.

Menggunakan AWS CLI

Kode berikut menghapus cache my-cache.

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

Tindakan CLI `delete-serverless-cache` hanya menghapus satu cache nirserver. Untuk menghapus beberapa cache, panggil `delete-serverless-cache` untuk setiap cache nirserver yang ingin dihapus. Anda tidak perlu menunggu satu cache nirserve 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 AWS CLI untuk topik ElastiCache `delete-serverless-cache`.

Anda sekarang dapat melanjutkan ke [Langkah Berikutnya](#).

Langkah Berikutnya

Untuk informasi selengkapnya tentang ElastiCache, lihat:

- [Bekerja dengan ElastiCache](#)
- [Penskalaan ElastiCache untuk Memcached](#)
- [Kuota untuk ElastiCache](#)
- [Praktik terbaik ElastiCache dan strategi caching](#)
- [Melihat peristiwa ElastiCache](#)

Tutorial: Mengonfigurasi fungsi Lambda untuk Mengakses Amazon ElastiCache di Amazon VPC

Dalam tutorial ini, Anda akan melakukan hal-hal berikut:

- Buat cache Amazon ElastiCache di Amazon Virtual Private Cloud (Amazon VPC) default Anda di wilayah us-east-1.
- Buat fungsi Lambda untuk mengakses cache ElastiCache. Saat membuat fungsi Lambda, Anda perlu menyediakan ID subnet di Amazon VPC dan grup keamanan VPC agar fungsi Lambda dapat mengakses sumber daya di VPC Anda. Sebagai ilustrasi dalam tutorial ini, fungsi Lambda menghasilkan UUID, menuliskannya ke cache, dan mengambilnya dari cache.
- Invokasi fungsi Lambda secara manual dan verifikasi bahwa fungsi tersebut mengakses cache ElastiCache di VPC Anda.

Important

Tutorial ini menggunakan Amazon VPC default di wilayah us-east-1 di akun Anda. Untuk informasi selengkapnya tentang Amazon VPC, lihat [Cara Mulai Menggunakan Amazon VPC](#) dan Panduan Pengguna Amazon VPC.

Topik

- [Langkah 1: Buat cache ElastiCache](#)
- [Langkah 2: Buat fungsi Lambda](#)
- [Langkah 3: Uji fungsi Lambda](#)

Mulai

[Langkah 1: Buat cache ElastiCache](#)

Langkah 1: Buat cache ElastiCache

Pada langkah ini, Anda membuat cache Amazon ElastiCache di Amazon Virtual Private Cloud default di wilayah us-east-1 di akun Anda menggunakan AWS CLI. Untuk informasi tentang pembuatan cache nirservers ElastiCache menggunakan konsol atau API ElastiCache, lihat [Membuat kluster](#) dalam Panduan Pengguna ElastiCache for Memcached.

AWS Management Console

Jalankan perintah AWS CLI berikut untuk membuat cache nirserver klaster Memcached baru di VPC default di wilayah us-east-1.

Linux

```
aws elasticache create-serverless-cache \  
--serverless-cache-name serverlessCacheForLambda \  
--region us-east-1 \  
--engine memcached
```

Windows

```
aws elasticache create-serverless-cache ^  
--serverless-cache-name serverlessCacheForLambda ^  
--region us-east-1 ^  
--engine memcached
```

Perhatikan bahwa nilai bidang Status diatur ke CREATING. Diperlukan waktu beberapa menit hingga ElastiCache menyelesaikan pembuatan klaster Anda.

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

Linux

```
aws elasticache describe-serverless-caches \  
--serverless-cache-name serverlessCacheforLambda \  
--region us-east-1
```

Windows

```
aws elasticache describe-serverless-caches ^  
--serverless-cache-name serverlessCacheforLambda ^  
--region us-east-1
```

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

Setelah membuat cache baru, lanjutkan ke [Langkah 2: Buat fungsi Lambda](#).

Langkah Selanjutnya:

[Langkah 2: Buat fungsi Lambda](#)

Langkah 2: Buat fungsi Lambda

Dalam langkah ini, lakukan hal berikut:

1. Buat paket deployment fungsi Lambda menggunakan contoh kode yang disediakan.
2. Buat peran IAM (peran eksekusi). Pada saat Anda mengunggah paket deployment, Anda perlu menentukan peran ini agar Lambda dapat mengambil peran ini lalu menjalankan fungsi tersebut atas nama Anda. Kebijakan izin memberi AWS Lambda izin untuk menyiapkan antarmuka jaringan elastis atau ENI guna mengaktifkan fungsi Lambda Anda untuk mengakses sumber daya di VPC. Dalam contoh ini, fungsi Lambda Anda mengakses kluster ElastiCache di VPC.
3. Buat fungsi Lambda dengan mengunggah paket deployment.

Langkah Selanjutnya

[Langkah 2.1: Buat paket deployment](#)

Langkah 2.1: Buat paket deployment

Saat ini contoh kode untuk fungsi Lambda hanya disediakan dalam Python.

Python

Contoh kode Python berikut membaca dan menuliskan item ke kluster ElastiCache Anda. Salin kode tersebut dan simpan ke dalam file bernama `app.py`. Pastikan untuk mengganti nilai `elasticache_config_endpoint` dalam kode dengan alamat titik akhir yang Anda salin pada langkah 1.

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

elasticache_config_endpoint = "serverlesscacheforlambda-ces85m.serverless.use1.cache.amazonaws.com"
target_port = 11211

context = ssl.create_default_context()
```

```
memcached_client = Client((elasticsearch_config_endpoint, target_port),
    tls_context=context)

def lambda_handler(event, context):

    # create a random UUID - this will be the sample element we add to the cache
    uuid_in = uuid.uuid4().hex

    # put the UUID to the cache
    memcached_client.set("uuid", uuid_in, expire=500, noreply=False)

    # get the item (UUID) from the cache
    result = memcached_client.get("uuid")
    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 Memcached.")
    else:
        raise Exception(f"Bad value retrieved. Expected {uuid_in}, got
        {decoded_result}")

    return "Fetched value from Memcached"
```

Kode ini menggunakan pustaka Python [pymemcache](#) untuk memasukkan item ke dalam cache Anda dan mengambilnya. Untuk membuat paket deployment yang berisi pymemcache, lakukan langkah-langkah berikut.

1. Di direktori proyek Anda yang berisi file kode sumber `app.py`, buat folder `package` untuk menginstal pustaka `pymemcache`.

```
mkdir package
```

2. Instal `pymemcache` menggunakan `pip`.

```
pip install --target ./package pymemcache
```

3. Buat file `.zip` yang berisi pustaka `pymemcache`. Di Linux atau macOS, jalankan perintah CLI berikut. Di Windows, gunakan utilitas `zip` pilihan Anda untuk membuat file `.zip` dengan pustaka `pymemcache` di root.

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

4. 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 Selanjutnya

[Langkah 2.2: Buat peran IAM \(peran eksekusi\)](#)

Langkah 2.2: Buat peran IAM (peran eksekusi)

Pada langkah ini, Anda membuat peran AWS Identity and Access Management (IAM) menggunakan jenis peran dan kebijakan akses standar berikut ini:

- Peran layanan AWS dari jenis AWS Lambda – Peran ini memberi AWS Lambda izin untuk mengambil peran.
- AWSLambdaVPCLambdaAccessExecutionRole – Ini adalah kebijakan izin akses yang Anda lampirkan pada peran tersebut. Kebijakan ini memberikan izin untuk tindakan EC2 yang diperlukan AWS Lambda untuk mengelola ENI. Anda dapat melihat kebijakan yang dikelola AWS ini di konsol IAM.

Untuk informasi selengkapnya tentang peran pengguna IAM, lihat [Peran \(Delegasi dan Federasi\)](#) dalam Panduan Pengguna IAM.

Gunakan prosedur berikut untuk membuat peran IAM.

Untuk membuat peran IAM (eksekusi).

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Pilih Peran lalu Buat peran.
 - Untuk Jenis entitas tepercaya, pilih Layanan AWS, dan untuk Kasus penggunaan, pilih Lambda. Tindakan ini memberi layanan AWS Lambda izin untuk mengambil peran. Pilih Berikutnya.

- Di bagian Tambahkan izin, cari **AWSLambdaVPCAccessExecutionRole** dan pilih kotak centang di samping kebijakan.
 - Pilih Berikutnya.
 - Di Nama Peran, gunakan nama yang unik dalam akun AWS Anda (misalnya, lambda-vpc-execution-role).
 - Pilih Buat peran.
3. Salin ARN peran. Anda akan membutuhkannya pada langkah berikutnya saat membuat fungsi Lambda.

Langkah Selanjutnya

[Langkah 2.3: Unggah paket deployment \(buat fungsi Lambda\)](#)

Langkah 2.3: Unggah paket deployment (buat fungsi Lambda)

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

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

Untuk opsi `role`, gunakan ARN peran eksekusi yang Anda buat di langkah 2.2. Untuk `vpc-config`, masukkan daftar yang dipisahkan koma yang berisi subnet VPC default dan ID grup keamanan VPC default Anda. Anda dapat menemukan nilai-nilai ini di [konsol Amazon VPC](#). Untuk menemukan subnet VPC default Anda, pilih VPC Anda, lalu pilih VPC default Akun AWS Anda. Untuk menemukan grup keamanan untuk VPC ini, di bagian Keamanan, pilih Grup keamanan. Pastikan Anda memilih wilayah `us-east-1`.

Untuk Linux, macOS, atau Unix:

```
aws lambda create-function \
--function-name AccessMemcached \
--region us-east-1 \
--zip-file fileb://my_deployment_package.zip \
--role arn:aws:iam::123456789012:role/lambda-vpc-execution-role \
--handler app.lambda_handler \
--runtime python3.11 \
--timeout 30 \
```

```
--vpc-config SubnetIds=comma-separated-vpc-subnet-ids,SecurityGroupIds=default-security-group-id \
```

Untuk Windows:

```
aws lambda create-function ^  
--function-name AccessMemcached ^  
--region us-east-1 ^  
--zip-file fileb://path-to/my_deployment_package.zip ^  
--role arn:aws:iam::123456789012:role/lambda-vpc-execution-role ^  
--handler app.lambda_handler ^  
--runtime python3.11 ^  
--timeout 30 ^  
--vpc-config SubnetIds=comma-separated-vpc-subnet-ids,SecurityGroupIds=default-security-group-id ^
```

Jika perlu, Anda dapat mengunggah file .zip ke bucket Amazon S3 di wilayah AWS yang sama, lalu menentukan nama bucket dan objek pada perintah di atas. Anda perlu mengganti parameter `--zip-file` dengan parameter `--code`, seperti yang ditunjukkan berikut ini:

```
--code S3Bucket=bucket-name,S3Key=zip-file-object-key
```

Anda juga dapat membuat fungsi Lambda menggunakan konsol AWS Lambda. Saat membuat fungsi, pilih VPC untuk Lambda lalu pilih subnet dan grup keamanan dari bidang yang tersedia.

Langkah Selanjutnya

[Langkah 3: Uji fungsi Lambda](#)

Langkah 3: Uji fungsi Lambda

Pada langkah ini, Anda menginvokasi fungsi Lambda secara manual menggunakan perintah `invoke`. Saat fungsi Lambda dijalankan, fungsi tersebut menghasilkan UUID dan menuliskannya ke kluster ElastiCache yang Anda tentukan dalam kode Lambda Anda. Fungsi Lambda kemudian mengambil item dari cache.

1. Invokasi fungsi Lambda (AccessMemCache) menggunakan perintah AWS Lambda `invoke`.

Untuk Linux, macOS, atau Unix:

```
aws lambda invoke \
```

```
--function-name AccessMemCache \  
--region us-east-1 \  
output.txt
```

Untuk Windows:

```
aws lambda invoke ^  
--function-name AccessMemCache ^  
--region us-east-1 ^  
output.txt
```

2. Verifikasikan bahwa fungsi Lambda berhasil dijalankan sebagai berikut:

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

```
Success: Inserted 05fcf2e4d6c942209acc89ea79b5b15e. Fetched  
05fcf2e4d6c942209acc89ea79b5b15e from Memcached.
```

- Tinjau hasilnya di konsol AWS Lambda.

Tutorial dan video ElastiCache

Tutorial berikut membahas tugas yang menarik bagi pengguna Amazon ElastiCache.

- [Video ElastiCache](#)
- [Tutorial: Mengonfigurasi Fungsi Lambda untuk Mengakses Amazon ElastiCache di Amazon VPC](#)

Video ElastiCache

Pada bagian berikut ini, Anda dapat menemukan video untuk membantu Anda mempelajari konsep dasar dan lanjutan Amazon ElastiCache. Untuk informasi tentang Pelatihan AWS, lihat [Pelatihan & Sertifikasi AWS](#).

Topik

- [Video Pengantar](#)
- [Video Lanjutan](#)

Video Pengantar

Video berikut memberikan pengantar Amazon ElastiCache.

Topik

- [AWS re:Invent 2020: Yang baru di Amazon ElastiCache](#)
- [AWS re:Invent 2019: Yang baru di Amazon ElastiCache](#)
- [AWS re:Invent 2017: Yang baru di Amazon ElastiCache](#)
- [DAT204—Membuat Aplikasi yang Dapat Diskalakan di Layanan AWS NoSQL \(re:Invent 2015\)](#)
- [DAT207—Mempercepat Performa Aplikasi dengan Amazon ElastiCache \(re:Invent 2013\) AWS](#)

AWS re:Invent 2020: Yang baru di Amazon ElastiCache

[AWS re:Invent 2020: Yang baru di Amazon ElastiCache](#)

AWS re:Invent 2019: Yang baru di Amazon ElastiCache

[AWS re:Invent 2019: Yang baru in Amazon ElastiCache](#)

AWS re:Invent 2017: Yang baru di Amazon ElastiCache

[AWS re:Invent 2017: Yang baru di Amazon ElastiCache](#)

DAT204—Membuat Aplikasi yang Dapat Diskalakan di Layanan AWS NoSQL (re:Invent 2015)

Dalam sesi ini, kita membahas manfaat basis data NoSQL dan mempelajari layanan NoSQL utama yang ditawarkan oleh AWS—Amazon DynamoDB dan Amazon ElastiCache. Kemudian, kita akan mendengar pendapat dari dua pelanggan terkemuka, Expedia dan Mapbox, tentang kasus penggunaan dan tantangan arsitektur mereka, dan cara mereka menanganinya menggunakan

layanan AWS NoSQL, termasuk 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—Membuat Aplikasi yang Dapat Diskalakan di Layanan AWS NoSQL \(re:Invent 2015\)](#)

[DAT207—Mempercepat Performa Aplikasi dengan Amazon ElastiCache \(re:Invent 2013\) AWS](#)

Dalam video ini, pelajari cara menggunakan Amazon ElastiCache untuk men-deploy sistem cache dalam memori dengan mudah untuk mempercepat performa aplikasi Anda. Kami menunjukkan cara menggunakan Amazon ElastiCache untuk meningkatkan latensi aplikasi dan mengurangi beban pada server basis data Anda. Kami juga akan menunjukkan cara membuat sebuah lapisan caching yang mudah dikelola dan diskalakan seiring pertumbuhan aplikasi Anda. Selama sesi ini, kita akan membahas berbagai skenario dan kasus penggunaan yang bermanfaat dengan mengaktifkan caching, serta mendiskusikan fitur yang disediakan oleh Amazon ElastiCache.

[DAT207 - Mempercepat Performa Aplikasi dengan Amazon ElastiCache \(re:Invent 2013\)](#)

Video Lanjutan

Video berikut membahas berbagai topik Amazon ElastiCache lanjutan.

Topik

- [Desain untuk keberhasilan dengan praktik terbaik Amazon ElastiCache \(re:Invent 2020\)](#)
- [Memaksimalkan aplikasi waktu nyata dengan Amazon ElastiCache \(Re:invent 2019\)](#)
- [Praktik terbaik: memigrasikan kluster Redis dari Amazon EC2 ke ElastiCache \(Re:invent 2019\)](#)
- [Menskalakan Platform Olahraga Fantasi dengan Amazon ElastiCache & Amazon Aurora STP11 \(re:invent 2018\)](#)
- [Redis yang Andal & Dapat Diskalakan di Cloud dengan Amazon ElastiCache \(re:invent 2018\)](#)
- [Mempelajari Lebih Dalam ElastiCache: Pola desain untuk Penyimpanan Data Dalam Memori \(re:invent 2018\)](#)
- [DAT305—Mempelajari Lebih Dalam Amazon ElastiCache \(re:Invent 2017\)](#)
- [DAT306—Mempelajari Lebih Dalam Amazon ElastiCache \(re:Invent 2016\)](#)
- [DAT407—Mempelajari Lebih Dalam Amazon ElastiCache \(re:Invent 2015\)](#)
- [SDD402—Mempelajari Lebih Dalam Amazon ElastiCache \(re:Invent 2014\)](#)
- [DAT307—Mempelajari Lebih Dalam Arsitektur dan Pola Desain Amazon ElastiCache \(re:Invent 2013\)](#)

Desain untuk keberhasilan dengan praktik terbaik Amazon ElastiCache (re:Invent 2020)

Seiring dengan pesatnya pertumbuhan aplikasi waktu nyata dan penting untuk bisnis yang dibangun di Redis, ketersediaan, skalabilitas, dan keamanan telah menjadi pertimbangan utama. Pelajari praktik terbaik untuk menyiapkan Amazon ElastiCache agar Anda berhasil dengan penskalaan online, ketersediaan tinggi di seluruh deployment Multi-AZ, dan konfigurasi keamanan.

[Desain untuk keberhasilan dengan praktik terbaik Amazon ElastiCache \(re:Invent 2020\)](#)

Memaksimalkan aplikasi waktu nyata 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 secara tradisional 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 bagaimana Anda dapat meningkatkan kinerja aplikasi waktu nyata dengan menggunakan Amazon ElastiCache dalam memori yang dikelola sepenuhnya untuk kinerja ekstrem, skalabilitas tinggi, ketersediaan, dan keamanan.

[Memaksimalkan aplikasi waktu nyata dengan Amazon ElastiCache \(Re:invent 2019\)](#)

Praktik terbaik: memigrasikan kluster Redis dari Amazon EC2 ke ElastiCache (Re:invent 2019)

Mengelola kluster Redis sendiri tidaklah mudah. Anda harus menyediakan perangkat keras, patch perangkat lunak, melakukan backup data, dan memantau beban kerja secara konstan. Dengan fitur Migrasi Online yang baru dirilis untuk Amazon ElastiCache, Anda kini dapat dengan mudah memindahkan data Anda dari Redis yang dihosting sendiri di Amazon EC2 ke Amazon ElastiCache yang terkelola sepenuhnya, dengan mode menonaktifkan kluster. Dalam sesi ini, Anda mempelajari alat Migrasi Online baru, melihat demo, dan, yang lebih penting, mempelajari praktik terbaik secara langsung untuk migrasi yang mulus ke Amazon ElastiCache.

[Praktik terbaik: memigrasikan kluster Redis dari Amazon EC2 ke ElastiCache \(Re:invent 2019\)](#)

Menskalakan Platform Olahraga Fantasi dengan Amazon ElastiCache & Amazon Aurora STP11 (re:invent 2018)

Dream11 adalah startup 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 diskusi ini, CTO Dream11, Amit Sharma, menjelaskan bagaimana perusahaan menggunakan Amazon Aurora dan Amazon ElastiCache untuk menangani lalu lintas flash, yang dapat menjadi tiga kali lipat dalam jendela waktu 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 ElastiCache & Amazon Aurora (STP11)

[Menskalakan Platform Olahraga Fantasi dengan Amazon ElastiCache & Amazon Aurora STP11 \(re:invent 2018\)](#)

Redis yang Andal & Dapat Diskalakan di Cloud dengan Amazon ElastiCache (re:invent 2018)

Sesi ini membahas fitur dan peningkatan dalam layanan kami yang kompatibel dengan Redis, yaitu Amazon ElastiCache for Redis. Kami membahas fitur utama, seperti Redis 5, skalabilitas dan peningkatan kinerja, keamanan dan kepatuhan, dan banyak lagi. Kami juga membahas fitur yang akan datang dan studi kasus pelanggan.

[Redis yang Andal & Dapat Diskalakan di Cloud dengan Amazon ElastiCache \(re:invent 2018\)](#)

Mempelajari Lebih Dalam ElastiCache: Pola desain untuk Penyimpanan Data Dalam Memori (re:invent 2018)

Dalam sesi ini, kami menyediakan informasi di belakang layar untuk belajar tentang desain dan arsitektur dari Amazon ElastiCache. Lihat pola desain umum dengan penawaran Redis dan Memcached kami dan cara pelanggan menggunakannya untuk pemrosesan data dalam memori untuk mengurangi latensi dan meningkatkan throughput aplikasi. Kami meninjau praktik terbaik, pola desain, dan anti-pola dari ElastiCache.

[Mempelajari Lebih Dalam ElastiCache: Pola desain untuk Penyimpanan Data Dalam Memori \(re:invent 2018\)](#)

DAT305—Mempelajari Lebih Dalam Amazon ElastiCache (re:Invent 2017)

Lihat di balik layar untuk mempelajari tentang desain dan arsitektur Amazon ElastiCache. Lihat pola desain umum dengan penawaran Redis dan Memcached kami dan cara pelanggan menggunakannya untuk pemrosesan data dalam memori untuk mengurangi latensi dan meningkatkan throughput aplikasi. Dalam video ini, kami meninjau praktik terbaik, pola desain, dan anti-pola dari ElastiCache.

Video ini memperkenalkan hal berikut:

- Resharding online ElastiCache for Redis
- Keamanan dan enkripsi ElastiCache
- ElastiCache for Redis versi 3.2.10

[DAT305—Mempelajari Lebih Dalam Amazon ElastiCache \(re:Invent 2017\)](#)

DAT306—Mempelajari Lebih Dalam Amazon ElastiCache (re:Invent 2016)

Lihat di balik layar untuk mempelajari tentang desain dan arsitektur Amazon ElastiCache. Lihat pola desain umum dengan penawaran Redis dan Memcached kami dan cara pelanggan menggunakannya untuk pemrosesan data dalam memori untuk mengurangi latensi dan meningkatkan throughput aplikasi. Selama sesi ini, kami meninjau praktik terbaik, pola desain, dan anti-pola dari ElastiCache.

[DAT306—Mempelajari Lebih Dalam Amazon ElastiCache \(re:Invent 2016\)](#)

DAT407—Mempelajari Lebih Dalam Amazon ElastiCache (re:Invent 2015)

Lihat di balik layar untuk mempelajari tentang desain dan arsitektur Amazon ElastiCache. Lihat pola desain umum dengan penawaran Redis dan Memcached kami dan cara 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 dari Amazon ElastiCache.

[DAT407—Mempelajari Lebih Dalam Amazon ElastiCache \(re:Invent 2015\)](#)

SDD402—Mempelajari Lebih Dalam Amazon ElastiCache (re:Invent 2014)

Dalam video ini, kita memeriksa kasus penggunaan caching umum, mesin Memcached dan Redis, pola yang membantu Anda menentukan mesin yang lebih baik untuk kebutuhan Anda, hashing konsisten, dan lainnya sebagai sarana untuk membangun aplikasi yang cepat dan dapat diskalakan. Frank Wiebe, Ilmuwan Utama di Adobe, merinci cara Adobe menggunakan Amazon ElastiCache untuk meningkatkan pengalaman pelanggan dan menskalakan bisnis mereka.

[DAT402—Mempelajari Lebih Dalam Amazon ElastiCache \(re:Invent 2014\)](#)

DAT307—Mempelajari Lebih Dalam Arsitektur dan Pola Desain Amazon ElastiCache (re:Invent 2013)

Dalam video ini, kita memeriksa caching, strategi caching, penskalaan ke luar, dan pemantauan. Kita juga membandingkan mesin Memcached dan Redis. Selama sesi ini, kami meninjau praktik terbaik, pola desain, dan anti-pola dari Amazon ElastiCache.

[DAT307—Mempelajari Lebih Dalam Arsitektur dan Pola Desain Amazon ElastiCache \(AWS re:Invent 2013\).](#)

Memilih wilayah dan zona ketersediaan

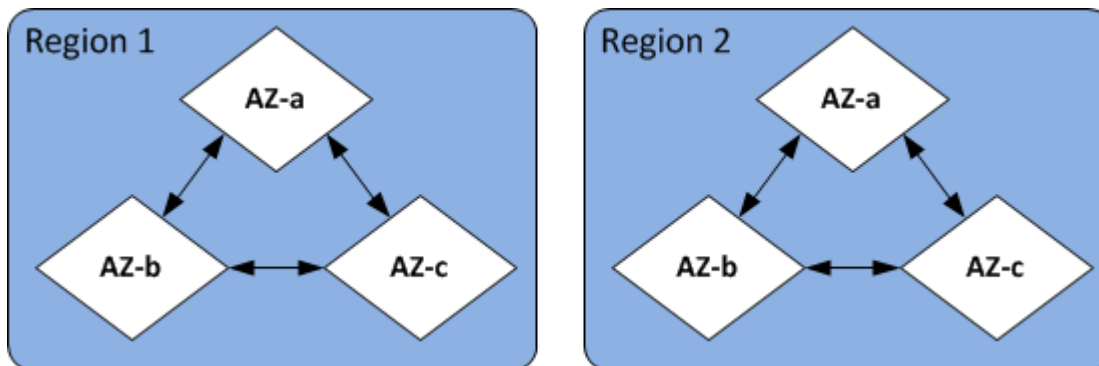
AWS Sumber daya komputasi awan ditempatkan di fasilitas pusat data yang sangat tersedia. Untuk memberikan skalabilitas dan keandalan tambahan, fasilitas pusat data ini ditempatkan di beberapa lokasi fisik yang berbeda. Lokasi ini dikategorikan berdasarkan wilayah dan Zona Ketersediaan.

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 adalah independen sepenuhnya. ElastiCache Aktivitas apa pun yang Anda mulai (misalnya, membuat kluster) hanya berjalan di wilayah default Anda saat ini.

Untuk membuat atau bekerja dengan kluster di wilayah tertentu, gunakan titik akhir layanan wilayah tersebut. Untuk titik akhir layanan, lihat [Wilayah dan titik akhir yang didukung](#).



Wilayah dan Zona Ketersediaan

Topik

- [Pertimbangan terkait Zona Ketersediaan](#)
- [Wilayah dan titik akhir yang didukung](#)
- [Menempatkan simpul Anda](#)
- [Menggunakan zona lokal dengan ElastiCache](#)
- [Menggunakan Outposts](#)

Pertimbangan terkait Zona Ketersediaan

Mendistribusikan simpul Memcached Anda ke beberapa Zona Ketersediaan di 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 cache 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`) ke benar.

 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 kluster 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 kluster.

Jumlah ID simpul cache yang terdaftar harus sama dengan jumlah simpul yang saat ini aktif dikurangi nilai dalam `NumCacheNodes`.

- (Opsional) Tetapkan `ApplyImmediately` (CLI: `--apply-immediately`) ke benar.

Jika Anda tidak menetapkan `ApplyImmediately` (CLI: `--apply-immediately`) ke benar, penghapusan simpul akan berlangsung pada masa pemeliharaan berikutnya.

Wilayah dan 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. ElastiCache Cache tanpa server secara otomatis mereplikasi data di beberapa zona ketersediaan (kecualius-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 di AZ yang berbeda untuk mencapai toleransi kesalahan. Untuk informasi selengkapnya tentang wilayah dan Zona Ketersediaan, lihat [Memilih wilayah dan zona ketersediaan](#) 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	

Nama Wilayah/Wilayah	Titik Akhir	Protokol	
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	
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	

Nama Wilayah/Wilayah	Titik Akhir	Protokol	
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	
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	

Nama Wilayah/Wilayah	Titik Akhir	Protokol	
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	
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	

Nama Wilayah/Wilayah	Titik Akhir	Protokol
Wilayah Tiongkok (Ningxia) cn-northwest-1	elasticache.cn-northwest-1.amazonaws.com.cn	HTTPS
Wilayah Asia Pacific (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
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 dari jenis simpul. Untuk tabel tipe node yang didukung menurut AWS Region, lihat [Jenis simpul yang didukung oleh Wilayah AWS](#).

Untuk tabel AWS produk dan layanan menurut wilayah, lihat [Produk dan Layanan menurut Wilayah](#).

Menempatkan simpul Anda

Amazon ElastiCache mendukung lokasi semua node cluster dalam satu atau beberapa Availability Zones (AZ). Selanjutnya, jika Anda memilih untuk menemukan node Anda di beberapa AZ (disarankan), ElastiCache memungkinkan Anda untuk memilih AZ untuk setiap node, atau memungkinkan untuk memilihnya ElastiCache untuk Anda.

Dengan menempatkan simpul di AZ yang berbeda, Anda menutup kemungkinan bahwa kegagalan, seperti pemadaman listrik, di satu AZ akan menyebabkan kegagalan di seluruh sistem Anda.

Anda dapat menentukan AZ untuk setiap simpul saat Anda membuat klaster atau dengan menambahkan simpul saat Anda mengubah klaster yang sudah ada. Untuk informasi selengkapnya, lihat informasi berikut:

- [Membuat klaster](#)
- [Memodifikasi sebuah cluster ElastiCache](#)
- [Menambahkan simpul ke klaster](#)

Menggunakan zona lokal dengan ElastiCache

Zona lokal adalah perluasan dari Wilayah AWS yang secara geografis dekat dengan pengguna Anda. Anda dapat memperluas virtual private cloud (VPC) apa pun dari Wilayah AWS induk menjadi Local Zones dengan membuat subnet baru dan menempatkannya di Local Zone itu. Saat Anda membuat subnet di Zona Lokal, VPC Anda diperluas ke Zona Lokal itu. Subnet di Zona Lokal beroperasi sama seperti subnet lain di VPC Anda.

Dengan menggunakan Local Zones, Anda dapat menempatkan sumber daya ElastiCache cluster di beberapa lokasi yang dekat dengan pengguna Anda.

Saat Anda membuat ElastiCache klaster, Anda dapat memilih subnet di Zona Lokal. Local Zone memiliki koneksi sendiri ke internet dan mendukung AWS Direct Connect. Jadi, sumber daya yang dibuat di Zona Lokal dapat melayani pengguna lokal dengan komunikasi latensi sangat rendah. Untuk informasi lain, lihat [Local Zones AWS](#).

Zona Lokal diwakili oleh kode Wilayah AWS diikuti dengan pengidentifikasi yang menunjukkan lokasi, misalnya `us-west-2-lax-1a`.

Saat ini, Local Zones yang tersedia adalah `us-west-2-lax-1a` dan `us-west-2-lax-1b`.

Batasan berikut berlaku untuk ElastiCache untuk Local Zones:

- Jenis simpul berikut didukung oleh Local Zones 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 pada konsol Amazon EC2.

Untuk informasi lain, lihat [Mengaktifkan Local Zones](#) dalam Panduan Pengguna Amazon EC2.

2. Membuat subnet di Zona Lokal.

Untuk informasi lain, lihat [Membuat subnet di VPC Anda](#) dalam Panduan Pengguna Amazon VPC.

3. Buat ElastiCache grup subnet di Zona Lokal.

Saat Anda membuat ElastiCache grup subnet, pilih grup Availability Zone untuk Zona Lokal.

Untuk informasi lain, lihat [Membuat grup subnet](#) di Panduan Pengguna .ElastiCache

4. Buat ElastiCache untuk memcached cluster yang menggunakan ElastiCache subnet di Zona Lokal.

Untuk informasi selengkapnya, lihat [Membuat klaster Memcached \(konsol\)](#).

Menggunakan Outposts

AWS Outposts adalah layanan terkelola penuh yang memperluas infrastruktur, layanan, API, dan alat AWS ke premise dari pelanggan. Dengan menyediakan akses lokal ke infrastruktur terkelola AWS, Outposts AWS memungkinkan pelanggan untuk membangun dan menjalankan aplikasi on premises menggunakan antarmuka pemrograman yang sama seperti di Wilayah AWS, sekaligus

menggunakan sumber daya komputasi dan penyimpanan lokal untuk latensi yang lebih rendah dan kebutuhan pemrosesan data lokal. Outpost adalah sebuah pengumpulan komputasi dan kapasitas penyimpanan AWS yang di-deploy di situs pelanggan. AWS mengoperasikan, memantau, dan mengelola kapasitas ini sebagai bagian dari Wilayah AWS. Anda dapat membuat subnet di Outpost Anda dan menentukannya saat Anda membuatnya AWS sumber daya seperti ElastiCache kluster.

Note

Dalam versi ini, batasan berikut berlaku:

- ElastiCache untuk Outposts hanya mendukung keluarga simpul M5 dan R5.
- Multi-AZ (cross replikasi Outpost tidak didukung).
- ElastiCache untuk Outposts tidak didukung di wilayah berikut: cn-north-1, cn-northeast-3 dan ap-northeast-3.

Menggunakan Outposts dengan konsol Memcached

1. Masuk ke AWS Management Console dan membuka ElastiCache konsol di <https://console.aws.amazon.com/elasticache/>.
2. Pada panel navigasi, pilih Memcached.
3. Pada Mesin kluster pilih Memcached.
4. Di bawah Lokasi, pilih On-Premise - Buat ElastiCache contoh pada AWS Outposts.

Membuat konfigurasi pilihan on-premises

Anda dapat memilih Outpost yang tersedia untuk menambahkan kluster cache atau, jika tidak ada Outposts yang tersedia, buatlah peran baru menggunakan langkah berikut:

Di bawah opsi On-Premises:

1. Di bawah Pengaturan Memcached:
 - a. Nama: Masukkan nama untuk kluster Memcached
 - b. Deskripsi: Memasukkan deskripsi untuk kluster Memcached.
 - c. Versi mesin compatibility: Versi mesin didasarkan pada AWS Wilayah Outpost

- d. Pelabuhan: Terima port default, 11211. Jika Anda mempunyai alasan untuk menggunakan port yang berbeda, ketikkan nomor port tersebut.
 - e. Grup parameter: Gunakan drop-down untuk memilih default atau grup parameter khusus.
 - f. Jenis Simpul: Instans yang tersedia didasarkan pada ketersediaan Outposts. Dari daftar tarik ke bawah, pilih Outposts dan kemudian pilih jenis simpul tersedia yang ingin Anda gunakan untuk kluster ini. Kemudian pilih Simpan.
 - g. Jumlah simpul: Masukkan jumlah simpul yang Anda inginkan dalam kluster Anda.
2. Di bawah Pengaturan Memcached lanjutan:
- 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 sesuai dengan Outpost (VPC).
 - Availability Zone atau Outpost: Pilih Outpost yang Anda gunakan.
 - ID Subnet: Pilih ID subnet yang tersedia untuk Outpost. Jika tidak ada ID subnet yang tersedia, Anda harus membuatnya. Untuk informasi lain, lihat [Membuat Subnet](#).
 - b. Pilih Buat.

Melihat detail kluster Outpost

Pada halaman daftar Memcached, pilih kluster yang dimiliki oleh Outpost AWS dan perhatikan hal-hal berikut ketika melihat Detail kluster:

- Availability Zone: Ini akan mewakili Outpost, menggunakan ARN (Amazon Resource Name) dan AWS Nomor Sumber Daya.
- Nama Outpost: Nama AWS Outpost.

Menggunakan Outposts dengan CLI AWS

Anda dapat menggunakan AWS Command Line Interface (AWS CLI) untuk mengontrol beberapa layanan AWS dari baris perintah dan mengotomatiskan layanan tersebut melalui skrip. Anda dapat menggunakan AWS CLI untuk operasi ad hoc (hanya berjalan satu kali).

Mengunduh dan membuat konfigurasi AWS CLI

AWS CLI berjalan di Windows, macOS, atau Linux. Gunakan prosedur berikut untuk mengunduh dan membuat konfigurasinya.

Untuk mengunduh, menginstal, dan membuat konfigurasi CLI

1. Unduh CLI AWS pada halaman web [Antamuka Baris Perintah AWS](#).
2. Ikuti petunjuknya untuk [MenginstalAWSCLI](#) dan [MengonfigurasiAWSCLI](#) di dalam AWS Command Line Interface Panduan Pengguna.

Menggunakan CLI AWS dengan Outpost

Menggunakan operasi CLI berikut untuk membuat klaster cache yang menggunakan Outposts:

- [create-cache-cluster](#)- Menggunakan operasi ini, `outpost-mode` parameter menerima nilai yang menentukan apakah simpul dalam klaster cache dibuat dalam satu Outpost atau pada 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 \  
  
```

Merancang dan mengelola kluster ElastiCache Anda sendiri untuk implementasi Memcached

Jika Anda memerlukan kontrol terperinci atas kluster ElastiCache, Anda dapat memilih untuk mendesain kluster Anda sendiri. ElastiCache memungkinkan Anda mengoperasikan kluster berbasis simpul, dengan memilih tipe simpul, jumlah simpul, dan penempatan simpul di seluruh Zona Ketersediaan AWS untuk kluster Anda. Karena ElastiCache adalah layanan yang dikelola sepenuhnya, ElastiCache secara otomatis mengelola penyediaan perangkat keras, pemantauan, penggantian simpul, dan penambalan perangkat lunak untuk kluster Anda.

Untuk informasi tentang cara menyiapkannya, lihat [Pengaturan](#). Untuk detail tentang cara mengelola, memperbarui, atau menghapus simpul atau kluster, lihat [Mengelola simpul](#). Untuk gambaran umum tentang komponen utama deployment Amazon ElastiCache saat Anda mendesain kluster ElastiCache Anda sendiri, lihat [konsep utama](#) ini.

Topik

- [ElastiCache untuk komponen dan fitur Memcached](#)
- [Mengelola kluster](#)
- [Mengelola simpul](#)

ElastiCache untuk komponen dan fitur Memcached

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

Topik

- [ElastiCache simpul](#)
- [ElastiCache untuk cluster Memcached](#)
- [AWS Wilayah dan zona ketersediaan](#)
- [ElastiCache untuk titik akhir Memcached](#)
- [ElastiCache kelompok parameter](#)
- [ElastiCache keamanan](#)
- [ElastiCache kelompok subnet](#)
- [ElastiCache untuk acara Memcached](#)

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 instans dari Memcached. Jika perlu, Anda dapat menaikkan atau menurunkan skala simpul di dalam klaster ke jenis instans yang berbeda. Untuk informasi selengkapnya, lihat [Penskalaan ElastiCache untuk Memcached](#).

Setiap simpul dalam klaster adalah tipe 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 tipe instans simpul yang didukung, lihat [Tipe 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 terpesan dapat menghemat uang Anda. Misalkan klaster Anda hampir digunakan setiap saat, dan Anda terkadang menambahkan simpul untuk menangani lonjakan penggunaan. Dalam hal ini, Anda dapat membeli sejumlah node cadangan untuk menjalankan sebagian besar waktu dan membeli pay-as-you-go node untuk waktu

yang kadang-kadang Anda butuhkan untuk menambahkan node. Untuk informasi selengkapnya tentang simpul terpesan, lihat [Simpul terpesan ElastiCache](#).

Mesin Memcached mendukung Penemuan Otomatis. Penemuan Otomatis adalah kemampuan program klien untuk otomatis mengidentifikasi semua simpul di kluster cache, serta untuk memulai dan memelihara koneksi ke semua simpul ini. Dengan Penemuan Otomatis, aplikasi Anda tidak perlu terhubung ke setiap simpul secara manual. Sebagai gantinya, aplikasi Anda terhubung ke titik akhir konfigurasi. Entri DNS titik akhir konfigurasi berisi entri CNAME untuk setiap titik akhir simpul cache. Jadi, dengan terhubung ke titik akhir konfigurasi, aplikasi Anda akan langsung memiliki informasi tentang semua simpul di kluster dan dapat terhubung ke semua simpul tersebut. Anda tidak perlu melakukan hard coding setiap titik akhir simpul cache ke dalam kode aplikasi Anda. Untuk informasi selengkapnya, lihat [Penemuan Otomatis](#).

Untuk informasi selengkapnya tentang simpul, lihat [Mengelola simpul](#).

ElastiCache untuk cluster Memcached

Kluster Memcached adalah pengelompokan logis dari satu atau beberapa [ElastiCache simpul](#). Data dipartisi di seluruh simpul dalam kluster Memcached.

Banyak ElastiCache operasi yang ditargetkan pada cluster:

- Membuat kluster
- Mengubah kluster
- Menghapus kluster
- Melihat elemen di kluster
- Menambahkan atau menghapus tag alokasi biaya ke dan dari kluster

Untuk informasi selengkapnya, lihat topik terkait berikut:

- [Mengelola kluster](#) dan [Mengelola simpul](#)

Informasi tentang kluster, simpul, dan operasi terkait.

- [AWS batas layanan: Amazon ElastiCache](#)

Informasi tentang ElastiCache batas, seperti jumlah maksimum node atau cluster.

Jika Anda perlu melampaui batas ini, buat permintaan Anda menggunakan [formulir permintaan node ElastiCache cache Amazon](#).

- [Mitigasi Kegagalan](#)

Informasi tentang peningkatan toleransi kesalahan kluster Anda.

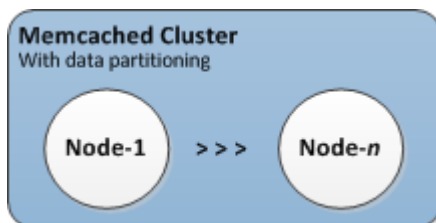
Konfigurasi kluster umum

Memcached mendukung hingga 300 node per pelanggan untuk setiap AWS Wilayah dengan setiap cluster memiliki 1-60 node. Data dipartisi di seluruh simpul di kluster 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 Zone (AZ) dalam Region cluster. AWS Dengan cara ini, kegagalan dalam satu AZ akan berdampak minimal pada keseluruhan kluster dan aplikasi. Untuk informasi selengkapnya, lihat [Mitigasi Kegagalan](#).

Seiring perubahan permintaan atas kluster 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 [Buat konfigurasi klien ElastiCache Anda untuk penyeimbangan beban yang efisien](#). Di diagram berikut, Anda dapat melihat contoh kluster Memcached simpul tunggal dan simpul banyak.



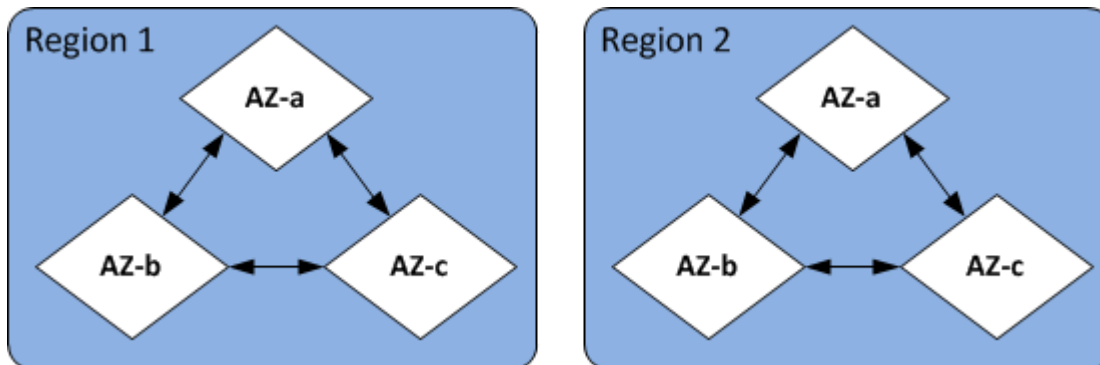
AWS Wilayah dan zona ketersediaan

Amazon ElastiCache untuk Memcached tersedia di beberapa AWS Wilayah di seluruh dunia. Dengan demikian, Anda dapat meluncurkan ElastiCache cluster di lokasi yang memenuhi persyaratan bisnis Anda. Misalnya, Anda dapat meluncurkan di AWS Wilayah terdekat dengan pelanggan Anda atau untuk memenuhi persyaratan hukum tertentu.

Secara default, AWS SDK, ElastiCache API AWS CLI, dan ElastiCache konsol mereferensikan wilayah AS-Barat (Oregon). Saat ElastiCache memperluas ketersediaan ke AWS Wilayah baru, titik

akhir baru untuk AWS Wilayah ini juga tersedia untuk digunakan dalam permintaan HTTP, AWS SDK, AWS CLI, dan konsol Anda. ElastiCache

Setiap AWS Wilayah dirancang untuk sepenuhnya terisolasi dari AWS Wilayah lain. Di dalam setiap wilayah terdapat beberapa Availability Zone. Dengan meluncurkan simpul Anda di Availability Zone yang berbeda, Anda dapat mencapai toleransi kesalahan sebesar mungkin. Untuk informasi selengkapnya tentang AWS Wilayah dan Availability Zone, lihat [Memilih wilayah dan zona ketersediaan](#).



Untuk informasi tentang AWS Wilayah yang didukung oleh ElastiCache dan titik akhirnya, lihat [Wilayah dan titik akhir yang didukung](#).

ElastiCache untuk titik akhir Memcached

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

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 [Titik akhir](#).

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 melakukannya, Anda dapat memastikan bahwa semua simpul dalam kluster dikonfigurasi dengan cara yang sama persis.

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

Untuk informasi lebih rinci tentang grup ElastiCache parameter, lihat [Mengonfigurasi parameter mesin menggunakan grup parameter](#).

ElastiCache keamanan

Untuk keamanan yang ditingkatkan, akses ElastiCache node dibatasi untuk aplikasi yang berjalan pada instans Amazon EC2 yang masuk daftar putih. Anda dapat mengontrol instans Amazon EC2 yang dapat mengakses kluster Anda dengan 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 kluster dari instans Amazon EC2 yang berjalan di subnet tertentu. Jika Anda memilih untuk menjalankan kluster di luar Amazon VPC, Anda dapat membuat grup keamanan untuk mengotorisasi instans Amazon EC2 yang berjalan di dalam grup keamanan Amazon EC2 tertentu.

ElastiCache kelompok subnet

Grup subnet adalah kumpulan subnet (biasanya privat) yang dapat Anda tetapkan untuk kluster Anda yang berjalan di lingkungan Amazon Virtual Private Cloud (Amazon VPC).

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

Untuk informasi selengkapnya tentang penggunaan grup subnet cache di lingkungan Amazon VPC, lihat [Amazon VPC dan keamanan ElastiCache](#), [Mengotorisasi akses](#), dan [Subnet dan grup subnet](#).

ElastiCache untuk acara Memcached

Saat peristiwa penting terjadi di cluster cache, ElastiCache kirimkan notifikasi ke topik Amazon SNS tertentu. Peristiwa penting dapat mencakup hal seperti kegagalan penambahan simpul, keberhasilan penambahan simpul, modifikasi grup keamanan, dan lain-lain. Dengan memantau peristiwa penting, Anda dapat mengetahui status kluster terbaru Anda dan dapat mengambil tindakan korektif sesuai peristiwa tersebut.

Untuk informasi lebih lanjut tentang ElastiCache acara, lihat [Pemantauan Amazon SNS dari peristiwa ElastiCache](#).

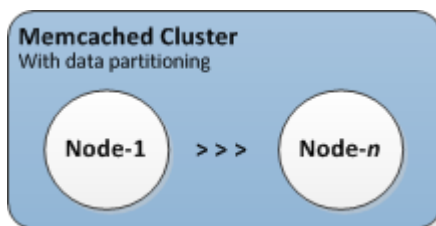
Mengelola klaster

Klaster adalah kumpulan dari satu atau beberapa simpul cache, yang semuanya menjalankan instans dari perangkat lunak mesin cache Memcached. Saat membuat klaster, Anda menentukan mesin dan versi yang akan digunakan oleh semua simpul.

Diagram berikut menggambarkan klaster Memcached secara umum. Cluster memcache berisi dari 1 hingga 60 node di mana Anda secara horizontal mempartisi data Anda.

Untuk meminta penambahan batas, lihat [Batas Layanan AWS](#) dan pilih jenis batas Simpul per klaster per jenis instans.

Klaster Memcached umumnya terlihat sebagai berikut.



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 klaster 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 instans Amazon EC2. Jika Anda meluncurkan klaster Anda di cloud privat virtual (VPC) berdasarkan layanan Amazon VPC, Anda dapat mengaksesnya dari luar AWS. Untuk informasi selengkapnya, lihat [Pola Akses untuk Mengakses Cache ElastiCache di Amazon VPC](#).

Untuk daftar versi Memcached yang didukung, lihat [Didukung ElastiCache untuk Versi Memcache](#).

Memilih jenis jaringan

ElastiCache mendukung Internet Protocol versi 4 dan 6 (IPv4 dan IPv6), memungkinkan Anda untuk mengonfigurasi klaster Anda untuk menerima:

- hanya koneksi IPv4,
- hanya koneksi IPv6,
- koneksi IPv4 dan IPv6 sekaligus (tumpukan ganda)

IPv6 didukung untuk beban kerja menggunakan mesin Memcached versi 1.6.6 dan seterusnya pada semua instans yang dibangun pada [sistem Nitro](#). Tidak ada biaya tambahan untuk mengakses ElastiCache melalui IPv6.

Note

Migrasi klaster yang dibuat sebelum ketersediaan IPV6/tumpukan ganda tidak didukung. Beralih antar jenis jaringan pada klaster yang baru dibuat juga tidak didukung.

Mengonfigurasi subnet untuk jenis jaringan

Jika Anda membuat klaster di Amazon VPC, Anda harus menentukan grup subnet. ElastiCache menggunakan grup subnet tersebut untuk memilih subnet dan alamat IP dalam subnet tersebut yang akan diasosiasikan dengan simpul Anda. Klaster ElastiCache memerlukan subnet tumpukan ganda dengan alamat IPv4 dan IPv6 yang ditetapkan untuk beroperasi dalam mode tumpukan ganda dan subnet khusus IPv6 untuk beroperasi sebagai IPv6 saja.

Menggunakan tumpukan ganda

Saat Anda membuat klaster cache dan memilih tumpukan ganda sebagai jenis jaringan, Anda perlu menunjuk jenis penemuan IP — baik IPv4 atau IPv6. Opsi default ElastiCache jenis jaringan dan penemuan IP adalah 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 mempertahankan kompatibilitas mundur dengan semua klien yang ada, penemuan IP digunakan, yang memungkinkan Anda memilih jenis IP (yaitu, IPv4 atau IPv6) untuk dinyatakan di protokol penemuan. Meskipun ini membatasi penemuan otomatis hanya untuk satu jenis IP,

tumpukan ganda bermanfaat berkat Penemuan Otomatis, karena memungkinkan migrasi (atau rollback) dari IPv4 ke jenis IP Discovery IPv6 tanpa waktu henti.

Klaster ElastiCache tumpukan ganda dengan TLS diaktifkan

Ketika TLS diaktifkan untuk klaster ElastiCache, fungsi penemuan klaster `config get cluster` akan menampilkan nama host, bukan IP. Nama host kemudian digunakan sebagai pengganti IP untuk terhubung ke klaster ElastiCache dan melakukan handshake TLS. Ini berarti bahwa klien tidak akan terpengaruh oleh parameter IP Discovery. Untuk klaster dengan TLS diaktifkan, parameter IP Discovery tidak berpengaruh pada protokol IP yang disukai. Sebagai gantinya, protokol IP yang digunakan akan ditentukan oleh protokol IP mana yang lebih disukai klien saat menyelesaikan nama host DNS.

Untuk contoh tentang cara mengonfigurasi preferensi protokol IP saat menyelesaikan nama host DNS, lihat [TLS mengaktifkan cluster tumpukan ElastiCache ganda](#).

Menggunakan AWS Management Console

Saat membuat klaster cache menggunakan AWS Management Console, di bawah Konektivitas, pilih jenis jaringan, baik IPv4, IPv6 atau Tumpukan ganda. Jika Anda memilih tumpukan ganda, Anda harus memilih jenis IP Discovery, baik IPv6 atau IPv4.

Untuk informasi selengkapnya, lihat [Membuat klaster Memcached \(konsol\)](#).

Menggunakan CLI

Saat membuat klaster cache menggunakan CLI, Anda menggunakan perintah [create-cache-cluster](#) dan menentukan parameter `NetworkType` dan `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
```

Identifikasi simpul di kluster Anda secara otomatis

Untuk kluster yang menjalankan mesin Memcached, ElastiCache mendukung Penemuan Otomatis—kemampuan program klien untuk secara otomatis mengidentifikasi semua simpul di kluster cache, serta untuk memulai dan memelihara koneksi ke semua simpul ini.

Note

Penemuan Otomatis ditambahkan untuk kluster cache yang berjalan pada Amazon ElastiCache Memcached.

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 kluster 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 jenis jaringan tumpukan ganda di kluster Anda, Penemuan Otomatis hanya akan menampilkan alamat IPv4 atau IPv6, bergantung pada alamat mana yang Anda pilih. Untuk informasi selengkapnya, lihat [Memilih jenis jaringan](#).

Semua simpul cache di kluster memelihara daftar metadata tentang semua simpul lainnya. Metadata ini diperbarui setiap kali simpul ditambahkan atau dihapus dari kluster.

Topik

- [Manfaat Penemuan Otomatis](#)
- [Cara Kerja Penemuan Otomatis](#)
- [Menggunakan Penemuan Otomatis](#)

- [Menghubungkan ke Simpul Cache Secara Manual](#)
- [Menambahkan Penemuan Otomatis ke Pustaka Klien Anda](#)
- [Klien ElastiCache dengan penemuan otomatis](#)

Manfaat Penemuan Otomatis

Penemuan Otomatis menawarkan manfaat berikut:

- Ketika Anda menambah jumlah simpul di dalam kluster cache, simpul yang baru akan mendaftarkan diri dengan titik akhir konfigurasi dan dengan semua simpul lainnya. Ketika Anda menghapus simpul dari kluster cache, simpul yang dihapus tersebut akan menghapus pendaftarannya sendiri. Dalam kedua kasus, semua simpul lain di kluster 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 kluster.
- Program klien melakukan polling kluster sekali setiap menit (interval ini dapat disesuaikan jika diperlukan). Jika ada perubahan pada konfigurasi kluster, seperti simpul yang baru atau dihapus, klien menerima daftar metadata yang diperbarui. Kemudian klien terhubung ke, atau memutuskan koneksi dari, simpul ini sesuai kebutuhan.

Penemuan Otomatis diaktifkan pada semua Kluster Cache ElastiCache Memcached. Anda tidak perlu memulai ulang simpul cache Anda yang mana pun untuk menggunakan fitur ini.

Cara Kerja Penemuan Otomatis

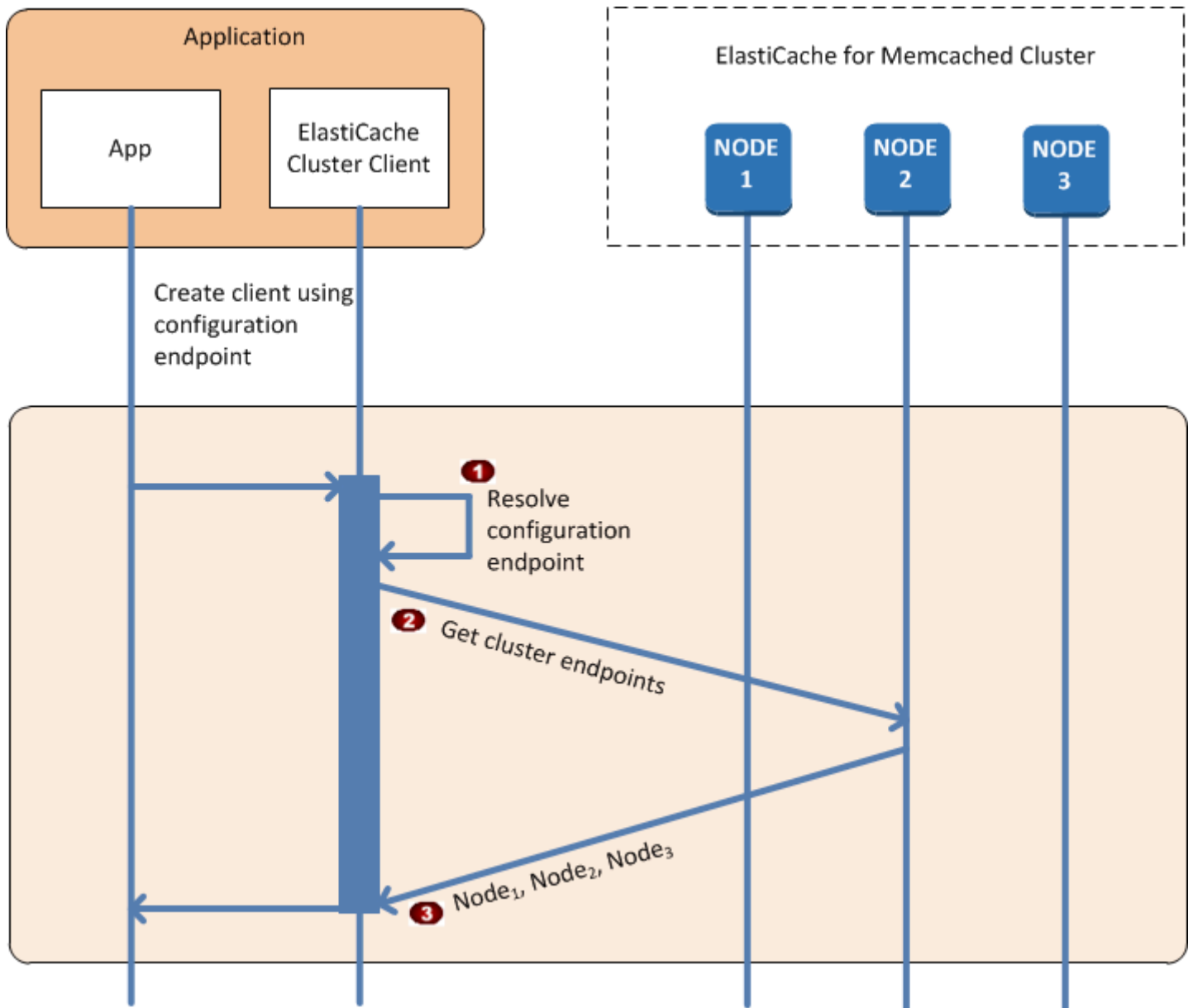
Topik

- [Menghubungkan ke Simpul Cache](#)
- [Operasi Klaster Normal](#)
- [Operasi Lainnya](#)

Bagian ini menjelaskan cara aplikasi klien menggunakan ElastiCache Cluster Client untuk mengelola koneksi simpul cache, dan berinteraksi dengan item data pada cache.

Menghubungkan 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 menjadi 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 kluster, simpul mana pun dapat meneruskan informasi konfigurasi ke klien jika diminta.

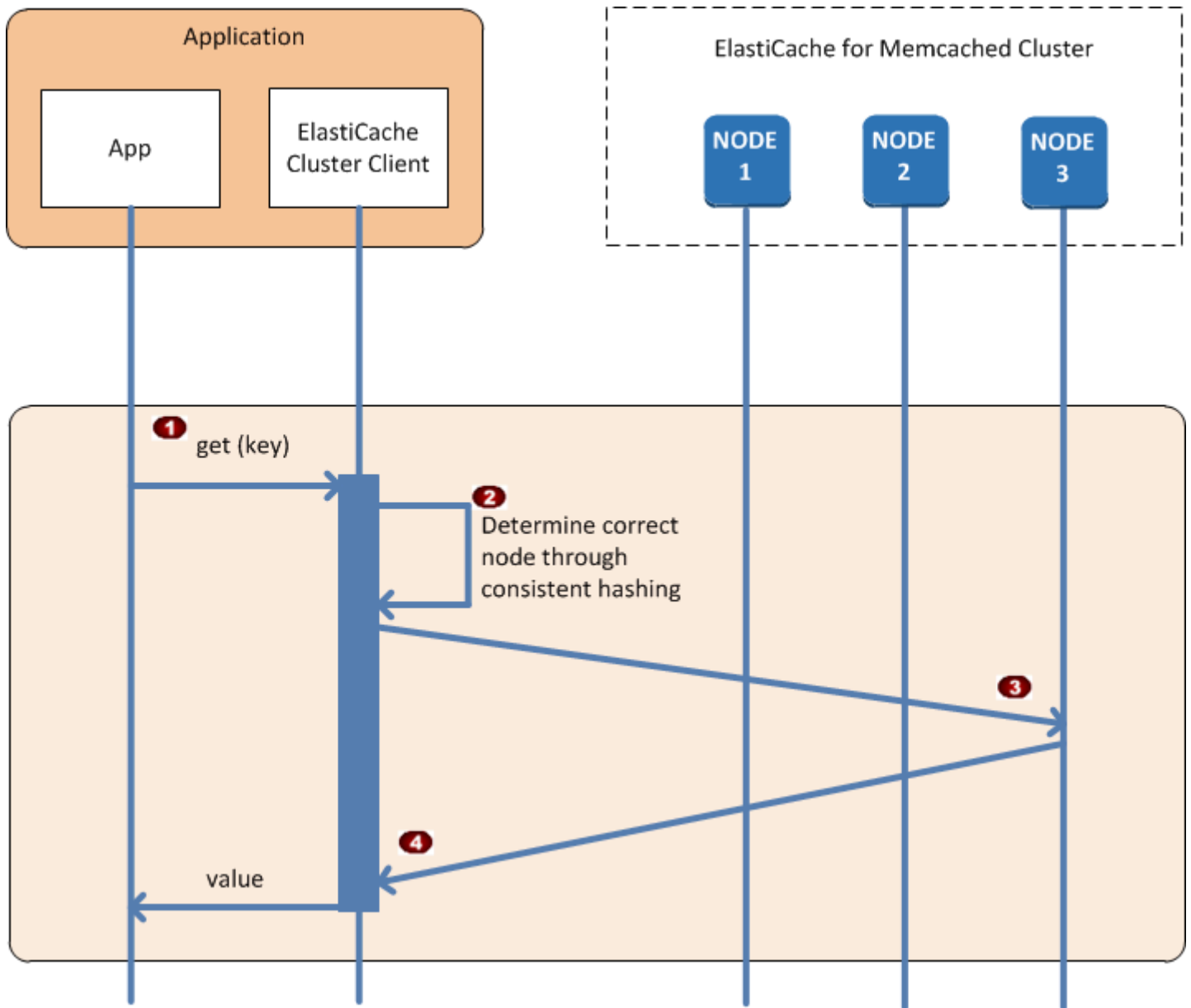
- Klien menerima daftar terkini yang berisi nama host simpul cache dan alamat IP. Klien kemudian dapat terhubung ke semua simpul lain dalam kluster.

Note

Program klien menyegarkan daftar nama host simpul cache dan alamat IP sekali setiap menit. Interval polling ini dapat disesuaikan jika perlu.

Operasi Kluster Normal

Ketika aplikasi telah terhubung ke semua simpul cache, ElastiCache Cluster Client menentukan simpul mana yang harus menyimpan item data individual, dan simpul mana yang harus dikueri untuk item data tersebut nanti. Diagram urutan berikut menunjukkan proses operasi kluster normal.



Proses Operasi Kluster 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 menggunakan 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 di dalam metadata dan aplikasi Anda hanya akan berinteraksi dengan simpul $n-1$ yang tersisa, seolah-olah simpul tersebut tidak ada.

Mengganti simpul

Jika simpul gagal, ElastiCache menghentikan simpul tersebut dan membuat 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 Penemuan Otomatis, ikuti langkah-langkah ini:

- [Langkah 1: Dapatkan Titik Akhir Konfigurasi](#)
- [Langkah 2: Unduh ElastiCache Cluster Client](#)
- [Langkah 3: Modifikasi Program Aplikasi Anda](#)

Langkah 1: Dapatkan Titik Akhir Konfigurasi

Untuk terhubung ke klaster, program klien harus mengetahui titik akhir konfigurasi klaster. Lihat topik [Menemukan Titik Akhir Klaster \(Konsol\)](#).

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 AWS CLI untuk 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"
    }
  ]
}
```

Langkah 2: Unduh ElastiCache Cluster Client

Untuk memanfaatkan Penemuan Otomatis, program klien harus menggunakan ElastiCache Cluster Client. ElastiCache Cluster Client tersedia untuk Java, PHP, dan .NET dan berisi semua logika yang diperlukan untuk menemukan dan terhubung ke semua simpul cache Anda.

Untuk mengunduh ElastiCache Cluster Client

1. Masuk ke Konsol Manajemen AWS dan buka konsol ElastiCache di <https://console.aws.amazon.com/elasticache/>.
2. Dari konsol ElastiCache, memilih Klien kluster ElastiCache lalu pilih Unduh.

Kode sumber ElastiCache Cluster Client untuk Java tersedia di <https://github.com/amazonwebservices/aws-elasticache-cluster-client-memcached-for-java>. Pustaka ini didasarkan pada klien Spymemcached yang populer. ElastiCache Cluster Client dirilis berdasarkan Lisensi Perangkat Lunak Amazon <https://aws.amazon.com/asl>. Anda bebas memodifikasi 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, Anda harus menginstalnya terlebih dahulu pada instans Amazon EC2 Anda. Untuk informasi selengkapnya, lihat [Menginstal ElastiCache Cluster Client untuk PHP](#).

Untuk klien yang didukung TLS, unduh biner dengan PHP versi 7.4 atau lebih tinggi. Untuk menggunakan ElastiCache Cluster Client untuk .NET, Anda harus menginstalnya terlebih dahulu pada instans Amazon EC2 Anda. Untuk informasi selengkapnya, lihat [Menginstal ElastiCache Cluster Client untuk .NET](#).

Langkah 3: Modifikasi Program Aplikasi Anda

Modifikasi 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 kluster, pastikan bahwa titik akhir memiliki “.cfg” di alamatnya, seperti ditunjukkan di sini. Jangan menggunakan CNAME atau titik akhir tanpa “.cfg” di dalamnya.

```
"mycluster.fnjyzo.cfg.use1.cache.amazonaws.com";
```

Kegagalan untuk secara eksplisit menentukan titik akhir konfigurasi kluster 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 tersambung ke titik akhir konfigurasi kluster dan menambahkan item data ke cache. Dengan Penemuan Otomatis, program ini terhubung ke semua simpul dalam kluster 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 tersambung ke titik akhir konfigurasi kluster dan menambahkan item data ke cache. Dengan Penemuan Otomatis, program ini akan terhubung ke semua simpul di kluster tanpa intervensi lebih lanjut.

Untuk menggunakan ElastiCache Cluster Client untuk PHP, Anda harus menginstalnya terlebih dahulu pada instans Amazon EC2 Anda. Untuk informasi selengkapnya, lihat [Menginstal ElastiCache Cluster Client 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 ElastiCache Cluster Client dengan TLS yang diaktifkan, lihat [Menggunakan enkripsi data bergerak dengan PHP dan Memcached](#).

Menggunakan ElastiCache Cluster Client untuk .NET

 Note

ElastiCache .NET Cluster Client sudah ditiadakan mulai Mei 2022.

Klien .NET untuk ElastiCache adalah sumber terbuka 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 ElastiCache Cluster Client untuk terhubung ke titik akhir konfigurasi kluster dan menambahkan item data ke cache. Dengan Penemuan Otomatis, program ini akan terhubung ke semua simpul di kluster 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 Simpul Cache Secara Manual

Jika program klien Anda tidak menggunakan Penemuan Otomatis, program ini dapat secara manual terhubung ke setiap simpul cache. 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 perintah AWS CLI `aws elasticache describe-cache-clusters` dengan parameter `--show-cache-node-info`.

Example

Cuplikan kode Java berikut menunjukkan cara terhubung ke semua simpul di kluster 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 kluster cache Anda dengan menambahkan atau menghapus simpul, Anda akan perlu memperbarui daftar simpul di dalam kode klien.

Menambahkan Penemuan Otomatis ke Pustaka Klien Anda

Informasi konfigurasi untuk Penemuan Otomatis disimpan secara redundan di setiap simpul kluster cache. Aplikasi klien dapat mengkueri simpul cache apa pun dan memperoleh informasi konfigurasi untuk semua simpul di kluster 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.

Mesin Cache 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 dan protokol biner Memcached oleh ElastiCache, dan diimplementasikan di dalam ElastiCache Cluster Client. 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 Penemuan Otomatis menggunakan protokol biner, lihat [Kode sumber untuk ElastiCache Cluster Client](#).

Sintaksis

```
config [sub-command] [key]
```

Opsi

Nama	Deskripsi	Diperlukan
sub-command		Ya

Nama	Deskripsi	Diperlukan
	Sub-perintah yang digunakan untuk berinteraksi dengan simpul cache. Untuk Penemuan Otomatis, sub-perintah ini adalah <code>get</code> .	
<code>key</code>	Kunci yang digunakan untuk menyimpan konfigurasi klaster. Untuk Penemuan Otomatis, kunci ini disebut <code>cluster</code> .	Ya

Untuk mendapatkan informasi konfigurasi klaster, gunakan perintah berikut:

```
config get cluster
```

Mesin Cache Versi Lebih Rendah dari 1.4.14

Untuk mendapatkan informasi konfigurasi klaster, gunakan perintah berikut:

```
get AmazonElastiCache:cluster
```

Note

Jangan mengutak-atik kunci "AmazonElastiCache:cluster", karena kunci ini menyimpan informasi konfigurasi klaster. Jika Anda melakukan perubahan pada kunci ini, maka klien dapat salah dikonfigurasi untuk jangka waktu singkat (tidak lebih dari 15 detik) sebelum ElastiCache melakukan pembaruan dengan informasi konfigurasi yang benar secara otomatis.

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 dimodifikasi 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.

Klien ElastiCache dengan penemuan otomatis

Bagian ini membahas cara menginstal dan mengonfigurasi klien PHP dan .NET ElastiCache.

Topik

- [Menginstal & mengompilasi klien klaster](#)
- [Mengonfigurasi Klien ElastiCache](#)

Menginstal & mengompilasi klien kluster

Bagian ini membahas cara menginstal, mengonfigurasi, dan mengompilasi klien kluster penemuan otomatis Amazon ElastiCache PHP dan .NET.

Topik

- [Menginstal ElastiCache Cluster Client untuk .NET](#)
- [Menginstal ElastiCache Cluster Client untuk PHP](#)
- [Mengompilasi kode sumber ElastiCache Cluster Client untuk PHP](#)

Menginstal ElastiCache Cluster Client untuk .NET

Note

ElastiCache .NET Cluster Client sudah ditiadakan mulai Mei 2022.

Anda dapat menemukan kode ElastiCache .NET Cluster Client sebagai kode sumber terbuka di <https://github.com/awslabs/elasticache-cluster-config-net>.

Bagian ini menjelaskan cara menginstal, memperbarui, dan menghapus komponen .NET untuk ElastiCache Cluster Client pada instans Amazon EC2. Untuk informasi selengkapnya, lihat [Penemuan otomatis](#). Untuk contoh kode .NET untuk menggunakan klien, lihat [Penemuan otomatis dengan DotNET](#).

Topik

- [Menginstal .NET](#)
- [Unduh ElastiCache .NET Cluster Client untuk ElastiCache](#)
- [Instal kompilasi AWS dengan NuGet](#)

Menginstal .NET

Anda harus memiliki .NET 3.5 atau yang lebih baru terinstal untuk menggunakan AWS SDK .NET for 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 .NET Cluster Client untuk ElastiCache

Untuk mengunduh ElastiCache .NET Cluster Client

1. Masuk ke AWS Management Console dan buka konsol ElastiCache di <https://console.aws.amazon.com/elasticache/>.
2. Pada panel navigasi, klik Klien klaster ElastiCache.
3. Pada daftar Unduh klien klaster Memcached ElastiCache, pilih .NET, lalu klik Unduh.

Instal kompilasi AWS dengan NuGet

NuGet adalah sistem manajemen paket untuk platform .NET. NuGet mengetahui dependensi kompilasi dan menginstal semua file yang diperlukan secara otomatis. Kompilasi yang diinstall dengan NuGet disimpan bersama solusi Anda, bukan di lokasi pusat seperti Program Files, sehingga Anda dapat menginstal versi khusus untuk suatu aplikasi tanpa menimbulkan masalah kompatibilitas.

Menginstal NuGet

NuGet dapat diinstal dari Installation Gallery di MSDN; lihat <https://visualstudiogallery.msdn.microsoft.com/27077b70-9dad-4c64-adcf-c7cf6bc9970c>. Jika Anda menggunakan Visual Studio 2010 atau yang lebih baru, NuGet terinstal secara otomatis.

Anda dapat menggunakan NuGet baik dari Solution Explorer maupun 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 melalui NuGet, seperti kompilasi AWSSDK dan AWS.Extensions, lihat situs web NuGet di <http://www.nuget.org>. Halaman untuk setiap paket menyertakan contoh baris perintah untuk menginstal paket menggunakan konsol dan daftar versi sebelumnya dari paket yang tersedia melalui NuGet.

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 ElastiCache Cluster Client untuk PHP

Bagian ini menjelaskan cara menginstal, memperbarui, dan menghapus komponen PHP untuk ElastiCache Cluster Client pada instans Amazon EC2. Untuk informasi selengkapnya tentang Penemuan Otomatis, lihat [Identifikasi simpul di klaster Anda secara otomatis](#). Untuk contoh kode PHP untuk menggunakan klien, lihat [Menggunakan ElastiCache Cluster Client untuk PHP](#).

Topik

- [Mengunduh paket penginstalan](#)
- [Bagi pengguna yang sudah memiliki ekstensi php-memcached terinstal](#)
- [Langkah penginstalan untuk pengguna baru](#)
- [Menghapus klien klaster PHP](#)

Mengunduh paket penginstalan

Untuk memastikan bahwa Anda menggunakan versi ElastiCache Cluster Client yang benar untuk PHP, Anda perlu mengetahui versi PHP yang diinstal pada instans Amazon EC2 Anda. Anda juga perlu mengetahui apakah instans Amazon EC2 Anda menjalankan Linux versi 64-bit atau 32-bit.

Untuk menentukan versi PHP yang diinstal pada instans Amazon EC2 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 ElastiCache Cluster Client untuk PHP](#).

Untuk menentukan arsitektur AMI Amazon EC2 Anda (64-bit atau 32-bit)

1. Masuk ke AWS Management Console dan buka konsol Amazon EC2 di <https://console.aws.amazon.com/ec2/>.
2. Pada daftar Instans, klik instans Amazon EC2 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 ElastiCache Cluster Client untuk PHP

1. Masuk ke AWS Management Console dan buka konsol ElastiCache di <https://console.aws.amazon.com/elasticache/>.
2. Dari konsol ElastiCache, pilih Klien klaster ElastiCache.
3. Dari daftar Unduh klien klaster Memcached ElastiCache, pilih ElastiCache Cluster Client yang sesuai dengan versi PHP dan arsitektur AMI Anda, lalu pilih tombol Unduh.

Untuk klien yang didukung TLS, unduh biner dengan PHP versi 7.4 atau lebih tinggi.

Bagi pengguna yang sudah memiliki ekstensi php-memcached terinstal

Untuk memperbarui instalasi **php-memcached**

1. Hapus instalasi sebelumnya dari ekstensi Memcached untuk PHP seperti yang dijelaskan pada topik [Menghapus klien klaster PHP](#).
2. Instal ekstensi php-memcached ElastiCache yang baru seperti dijelaskan sebelumnya di [Langkah penginstalan untuk pengguna baru](#).

Langkah penginstalan untuk pengguna baru

Topik

- [Menginstal PHP 7.x - 8.x untuk pengguna baru](#)
- [Menginstal PHP 5.x untuk pengguna baru](#)

Menginstal PHP 7.x - 8.x untuk pengguna baru

Topik

- [Untuk menginstal PHP 7.x - 8.x di AMI Amazon Linux 2](#)
- [Untuk menginstal PHP 7.x - 8.x di AMI Amazon Linux 201609](#)
- [Untuk menginstal PHP 7.x - 8.x di AMI SUSE Linux 15](#)
- [Untuk menginstal PHP 7.x - 8.x di AMI Ubuntu 22.04](#)

Untuk menginstal PHP 7.x - 8.x di AMI Amazon Linux 2

Note

Jika perlu, ganti *PHP-7.x* dengan versi yang Anda gunakan.

1. Luncurkan instans baru dari AMI.
2. Jalankan perintah berikut:

```
sudo yum install gcc-c++ zlib-devel
```

3. Instal PHP 7.x menggunakan `amazon-linux-extras`

Dengan Amazon Linux 2, Anda dapat menggunakan Pustaka Extras untuk menginstal pembaruan aplikasi dan perangkat lunak pada instans Anda. Pembaruan perangkat lunak ini dikenal sebagai topik. Anda dapat menginstal versi spesifik dari topik atau menghilangkan informasi versi untuk menggunakan versi yang terbaru. Untuk informasi selengkapnya, [Pustaka Extras \(Amazon Linux 2\)](#).

Untuk melakukannya, ikuti langkah-langkah berikut:

- a. Pertama, verifikasi bahwa `amazon-linux-extras` sudah terinstal.

- b. Jika belum terinstal, gunakan perintah berikut untuk menginstal:

```
sudo yum install -y amazon-linux-extras
```

- c. Konfirmasikan bahwa topik PHP 7.x tersedia di mesin Amazon Linux 2:

```
sudo amazon-linux-extras | grep php
```

- d. Dari output, tinjau semua topik PHP 7 dan pilih versi yang Anda inginkan:

```
sudo amazon-linux-extras enable php7.x
```

- e. Instal paket PHP dari repositori. Contoh:

```
sudo yum clean metadata
```

```
sudo yum install php php-devel
```

4. Unduh Amazon ElastiCache Cluster Client.

- Buka konsol ElastiCache di <https://console.aws.amazon.com/elasticache/>.

Di dasbor ElastiCache, buka Klien kluster ElastiCache lalu pilih versi PHP7 yang Anda inginkan.

- Dari baris perintah, ganti PHP-7.X dengan versi PHP yang diinginkan, dan ganti ARCH dengan arsitektur yang diinginkan (X86 atau arm) dan untuk PHP >= 7.4, ganti OpenSSL dengan versi OpenSSL yang diinginkan (openssl1.1 atau openssl3). Jika Anda menggunakan PHP > 7.4, hapus akhiran OpenSSL.

```
wget https://elasticache-downloads.s3.amazonaws.com/ClusterClient/PHP-7.X/  
latest-64bit-<ARCH>-<OpenSSL>
```

5. Gunakan tar -zxvf untuk mengekstrak file unduhan.

```
tar -zxvf latest-64bit-<ARCH>-<OpenSSL>
```

6. Dengan izin root, salin file artefak `amazon-elasticache-cluster-client.so` yang diekstrak ke `/usr/lib64/php/modules`.


```
sudo mv amazon-elasticache-cluster-client.so /usr/lib64/php/modules/
```

7. Tambahkan `extension=amazon-elasticache-cluster-client.so` ke file `/etc/php.ini`

8. Jika Anda mengunduh ElastiCache Cluster Client dengan PHP 7.4 atau lebih tinggi, instal OpenSSL 1.1.x atau lebih tinggi. Petunjuk penginstalan untuk OpenSSL 1.1.1:

```
sudo yum -y update
sudo yum install -y make gcc perl-core pcre-devel wget zlib-devel
wget https://www.openssl.org/source/openssl-1.1.1c.tar.gz
tar xvf openssl-1.1.1c.tar.gz
cd openssl-1.1.1c
./config
make
sudo make install
sudo ln -s /usr/local/lib64/libssl.so.1.1 /usr/lib64/libssl.so.1.1
```

Untuk menginstal PHP 7.x - 8.x di AMI Amazon Linux 201609

 Note

Jika perlu, ganti *php7.x* dengan versi yang Anda gunakan.

1. Luncurkan instans baru dari AMI. Untuk informasi selengkapnya, lihat [Langkah 1: Luncurkan instans](#) dalam Panduan Pengguna Amazon EC2.
2. Jalankan perintah berikut:

```
sudo yum install gcc-c++
```

3. Instal PHP.

```
sudo yum install php7.x
```

4. Unduh Amazon ElastiCache Cluster Client.

```
wget https://elasticache-downloads.s3.amazonaws.com/ClusterClient/PHP-7.x/
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.x/modules/`.

```
sudo mv artifact/amazon-elasticache-cluster-client.so /usr/lib64/php/7.x/modules/
```

7. Buat file `50-memcached.ini`.

```
echo "extension=amazon-elasticache-cluster-client.so" | sudo tee --append /etc/php-7.x.d/50-memcached.ini
```

8. Mulai atau mulai ulang server Apache Anda.

```
sudo /etc/init.d/httpd start
```

Untuk menginstal PHP 7.x - 8.x di AMI SUSE Linux 15

Note

Jika perlu, ganti `php7.x` dengan versi yang Anda gunakan.

1. Luncurkan instans baru dari AMI.
2. Jalankan perintah berikut:

```
sudo zypper refresh
sudo zypper update -y
sudo zypper install gcc
```

3. Instal PHP.

```
sudo yum install php7.x
```

atau

```
sudo zypper addrepo //download.opensuse.org/repositories/devel:/languages:/php/  
openSUSE_Leap_15.3/ php
```

4. Unduh Amazon ElastiCache Cluster Client, lalu ganti <ARCH> dengan arsitektur yang diinginkan (X86 atau arm). SUSE 15 hadir dengan OpenSSL1.1 bawaan, jadi untuk PHP >= 7.4, pilih biner klien dengan OpenSSL1. Jika Anda menggunakan PHP < 7.4, hapus akhiran OpenSSL.

```
wget https://elasticache-downloads.s3.amazonaws.com/ClusterClient/PHP-7.x/  
latest-64bit-<ARCH>-openssl1.1
```

5. Ekstrak file latest-64bit.

```
tar -zxvf latest-64bit-<ARCH>-openssl1.1
```

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
```

Untuk menginstal PHP 7.x - 8.x di AMI Ubuntu 22.04

Note

Jika perlu, ganti *php7.x* dengan versi yang Anda gunakan.

1. Luncurkan instans baru dari AMI.

2. Jalankan perintah berikut:

```
sudo apt-get update
sudo apt-get install gcc g++ make zlib1g zlib1g-dev
```

3. Instal PHP.

a. Petunjuk penginstalan untuk PHP 8.1:

```
sudo apt install php8.1-cli php8.1-dev
```

b. Petunjuk penginstalan untuk PHP 7.4:

```
sudo apt -y install software-properties-common
sudo add-apt-repository ppa:ondrej/php
sudo apt-get update
sudo apt -y install php7.4
```

4. Unduh Amazon ElastiCache Cluster Client, lalu ganti <ARCH> dengan arsitektur yang diinginkan (X86 atau arm). Ubuntu 22.04 hadir dengan OpenSSL3 bawaan, jadi untuk PHP >= 7.4, pilih biner klien dengan OpenSSL3. Jika Anda menggunakan PHP < 7.4, hapus akhiran OpenSSL.

```
wget https://elasticache-downloads.s3.amazonaws.com/ClusterClient/PHP-7.x/
latest-64bit-<ARCH>-openssl3
```

5. Ekstrak file latest-64bit.

```
tar -zxvf latest-64bit-<ARCH>-openssl3
```

6. Dengan izin root, salin file artefak `amazon-elasticache-cluster-client.so` yang diekstrak ke direktori ekstensi php `/usr/lib/php/20190902`. Jika direktori ekstensi ini tidak ada, Anda dapat menemukannya dengan menjalankan: `php -i | grep extension_dir`
7. Sisipkan baris `extension=amazon-elasticache-cluster-client.so` ke file `/etc/php/7.x/cli/php.ini`.

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 paket php-memcached yang benar untuk instans Amazon EC2 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. Pada contoh ini, ganti *X.Y.Z* dengan nomor versi yang sebenarnya:

```
$ sudo pecl install /home/AmazonElastiCacheClusterClient-X.Y.Z-PHP54-64bit.tgz
```

Note

Pastikan untuk menggunakan versi terbaru artefak penginstalan.

5. Dengan izin root/sudo, tambahkan file baru bernama `memcached.ini` di direktori `/etc/php.d`, dan sisipkan "extension=amazon-elasticache-cluster-client.so" ke 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 paket php-memcached yang benar untuk instans Amazon EC2 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 izin root/sudo, tambahkan file baru bernama memcached.ini di direktori /etc/php.d, dan sisipkan extension=amazon-elasticache-cluster-client.so ke 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 paket php-memcached yang benar untuk instans Amazon EC2 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 izin root/sudo, tambahkan file baru bernama `memcached.ini` ke dalam direktori `/etc/php5/cli/conf.d`, dan sisipkan `"extension=<absolute path to amazon-elasticache-cluster-client.so>"` ke 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 paket php-memcached yang benar untuk instans Amazon EC2 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 izin root/sudo, tambahkan file baru bernama `memcached.ini` di direktori `/etc/php5/conf.d`, dan sisipkan **`extension=amazon-elasticache-cluster-client.so`** ke 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 di dalam ekstensi. Selain itu, verifikasi bahwa PHP yang digunakan adalah versi yang didukung. Kami mendukung versi 5.3 hingga 5.5.

Distribusi Linux lainnya

Pada beberapa sistem, terutama CentOS7 dan Red Hat Enterprise Linux (RHEL) 7.1, `libsas12.so.3` telah menggantikan `libsas12.so.2`. Pada sistem tersebut, ketika Anda memuat ElastiCache Cluster Client, sistem akan mencoba, tetapi gagal menemukan dan memuat `libsas12.so.2`. Untuk mengatasi masalah ini, buat tautan simbolis ke `libsas12.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 lama PHP 7 atau lebih tinggi](#)
- [Menghapus versi lama PHP 5](#)

Menghapus versi lama PHP 7 atau lebih tinggi

Untuk menghapus versi lama PHP 7 atau lebih tinggi

1. Hapus file `amazon-elasticache-cluster-client.so` dari direktori pustaka PHP yang sesuai seperti yang ditunjukkan sebelumnya pada petunjuk penginstalan. 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 ElastiCache Cluster Client untuk PHP

Bagian ini membahas cara mendapatkan dan mengompilasi kode sumber ElastiCache Cluster Client untuk PHP.

Ada dua paket yang perlu ditarik dari GitHub dan dikompilasi; [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

Pustaka prasyarat

- OpenSSL 1.1.0 atau lebih tinggi (kecuali dukungan TLS dinonaktifkan oleh `./configure --disable-tls`).
- SASL (libsasl2, kecuali dukungan SASL dinonaktifkan oleh `./configure --disable-sasl`).

Untuk mengompilasi pustaka `aws-elasticache-cluster-client-libmemcached`

1. Luncurkan instans Amazon EC2.
2. Instal dependensi pustaka.
 - Di AMI Amazon Linux 201509/AMI Amazon Linux 2

```
sudo yum -y update
sudo yum install gcc gcc-c++ autoconf libevent-devel make perl-core pcre-devel
wget zlib-devel
// Install OpenSSL 1.1.1
wget https://www.openssl.org/source/openssl-1.1.1c.tar.gz
tar xvf openssl-1.1.1c.tar.gz
cd openssl-1.1.1c
./config
make
sudo make install
sudo ln -s /usr/local/lib64/libssl.so.1.1 /usr/lib64/libssl.so.1.1
```

- Di AMI Ubuntu 14.04 (tidak diperlukan untuk versi Ubuntu yang disertai OpenSSL \geq 1.1)

```
sudo apt-get update

sudo apt-get install libevent-dev gcc g++ make autoconf libsasl2-dev

// Install OpenSSL 1.1.1
wget https://www.openssl.org/source/openssl-1.1.1c.tar.gz
tar xvf openssl-1.1.1c.tar.gz
cd openssl-1.1.1c
./config
make
sudo make install
sudo ln -s /usr/local/lib/libssl.so.1.1 /usr/lib/x86_64-linux-gnu/libssl.so.1.1
```

3. Tarik repositori dan kompilasikan kode.

```
git clone https://github.com/aws-labs/aws-elasticache-cluster-client-libmemcached.git
cd aws-elasticache-cluster-client-libmemcached
touch configure.ac aclocal.m4 configure Makefile.am Makefile.in
mkdir BUILD
cd BUILD
../configure --prefix=<libmemcached-install-directory> --with-pic --disable-sasl
```

Jika menjalankan `../configure` gagal menemukan `libssl` (pustaka OpenSSL), mungkin variabel lingkungan `PKG_CONFIG_PATH` perlu diubah:

```
PKG_CONFIG_PATH=/path/to/ssl/lib/pkgconfig ../configure --prefix=<libmemcached-install-directory> --with-pic --disable-sasl
```

Alternatifnya, jika Anda tidak menggunakan TLS, Anda dapat menonaktifkannya dengan menjalankan:

```
make
sudo make install
../configure --prefix=<libmemcached-install-directory> --with-pic --disable-sasl --disable-tls
```

Mengompilasi klien penemuan otomatis ElastiCache Memcached untuk PHP

Bagian berikut menjelaskan cara mengompilasi ElastiCache Memcached Auto Discovery Client

Topik

- [Mengompilasi klien ElastiCache Memcached untuk PHP 7 atau lebih tinggi](#)
- [Mengompilasi klien ElastiCache Memcached untuk PHP 5](#)

Mengompilasi klien ElastiCache Memcached untuk PHP 7 atau lebih tinggi

Ganti PHP-7.x dengan versi yang Anda gunakan.

Instal PHP:

```
sudo yum install -y amazon-linux-extras
sudo amazon-linux-extras enable php7.x
```

Jalankan rangkaian perintah berikut dalam direktori kode.

```
git clone https://github.com/awslabs/aws-elasticache-cluster-client-memcached-for-
php.git
cd aws-elasticache-cluster-client-memcached-for-php
phpize
mkdir BUILD
CD BUILD
../configure --with-libmemcached-dir=<libmemcached-install-directory> --disable-
memcached-sasl
```

Jika menjalankan `../configure` gagal menemukan `libssl` (pustaka OpenSSL), mungkin perlu variabel lingkungan `PKG_CONFIG_PATH` diubah ke direktori file `.PC` OpenSSL:

```
PKG_CONFIG_PATH=/path/to/ssl/lib/pkgconfig ../configure --with-libmemcached-dir=<path
to libmemcached build directory> --disable-memcached-sasl
```

Alternatifnya, jika Anda tidak menggunakan TLS, Anda dapat menonaktifkannya dengan menjalankan:

```
make
make install
```

```
../configure --with-libmemcached-dir=<path to libmemcached build directory> --disable-memcached-sasl --disable-memcached-tls
```

Note

Anda secara statis dapat menghubungkan pustaka libmemcached ke biner PHP sehingga dapat di-porting 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/aws-labs/aws-elasticache-cluster-client-memcached-for-php/tree/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
```

Mengonfigurasi Klien ElastiCache

Klaster ElastiCache memiliki kepatuhan protokol dengan Memcached. Kode, aplikasi, dan alat paling populer yang Anda gunakan saat ini dengan lingkungan Memcached yang sudah ada akan berfungsi secara lancar dengan layanan ini.

Bagian ini membahas pertimbangan khusus untuk terhubung ke simpul cache di ElastiCache.

Topik

- [Menemukan titik akhir dan nomor port simpul](#)
- [Menghubungkan untuk menggunakan penemuan otomatis](#)
- [Nama DNS dan IP yang mendasari](#)

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 manajemen Amazon ElastiCache](#) dan pilih mesin yang berjalan di klaster 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 dan nomor port simpul cache (API ElastiCache)

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 kluster, bukan titik akhir individual untuk setiap simpul cache. Untuk informasi selengkapnya, lihat [Identifikasi simpul di kluster Anda secara otomatis](#).

Note

Pada saat ini, Penemuan Otomatis hanya tersedia untuk kluster cache yang menjalankan Memcached.

Nama DNS dan IP yang mendasari

Klien memelihara daftar server yang berisi alamat dan port dari server yang menampung data cache. Saat menggunakan ElastiCache, API `DescribeCacheClusters` (atau utilitas baris perintah `describe-cache-clusters`) menampilkan 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.

Penginstalan VPC

ElastiCache memastikan bahwa nama DNS dan alamat IP dari simpul cache tetap sama ketika simpul cache dipulihkan dalam kasus kegagalan.

Penginstalan Non-VPC

ElastiCache memastikan bahwa nama DNS simpul cache tidak berubah ketika simpul cache dipulihkan dalam kasus kegagalan; namun, alamat IP yang mendasari simpul cache dapat berubah.

Kebanyakan pustaka klien mendukung koneksi simpul cache persisten secara default. Kami merekomendasikan untuk 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.

Menyiapkan klaster

Berikut ini, terdapat petunjuk tentang cara membuat klaster menggunakan konsol ElastiCache, AWS CLI, atau API ElastiCache.

Anda juga dapat membuat klaster ElastiCache 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, sebaiknya lakukan pekerjaan persiapan tertentu agar Anda tidak perlu melakukan peningkatan atau melakukan perubahan.

Topik

- [Menentukan kebutuhan Anda](#)
- [Memilih ukuran simpul Anda](#)

Menentukan kebutuhan Anda

Persiapan

Mengetahui jawaban atas pertanyaan berikut membantu memudahkan proses pembuatan kluster Anda:

Apakah Anda ingin menggunakan layanan ElastiCache tanpa server atau berbasis instance?

Jika Anda ingin menggunakan cache nirserver, cukup pastikan bahwa Anda telah mengonfigurasi VPC, subnet, dan grup keamanan Anda dengan benar. Untuk detail selengkapnya, lihat [Pola Akses untuk Mengakses Cache ElastiCache di Amazon VPC](#). Jika Anda ingin menggunakan berbasis instance ElastiCache, lanjutkan membaca.

- Jenis instans simpul apa yang dibutuhkan?

Untuk panduan terkait cara memilih jenis simpul instans, lihat [Memilih ukuran simpul Memcached Anda](#).

- Apakah kluster Anda diluncurkan di cloud privat virtual (VPC) berdasarkan Amazon VPC?

Important

Jika Anda akan meluncurkan kluster di VPC, pastikan untuk membuat grup subnet di VPC yang sama sebelum Anda mulai membuat kluster. Untuk informasi selengkapnya, lihat [Subnet dan grup subnet](#).

ElastiCache dirancang untuk diakses dari dalam AWS menggunakan Amazon EC2. Namun, jika Anda meluncurkan di VPC berdasarkan Amazon VPC dan kluster Anda berada di VPC, Anda dapat menyediakan akses dari luar AWS. Untuk informasi selengkapnya, lihat [Pola Akses untuk Mengakses Cache ElastiCache di Amazon VPC](#).

- Apakah Anda perlu menyesuaikan nilai parameter tertentu?

Jika perlu, buat grup parameter khusus. Untuk informasi selengkapnya, lihat [Membuat grup parameter](#).

- 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

- [Persyaratan memori dan prosesor](#)
- [Konfigurasi kluster Memcached](#)
- [Persyaratan penskalaan](#)
- [Persyaratan akses](#)
- [Persyaratan Wilayah, Availability Zone, dan Zona Lokal](#)

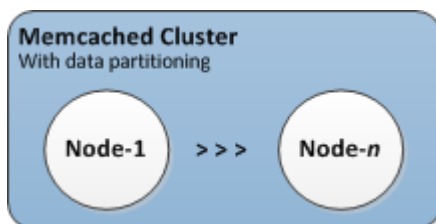
Persyaratan memori dan prosesor

Blok bangunan dasar Amazon ElastiCache adalah simpul. Simpul dikonfigurasi secara tunggal atau dalam pengelompokan untuk membentuk kluster. Saat menentukan jenis simpul yang akan digunakan untuk kluster Anda, pertimbangkan konfigurasi simpul kluster dan jumlah data yang harus disimpan.

Mesin Memcached adalah multi-threaded. Jadi, jumlah core dari simpul berdampak pada daya komputasi yang tersedia untuk kluster.

Konfigurasi kluster Memcached

ElastiCache untuk cluster Memcached terdiri dari 1 hingga 60 node. Data di dalam kluster Memcached dipartisi di seluruh simpul di kluster tersebut. Aplikasi Anda terhubung dengan kluster Memcached menggunakan alamat jaringan yang disebut Titik Akhir. Setiap simpul dalam kluster Memcached memiliki titik akhir sendiri yang digunakan oleh aplikasi Anda untuk membaca dari atau menulis ke simpul tertentu. Selain titik akhir simpul, kluster Memcached itu sendiri memiliki titik akhir yang disebut titik akhir konfigurasi. Aplikasi Anda dapat menggunakan titik akhir ini untuk membaca dari atau menulis ke kluster, dan membiarkan [Identifikasi simpul di kluster Anda secara otomatis](#) menentukan simpul tempat proses baca atau proses tulis akan dilakukan.



Untuk informasi selengkapnya, lihat [Mengelola kluster](#).

Persyaratan penskalaan

Semua klaster dapat dinaikkan skalanya dengan membuat klaster baru yang memiliki jenis simpul baru yang lebih besar. Saat Anda menaikkan skala klaster Memcached, klaster baru dimulai dalam keadaan 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 untuk Memcached](#) dalam panduan ini.

Persyaratan akses

Secara desain, ElastiCache cluster Amazon diakses dari instans Amazon EC2. Akses jaringan ke ElastiCache cluster terbatas pada akun yang membuat cluster. Oleh karena itu, sebelum Anda dapat mengakses klaster dari instans Amazon EC2, Anda harus memberikan otorisasi pada instans Amazon EC2 untuk mengakses klaster tersebut. Langkah untuk melakukan hal ini bervariasi, tergantung apakah Anda meluncurkan ke EC2-VPC atau EC2-Classic.

Jika Anda meluncurkan klaster Anda ke EC2-VPC, Anda perlu memberikan izin masuk jaringan ke klaster tersebut. Jika meluncurkan klaster ke EC2-Classic, Anda harus memberikan grup keamanan Amazon Elastic Compute Cloud yang terkait dengan instans akses ke grup keamanan Anda ElastiCache . Untuk instruksi yang lebih lengkap, lihat [Mengakses klaster Anda](#) dalam panduan ini.

Persyaratan Wilayah, Availability Zone, dan Zona Lokal

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, menemukan simpul Anda di Availability Zone atau di Local Zones yang berbeda dapat mengurangi dampak kegagalan pada klaster Anda.

Untuk informasi selengkapnya, lihat hal berikut:

- [Wilayah dan zona ketersediaan](#)
- [Zona lokal](#)
- [Mitigasi Kegagalan](#)

Memilih ukuran simpul Anda

Ukuran simpul yang Anda pilih untuk klaster Anda memengaruhi biaya, performa, dan toleransi kesalahan.

Memilih ukuran simpul Memcached Anda

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 pengganti ini akan diisi ulang. Untuk mengurangi dampak ketersediaan ini, sebarkan kapasitas memori dan komputasi Anda ke lebih banyak simpul dengan kapasitas yang lebih kecil, daripada menggunakan sedikit simpul yang berkapasitas tinggi.

Dalam skenario saat 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 USD	11	35,42	22	538,56 USD
cache.m4.large	6,42	2	0,156 USD	6	38,52	12	673,92 USD
cache.m4.xlarge	14,28	4	0,311 USD	3	42,84	12	671,76 USD
cache.m5.xlarge	12,93	4	0,311 USD	3	38,81	12	671,76 USD
cache.m6g.large	6,85	2	0,147 USD	6	41,1	12	635 USD
cache.r4.large	12,3	2	0,228 USD	3	36,9	6	492,48 USD
cache.r5.large	13,07	2	0,216 USD	3	39,22	6	466,56 USD
cache.r6g.large	13,07	2	0,205 USD	3	42,12	6	442 USD

* 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, tetapi kapasitas komputasi dan biaya yang berbeda. Untuk membandingkan biaya opsi spesifik Anda, lihat [Harga Amazon ElastiCache](#).

Untuk klaster yang menjalankan Memcached, beberapa memori yang tersedia pada setiap simpul digunakan untuk overhead koneksi. Untuk informasi selengkapnya, lihat [Overhead koneksi Memcached](#)

Penggunaan beberapa simpul membutuhkan penyebaran kunci di seluruh simpul ini. Setiap simpul memiliki titik akhir sendiri. Untuk pengelolaan titik akhir yang mudah, Anda dapat menggunakan fitur Penemuan Otomatis ElastiCache, yang memungkinkan program klien secara otomatis mengidentifikasi semua simpul dalam suatu klaster. Untuk informasi selengkapnya, lihat [Identifikasi simpul di klaster Anda secara otomatis](#).

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 metrik ElastiCache yang diterbitkan ke Amazon CloudWatch. Untuk informasi selengkapnya tentang metrik CloudWatch untuk ElastiCache, lihat [Memantau penggunaan dengan Metrik CloudWatch](#). 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 klaster

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, memperbarui daftar titik akhir seiring penambahan dan penghapusan simpul. Aplikasi Anda hanya memerlukan titik akhir konfigurasi agar dapat berinteraksi dengan klaster. Untuk informasi selengkapnya tentang penemuan otomatis, lihat [Identifikasi simpul di klaster Anda secara otomatis](#).

Untuk membuat klaster Memcached, ikuti langkah-langkah dalam [Langkah 1: Buat Cache](#)

Setelah status klaster Anda menjadi tersedia, Anda dapat memberikan Amazon EC2 akses ke klaster tersebut, terhubung dengannya, dan mulai menggunakannya. Untuk informasi selengkapnya, lihat [Mengakses klaster Anda](#) dan [Menghubungkan ke Simpul Cache Secara Manual](#).

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 klaster](#).

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 durasi saat klaster aktif, meskipun Anda tidak sedang aktif menggunakannya. Untuk menghentikan tagihan biaya untuk klaster ini, Anda harus menghapusnya. Lihat [Menghapus klaster](#).

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 (ElastiCache API)

Untuk membuat cluster menggunakan ElastiCache API, gunakan `CreateCacheCluster` tindakan.

Important

Setelah klaster Anda tersedia, Anda akan ditagih untuk setiap jam atau durasi saat klaster aktif, meskipun Anda tidak sedang menggunakannya. Untuk menghentikan tagihan biaya untuk klaster ini, Anda harus menghapusnya. Lihat [Menghapus klaster](#).

Membuat cluster cache Memcached (API) ElastiCache

Kode berikut membuat cluster Memcached dengan 3 node (ElastiCache API).

Jeda baris ditambahkan agar 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 klaster

Anda dapat melihat informasi detail tentang satu atau lebih klaster menggunakan konsol ElastiCache, AWS CLI, atau API ElastiCache.

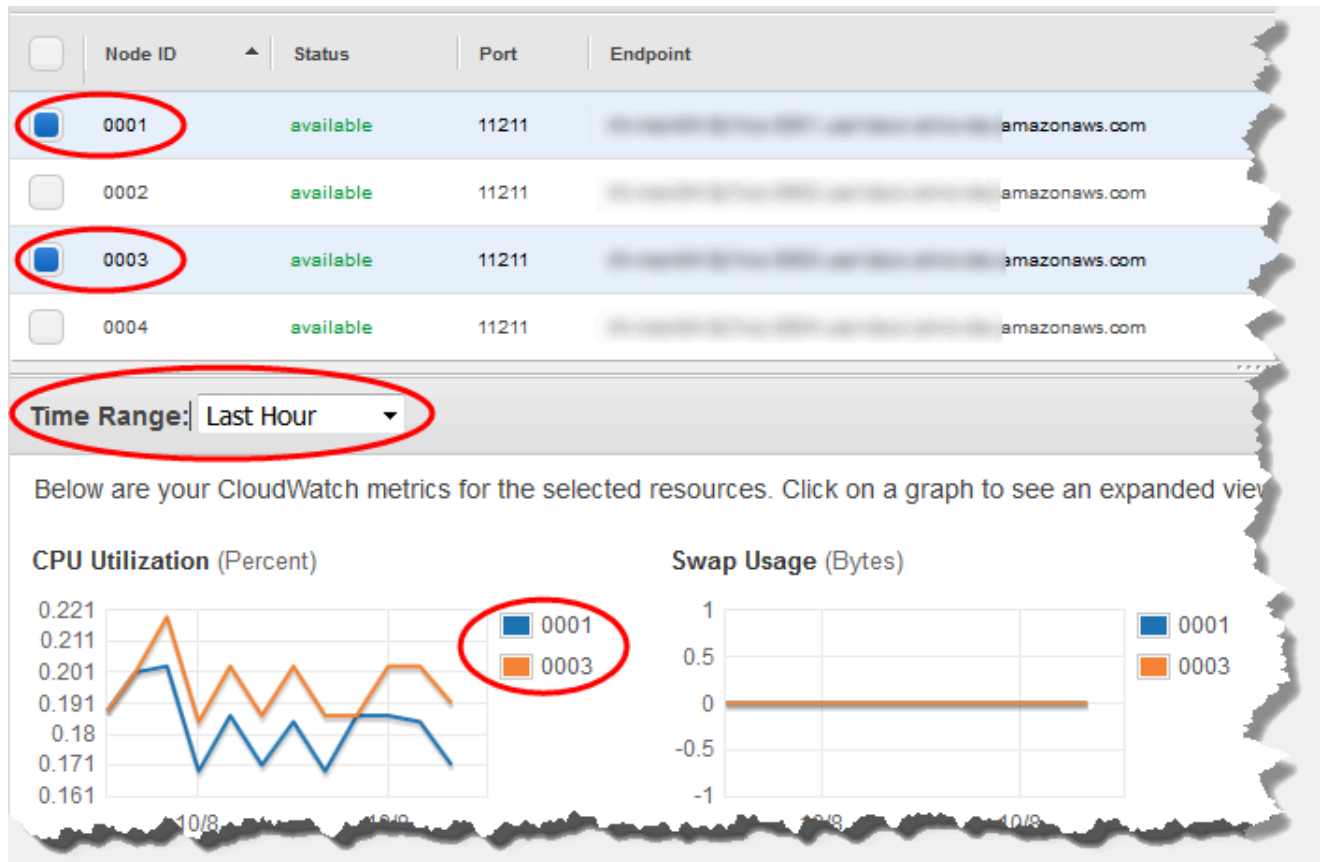
Melihat Detail Klaster (Konsol)

Anda dapat melihat detail klaster Memcached menggunakan konsol ElastiCache, AWS CLI untuk ElastiCache, atau API ElastiCache.

Prosedur berikut memerinci cara untuk melihat detail klaster Memcached menggunakan konsol ElastiCache.

Untuk melihat detail klaster Memcached

1. Masuk ke AWS Management Console dan buka konsol Amazon ElastiCache di <https://console.aws.amazon.com/elasticache/>.
2. Dari daftar di sudut kanan atas, pilih Wilayah AWS yang Anda minati.
3. Pada dasbor konsol ElastiCache, memilih Memcached. Melakukan hal ini akan menampilkan daftar semua klaster yang menjalankan versi Memcached apa pun.
4. Untuk melihat detail klaster, pilih kotak di sebelah kiri nama klaster.
5. Untuk melihat informasi simpul:
 - a. Pilih nama klaster.
 - b. Pilih tab Simpul.
 - c. Untuk melihat metrik pada satu simpul atau lebih, pilih kotak di sebelah kiri ID Simpul, lalu pilih rentang waktu untuk metrik tersebut dari daftar Rentang waktu. Jika beberapa simpul dipilih, grafik overlay akan dibuat.



Metrik selama satu jam terakhir untuk dua simpul Memcached

Melihat detail klaster (AWS CLI)

Anda dapat melihat detail untuk klaster menggunakan perintah AWS CLI `describe-cache-clusters`. Jika parameter `--cache-cluster-id` dihilangkan, detail untuk maksimal `--max-items` klaster akan ditampilkan. Jika parameter `--cache-cluster-id` disertakan, detail untuk klaster yang ditentukan akan ditampilkan. Anda dapat membatasi jumlah catatan yang ditampilkan dengan parameter `--max-items`.

Kode berikut menampilkan daftar detail untuk `my-cluster`.

```
aws elasticache describe-cache-clusters --cache-cluster-id my-cluster
```

Kode berikut menampilkan daftar detail untuk maksimal 25 klaster.

```
aws elasticache describe-cache-clusters --max-items 25
```

Example

Untuk Linux, macOS, atau Unix:

```
aws elasticache describe-cache-clusters \  
  --cache-cluster-id my-cluster \  
  --show-cache-node-info
```

Untuk Windows:

```
aws elasticache describe-cache-clusters ^  
  --cache-cluster-id my-cluster ^  
  --show-cache-node-info
```

Operasi ini menghasilkan output seperti yang berikut ini (format JSON):

```
{  
  "CacheClusters": [  
    {  
      "Engine": "memcached",  
      "CacheNodes": [  
        {  
          "CacheNodeId": "0001",  
          "Endpoint": {  
            "Port": 11211,  
            "Address": "my-cluster.7ef-  
example.0001.usw2.cache.amazonaws.com"  
          },  
          "CacheNodeStatus": "available",  
          "ParameterGroupStatus": "in-sync",  
          "CacheNodeCreateTime": "2016-09-21T16:28:28.973Z",  
          "CustomerAvailabilityZone": "us-west-2b"  
        },  
        {  
          "CacheNodeId": "0002",  
          "Endpoint": {  
            "Port": 11211,  
            "Address": "my-cluster.7ef-  
example.0002.usw2.cache.amazonaws.com"  
          },  
          "CacheNodeStatus": "available",  
          "ParameterGroupStatus": "in-sync",  
          "CacheNodeCreateTime": "2016-09-21T16:28:28.973Z",
```

```
        "CustomerAvailabilityZone": "us-west-2b"
    },
    {
        "CacheNodeId": "0003",
        "Endpoint": {
            "Port": 11211,
            "Address": "my-cluster.7ef-
example.0003.usw2.cache.amazonaws.com"
        },
        "CacheNodeStatus": "available",
        "ParameterGroupStatus": "in-sync",
        "CacheNodeCreateTime": "2016-09-21T16:28:28.973Z",
        "CustomerAvailabilityZone": "us-west-2b"
    }
],
"CacheParameterGroup": {
    "CacheNodeIdsToReboot": [],
    "CacheParameterGroupName": "default.memcached1.4",
    "ParameterApplyStatus": "in-sync"
},
"CacheClusterId": "my-cluster",
"PreferredAvailabilityZone": "us-west-2b",
"ConfigurationEndpoint": {
    "Port": 11211,
    "Address": "my-cluster.7ef-example.cfg.usw2.cache.amazonaws.com"
},
"CacheSecurityGroups": [],
"CacheClusterCreateTime": "2016-09-21T16:28:28.973Z",
"AutoMinorVersionUpgrade": true,
"CacheClusterStatus": "available",
"NumCacheNodes": 3,
"ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
"SecurityGroups": [
    {
        "Status": "active",
        "SecurityGroupId": "sg-dbe93fa2"
    }
],
"CacheSubnetGroupName": "default",
"EngineVersion": "1.4.24",
"PendingModifiedValues": {},
"PreferredMaintenanceWindow": "sat:09:00-sat:10:00",
"CacheNodeType": "cache.m3.medium"
```

```
    }  
  ]  
}
```

Untuk informasi selengkapnya, lihat AWS CLI untuk topik ElastiCache [describe-cache-clusters](#).

Melihat detail klaster (API ElastiCache)

Anda dapat melihat detail untuk klaster menggunakan tindakan `DescribeCacheClusters` API ElastiCache. Jika parameter `CacheClusterId` disertakan, detail untuk klaster tertentu akan dikembalikan. Jika parameter `CacheClusterId` dihilangkan, detail hingga `MaxRecords` klaster (defaultnya 100) akan dikembalikan. Nilai untuk `MaxRecords` tidak boleh kurang dari 20 atau lebih besar dari 100.

Kode berikut menampilkan 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 mencantumkan detail hingga 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 lebih lanjut, lihat topik referensi API ElastiCache [DescribeCacheClusters](#).

Memodifikasi sebuah cluster ElastiCache

Selain menambahkan atau menghapus simpul dari klaster, ada waktu ketika Anda perlu membuat perubahan lain pada klaster yang ada, seperti, menambahkan grup keamanan, mengubah periode 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 ini diterapkan pada klaster secara langsung atau setelah klaster diaktifkan ulang. Hal 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 Memcached](#). Untuk informasi tentang cara melakukan boot ulang simpul klaster, lihat [Melakukan boot ulang klaster](#).

Menggunakan 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:
 - Kompatibilitas Versi Mesin
 - Grup Keamanan VPC
 - Grup Parameter
 - Periode Pemeliharaan
 - Topik untuk Notifikasi SNS

Kotak Terapkan Segera hanya berlaku untuk perubahan versi mesin. 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 perubahan periode pemeliharaan, akan diterapkan segera.

7. Pilih Ubah.

Menggunakan AWS CLI

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 meng-upgrade ke versi mesin yang lebih baru. Untuk informasi selengkapnya tentang cara melakukannya, lihat [Versi mesin dan pemutakhiran](#). Namun, Anda tidak dapat men-downgrade 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 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 selengkapnya, lihat ElastiCache topik AWS CLI untuk [modify-cache-cluster](#).

Menggunakan ElastiCache API

Anda dapat memodifikasi klaster 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 meng-upgrade ke versi mesin yang lebih baru. Untuk informasi selengkapnya tentang cara melakukannya, lihat [Versi mesin dan pemutakhiran](#). Namun, Anda tidak dapat men-downgrade ke versi mesin yang lebih lama kecuali dengan menghapus klaster yang ada dan membuatnya lagi.

Jeda baris ditambahkan agar 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](#).

Melakukan boot ulang klaster

Beberapa perubahan mengharuskan klaster di-boot ulang agar perubahan dapat diterapkan. Misalnya, untuk beberapa parameter, mengubah nilai parameter di dalam grup parameter hanya diterapkan setelah boot ulang.

Saat Anda melakukan boot ulang klaster, klaster akan membersihkan semua data dan memulai ulang mesinnya. Selama proses ini Anda tidak dapat mengakses klaster. Karena klaster membersihkan semua datanya, saat klaster tersedia lagi, Anda memulai dengan klaster yang kosong.

Anda dapat melakukan boot ulang klaster menggunakan konsol ElastiCache, AWS CLI, atau API ElastiCache. Entah Anda menggunakan konsol ElastiCache, AWS CLI, atau API ElastiCache, Anda hanya dapat memulai boot ulang klaster tunggal. Untuk melakukan boot ulang beberapa klaster, Anda harus mengulangi proses atau operasi.

Menggunakan AWS Management Console

Anda dapat melakukan boot ulang klaster menggunakan konsol ElastiCache.

Untuk melakukan boot ulang klaster (konsol)

1. Masuk ke AWS Management Console dan buka konsol ElastiCache di <https://console.aws.amazon.com/elasticache/>.
2. Dari daftar di sudut kanan atas, pilih Wilayah AWS 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 akan di-boot ulang dengan memilih tombol di sebelah kiri nama klaster.

Pilih Tindakan, lalu pilih Boot ulang.

Jika Anda memilih lebih dari satu klaster, tombol Boot ulang menjadi tidak aktif.

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 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 semua ID simpul dari klaster. Untuk informasi selengkapnya, lihat [reboot-cache-cluster](#).

Menggunakan API ElastiCache

Untuk melakukan boot ulang klaster menggunakan API ElastiCache, gunakan tindakan `RebootCacheCluster`.

Untuk melakukan boot ulang simpul tertentu dalam klaster, gunakan `CacheNodeIdsToReboot` untuk menampilkan 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 semua ID simpul dari klaster. Untuk informasi selengkapnya, lihat

[RebootCacheCluster](#).

Menambahkan simpul ke klaster

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 [Buat konfigurasi klien ElastiCache Anda untuk penyeimbangan beban yang efisien](#).

Anda dapat menggunakan Konsol Manajemen ElastiCache, AWS CLI, atau API ElastiCache untuk menambahkan simpul ke klaster Anda.

Menggunakan AWS Management Console

Topik

- [Untuk menambahkan simpul ke klaster \(konsol\)](#)

Untuk menambahkan simpul ke klaster (konsol)

Prosedur berikut dapat digunakan untuk menambahkan simpul ke klaster.

1. Masuk ke AWS Management Console dan buka konsol ElastiCache 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.
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	<p>Permintaan penghapusan baru, tertunda atau langsung, akan menggantikan permintaan penghapusan yang tertunda.</p> <p>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.</p>
Skenario 2	Hapus	Buat	<p>Permintaan pembuatan baru, tertunda atau langsung, akan menggantikan permintaan penghapusan tertunda.</p> <p>Misalnya, jika ada penghapusan tertunda untuk simpul 0001, 0003, dan 0007 dan permintaan baru untuk membuat simpul dikirimkan, simpul baru akan dibuat dan simpul 0001, 0003, dan 0007 tidak akan dihapus.</p>
Skenario 3	Buat	Hapus	<p>Permintaan penghapusan baru, tertunda atau langsung, akan menggantikan permintaan pembuatan tertunda.</p> <p>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.</p>

Skenario	Operasi Tertunda	Permintaan Baru	Hasil
Skenario 4	Buat	Buat	<p>Permintaan pembuatan baru ditambahkan ke permintaan pembuatan yang tertunda.</p> <p>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.</p> <div style="border: 1px solid #f08080; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p>⚠ Important</p> <p>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.</p> </div>

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 akan dapat digunakan.

Menggunakan AWS CLI

Untuk menambahkan simpul ke klaster menggunakan AWS CLI, gunakan operasi AWS CLI `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 kluster ini setelah perubahan diterapkan. Untuk menambahkan simpul ke kluster ini, `--num-cache-nodes` harus lebih besar dari jumlah simpul saat ini di dalam kluster. Jika nilai ini kurang dari jumlah simpul saat ini, ElastiCache mengharapkan parameter `cache-node-ids-to-remove` dan daftar simpul yang akan dihapus dari kluster. Untuk informasi selengkapnya, lihat [Menggunakan AWS CLI](#).
- `--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",
  }
}

```

Untuk informasi selengkapnya, lihat topik AWS CLI [modify-cache-cluster](#).

Menggunakan API ElastiCache

Untuk menambahkan simpul ke klaster (API ElastiCache)

- Panggil operasi API `ModifyCacheCluster` dengan parameter berikut ini:
 - `CacheClusterId` – ID dari klaster yang ingin ditambahi simpul.
 - `NumCacheNodes` – Parameter `NumCacheNodes` 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 di dalam klaster. Jika nilai ini kurang dari jumlah simpul saat ini, ElastiCache mengharapkan parameter `CacheNodeIdsToRemove` dengan daftar simpul yang akan dihapus dari klaster (lihat [Menggunakan API ElastiCache](#)).
 - `ApplyImmediately` – Menentukan apakah akan menambahkan simpul ini secara langsung atau pada periode pemeliharaan berikutnya.
 - `Region` – Menentukan Wilayah AWS untuk klaster yang ingin ditambahi simpul.

Contoh berikut menunjukkan panggilan untuk menambahkan simpul ke kluster.

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 API ElastiCache [ModifyCacheCluster](#).

Menghapus simpul dari klaster

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 [Buat konfigurasi klien ElastiCache Anda untuk penyeimbangan beban yang efisien](#).

Anda dapat menghapus simpul dari klaster menggunakan AWS Management Console, AWS CLI, atau API ElastiCache.

Menggunakan AWS Management Console

Untuk menghapus simpul dari klaster (konsol)

1. Masuk ke AWS Management Console dan buka konsol ElastiCache di <https://console.aws.amazon.com/elasticache/>.
2. Dari daftar di sudut kanan atas, pilih Wilayah AWS dari klaster yang ingin dihapus simpulnya.
3. Di panel navigasi, pilih mesin yang berjalan pada klaster yang ingin dihapus simpulnya.

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. Dengan konsol ElastiCache, Anda hanya dapat menghapus satu simpul pada satu waktu. Jadi, jika Anda memilih beberapa simpul, 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.

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.

Skenario	Operasi Tertunda	Permintaan Baru	Hasil
			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	<p>Permintaan pembuatan baru, tertunda atau langsung, akan menggantikan permintaan penghapusan tertunda.</p> <p>Misalnya, jika ada penghapusan tertunda untuk simpul 0001, 0003, dan 0007 dan permintaan baru untuk membuat simpul dikirimkan, simpul baru akan dibuat dan simpul 0001, 0003, dan 0007 tidak akan dihapus.</p>
Skenario 3	Buat	Hapus	<p>Permintaan penghapusan baru, tertunda atau langsung, akan menggantikan permintaan pembuatan tertunda.</p> <p>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.</p>

Skenario	Operasi Tertunda	Permintaan Baru	Hasil
Skenario 4	Buat	Buat	<p>Permintaan pembuatan baru ditambahkan ke permintaan pembuatan yang tertunda.</p> <p>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.</p> <div style="border: 1px solid #f08080; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p>⚠ Important</p> <p>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.</p> </div>

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

1. Identifikasi ID simpul yang ingin dihapus. Untuk informasi selengkapnya, lihat [Melihat detail klaster](#).
2. Gunakan operasi CLI `modify-cache-cluster` dengan daftar simpul yang akan dihapus, seperti dalam contoh berikut.

Untuk menghapus simpul dari klaster menggunakan antarmuka baris perintah, gunakan perintah `modify-cache-cluster` dengan parameter berikut:

- `--cache-cluster-id` – ID dari klaster cache yang ingin dihapus simpulnya.
- `--num-cache-nodes` – Parameter `--num-cache-nodes` menentukan jumlah simpul yang diinginkan dalam klaster ini setelah perubahan diterapkan.

- `--cache-node-ids-to-remove` – Daftar ID simpul yang ingin dihapus dari kluster ini.
- `--apply-immediately` atau `--no-apply-immediately` – Menentukan apakah akan menghapus simpul ini dengan segera atau pada periode pemeliharaan berikutnya.
- `--region` – Menentukan Wilayah AWS dari kluster yang ingin dihapus simpulnya.

Contoh berikut segera menghapus simpul 0001 dari kluster my-cluster.

Untuk Linux, macOS, atau Unix:

```
aws elasticache modify-cache-cluster \  
  --cache-cluster-id my-cluster \  
  --num-cache-nodes 2 \  
  --cache-node-ids-to-remove 0001 \  
  --region us-east-2 \  
  --apply-immediately
```

Untuk Windows:

```
aws elasticache modify-cache-cluster ^  
  --cache-cluster-id my-cluster ^  
  --num-cache-nodes 2 ^  
  --cache-node-ids-to-remove 0001 ^  
  --region us-east-2 ^  
  --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-east-2b",  
    "ConfigurationEndpoint": {  
      "Port": 11211,  
      "Address": "rlh-mem000.7ef-example.cfg.usw2.cache.amazonaws.com"  
    }  
  }  
}
```



```

    },
    "CacheSecurityGroups": [],
    "CacheClusterCreateTime": "2016-09-21T16:28:28.973Z", 9dcv5r
    "AutoMinorVersionUpgrade": true,
    "CacheClusterStatus": "modifying",
    "NumCacheNodes": 3,
    "ClientDownloadLandingPage": "https://console.aws.amazon.com/elasticache/
home#client-download:",
    "SecurityGroups": [
      {
        "Status": "active",
        "SecurityGroupId": "sg-dbe93fa2"
      }
    ],
    "CacheSubnetGroupName": "default",
    "EngineVersion": "1.4.24",
    "PendingModifiedValues": {
      "NumCacheNodes": 2,
      "CacheNodeIdsToRemove": [
        "0001"
      ]
    }
  },
  "PreferredMaintenanceWindow": "sat:09:00-sat:10:00",
  "CacheNodeType": "cache.m3.medium",
}
}

```

Untuk informasi selengkapnya, lihat topik AWS CLI [describe-cache-cluster](#) dan [modify-cache-cluster](#).

Menggunakan API ElastiCache

Untuk menghapus simpul menggunakan API ElastiCache, panggil operasi API `ModifyCacheCluster` dengan ID kluster cache dan daftar simpul yang akan dihapus, seperti yang ditunjukkan berikut:

- `CacheClusterId` – ID dari kluster cache yang ingin dihapus simpulnya.
- `NumCacheNodes` – Parameter `NumCacheNodes` menentukan jumlah simpul yang diinginkan dalam kluster ini setelah perubahan diterapkan.
- `CacheNodeIdsToRemove.member.n` – Daftar ID simpul yang akan dihapus dari kluster.

- `CacheNodeIdsToRemove.member.1=0004`
- `CacheNodeIdsToRemove.member.1=0005`
- `ApplyImmediately` – Menentukan apakah akan menghapus simpul ini dengan segera atau pada periode pemeliharaan berikutnya.
- `Region` – Menentukan Wilayah AWS dari kluster yang ingin dihapus simpulnya.

Contoh berikut segera menghapus simpul 0004 dan 0005 dari kluster 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 API ElastiCache [ModifyCacheCluster](#).

Membatalkan operasi penambahan atau penghapusan simpul yang tertunda

Jika Anda memilih untuk tidak langsung menerapkan perubahan, operasi memiliki status tertunda hingga dilakukan pada periode pemeliharaan berikutnya. Anda dapat membatalkan setiap operasi tertunda.

Untuk membatalkan operasi yang tertunda

1. Masuk ke AWS Management Console dan buka konsol ElastiCache di <https://console.aws.amazon.com/elasticache/>.
2. Dari daftar di sudut kanan atas, pilih Wilayah AWS yang ingin Anda membatalkan operasi penambahan atau penghapusan simpul yang tertunda.
3. Di panel navigasi, pilih mesin yang berjalan di klaster yang memiliki operasi tertunda yang ingin dibatalkan. Daftar klaster yang menjalankan mesin yang dipilih akan muncul.
4. Dalam daftar klaster, pilih nama klaster, bukan kotak di sebelah kiri nama klaster, yang memiliki operasi tertunda yang ingin Anda batalkan.
5. Untuk menentukan operasi apa yang tertunda, pilih tab Deskripsi dan periksa untuk melihat berapa banyak pembuatan tertunda atau penghapusan ditampilkan. Anda tidak dapat memiliki kedua pembuatan tertunda dan penghapusan tertunda.
6. Pilih tab Simpul.
7. Untuk membatalkan semua operasi yang tertunda, klik Batalkan Penundaan. Kotak dialog Batalkan Penundaan akan muncul.
8. Konfirmasi bahwa Anda ingin membatalkan semua operasi yang tertunda dengan memilih tombol Batalkan Penundaan, atau untuk mempertahankan operasi, pilih Batalkan.

Menghapus klaster

Selama klaster dalam status tersedia, Anda akan dikenakan biaya, terlepas dari apakah Anda secara aktif menggunakannya atau tidak. Untuk menghentikan biaya, hapus klaster tersebut.

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 konsol Amazon ElastiCache di <https://console.aws.amazon.com/elasticache/>.
2. Pada dasbor konsol ElastiCache, pilih mesin yang dijalankan oleh klaster yang ingin dihapus.

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 klaster saja pada satu waktu dari konsol ElastiCache. Memilih beberapa klaster akan menonaktifkan operasi hapus.

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 di dalam daftar klaster, Anda berhenti dikenakan biaya untuk itu.

Menggunakan AWS CLI

Kode berikut menghapus klaster cache `my-cluster`.

```
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 topik AWS CLI untuk ElastiCache [delete-cache-cluster](#).

Menggunakan API ElastiCache

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 lebih lanjut, lihat topik referensi API ElastiCache [DeleteCacheCluster](#).

Mengakses klaster Anda

Instans Amazon ElastiCache Anda dirancang untuk diakses melalui instans Amazon EC2.

Jika Anda meluncurkan instans ElastiCache Anda di Amazon Virtual Private Cloud (Amazon VPC), Anda dapat mengakses instans ElastiCache Anda dari instans Amazon EC2 di Amazon VPC yang sama. Anda juga dapat menggunakan peering VPC untuk mengakses instans ElastiCache Anda dari Amazon EC2 di Amazon VPC yang berbeda.

Jika Anda meluncurkan instans ElastiCache Anda di EC2 Classic, Anda mengizinkan instans EC2 untuk mengakses klaster Anda dengan memberikan akses ke grup keamanan cache Anda kepada grup keamanan Amazon EC2 yang terkait dengan instans tersebut. Secara default, akses ke klaster dibatasi pada akun yang meluncurkan klaster tersebut.

Topik

- [Memberikan akses ke klaster Anda](#)

Memberikan akses ke klaster Anda

Anda meluncurkan klaster Anda ke EC2-VPC

Jika Anda meluncurkan klaster ke Amazon Virtual Private Cloud (Amazon VPC), Anda dapat terhubung ke klaster ElastiCache Anda hanya dari instans Amazon EC2 yang berjalan di Amazon VPC yang sama. Dalam hal ini, Anda akan perlu memberikan izin masuk jaringan kepada klaster.


Note

Pastikan Anda telah mengaktifkan Local Zones jika Anda menggunakannya. Untuk informasi lain, lihat [Mengaktifkan Local Zones](#). Dengan mengaktifkannya, VPC Anda diperluas ke Zona Lokal dan VPC Anda akan memperlakukan subnet seperti subnet apa pun di Availability Zone 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 konsol Amazon EC2 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 penggunaan ElastiCache, 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 khusus.
 - d. Di kotak Rentang port, ketik nomor port untuk simpul kluster Anda. Nomor ini harus sama dengan yang Anda tentukan saat meluncurkan kluster. Port default untuk Memcached adalah **11211**.
 - e. Di kotak Sumber, pilih Di mana saja yang memiliki rentang port (0.0.0.0/0). Dengan memilih opsi ini, setiap instans Amazon EC2 yang Anda luncurkan di Amazon VPC Anda dapat terhubung ke simpul ElastiCache Anda.

 Important

Membuka kluster ElastiCache untuk 0.0.0.0/0 tidak mengekspos kluster ke Internet karena kluster tidak memiliki alamat IP publik sehingga kluster tidak dapat diakses dari luar VPC. Namun, grup keamanan default dapat diterapkan ke instans Amazon EC2 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 buat Grup Keamanan VPC yang akan digunakan secara eksklusif oleh ElastiCache. Untuk informasi selengkapnya, lihat [Grup Keamanan Khusus](#).

- f. Pilih Simpan.

Saat Anda meluncurkan instans Amazon EC2 ke Amazon VPC Anda, instans tersebut dapat terhubung ke kluster ElastiCache Anda.

Mengakses sumber daya ElastiCache dari luar AWS

Amazon ElastiCache adalah layanan AWS yang menyediakan penyimpanan nilai-kunci dalam memori berbasis cloud. Layanan ini dirancang untuk diakses secara eksklusif dari dalam AWS. Namun, jika klaster ElastiCache di-host di dalam VPC, Anda dapat menggunakan instans Network Address Translation (NAT) untuk menyediakan akses luar.

Persyaratan

Persyaratan berikut harus dipenuhi agar Anda dapat mengakses sumber daya ElastiCache 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 di dalam VPC.

Pertimbangan

Perhatikan pertimbangan berikut saat mengakses sumber daya ElastiCache 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 proxy antara klien dan beberapa klaster. Penambahan proxy 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 mempertahankan 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 sumber daya ElastiCache dari luar AWS

Prosedur berikut menunjukkan cara terhubung ke sumber daya ElastiCache Anda menggunakan instans NAT.

Langkah-langkah ini mengasumsikan sebagai 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 kluster Memcached dengan:
 - Alamat IP – 10.0.1.230
 - Port Memcached default – 11211
 - Grup keamanan – *10\0\0\0.55*
- 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 resource ElastiCache 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 tipe simpul cache.m1.small. Anda harus memilih ukuran simpul berdasarkan kebutuhan Anda. Anda harus menggunakan EC2 NAT AMI agar dapat mengakses ElastiCache dari luar AWS.

Untuk informasi tentang cara membuat instans 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 cluster harus memiliki aturan berikut:

- Dua aturan masuk
 - Satu untuk mengizinkan koneksi TCP dari klien tepercaya ke setiap port cache yang diteruskan dari instans NAT (11211 - 11213).
 - Sedetik untuk mengizinkan akses SSH ke klien tepercaya.

Grup keamanan instans NAT - aturan masuk

Tipe	Protokol	Rentang port	Sumber
Aturan TCP Kustom	TCP	11211-11213	198.51.100.27/32
SSH	TCP	%22	198.51.100.27/32

- Aturan keluar untuk mengizinkan koneksi TCP ke port cache (11211).

Grup keamanan instans NAT - aturan keluar

Tipe	Protokol	Rentang port	Tujuan
Aturan TCP Kustom	TCP	11211	sg-ce56b7a9 (Grup Keamanan instans klaster)

- Aturan masuk untuk grup keamanan klaster yang mengizinkan koneksi TCP dari instans NAT ke port cache (11211).

Grup keamanan instans kluster - aturan masuk

Tipe	Protokol	Rentang port	Sumber
Aturan TCP Kustom	TCP	11211	sg-bd56b7da (Grup Keamanan NAT)

3. Validasi aturan.

- Konfirmasikan bahwa klien tepercaya dapat melakukan SSH ke instans NAT.
- Konfirmasikan bahwa klien tepercaya dapat terhubung ke kluster dari instans NAT.

4. Tambahkan aturan iptables ke instans NAT.

Aturan iptables harus ditambahkan ke tabel NAT untuk setiap simpul di kluster untuk meneruskan port cache dari instans NAT ke simpul kluster. 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 kluster. Misalnya, jikaq bekerja dengan kluster 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
```

5. Konfirmasikan bahwa klien tepercaya dapat terhubung ke kluster.

Klien tepercaya harus terhubung ke EIP yang terkait dengan instans NAT dan port kluster yang sesuai dengan simpul kluster 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
```

6. 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 Cache ElastiCache di Amazon VPC](#)
- [Mengakses Cache ElastiCache dari Aplikasi yang Berjalan di Pusat Data Pelanggan](#)
- [Instans NAT](#)
- [Mengonfigurasi Klien ElastiCache](#)
- [Ketersediaan Tinggi untuk Instans NAT Amazon VPC: Contoh](#)

Menemukan titik akhir koneksi

Aplikasi Anda terhubung ke klaster Anda menggunakan titik akhir. Titik akhir adalah alamat unik dari simpul atau klaster.

Titik akhir mana yang digunakan

Untuk cache nirserver ElastiCache dengan Memcached, cukup dapatkan titik akhir DNS dan port klaster dari konsol.

Dari AWS CLI, gunakan perintah `describe-serverless-caches` untuk memperoleh informasi Titik Akhir.

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 kluster Memcached berbasis instans, jika Anda menggunakan Penemuan Otomatis, maka Anda dapat menggunakan titik akhir konfigurasi kluster 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 proses baca dan tulis. Anda juga harus melacak titik akhir saat menambahkan dan menghapus simpul.

Bagian berikut memandu Anda menemukan titik akhir yang Anda perlukan untuk mesin yang sedang Anda jalankan.

Menemukan Titik Akhir Klaster (Konsol)

Semua titik akhir Memcached adalah titik akhir baca/tulis. Untuk terhubung ke simpul di 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 simpul](#).

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)

Anda dapat menggunakan AWS CLI untuk Amazon ElastiCache guna menemukan titik akhir untuk simpul dan klaster.

Topik

- [Menemukan Titik Akhir untuk Simpul dan Klaster \(AWS CLI\)](#)

Menemukan Titik Akhir untuk Simpul dan Klaster (AWS CLI)

Anda dapat menggunakan AWS CLI guna menemukan titik akhir untuk klaster dan simpulnya dengan perintah `describe-cache-clusters`. 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",
  }
]
}
```

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` di dalam file `php.ini` untuk parameter `session.save_path`.

Untuk informasi selengkapnya, lihat topik [describe-cache-clusters](#).

Menemukan Titik Akhir (API ElastiCache)

Anda dapat menggunakan API Amazon ElastiCache guna menemukan titik akhir untuk simpul dan klaster.

Topik

- [Menemukan Titik Akhir untuk Simpul dan Klaster \(API ElastiCache\)](#)

Menemukan Titik Akhir untuk Simpul dan Klaster (API ElastiCache)

Anda dapat menggunakan API ElastiCache guna menemukan titik akhir untuk klaster dan simpulnya dengan perintah `DescribeCacheClusters`. 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

Perintah berikut mengambil titik akhir konfigurasi (`ConfigurationEndpoint`) dan titik akhir simpul individual (`Endpoint`) untuk klaster Memcached `mycluster`.

```
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` di dalam file `php.ini` untuk parameter `session.save_path`.

Mengelola simpul

Simpul adalah komponen penyusun terkecil deployment Amazon ElastiCache. Simpul adalah potongan RAM berukuran tetap yang aman dan terpasang ke jaringan. Setiap simpul menjalankan mesin yang dipilih ketika kluster dibuat atau terakhir diubah. Setiap simpul mempunyai nama dan port Layanan Nama Domain (DNS) sendiri. Beberapa tipe simpul ElastiCache didukung, masing-masing dengan jumlah memori terkait dan daya komputasi yang bervariasi.

Untuk diskusi yang lebih terperinci tentang ukuran simpul yang akan digunakan, lihat [Memilih ukuran simpul Memcached Anda](#).

Topik

- [Melihat Status ElastiCache Node](#)
- [Menghubungkan ke simpul](#)
- [Tipe simpul yang didukung](#)
- [Mengganti simpul](#)
- [Simpul terpesan ElastiCache](#)
- [Memigrasikan simpul generasi sebelumnya](#)

Beberapa operasi penting yang melibatkan simpul adalah sebagai berikut:

- [Menambahkan simpul ke kluster](#)
- [Menghapus simpul dari kluster](#)
- [Penskalaan ElastiCache untuk Memcached](#)
- [Menemukan titik akhir koneksi](#)

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	Dikenakan Biaya	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: <ul style="list-style-type: none">• ElastiCache Node sedang ditambah sebagai bagian dari update 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	Dikenakan Biaya	Deskripsi
		<ul style="list-style-type: none">• ElastiCache Cluster sedang ditingkatkan atau diperkecil.• Konfigurasi pengiriman log sedang dimodifikasi untuk ElastiCache Cluster.• Operasi penghapusan untuk ElastiCache node sedang tertunda.• Kata sandi ElastiCache for Redis sedang diperbarui/ diputar menggunakan 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	Dikenakan Biaya	Deskripsi
<code>incompatible_parameters</code>	Tidak ditagih	<p>Amazon tidak ElastiCache dapat memulai node karena parameter yang ditentukan dalam grup parameter node tidak kompatibel dengan node. Kembalikan perubahan parameter atau buat parameter kompatibel dengan simpul untuk mendapatkan akses ke simpul Anda. Untuk informasi selengkapnya tentang parameter yang tidak kompatibel, periksa daftar Peristiwa untuk ElastiCache node.</p>
<code>incompatible_network</code>	Tidak ditagih	<p>Status jaringan yang tidak kompatibel menunjukkan satu atau beberapa hal berikut berlaku untuk node Amazon: ElastiCache</p> <ul style="list-style-type: none">• 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	Dikenakan Biaya	Deskripsi
restore_failed	Tidak ditagih	<p>Status gagal pemulihan menunjukkan salah satu dari berikut ini benar dari simpul Amazon: ElastiCache</p> <ul style="list-style-type: none">• 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, lihat Memigrasikan simpul generasi sebelumnya.• Snapshot RDB yang ditentukan gagal dipulihkan.• AWS Akun untuk ElastiCache cluster telah ditangguhkan.• Node gagal dan tidak dapat dipulihkan.

Jenis	Dikenakan Biaya	Deskripsi
snapshotting	Ditagih	ElastiCache sedang membuat snapshot dari node ElastiCache for Redis.

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 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.

Menghubungkan ke simpul

Sebelum mencoba untuk terhubung ke kluster Memcached, Anda harus memiliki titik akhir untuk simpul. Untuk menemukan titik akhir, lihat yang berikut ini:

- [Menemukan Titik Akhir Kluster \(Konsol\)](#)
- [Menemukan Titik Akhir \(AWS CLI\)](#)
- [Menemukan Titik Akhir \(API ElastiCache\)](#)

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. Hubungkan ke instans Amazon EC2 Anda menggunakan utilitas koneksi pilihan Anda.

Note

Untuk petunjuk tentang cara menghubungkan ke instans Amazon EC2, lihat [Panduan Memulai Amazon EC2](#).

2. Unduh dan instal utilitas telnet di instans Amazon EC2 Anda. Pada prompt perintah instans Amazon EC2 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 instans Amazon EC2 Anda, ketik perintah berikut, dengan mengganti titik akhir simpul 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 contohnya.

```
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
>
```


Tipe simpul yang didukung

Untuk informasi tentang ukuran simpul yang akan digunakan, lihat [Memilih ukuran simpul Memcached Anda](#).

ElastiCache mendukung jenis simpul berikut. Secara umum, tipe generasi saat ini memberikan lebih banyak memori dan daya komputasi dengan biaya lebih rendah dibandingkan dengan tipe generasi sebelumnya yang setara.

Untuk informasi selengkapnya tentang detail performa untuk setiap tipe simpul, lihat [Tipe Instans Amazon EC2](#).

- Tujuan umum:
- Generasi saat ini:

Tipe simpul M6g (tersedia hanya untuk mesin Memcached versi 1.5.16 ke atas).

cache.m6g.large, cache.m6g.xlarge, cache.m6g.2xlarge, cache.m6g.4xlarge, cache.m6g.8xlarge, cache.m6g.12xlarge, cache.m6g.16xlarge

Note

Untuk ketersediaan wilayah, lihat [Jenis simpul yang didukung oleh Wilayah AWS](#).

Tipe simpul M5: cache.m5.large, cache.m5.xlarge, cache.m5.2xlarge, cache.m5.4xlarge, cache.m5.12xlarge, cache.m5.24xlarge

Tipe simpul M4: cache.m4.large, cache.m4.xlarge, cache.m4.2xlarge, cache.m4.4xlarge, cache.m4.10xlarge

Tipe simpul T4g (tersedia hanya untuk mesin Memcached versi 1.5.16 ke atas).

cache.t4g.micro, cache.t4g.small, cache.t4g.medium

Tipe simpul T3: cache.t3.micro, cache.t3.small, cache.t3.medium

Tipe simpul T2: cache.t2.micro, cache.t2.small, cache.t2.medium

- Generasi sebelumnya: (tidak disarankan. Klaster yang ada masih didukung tetapi pembuatan klaster baru tidak didukung untuk tipe ini.)

Tipe simpul T1: `cache.t1.micro`

Tipe simpul M1: `cache.m1.small`, `cache.m1.medium`, `cache.m1.large`,
`cache.m1.xlarge`

Tipe simpul M3: `cache.m3.medium`, `cache.m3.large`, `cache.m3.xlarge`,
`cache.m3.2xlarge`


- Komputasi yang dioptimalkan:
 - Generasi sebelumnya: (tidak disarankan)

Tipe simpul C1: `cache.c1.xlarge`

- Memori yang dioptimalkan
 - Generasi saat ini:

(Tipe simpul R6g tersedia hanya untuk mesin Memcached versi 1.5.16 ke atas.)

Tipe simpul R6g: `cache.r6g.large`, `cache.r6g.xlarge`, `cache.r6g.2xlarge`,
`cache.r6g.4xlarge`, `cache.r6g.8xlarge`, `cache.r6g.12xlarge`,
`cache.r6g.16xlarge`

 Note

Untuk ketersediaan wilayah, lihat [Jenis simpul yang didukung oleh Wilayah AWS](#).

Tipe simpul R5: `cache.r5.large`, `cache.r5.xlarge`, `cache.r5.2xlarge`,
`cache.r5.4xlarge`, `cache.r5.12xlarge`, `cache.r5.24xlarge`

Tipe simpul R4: `cache.r4.large`, `cache.r4.xlarge`, `cache.r4.2xlarge`,
`cache.r4.4xlarge`, `cache.r4.8xlarge`, `cache.r4.16xlarge`

- Generasi sebelumnya: (tidak disarankan)

Tipe simpul M2: `cache.m2.xlarge`, `cache.m2.2xlarge`, `cache.m2.4xlarge`

Tipe simpul R3: `cache.r3.large`, `cache.r3.xlarge`, `cache.r3.2xlarge`,
`cache.r3.4xlarge`, `cache.r3.8xlarge`

- Jaringan yang dioptimalkan:

- Generasi saat ini:

(Tipe simpul C7gn tersedia hanya untuk mesin Memcached versi 1.6.6 ke atas.)

Tipe simpul C7gn: `cache.c7gn.large`, `cache.c7gn.xlarge`, `cache.c7gn.2xlarge`, `cache.c7gn.4xlarge`, `cache.c7gn.8xlarge`, `cache.c7gn.12xlarge`, `cache.c7gn.16xlarge`

Generasi Saat Ini

Tabel berikut menunjukkan bandwidth acuan dan lonjakan untuk tipe instans yang menggunakan mekanisme kredit I/O jaringan untuk melonjak di atas bandwidth acuannya.

Note

Tipe instans dengan performa jaringan yang dapat dilonjatkan menggunakan mekanisme kredit I/O jaringan untuk melampaui bandwidth dasar mereka dengan upaya terbaik.

Umum

Jenis instans	Versi Memcached minimum yang didukung	Bandwidth dasar (Gbps)	Bandwidth burst (Gbps)
<code>cache.m7g.large</code>		0,937	12,5
<code>cache.m7g.xlarge</code>		1,876	12,5
<code>cache.m7g.2xlarge</code>		3,75	15
<code>cache.m7g.4xlarge</code>		7,5	15
<code>cache.m7g.8xlarge</code>		15	T/A
<code>cache.m7g.12xlarge</code>		22,5	T/A
<code>cache.m7g.16xlarge</code>		30	T/A

Jenis instans	Versi Memcached minimum yang didukung	Bandwidth dasar (Gbps)	Bandwidth burst (Gbps)
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	T/A
cache.m6g.12xlarge	1.5.16	20	T/A
cache.m6g.16xlarge	1.5.16	25	T/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	T/A	T/A
cache.m5.24xlarge	1.5.16	T/A	T/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

Jenis instans	Versi Memcached minimum yang didukung	Bandwidth dasar (Gbps)	Bandwidth burst (Gbps)
cache.t4g.small	1.5.16	0,128	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	0,128	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	0,128	1,024
cache.t2.medium	1.5.16	0,256	1,024

Memori yang dioptimalkan

Jenis instans	Versi minimum yang didukung	Bandwidth dasar (Gbps)	Bandwidth burst (Gbps)
cache.r7g.large		0,937	12,5
cache.r7g.xlarge		1,876	12,5
cache.r7g.2xlarge		3,75	15
cache.r7g.4xlarge		7,5	15
cache.r7g.8xlarge		15	T/A
cache.r7g.12xlarge		22,5	T/A
cache.r7g.16xlarge		30	T/A
cache.r6g.large	1.5.16	0,75	10,0

Jenis instans	Versi minimum yang didukung	Bandwidth dasar (Gbps)	Bandwidth burst (Gbps)
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	T/A
cache.r6g.12xlarge	1.5.16	20	T/A
cache.r6g.16xlarge	1.5.16	25	T/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	T/A
cache.r5.24xlarge	1.5.16	25	T/A
cache.r4.large	1.5.16	0,75	10,0
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	T/A
cache.r4.16xlarge	1.5.16	25	T/A

Jaringan yang dioptimalkan

Jenis instans	Versi minimum yang didukung	Bandwidth dasar (Gbps)	Bandwidth burst (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	T/A
cache.c7gn.8xlarge	1.6.6	100	T/A
cache.c7gn.12xlarge	1.6.6	150	T/A
cache.c7gn.16xlarge	1.6.6	200	T/A

Jenis simpul yang didukung oleh Wilayah AWS

Jenis simpul yang didukung dapat bervariasi antar Wilayah AWS. Untuk detail selengkapnya, lihat [harga Amazon ElastiCache](#).

instans Performa yang Dapat Melonjak

Anda dapat meluncurkan simpul cache tujuan umum T4g, T3-Standard, dan T2-Standard yang dapat melonjak di Amazon ElastiCache. Simpul ini menyediakan tingkat performa CPU dasar dengan kemampuan untuk menangani lonjakan penggunaan CPU kapan pun hingga kredit terakumulasi habis. Kredit CPU menyediakan performa inti CPU penuh selama satu menit.

Simpul T4g, T3 dan T2 dari Amazon ElastiCache dikonfigurasi sebagai standar dan sesuai untuk beban kerja dengan penggunaan CPU rata-rata yang secara konsisten di bawah performa dasar instans. Untuk melonjak di atas batas dasar, simpul menggunakan kredit yang telah diakumulasi dalam saldo kredit CPU. Jika simpul kehabisan kredit yang dikumpulkan, performa secara bertahap diturunkan ke tingkat performa dasar. Penurunan bertahap ini memastikan simpul tidak mengalami penurunan performa yang tajam saat saldo kredit CPU yang dikumpulkannya terpakai. Untuk informasi selengkapnya, lihat [Kredit CPU dan Performa Dasar untuk Instans Performa yang Dapat Melonjak](#) di Panduan Pengguna Amazon EC2.

Tabel berikut mencantumkan jenis simpul performa yang dapat melonjak, tingkat di mana kredit CPU dihasilkan per jam. Hal ini juga menunjukkan jumlah maksimum kredit CPU yang dihasilkan yang dapat dikumpulkan oleh simpul dan jumlah vCPUs per simpul. Selain itu, tabel menunjukkan tingkat performa dasar sebagai persentase dari performa inti penuh (menggunakan satu vCPU tunggal).


Kredit CPU dihasilkan per jam	Kredit maksimum yang dihasilkan yang dapat dikumpulkan*	vCPU	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
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

Kredit CPU dihasilkan per jam	Kredit maksimum yang dihasilkan yang dapat dikumpulkan*	vCPU	Performa dasar per vCPU	Memori (GiB)	Performa jaringan
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 yang memiliki lebih dari satu vCPU. Untuk ini, hitung pemanfaatan CPU garis dasar untuk simpul dengan cara mengalikan persentase vCPU dengan jumlah vCPUs.

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.

Informasi Terkait

- [Fitur dan Detail Produk Amazon ElastiCache](#)
- [Parameter Khusus Jenis Simpul Memcached](#)

Mengganti simpul

Amazon ElastiCache for Memcached sering meningkatkan armadanya dengan patch dan peningkatan yang diterapkan di instans dengan lancar. Namun, dari waktu ke waktu kita perlu meluncurkan ulang simpul ElastiCache untuk Memcached Anda untuk menerapkan pembaruan OS yang 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 menerima terus peringatan yang menunjukkan bahwa penggantian simpul sedang terjadi. Jika Anda telah mengurangi kebutuhan pemeliharaan secara manual, Anda dapat mengabaikan peringatan ini.

Note

Penggantian simpul cache yang secara otomatis dihasilkan 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 ketika ElastiCache menjadwalkan penggantian salah satu simpul Memcached Anda.

- Jangan lakukan apa pun – Jika Anda tidak melakukan apa pun, ElastiCache mengganti simpul seperti yang terjadwal. Saat ElastiCache secara otomatis menggantikan simpul dengan simpul baru, simpul baru pada awalnya akan kosong.
- Ubah periode pemeliharaan Anda - Untuk peristiwa pemeliharaan terjadwal, Anda menerima email atau peristiwa notifikasi dari ElastiCache. Dalam hal ini, jika Anda mengubah periode pemeliharaan Anda sebelum waktu penggantian terjadwal, simpul Anda sekarang digantikan pada waktu yang baru. Untuk informasi selengkapnya, lihat [Memodifikasi sebuah cluster ElastiCache](#).

Note

Kemampuan untuk mengubah jendela penggantian Anda dengan memindahkan jendela perawatan hanya tersedia jika notifikasi ElastiCache mencakup periode pemeliharaan. Jika

notifikasi tersebut tidak menyertakan periode pemeliharaan, Anda tidak dapat mengubah periode penggantian.

Misalnya, anggaplah pemeliharaan dilakukan pada Kamis, 9 November, pukul 15.00 dan periode pemeliharaan berikutnya adalah Jumat, 10 November, pukul 17.00. Berikut adalah tiga skenario beserta hasilnya:

- Anda mengubah periode pemeliharaan Anda ke Jumat pukul 16.00, setelah tanggal dan waktu saat ini dan sebelum periode pemeliharaan terjadwal berikutnya. Simpul akan 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 akan 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](#).

- Ganti simpul secara manual – Jika Anda perlu menggantikan simpul sebelum jendela pemeliharaan berikutnya, ganti simpul secara manual.

Jika Anda secara manual menggantikan simpul, kunci akan didistribusikan kembali. Distribusi kembali ini menyebabkan cache meleset.

Untuk secara manual mengganti simpul Memcached

1. Hapus simpul yang dijadwalkan untuk penggantian. Untuk petunjuk, lihat [Menghapus simpul dari klaster](#).
2. Tambahkan simpul baru ke klaster. Untuk petunjuk, lihat [Menambahkan simpul ke klaster](#).
3. Jika Anda tidak menggunakan penemuan otomatis di klaster ini, lihat aplikasi Anda dan ganti setiap instans dari titik akhir dari simpul lama dengan titik akhir dari simpul baru.

Simpul terpesan ElastiCache

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.

Untuk melihat apakah simpul terpesan menghemat biaya kasus penggunaan Anda, pertama-tama tentukan terlebih dahulu 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 terpesan dan simpul Sesuai Permintaan dalam kluster Anda. Untuk informasi harga, lihat [Harga Amazon ElastiCache](#).

Note

Simpul terpesan bersifat tidak fleksibel; simpul tersebut hanya diterapkan untuk jenis instans yang persis Anda pesan.

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 terlebih dahulu 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 kluster Anda. Untuk informasi harga, lihat [Harga Amazon ElastiCache](#).

Wilayah AWS, tipe simpul, dan durasi jangka waktu harus dipilih saat pembelian dan tidak dapat diubah nanti.

Anda dapat menggunakan API AWS Management Console, AWS CLI, atau ElastiCache untuk membuat daftar dan membeli penawaran simpul terpesan yang tersedia.

Untuk informasi selengkapnya tentang simpul terpesan, lihat [Simpul Terpesan Amazon ElastiCache](#).

Topik

- [Penawaran simpul terpesan standar](#)
- [Penawaran simpul terpesan warisan](#)
- [Mendapatkan info tentang penawaran simpul terpesan](#)
- [Membeli simpul terpesan](#)
- [Mendapatkan info tentang simpul terpesan Anda](#)

Penawaran simpul terpesan standar

Saat membeli instans simpul terpesan (RI) di Amazon ElastiCache, Anda membeli komitmen untuk mendapatkan tarif diskon pada jenis instans simpul dan Wilayah AWS tertentu selama durasi instans simpul terpesan. Untuk menggunakan instans simpul terpesan Amazon ElastiCache, Anda perlu membuat instans simpul ElastiCache yang baru, seperti yang Anda lakukan untuk instans sesuai permintaan.

Instans simpul baru yang Anda buat harus sama persis dengan spesifikasi instans simpul terpesan. Jika spesifikasi instans simpul baru sama dengan instans simpul terpesan yang sudah ada untuk akun Anda, Anda akan ditagih dengan tarif diskon yang ditawarkan untuk instans simpul terpesan. Jika tidak, instans simpul ditagih dengan tarif sesuai permintaan. RI standar ini tersedia dari rangkaian instans R5 dan M5 ke atas.

Note

Ketiga jenis penawaran yang dibahas berikutnya tersedia dalam jangka waktu satu tahun dan tiga tahun.

Jenis Penawaran

RI Tidak Ada Di Muka menyediakan akses ke instans ElastiCache terpesan tanpa memerlukan pembayaran di muka. Instans ElastiCache terpesan Tidak Ada Di Muka Anda mengenakan tagihan dengan tarif per jam yang didiskon untuk setiap jam dalam jangka waktu yang ditentukan, terlepas dari penggunaan.

RI Di Muka Sebagian mengharuskan sebagian instans ElastiCache terpesan 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.

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 acuan 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 simpul 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 atau tidak.

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 simpul 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 pemanfaatan 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 menggunakan simpul terpesan Pemanfaatan Ringan, Anda membayar biaya satu kali yang diikuti dengan biaya pemanfaatan 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 selama 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 keseluruhan tertinggi jika Anda berencana untuk menjalankan lainnya 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 [Harga Amazon ElastiCache](#).

Mendapatkan info tentang penawaran simpul terpesan

Sebelum Anda membeli simpul terpesan, Anda bisa mendapatkan informasi tentang penawaran simpul terpesan yang tersedia.

Contoh-contoh berikut menunjukkan cara mendapatkan harga dan informasi tentang penawaran simpul terpesan yang tersedia menggunakan API AWS Management Console, AWS CLI, dan ElastiCache.

Topik

- [Mendapatkan info tentang penawaran simpul terpesan \(Konsol\)](#)
- [Mendapatkan info tentang penawaran simpul terpesan \(AWS CLI\)](#)
- [Mendapatkan info tentang penawaran simpul terpesan \(API ElastiCache\)](#)

Mendapatkan info tentang penawaran simpul terpesan (Konsol)

Untuk mendapatkan harga dan informasi lainnya tentang penawaran klaster terpesan 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 konsol ElastiCache di <https://console.aws.amazon.com/elasticache/>.
2. Di panel navigasi, pilih Simpul Terpesan.
3. Pilih Beli Simpul terpesan.
4. Untuk Mesin, pilih Memcached.
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 Batalan untuk menghindari pembelian simpul dan biaya.

Mendapatkan info tentang penawaran simpul terpesan (AWS CLI)

Untuk mendapatkan harga dan informasi lainnya tentang penawaran simpul terpesan yang tersedia, ketikkan perintah berikut pada prompt perintah:

```
aws elasticache describe-reserved-cache-nodes-offerings
```

Operasi ini menghasilkan output seperti yang berikut ini (format JSON):

```
{
  "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-offerings](#) dalam Referensi AWS CLI.

Mendapatkan info tentang penawaran simpul terpesan (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 [DescribeReservedCacheNodesOfferings](#) dalam Referensi API ElastiCache.

Membeli simpul terpesan

Contoh-contoh berikut menunjukkan cara membeli penawaran simpul terpesan menggunakan API AWS Management Console, AWS CLI, dan ElastiCache.

Important

Jika contoh di bagian ini diikuti, biaya akan dikenakan pada akun AWS yang tidak dapat Anda kembalikan.

Topik

- [Membeli simpul terpesan \(Konsol\)](#)
- [Membeli simpul terpesan AWS CLI](#)
- [Membeli simpul terpesan \(API ElastiCache\)](#)

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 simpul terpesan berdasarkan ID penawaran.

Untuk membeli simpul terpesan

1. Masuk ke AWS Management Console dan buka konsol ElastiCache di <https://console.aws.amazon.com/elasticache/>.
2. Pada daftar navigasi, pilih tautan Simpul Terpesan.
3. Pilih tombol Beli simpul terpesan.
4. Untuk Mesin, pilih Memcached.
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 kluster 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 ini menghasilkan output seperti yang 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](#) dalam Referensi AWS CLI.

Membeli simpul terpesan (API ElastiCache)

Contoh berikut menunjukkan pembelian penawaran simpul terpesan yang spesifik, 649fd0c8-cf6d-47a0-bfa6-060f8e75e95f, dengan ID kluster 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 [PurchaseReservedCacheNodesOffering](#) dalam Referensi API ElastiCache.

Mendapatkan info tentang simpul terpesan Anda

Anda bisa mendapatkan informasi tentang simpul terpesan yang telah Anda beli menggunakan AWS Management Console, AWS CLI, dan API ElastiCache.

Topik

- [Mendapatkan info tentang simpul terpesan Anda \(Konsol\)](#)
- [Mendapatkan info tentang simpul terpesan Anda \(AWS CLI\)](#)
- [Mendapatkan info tentang simpul terpesan Anda \(API ElastiCache\)](#)

Mendapatkan info tentang simpul terpesan Anda (Konsol)

Prosedur berikut menjelaskan cara menggunakan AWS Management Console untuk mendapatkan informasi tentang simpul terpesan yang Anda beli.

Untuk mendapatkan informasi tentang simpul terpesan yang dibeli

1. Masuk ke AWS Management Console dan buka konsol ElastiCache 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 simpul terpesan untuk akun AWS Anda, ketikkan 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",
```

```
"Duration": "31536000",
"ProductDescription": "memcached",
"OfferingType": "Medium Utilization",
"MaxRecords": 0
}
```

Untuk informasi selengkapnya, lihat [describe--reserved-cache-nodes](#) dalam Referensi AWS CLI.

Mendapatkan info tentang simpul terpesan Anda (API ElastiCache)

Untuk mendapatkan informasi tentang simpul terpesan untuk akun AWS Anda, panggil operasi DescribeReservedCacheNodes.

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 [DescribeReservedCacheNodes](#) dalam Referensi API ElastiCache.

Memigrasikan simpul generasi sebelumnya

Simpul generasi sebelumnya adalah jenis simpul yang sedang dihentikan secara bertahap. Jika Anda tidak memiliki klaster yang menggunakan jenis simpul generasi sebelumnya, ElastiCache tidak mendukung pembuatan klaster baru dengan jenis simpul tersebut.

Karena jumlah terbatas jenis simpul generasi sebelumnya, kami tidak dapat menjamin pengganti sukses ketika simpul menjadi tidak sehat di klaster Anda. Dalam skenario seperti tersebut, ketersediaan klaster Anda mungkin berdampak negatif.

Sebaiknya migrasikan klaster Anda ke tipe simpul baru untuk ketersediaan dan kinerja yang lebih baik. Untuk tipe simpul yang direkomendasikan untuk dimigrasikan, lihat [Jalur Peningkatan](#). Untuk daftar lengkap jenis simpul yang didukung dan jenis simpul generasi sebelumnya di ElastiCache, lihat [Tipe simpul yang didukung](#).

Memigrasikan simpul di klaster Memcached

Untuk memigrasikan ElastiCache untuk Memcached ke tipe simpul yang berbeda, Anda harus membuat klaster baru, yang selalu dimulai dengan kosong sehingga aplikasi Anda dapat terisi.

Untuk memigrasikan ElastiCache Anda untuk jenis simpul klaster Memcached menggunakan Konsol ElastiCache:

- Buat klaster baru dengan tipe 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 Klaster \(Konsol\)](#)
- Hapus klaster lama. Untuk informasi selengkapnya, lihat [Menghapus klaster](#)

Bekerja dengan ElastiCache

Di bagian ini Anda dapat menemukan detail tentang cara mengelola berbagai komponen ElastiCache implementasi Anda.

Topik

- [Versi mesin dan peningkatannya](#)
- [Praktik terbaik ElastiCache dan strategi caching](#)
- [Mengelola kluster yang dirancang sendiri](#)
- [Penskalaan ElastiCache untuk Memcached](#)
- [Menandai sumber daya ElastiCache Anda](#)
- [Menggunakan Lensa Amazon ElastiCache Well-Architected](#)
- [Langkah-langkah pemecahan masalah umum dan praktik terbaik](#)
- [Langkah pemecahan masalah tambahan](#)

Versi mesin dan peningkatannya

Bagian ini mencakup versi mesin Memcached yang didukung dan cara untuk meningkatkannya.

Topik

- [Versi ElastiCache for Memcached yang Didukung](#)
- [Versi mesin dan pemutakhiran](#)
- [Cara meningkatkan versi mesin](#)

Versi ElastiCache for Memcached yang Didukung

ElastiCache mendukung versi Memcached berikut dan melakukan peningkatan ke versi yang lebih baru. Saat meningkatkan ke versi yang lebih baru, perhatikan kondisi yang jika tidak terpenuhi dapat menyebabkan peningkatan Anda gagal.

Versi ElastiCache for Memcached

- [Memcached versi 1.6.22](#)
- [Memcached versi 1.6.17](#)
- [Memcached versi 1.6.12](#)
- [Memcached versi 1.6.6](#)
- [Memcached versi 1.5.16](#)
- [Memcached versi 1.5.10](#)
- [Memcached versi 1.4.34](#)
- [Memcached versi 1.4.33](#)
- [Memcached versi 1.4.24](#)
- [Memcached versi 1.4.14](#)
- [Memcached versi 1.4.5](#)

Memcached versi 1.6.22

ElastiCache for 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 selengkapnya, lihat [ReleaseNotes1622](#) pada Memcached di GitHub.

Memcached versi 1.6.17

ElastiCache for Memcached menambahkan dukungan untuk Memcached versi 1.6.17. Ini tidak menyertakan fitur baru, tetapi mencakup perbaikan bug dan pembaruan kumulatif dari [Memcached 1.6.17](#).

Untuk informasi selengkapnya, lihat [CatatanRilis1617](#) pada Memcached di GitHub.

Memcached versi 1.6.12

ElastiCache for Memcached menambahkan dukungan untuk Memcached versi 1.6.12 dan enkripsi in-transit. Ini juga mencakup perbaikan bug dan pembaruan kumulatif dari [Memcached 1.6.6](#).

Untuk informasi selengkapnya, lihat [CatatanRilis1612](#) pada Memcached di GitHub.

Memcached versi 1.6.6

ElastiCache for Memcached menambahkan dukungan untuk Memcached versi 1.6.6. Ini tidak menyertakan fitur baru, tetapi mencakup perbaikan bug dan pembaruan kumulatif dari [Memcached 1.5.16](#). ElastiCache for Memcached tidak menyertakan dukungan untuk [Extstore](#).

Untuk informasi selengkapnya, lihat [CatatanRilis166](#) pada Memcached di GitHub.

Memcached versi 1.5.16

ElastiCache for Memcached menambahkan dukungan untuk Memcached versi 1.5.16. Ini tidak menyertakan fitur baru, tetapi mencakup perbaikan bug dan pembaruan kumulatif dari [Memcached 1.5.14](#) dan [Memcached 1.5.15](#).

Untuk informasi selengkapnya, lihat [Catatan Rilis Memcached 1.5.16](#) pada Memcached di GitHub.

Memcached versi 1.5.10

ElastiCache for Memcached versi 1.5.10 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 ditambahkan sejak ElastiCache for Memcached versi 1.4.34 meliputi hal berikut:

- Perbaikan kumulatif, seperti multiget 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.
- Kinerja yang lebih baik dan perbaikan overhead memori dengan mengurangi persyaratan memori per-item sebanyak beberapa byte.

Untuk informasi selengkapnya, lihat [Catatan Rilis Memcached 1.5.10](#) pada Memcached di GitHub.

Memcached versi 1.4.34

ElastiCache for Memcached versi 1.4.34 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 selengkapnya, lihat [Catatan Rilis Memcached 1.4.34](#) pada Memcached di GitHub.

Memcached versi 1.4.33

Perbaikan Memcached ditambahkan sejak versi 1.4.24 meliputi hal berikut:

- Kemampuan untuk membuang semua metadata untuk kelas slab tertentu, daftar kelas slab, atau semua kelas slab. Untuk informasi selengkapnya, lihat [Catatan rilis Memcached 1.4.31](#).
- Peningkatan dukungan untuk item besar melebihi default 1 megabyte. Untuk informasi selengkapnya, lihat [Catatan rilis Memcached 1.4.29](#).
- Kemampuan untuk menentukan berapa lama klien dapat diam sebelum diminta untuk menutup.

Kemampuan untuk secara dinamis meningkatkan jumlah memori yang tersedia untuk Memcached tanpa harus memulai ulang klaster. Untuk informasi selengkapnya, lihat [Catatan rilis Memcached 1.4.27](#).

- Pembuatan log dari fetchers, mutations, dan evictions didukung saat ini. Untuk informasi selengkapnya, lihat [Catatan rilis Memcached 1.4.26](#).
- Memori yang sudah dibebaskan dapat diklaim kembali ke dalam kolam global dan ditempatkan ulang ke kelas slab baru. Untuk informasi selengkapnya, lihat [Catatan rilis Memcached 1.4.25](#).
- Beberapa perbaikan bug.
- Beberapa perintah dan parameter baru. Untuk daftarnya, lihat [Parameter yang ditambahkan dalam Memcached 1.4.33](#).

Memcached versi 1.4.24

Perbaikan Memcached ditambahkan sejak versi 1.4.14 meliputi hal berikut:

- Pengelolaan yang baru-baru ini paling sedikit digunakan (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 dalam Memcached 1.4.24](#).
- Beberapa perbaikan bug.

Memcached versi 1.4.14

Perbaikan Memcached ditambahkan sejak versi 1.4.5 meliputi hal berikut:

- Kemampuan penyeimbangan kembali slab yang ditingkatkan.
- Peningkatan kinerja dan skalabilitas.
- Memperkenalkan perintah touch untuk memperbarui waktu kedaluwarsa dari item yang telah 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.

Memcached versi 1.4.5

Memcached versi 1.4.5 adalah mesin awal dan versi yang didukung oleh Amazon ElastiCache for Memcached.

Versi mesin dan pemutakhiran

Versi UTAMA adalah untuk perubahan API yang tidak kompatibel dan versi MINOR untuk fungsionalitas baru yang ditambahkan dengan cara yang kompatibel dengan versi lama. Versi PATCH adalah untuk perbaikan bug yang kompatibel dengan versi lama dan perubahan non-fungsional.

Manajemen versi untuk ElastiCache Serverless

ElastiCache Serverless 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 mengirimkan notifikasi kepada Anda di konsol dan peristiwa di EventBridge. Anda dapat memilih untuk meningkatkan cache Anda ke versi utama terbaru dengan memodifikasi cache menggunakan Konsol, CLI, atau API, dan memilih versi mesin terbaru.

Manajemen versi untuk kluster ElastiCache yang dirancang sendiri

Saat bekerja dengan kluster ElastiCache yang dirancang sendiri, Anda dapat mengontrol waktu peningkatan perangkat lunak yang menjalankan kluster cache Anda ke versi baru yang didukung oleh ElastiCache. Anda dapat mengontrol kapan harus meningkatkan cache Anda ke versi UTAMA, MINOR, dan PATCH terbaru yang tersedia. Anda dapat memulai peningkatan versi mesin pada grup kluster atau replikasi Anda dengan mengubahnya dan menentukan versi mesin baru.

Anda dapat mengontrol syarat dan waktu perangkat lunak yang mematuhi protokol dan menjalankan kluster cache Anda dimutakhirkan 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.

Untuk melakukan peningkatan ke versi Memcached yang lebih baru, ubah kluster cache Anda dengan menentukan versi mesin baru yang ingin digunakan. Pemutakhiran 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 [Mengubah kluster](#).

Anda perlu mengetahui persyaratan berikut saat melakukan peningkatan dari versi Memcached yang lebih lama 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](#).

Pertimbangan peningkatan saat bekerja dengan klaster yang dirancang sendiri

Note

Pertimbangan berikut hanya berlaku saat meningkatkan klaster yang dirancang sendiri. Pertimbangan ini tidak berlaku untuk ElastiCache Serverless.

Saat meningkatkan klaster yang dirancang sendiri, pertimbangkan hal berikut

- Pengelolaan versi mesin dirancang agar Anda dapat memiliki kontrol sebanyak mungkin terkait cara melakukan patching. Namun, ElastiCache berhak untuk melakukan patching pada klaster atas nama Anda jika terjadi kerentanan keamanan yang penting dalam perangkat lunak cache atau sistem.
- Karena mesin Memcached tidak mendukung persistensi, peningkatan versi mesin Memcached merupakan proses disruptif yang menghilangkan semua data cache di klaster.

Cara memutakhirkan versi mesin

Untuk memulai peningkatan versi pada klaster, Anda perlu mengubahnya dan menentukan versi mesin yang lebih baru. Anda dapat melakukannya dengan menggunakan konsol ElastiCache, AWS CLI, atau API ElastiCache:

- Untuk menggunakan AWS Management Console, lihat – [Mengubah klaster melalui konsol](#).

- Untuk menggunakan AWS CLI, lihat [Mengubah klaster dengan CLI](#).
- Untuk menggunakan API ElastiCache, lihat [Mengubah klaster melalui API](#).

Cara meningkatkan versi mesin

Untuk memulai peningkatan versi pada klaster, Anda perlu mengubahnya dan menentukan versi mesin yang lebih baru. Anda dapat melakukannya menggunakan konsol ElastiCache, AWS CLI, atau API ElastiCache:

- Untuk menggunakan AWS Management Console, lihat - [Menggunakan AWS Management Console](#).
- Untuk menggunakan AWS CLI, lihat [Menggunakan AWS CLI](#).
- Untuk menggunakan API ElastiCache, lihat [Menggunakan ElastiCache API](#).

Praktik terbaik ElastiCache 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 dengan klien Memcached](#)
- [Perintah Memcached yang didukung](#)
- [Strategi Pembuatan Cache](#)

Praktik terbaik dengan klien Memcached

Untuk mempelajari lebih lanjut praktik terbaik cara menangani sumber daya ElastiCache dengan pustaka klien Memcached sumber terbuka yang umum digunakan, lihat topik di bawah ini.

Topik

- [Buat konfigurasi klien ElastiCache Anda untuk penyeimbangan beban yang efisien](#)
- [Contoh klien IPv6](#)

Buat konfigurasi klien ElastiCache Anda untuk penyeimbangan beban yang efisien

Note

Bagian ini berlaku untuk klaster Memcached multisimpul yang dirancang sendiri.

Untuk menggunakan beberapa simpul ElastiCache Memcached secara efektif, Anda harus dapat menyebarkan kunci cache Anda ke seluruh simpul. Cara mudah untuk menyeimbangkan beban kluster dengan simpul n adalah dengan menghitung hash dari kunci objek dan hitung modulus dari hasilnya dengan $n - \text{hash}(\text{key}) \bmod n$. Nilai yang dihasilkan (0 sampai $n-1$) adalah jumlah simpul tempat Anda menempatkan objek.

Pendekatan ini sederhana dan bekerja dengan baik selama jumlah simpul (n) adalah konstan. Namun, setiap kali Anda menambahkan atau menghapus simpul dari kluster, 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 sejumlah besar cache awal meleset, terutama seiring bertambahnya jumlah simpul. Dalam kondisi terbaik, penskalaan dari 1 hingga 2 simpul menghasilkan pemindahan kunci $(2-1) / 2$ (50 persen). Penskalaan dari 9 hingga 10 simpul menghasilkan pemindahan kunci $(10-1) / 10$ (90 persen). Jika Anda menaikkan skala karena lonjakan lalu lintas, Anda tidak ingin banyak cache meleset. Banyaknya jumlah cache yang meleset mengakibatkan pencarian ke basis data, yang sudah kelebihan beban karena lonjakan lalu lintas.

Solusi untuk dilema ini adalah melakukan hashing secara konsisten. Hashing konsisten menggunakan algoritme sehingga setiap kali simpul ditambahkan atau dihapus dari kluster, jumlah kunci yang harus dipindahkan adalah kira-kira $1 / n$ (dengan n adalah jumlah baru simpul). Dalam kondisi terburuk, penskalaan dari 1 hingga 2 simpul menghasilkan pemindahan kunci $1/2$ (50 persen). Penskalaan dari 9 hingga 10 simpul menghasilkan pemindahan kunci $1/10$ (10 persen).

Sebagai pengguna, Anda mengontrol jenis algoritma hashing yang digunakan untuk kluster multisimpul. Sebaiknya buat konfigurasi pada 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 yang konsisten dan cara menerapkannya.

Jika Anda bekerja dengan Java, PHP, atau .NET, sebaiknya Anda menggunakan salah satu dari pustaka klien Amazon ElastiCache.

Hashing konsisten Menggunakan Java

Klien Java ElastiCache Memcached didasarkan pada klien Java `spymemcached` sumber terbuka, yang memiliki kemampuan hashing konsisten bawaan. Pustaka meliputi kelas `KetamaConnectionFactory` yang mengimplementasikan hashing konsisten. Secara default, hashing konsisten dinonaktifkan pada `spymemcached`.

Untuk informasi selengkapnya, lihat dokumentasi `KetamaConnectionFactory` di [KetamaConnectionFactory](#).

Hashing konsisten menggunakan PHP

Klien PHP Memcached ElastiCache adalah pembungkus di sekitar pustaka 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 waktu aktif untuk PHP Memcached di <http://php.net/manual/en/memcached.configuration.php>. Perhatikan secara khusus parameter `memcached.sess_consistent_hash`.

Hashing konsisten menggunakan .NET

Klien .NET ElastiCache Memcached adalah pembungkus di sekitar Enyim Memcached. Secara default, hashing konsisten diaktifkan oleh klien Enyim Memcached.

Untuk informasi selengkapnya, lihat dokumentasi `memcached/locator` di <https://github.com/enyim/EnyimMemcached/wiki/MemcachedClient-Configuration#user-content-memcachedlocator>.

Contoh klien IPv6

Note

Bagian ini berlaku untuk klaster Memcached yang dirancang sendiri.

ElastiCache kompatibel dengan Memcached sumber terbuka. Ini berarti bahwa klien open source untuk Memcached yang mendukung koneksi IPv6 harus dapat terhubung ke IPv6 diaktifkan untuk cluster Memcached. ElastiCache Selain itu, klien berikut secara khusus telah divalidasi untuk berfungsi dengan semua konfigurasi jenis jaringan yang didukung:

Berikut ini adalah praktik terbaik untuk berinteraksi dengan sumber daya berkemampuan IPv6 dengan pustaka ElastiCache klien sumber terbuka yang umum digunakan. Anda dapat melihat [praktik terbaik yang ada untuk berinteraksi dengan ElastiCache](#) rekomendasi tentang mengonfigurasi

klien untuk ElastiCache sumber daya. Namun, ada beberapa peringatan yang perlu diperhatikan saat berinteraksi dengan sumber daya dengan IPv6 aktif.

Klien yang divalidasi

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

Mengonfigurasi protokol pilihan untuk kluster tumpukan ganda

Untuk kluster Memcached, Anda dapat mengontrol protokol yang akan digunakan klien untuk terhubung ke simpul di kluster dengan parameter Penemuan IP. Parameter Penemuan IP dapat diatur ke 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 kluster Memcached.

Perubahan pada Penemuan IP tidak akan mengakibatkan waktu henti untuk klien yang terhubung. Namun, perubahan ini akan memakan waktu untuk disebarakan.

Pantau output `getAvailableNodeEndpoints` untuk Java, sementara untuk Php, pantau output dari `getServerList`. Setelah output dari fungsi-fungsi ini melaporkan IP untuk semua simpul di kluster yang menggunakan protokol yang diperbarui, berarti perubahan telah selesai disebarakan.

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.toList());

    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 kluster menggunakan protokol IP baru setelah perubahan terdeteksi dalam output perintah penemuan kluster. Namun, hal ini tergantung pada implementasi klien.

TLS mengaktifkan cluster tumpukan ElastiCache ganda

Saat TLS diaktifkan untuk ElastiCache cluster, fungsi penemuan kluster mengembalikan nama host alih-alih IP. Nama host kemudian digunakan sebagai pengganti IP untuk terhubung ke ElastiCache cluster dan melakukan jabat tangan TLS. Hal ini berarti bahwa klien tidak akan terpengaruh oleh

parameter Penemuan IP. Untuk kluster 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 IPv4 dan IPv6, Java secara default akan lebih memilih IPv4 daripada IPv6 untuk kompatibilitas mundur. Namun, preferensi protokol IP dapat dikonfigurasi melalui argumen JVM. Untuk memilih IPv4, JVM menerima `-Djava.net.preferIPv4Stack=true` dan untuk memilih IPv6, JVM mengatur `-Djava.net.preferIPv6Stack=true`. Pengaturan `-Djava.net.preferIPv4Stack=true` berarti bahwa JVM tidak akan lagi membuat koneksi IPv6.

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 lebih memilih IPv6 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 memodifikasi 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 memilih IPv4 saat menghubungkan ke ElastiCache kluster, perbarui prioritas untuk rentang CIDR yang mencakup IP cluster berada di atas prioritas untuk koneksi IPv6 default. Secara default, koneksi IPv6 memiliki prioritas 40. Misalnya, anggaplah kluster berada di subnet dengan CIDR `172.31.0.0/16`, konfigurasi di bawah ini akan menyebabkan klien lebih memilih koneksi IPv4 ke kluster tersebut.

```
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
```

Detail selengkapnya tentang `gai.conf` tersedia di [Halaman utama 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 memodifikasi file `gai.conf` pada host Linux.

Hal ini akan memperbarui kebijakan preferensi untuk memilih koneksi IPv4 daripada koneksi IPv6 untuk rentang CIDR yang ditentukan. Misalnya, dengan asumsi bahwa kluster berada dalam subnet dengan CIDR `172.31.0.0/16`, eksekusi `netsh interface ipv6 set prefix ::ffff:172.31.0.0:0/112 100 15` akan menghasilkan tabel prioritas berikut yang akan menyebabkan klien lebih memilih IPv4 saat terhubung ke kluster.

```
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
```

Perintah Memcached yang didukung

ElastiCache Serverless for Memcached mendukung semua [perintah](#) memcached di memcached 1.6 sumber terbuka kecuali untuk yang berikut ini:

- Koneksi klien memerlukan TLS, akibatnya protokol UDP tidak didukung.
- Protokol biner tidak didukung, karena [tidak lagi digunakan](#) secara resmi 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`- modifikasi untuk pengaturan crawler LRU dan LRU
 - `watch`- menonton log server memcached
 - `verbosity`- mengkonfigurasi tingkat log server

Strategi Pembuatan Cache

Pada topik berikut ini, Anda dapat menemukan strategi untuk mengisi dan memelihara cache Anda.

Strategi yang akan diterapkan untuk mengisi dan memelihara cache Anda akan tergantung pada data apa yang dibuat cache dan pola akses ke data itu. Misalnya, Anda kemungkinan tidak ingin menggunakan strategi yang sama untuk papan peringkat 10 teratas di situs game dan berita yang sedang tren. Di bagian yang tersisa dari bagian ini, kita akan membahas strategi pemeliharaan cache yang umum dengan kelebihan dan kekurangannya.

Topik

- [Pemuatan malas](#)
- [Write-through](#)
- [Menambahkan TTL](#)
- [Topik terkait](#)

Pemuatan malas

Seperti namanya, pemuatan malas 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 (basis data) yang diaksesnya. Setiap kali aplikasi Anda meminta data, aplikasi pertama-tama akan membuat permintaan ke cache ElastiCache. 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 kena terjadi ketika data ada di dalam cache dan tidak kedaluwarsa:

1. Aplikasi Anda meminta data dari cache.
2. Cache mengembalikan data ke aplikasi.

Cache meleset terjadi ketika data tidak berada di dalam cache dan kedaluwarsa:

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 pemuatan malas

Kelebihan dari pemuatan malas adalah seperti berikut:

- Hanya data yang diminta yang dibuat cache.

Karena sebagian besar data tidak pernah diminta, pemuatan malas menghindari membuat cache terisi penuh oleh 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 terjadi cache meleset mengakibatkan kueri ke basis data. Pada saat yang sama, salinan data ditambahkan ke cache sehingga permintaan berikutnya diambil dari cache.

Kelebihan dari pemuatan malas adalah seperti berikut:

- Terjadi kerugian akibat cache meleset. Setiap cache meleset menyebabkan tiga perjalanan:
 1. Permintaan awal untuk data dari cache
 2. Kueri basis data untuk data
 3. Menulis data ke cache

Kejadian meleset ini dapat menyebabkan penundaan yang terasa pada data yang masuk ke aplikasi.

- Data yang usang.

Jika data ditulis ke cache hanya bila ada cache meleset, maka data di dalam cache dapat menjadi usang. Hal ini terjadi karena ada tidak ada pembaruan pada cache ketika data berubah di dalam basis data. Untuk mengatasi masalah ini, Anda dapat menggunakan strategi [Write-through](#) dan [Menambahkan TTL](#).

Contoh kode semu pemuatan malas

Berikut adalah contoh kode semu logika pemuatan malas.

```
// *****  
// 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, maka data dalam cache selalu yang terbaru.

- Kerugian tulis vs kerugian baca.

Setiap proses tulis menyangkut dua perjalanan:

1. Tulis ke cache
2. 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 [pemuatan malas](#) dengan write-through.

- Churn Cache.

Sebagian besar data tidak pernah dibaca, yang merupakan pemborosan sumber daya. Dengan [menambahkan nilai waktu untuk tayang \(TTL\)](#), 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

Pemuatan malas 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 waktu untuk tayang (TTL) untuk setiap proses tulis, Anda dapat memiliki kelebihan dari setiap strategi. Pada saat yang sama, Anda dapat dan secara garis besar berhasil menghindari cache menjadi penuh dengan data tambahan.

Waktu untuk tayang (TTL) adalah nilai integer yang menentukan jumlah detik hingga kunci kedaluwarsa. Memcached menentukan nilai ini dalam detik. Ketika sebuah aplikasi mencoba untuk membaca kunci yang kedaluwarsa, maka perlakuannya adalah seolah-olah kunci itu tidak ditemukan. Terjadi kueri ke basis data untuk meminta kunci dan cache yang diperbarui. Pendekatan ini tidak menjamin sebuah nilai tidak menjadi usang. Namun, itu menjaga data agar tidak terlalu usang dan mensyaratkan nilai di dalam cache agar kadang-kadang disegarkan kembali dari basis data.

Untuk informasi selengkapnya, lihat [perintah set Memcached](#).

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 pemuatan malas 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 untuk Memcached](#)

Mengelola klaster yang dirancang sendiri

Bagian-bagian ini berisi topik-topik yang membantu Anda mengelola klaster yang dirancang sendiri.

Note

Topik-topik ini tidak berlaku untuk ElastiCache Nirserver.

Topik

- [Mengelola pemeliharaan](#)
- [Mengonfigurasi parameter mesin menggunakan grup parameter](#)

Mengelola pemeliharaan

Setiap klaster memiliki periode pemeliharaan mingguan yang perubahan sistemnya diterapkan. Jika Anda tidak menentukan jendela pemeliharaan yang diinginkan saat Anda membuat atau mengubah klaster, maka ElastiCache menetapkan jendela pemeliharaan 60 menit pada hari yang dipilih secara acak dalam seminggu.

Periode pemeliharaan 60 menit dipilih secara acak dari blok 8 jam waktu per wilayah. Tabel berikut mencantumkan blok waktu untuk tiap wilayah dari jendela waktu default yang ditetapkan. Anda dapat memilih jendela perawatan yang disukai di luar blok jendela pemeliharaan wilayah.

Kode Wilayah	Nama wilayah	Jendela 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

Kode Wilayah	Nama wilayah	Jendela Pemeliharaan Wilayah
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 Pacific (Hong Kong)	13.00–21.00 UTC
ap-southeast-2	Wilayah Asia Pasifik (Sydney)	12.00–20.00 UTC
eu-west-3	Wilayah EU (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
sa-east-1	Wilayah Amerika Selatan (Sao 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	Wilayah AWS GovCloud (US)	06.00–14.00 UTC
us-west-1	Wilayah AS Barat (California Utara)	06.00–14.00 UTC
us-west-2	Wilayah AS Barat (Oregon)	06.00–14.00 UTC

Mengubah Jendela Pemeliharaan Klaster Anda

Periode pemeliharaan harus berada dalam waktu penggunaan terendah, sehingga kemungkinan memerlukan modifikasi dari waktu ke waktu. Anda dapat mengubah klaster Anda untuk menentukan rentang waktu hingga durasi 24 jam yang selama waktu tersebut kegiatan pemeliharaan yang Anda minta akan dilakukan. Setiap perubahan klaster modifikasi yang Anda minta yang ditangguhkan atau ditunda terjadi dalam kurun waktu ini.

Note

Jika Anda ingin segera menerapkan modifikasi jenis simpul dan/atau peningkatan mesin menggunakan AWS Management Console pilih kotak Terapkan Segera. Jika tidak, modifikasi ini akan diterapkan selama masa pemeliharaan terjadwal berikutnya. Untuk menggunakan API, lihat [modify-replication-group](#) atau [modify-cache-cluster](#).

Informasi selengkapnya

Untuk informasi tentang jendela pemeliharaan dan penggantian simpul, lihat yang berikut ini:

- [Pemeliharaan ElastiCache](#)—FAQ tentang pemeliharaan dan penggantian simpul
- [Mengganti simpul](#)—Mengelola penggantian simpul
- [Memodifikasi sebuah cluster ElastiCache](#)—Mengubah jendela pemeliharaan klaster

Mengonfigurasi parameter mesin menggunakan grup parameter

Amazon ElastiCache menggunakan parameter untuk mengontrol properti runtime simpul dan klaster Anda. Secara umum, versi mesin yang lebih baru mencakup parameter tambahan untuk mendukung fungsionalitas yang lebih baru. Untuk mengetahui tabel parameter, lihat [Parameter spesifik Memcached](#).

Seperti yang Anda harapkan, beberapa nilai parameter, seperti `maxmemory`, ditentukan oleh mesin dan jenis simpul. Untuk mengetahui tabel nilai parameter ini berdasarkan jenis simpul, lihat [Parameter khusus jenis simpul Memcached](#).

Note

Untuk mengetahui daftar parameter khusus Memcached, lihat [Parameter Khusus Memcached](#).

Topik

- [Manajemen parameter](#)
- [Tingkatan grup parameter cache](#)
- [Membuat grup parameter](#)
- [Menampilkan daftar grup parameter berdasarkan nama](#)
- [Menampilkan daftar nilai grup parameter](#)
- [Mengubah grup parameter](#)
- [Menghapus grup parameter](#)
- [Parameter spesifik Memcached](#)

Manajemen parameter

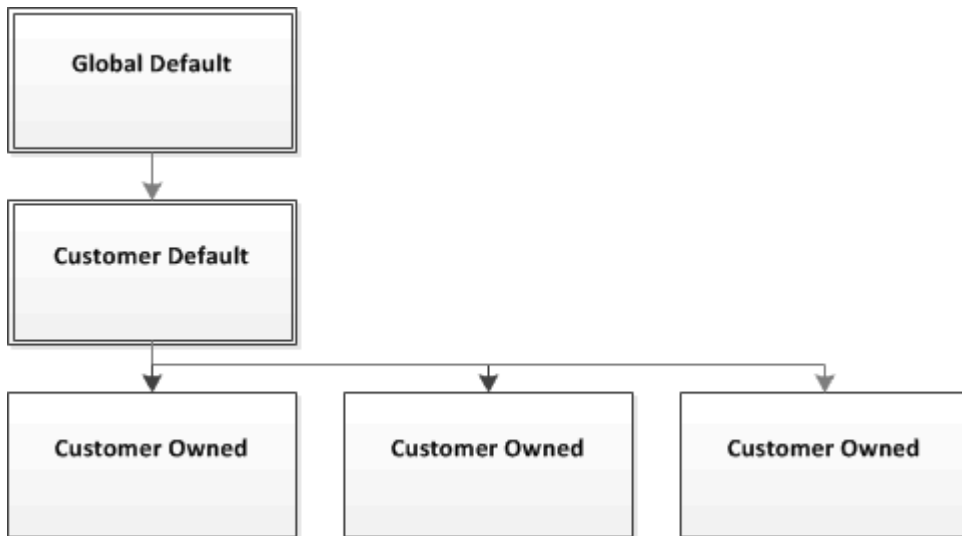
Parameter dikelompokkan bersama ke grup parameter bernama untuk manajemen parameter yang lebih mudah. Grup parameter mewakili kombinasi nilai tertentu untuk parameter yang diteruskan ke perangkat lunak mesin selama penyalaan mesin. Nilai ini menentukan cara mesin berproses pada setiap simpul berperilaku pada saat runtime. Nilai parameter pada grup parameter tertentu berlaku untuk semua simpul yang terkait dengan grup tersebut, terlepas dari asal kluster tersebut.

Untuk menyempurnakan kinerja kluster, Anda dapat mengubah beberapa nilai parameter atau mengubah grup parameter kluster.

- Anda tidak dapat mengubah atau menghapus grup parameter default. Jika membutuhkan nilai parameter khusus, Anda harus membuat grup parameter khusus.
- Keluarga grup parameter dan kluster yang menjadi destinasi penetapan harus kompatibel. Misalnya, jika kluster menjalankan Memcached versi 1.4.8, Anda hanya dapat menggunakan grup parameter, default atau khusus, dari keluarga Memcached 1.4.
- Jika Anda mengubah grup parameter kluster, nilai untuk setiap parameter yang dapat diubah bersyarat harus sama di kedua grup parameter baru dan saat ini.
- Saat Anda mengubah parameter kluster, perubahan tersebut akan segera diterapkan pada kluster tersebut. Ini berlaku terlepas dari apakah Anda mengubah grup parameter kluster itu sendiri atau nilai parameter di dalam grup parameter kluster. Untuk menentukan waktu penerapan perubahan parameter tertentu, lihat kolom Perubahan Berlaku dalam tabel untuk [Parameter spesifik Memcached](#). Untuk informasi tentang boot ulang simpul kluster, lihat [Boot ulang kluster](#).

Tingkatan grup parameter cache

Amazon ElastiCache memiliki tiga tingkatan grup parameter cache seperti yang ditunjukkan berikut.



Tingkatan grup parameter Amazon ElastiCache

Default Global

Grup parameter root tingkat atas untuk semua pelanggan Amazon ElastiCache di wilayah terkait.

Grup parameter cache default global:

- Disimpan 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 kluster pelanggan yang kompatibel.
- Dapat diubah oleh pelanggan untuk membuat grup parameter cache khusus.

Tidak semua nilai parameter dapat diubah. Untuk informasi selengkapnya, lihat [Parameter spesifik Memcached](#).

Membuat grup 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 konsol ElastiCache, AWS CLI, atau API ElastiCache.

Membuat grup parameter (Konsol)

Prosedur berikut menunjukkan cara membuat grup parameter menggunakan konsol ElastiCache.

Untuk membuat grup parameter menggunakan konsol ElastiCache

1. Masuk ke AWS Management Console dan buka konsol ElastiCache 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. 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.

Rangkaian grup parameter, misalnya memcached1.4 , menentukan parameter aktual dalam grup parameter Anda dan nilai awalnya. Rangkaian grup parameter harus sesuai dengan mesin dan versi kluster.

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 rangkaian 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.
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 rangkaian. Untuk mengubah nilai default, Anda harus mengubah grup parameter. Untuk informasi selengkapnya, lihat [Mengubah grup parameter](#).

Membuat grup 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` – Rangkaian mesin dan versi untuk grup parameter.
 - `--description` – Deskripsi yang diberikan pengguna untuk grup parameter.

Example

Contoh berikut membuat grup parameter bernama myMem14 menggunakan rangkaian 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"  
  }  
}
```

Ketika grup parameter dibuat, grup parameter ini akan memiliki nilai default rangkaian. Untuk mengubah nilai default, Anda harus mengubah grup parameter. Untuk informasi selengkapnya, lihat [Mengubah grup parameter](#).

Untuk informasi selengkapnya, lihat [create-cache-parameter-group](#).

Membuat grup parameter (API ElastiCache)

Untuk membuat grup parameter menggunakan API ElastiCache, gunakan tindakan `CreateCacheParameterGroup` 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` – Rangkaian mesin dan versi untuk grup parameter. Misalnya, `memcached1.4`.
- `Description` – Deskripsi yang diberikan pengguna untuk grup parameter.

Example

Contoh berikut membuat grup parameter bernama `myMem14` menggunakan rangkaian `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>
```

Ketika grup parameter dibuat, grup parameter ini akan memiliki nilai default rangkaian. Untuk mengubah nilai default, Anda harus mengubah grup parameter. Untuk informasi selengkapnya, lihat [Mengubah grup parameter](#).

Untuk informasi selengkapnya, lihat [CreateCacheParameterGroup](#).

Menampilkan daftar grup parameter berdasarkan nama

Anda dapat menampilkan daftar grup parameter menggunakan konsol ElastiCache, AWS CLI, atau API ElastiCache.

Menampilkan daftar grup parameter berdasarkan nama (Konsol)

Prosedur berikut menunjukkan cara melihat daftar grup parameter menggunakan konsol ElastiCache.

Untuk menampilkan daftar grup parameter menggunakan konsol ElastiCache

1. Masuk ke AWS Management Console dan buka konsol ElastiCache di <https://console.aws.amazon.com/elasticache/>.
2. Untuk melihat daftar semua grup parameter yang tersedia, pilih Grup Parameter di panel navigasi sebelah kiri.

Menampilkan daftar grup parameter berdasarkan nama (AWS CLI)

Untuk menghasilkan daftar grup 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 ini, nama, rangkaian, 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, dengan menampilkan daftar nama, rangkaian, dan deskripsi untuk grup parameter.

```
{
  "CacheParameterGroups": [
    {
      "CacheParameterGroupName": "myMem14",
      "CacheParameterGroupFamily": "memcached1.4",
      "Description": "My first parameter group"
    }
  ]
}
```

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, dengan menampilkan daftar nama, rangkaian, deskripsi dan, pada redis5.6, menunjukkan apakah grup parameter merupakan 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](#).

Menampilkan daftar grup parameter berdasarkan nama (API ElastiCache)

Untuk menghasilkan daftar grup parameter menggunakan API ElastiCache, gunakan tindakan `DescribeCacheParameterGroups`. 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 ini, nama, rangkaian, 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, rangkaian, 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, rangkaian, 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>
```

Untuk informasi selengkapnya, lihat [DescribeCacheParameterGroups](#).

Menampilkan daftar nilai grup parameter

Anda dapat menampilkan daftar parameter dan nilai grup parameter menggunakan konsol ElastiCache, AWS CLI, atau API ElastiCache.

Menampilkan daftar nilai grup parameter (Konsol)

Prosedur berikut menunjukkan cara menampilkan daftar parameter dan nilainya untuk grup parameter menggunakan konsol ElastiCache.

Untuk menampilkan daftar parameter dan nilai grup parameter menggunakan konsol ElastiCache

1. Masuk ke AWS Management Console dan buka konsol ElastiCache 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 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 menampilkan daftar parameter dan nilai grup parameter menggunakan AWS CLI, gunakan perintah `describe-cache-parameters`.

Example

Contoh kode berikut menampilkan daftar semua parameter 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
```

Untuk informasi selengkapnya, lihat [describe-cache-parameters](#).

Menampilkan daftar nilai grup parameter (API ElastiCache)

Untuk menampilkan daftar parameter dan nilai grup parameter menggunakan API ElastiCache, gunakan tindakan DescribeCacheParameters.

Example

Contoh kode berikut menampilkan daftar semua parameter 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>
```

Untuk informasi selengkapnya, lihat [DescribeCacheParameters](#).

Mengubah grup parameter

Important

Anda tidak dapat mengubah grup parameter default.

Anda dapat mengubah beberapa nilai parameter di grup parameter. Nilai parameter ini diterapkan ke kluster yang terkait dengan grup parameter. Untuk informasi selengkapnya tentang kapan perubahan nilai parameter diterapkan ke grup parameter, lihat [Parameter spesifik Memcached](#).

Mengubah grup parameter (Konsol)

Prosedur berikut menunjukkan cara mengubah nilai parameter `binding_protocol` menggunakan konsol ElastiCache. Anda akan menggunakan prosedur yang sama untuk mengubah nilai parameter apa pun.

Untuk mengubah nilai parameter menggunakan konsol ElastiCache

1. Masuk ke AWS Management Console dan buka konsol ElastiCache 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. Di layar Edit Grup Parameter:, gulir menggunakan panah kiri dan kanan sampai Anda menemukan parameter `cluster-enabled`, lalu ketik `yes` di kolom Nilai.
7. Pilih Simpan Perubahan.
8. Untuk menemukan nama parameter yang Anda ubah, lihat [Parameter spesifik Memcached](#). Jika perubahan parameter diterapkan Setelah mulai ulang, boot ulang setiap klaster yang menggunakan grup parameter ini. Untuk informasi selengkapnya, lihat [Boot ulang klaster](#).

Mengubah grup parameter (AWS CLI)

Untuk mengubah nilai parameter menggunakan AWS CLI, gunakan perintah `modify-cache-parameter-group`.

Example

Untuk menemukan nama dan nilai yang diizinkan dari parameter yang ingin Anda ubah, lihat [Parameter spesifik Memcached](#)

Kode sampel 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"
}
```

Untuk informasi selengkapnya, lihat [modify-cache-parameter-group](#).

Mengubah grup parameter (API ElastiCache)

Untuk mengubah nilai parameter grup parameter menggunakan API ElastiCache, gunakan tindakan `ModifyCacheParameterGroup`.

Example

Untuk menemukan nama dan nilai yang diizinkan dari parameter yang ingin Anda ubah, lihat [Parameter spesifik Memcached](#)

Kode sampel 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>
```

Untuk informasi selengkapnya, lihat [ModifyCacheParameterGroup](#).

Menghapus grup parameter

Anda dapat membuat grup parameter menggunakan konsol ElastiCache, AWS CLI, atau API ElastiCache.

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 konsol ElastiCache

1. Masuk ke AWS Management Console dan buka konsol ElastiCache 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 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 myMem14
```

Untuk Windows:

```
aws elasticache delete-cache-parameter-group ^  
  --cache-parameter-group-name myMem14
```

Untuk informasi selengkapnya, lihat [delete-cache-parameter-group](#).

Menghapus grup parameter (API ElastiCache)

Untuk menghapus grup parameter menggunakan API ElastiCache, gunakan tindakan `DeleteCacheParameterGroup`. Untuk grup parameter yang akan dihapus, grup parameter yang ditentukan berdasarkan `CacheParameterGroupName` tidak boleh memiliki kluster yang terkait dengannya, dan juga tidak dapat berupa grup parameter default.

Example

Contoh kode 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>
```

Untuk informasi selengkapnya, lihat [DeleteCacheParameterGroup](#).

Parameter spesifik Memcached

Jika Anda tidak menentukan grup parameter untuk kluster Redis, 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 menentukannya ke kluster Anda kapan saja. Untuk informasi selengkapnya, lihat [Membuat grup parameter](#).

Topik

- [Perubahan Memcached 1.6.17](#)
- [Parameter yang ditambahkan dalam Memcached 1.6.6](#)
- [Perubahan parameter Memcached 1.5.10](#)
- [Parameter yang ditambahkan dalam Memcached 1.4.34](#)
- [Parameter yang ditambahkan dalam Memcached 1.4.33](#)
- [Parameter yang ditambahkan dalam Memcached 1.4.24](#)
- [Parameter yang ditambahkan dalam 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 mengaktifkan/menonaktifkan `lru_crawler` saat runtime melalui perintah. Aktifkan/nonaktifkan `lru_crawler` dengan memodifikasi grup parameter kustom Anda.

Parameter yang ditambahkan dalam Memcached 1.6.6


Untuk Memcached 1.6.6, tidak ada parameter tambahan yang didukung.

Rangkaian grup parameter: `memcached1.6`

Perubahan parameter Memcached 1.5.10

Untuk Memcached 1.5.10, parameter tambahan berikut didukung.

Rangkaian grup parameter: `memcached1.5`

Nama	Detail	Deskripsi
no_modern	Default: 1 Tipe: boolean Dapat diubah: Ya Nilai yang Diizinkan: 0,1 Perubahan Berlaku: Saat peluncuran	<p>Alias untuk menonaktifkan perintah <code>slab_reassign</code> , <code>slab_automove</code> , <code>lru_crawler</code> , <code>lru_maintainer</code> , <code>maxconns_fast</code> . No modern juga menetapkan <code>hash_algorithm</code> untuk <code>jenkins</code> dan memungkinkan inlining ASCII VALUE. Berlaku untuk memcached 1.5 dan lebih tinggi. Untuk mengembalikan ke modern, Anda harus menonaktifkan parameter ini dan meluncurkannya kembali, yang secara otomatis akan mengaktifkan <code>slab_reassign</code> , <code>slab_automove</code> , <code>lru_crawler</code> , <code>lru_maintainer</code> , dan <code>maxconns_fast</code> .</p> <div data-bbox="1008 1224 1507 1885"><p> Note</p><p>Nilai konfigurasi default untuk parameter ini telah diubah dari 0 ke 1 mulai 20 Agustus 2021. Nilai default yang diperbarui akan diterapkan secara otomatis oleh pengguna ElastiCache baru untuk setiap wilayah setelah 20 Agustus 2021. Pengguna ElastiCache yang sudah ada di wilayah sebelum</p></div>

Nama	Detail	Deskripsi
		<p>20 Agustus 2021 perlu memodifikasi grup parameter kustom mereka secara manual untuk menerapkan perubahan baru ini.</p>
inline_ascii_resp	<p>Default: 0</p> <p>Tipe: boolean</p> <p>Dapat diubah: Ya</p> <p>Nilai yang Diizinkan: 0,1</p> <p>Perubahan Berlaku: Saat peluncuran</p>	<p>Menyimpan angka dari respons VALUE, di dalam item, menggunakan hingga 24 byte. Perlambatan kecil untuk get ASCII, faster ditetapkan.</p>

Untuk Memcached 1.5.10, parameter berikut dihapus.

Nama	Detail	Deskripsi
expirezero_does_no_t_evict	<p>Default: 0</p> <p>Tipe: boolean</p> <p>Dapat diubah: Ya</p> <p>Nilai yang Diizinkan: 0,1</p> <p>Perubahan Berlaku: Saat peluncuran</p>	<p>Tidak lagi didukung di versi ini.</p>
modern	<p>Default: 1</p>	

Nama	Detail	Deskripsi
	<p>Tipe: boolean</p> <p>Dapat Diubah: Ya (memerlukan peluncuran ulang jika diatur ke <code>no_modern</code>)</p> <p>Nilai yang Diizinkan: 0,1</p> <p>Perubahan Berlaku: Saat peluncuran</p>	<p>Tidak lagi didukung di versi ini. Dimulai dari versi ini, <code>no-modern</code> diaktifkan secara default pada setiap peluncuran atau peluncuran ulang.</p>

Parameter yang ditambahkan dalam Memcached 1.4.34

Untuk Memcached 1.4.34, tidak ada parameter tambahan yang didukung.

Rangkaian grup parameter: `memcached1.4`

Parameter yang ditambahkan dalam Memcached 1.4.33

Untuk Memcached 1.4.33, parameter tambahan berikut didukung.

Rangkaian grup parameter: `memcached1.4`

Nama	Detail	Deskripsi
<code>modern</code>	<p>Default: diaktifkan</p> <p>Tipe: boolean</p> <p>Dapat diubah: Ya</p> <p>Perubahan Berlaku: Saat peluncuran</p>	<p>Alias untuk beberapa fitur. Pengaktifan <code>modern</code> setara dengan mengaktifkan perintah berikut dan menggunakan algoritma hash murmur3: <code>slab_reassign</code>, <code>slab_auto move</code>, <code>lru_crawler</code>, <code>lru_maintainer</code>, <code>maxconns_</code></p>

Nama	Detail	Deskripsi
		fast , dan hash_algorithm=murmur3 .
watch	Default: diaktifkan Tipe: 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) Tipe: integer Dapat diubah: Ya Perubahan Berlaku: Saat Peluncuran	Jumlah minimum detik saat klien akan diizinkan untuk idle sebelum diminta untuk tutup. Rentang nilai: 0 hingga 86400.

Nama	Detail	Deskripsi
track_sizes	Default: dinonaktifkan Tipe: 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) Tipe: integer Dapat diubah: Ya Perubahan Berlaku: Saat Peluncuran	Perintah watch mengaktifkan logging stream untuk Memcached . Namun watch dapat menghapus log jika tingkat pengosongan, mutasi, atau pengambilan cukup tinggi untuk menyebabkan buffer logging menjadi penuh. Dalam situasi tersebut, pengguna dapat meningkatkan ukuran buffer untuk mengurangi kemungkinan kehilangan log.
worker_logbuf_size	Default: 64 (KB) Tipe: integer Dapat diubah: Ya Perubahan Berlaku: Saat Peluncuran	Perintah watch mengaktifkan logging stream untuk Memcached . Namun watch dapat menghapus log jika tingkat pengosongan, mutasi, atau pengambilan cukup tinggi untuk menyebabkan logging buffer menjadi penuh. Dalam situasi tersebut, pengguna dapat meningkatkan ukuran buffer untuk mengurangi kemungkinan kehilangan log.

Nama	Detail	Deskripsi
<code>slab_chunk_max</code>	Default: 524288 (byte) Tipe: integer Dapat diubah: Ya Perubahan Berlaku: Saat Peluncuran	Menentukan ukuran maksimum slab. Pengaturan ukuran slab lebih kecil membuat penggunaan memori lebih efisien. Item yang lebih besar dari <code>slab_chunk_max</code> akan dibagi menjadi beberapa slab.
<code>lru_crawler metadump [all 1 2 3]</code>	Default: dinonaktifkan Tipe: boolean Dapat diubah: Ya Perubahan Berlaku: Segera	jika <code>lru_crawler</code> diaktifkan, perintah ini menghapus semua kunci. <code>all 1 2 3</code> - semua slab, atau tentukan nomor slab tertentu



Parameter yang ditambahkan dalam Memcached 1.4.24


Untuk Memcached 1.4.24, parameter tambahan berikut didukung.

Rangkaian grup parameter: memcached1.4

Nama	Detail	Deskripsi
<code>disable_flush_all</code>	Default: 0 (dininaktifkan) Tipe: boolean Dapat diubah: Ya Perubahan Berlaku: Saat peluncuran	Tambahkan parameter (-F) untuk menonaktifkan <code>flush_all</code> . Berguna jika Anda tidak ingin dapat menjalankan flush penuh pada instans produksi. Nilai: 0, 1 (pengguna dapat melakukan <code>flush_all</code> jika nilai adalah 0).

Nama	Detail	Deskripsi
hash_algorithm	Default: jenkins Tipe: string Dapat diubah: Ya Perubahan Berlaku: Saat peluncuran	Algoritma hash yang akan digunakan. Nilai yang diizinkan: murmur3 dan jenkins.

Nama	Detail	Deskripsi
<code>lru_crawler</code>	<p>Default: 0 (dinoaktifkan)</p> <p>Tipe: boolean</p> <p>Dapat diubah: Ya</p> <p>Perubahan Berlaku: Setelah pengaktifan ulang</p> <div data-bbox="651 684 971 1430"><p> Note</p><p>Anda dapat mengaktifkan <code>lru_crawler</code> untuk sementara pada saat runtime dari baris perintah. Untuk informasi selengkapnya, lihat kolom Deskripsi.</p></div>	<p>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.</p> <p>Untuk mengaktifkan sementara, jalankan <code>lru_crawler enable</code> di baris perintah.</p> <p><code>lru_crawler 1,3,5</code> merayapi kelas slab 1, 3, 5 dengan mencari item yang kedaluwarsa untuk ditambahkan ke daftar bebas.</p> <p>Nilai: 0,1</p> <div data-bbox="1008 1052 1507 1799"><p> Note</p><p>Pengaktifan <code>lru_crawler</code> 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 Mengubah grup parameter</p></div>

Nama	Detail	Deskripsi
<code>lru_maintainer</code>	<p>Default: 0 (dinonaktifkan)</p> <p>Tipe: boolean</p> <p>Dapat diubah: Ya</p> <p>Perubahan Berlaku: Saat peluncuran</p>	<p>Sebuah thread latar belakang yang mengacak item di antara LRU seiring kapasitas tercapai. Nilai: 0, 1.</p>
<code>expirezero_does_not_evict</code>	<p>Default: 0 (dinonaktifkan)</p> <p>Tipe: boolean</p> <p>Dapat diubah: Ya</p> <p>Perubahan Berlaku: Saat peluncuran</p>	<p>Ketika digunakan dengan <code>lru_maintainer</code>, menjadikan item yang memiliki waktu kedaluwarsa 0 tidak dapat dikosongkan.</p> <div style="border: 1px solid #f08080; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> Warning</p> <p>Hal ini dapat memenuhi memori yang tersedia untuk item lainnya yang dapat dikosongkan.</p> </div> <p>Dapat diatur untuk mengabaikan <code>lru_maintainer</code>.</p>

Parameter yang ditambahkan dalam Memcached 1.4.14

Untuk Memcached 1.4.14, parameter tambahan berikut didukung.

Rangkaian grup parameter: `memcached1.4`

Parameter yang ditambahkan dalam Memcached 1.4.14

Nama	Deskripsi
<code>config_max</code>	Jumlah maksimum entri konfigurasi ElastiCache.
<code>config_size_max</code>	Ukuran maksimum entri konfigurasi, dalam byte.
<code>hashpower_init</code>	Ukuran awal tabel hash ElastiCache, dinyatakan sebagai pangkat dua. Default-nya adalah 16 (2^{16}), atau 65536 kunci.
<code>maxconns_fast</code>	Mengubah cara permintaan koneksi baru ditangani ketika batas koneksi maksimum tercapai. Jika parameter ini diatur ke 0 (nol), koneksi baru ditambahkan ke antrian backlog dan akan menunggu sampai koneksi lain ditutup. Jika parameter diatur ke 1, ElastiCache mengirimkan kesalahan ke klien dan segera menutup koneksi.

Nama	Deskripsi
slab_automove	<p>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, ElastiCache akan memindahkan slab secara agresif setiap kali ada pengosongan. (Mode ini tidak direkomendasikan kecuali untuk tujuan pengujian.)</p>

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

Rangkaian grup parameter: memcached1.4

Untuk Memcached 1.4.5, parameter tambahan berikut didukung.

Parameter yang ditambahkan dalam Memcached 1.4.5

Nama	Detail	Deskripsi
backlog_queue_limit	Default: 1024 Tipe: integer Dapat diubah: Tidak	Batas antrean backlog.
binding_protocol	Default: otomatis Tipe: string	Protokol pengikatan. Nilai yang diizinkan adalah <code>ascii</code> dan <code>auto</code> .

Nama	Detail	Deskripsi
	Dapat diubah: Ya Perubahan Berlaku: Setelah pengaktifan ulang	Untuk panduan dalam mengubah nilai <code>binding_protocol</code> , lihat Mengubah grup parameter .
<code>cas_disabled</code>	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.
<code>chunk_size</code>	Default: 48 Tipe: integer Dapat diubah: Ya Perubahan Berlaku: Setelah pengaktifan ulang	Jumlah minimum, dalam byte, ruang untuk mengalokasikan kunci, nilai, dan bendera item terkecil.
<code>chunk_size_growth_factor</code>	Default: 1,25 Tipe: Float Dapat diubah: Ya Perubahan Berlaku: Setelah pengaktifan ulang	Faktor pertumbuhan yang mengontrol ukuran setiap potongan Memcached berturut-turut; setiap potongan akan <code>chunk_size_growth_factor</code> kali lebih besar dari potongan sebelumnya.
<code>error_on_memory_exhausted</code>	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
<code>large_memory_pages</code>	<p>Default: 0 (salah)</p> <p>Jenis: Boolean</p> <p>Dapat diubah: Tidak</p>	Jika 1 (benar), ElastiCache akan mencoba menggunakan halaman memori yang besar.
<code>lock_down_paged_memory</code>	<p>Default: 0 (salah)</p> <p>Jenis: Boolean</p> <p>Dapat diubah: Tidak</p>	Jika 1 (benar), ElastiCache akan mengunci semua memori yang di-paging.
<code>max_item_size</code>	<p>Default: 1048576</p> <p>Tipe: integer</p> <p>Dapat diubah: Ya</p> <p>Perubahan Berlaku: Setelah pengaktifan ulang</p>	Ukuran, dalam byte, item terbesar yang dapat disimpan dalam klaster.
<code>max_simultaneous_connections</code>	<p>Default: 65000</p> <p>Tipe: integer</p> <p>Dapat diubah: Tidak</p>	Jumlah maksimum koneksi bersamaan.
<code>maximize_core_file_limit</code>	<p>Default: 0 (salah)</p> <p>Jenis: Boolean</p> <p>Dapat diubah:</p> <p>Perubahan Berlaku: Setelah pengaktifan ulang</p>	Jika 1 (benar), ElastiCache akan memaksimalkan batas file inti.

Nama	Detail	Deskripsi
<code>memcached_connections_overhead</code>	Default: 100 Tipe: integer Dapat diubah: Ya Perubahan Berlaku: Setelah pengaktifan ulang	Jumlah memori yang akan disimpan untuk koneksi Memcached dan berbagai overhead lainnya. Untuk informasi tentang parameter ini, lihat Overhead koneksi Memcached .
<code>requests_per_event</code>	Default: 20 Tipe: 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 klaster Anda menggunakan grup parameter default, ElastiCache akan mengatur nilai untuk `memcached_connections_overhead` ke 13 MB.
- 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 setiap jenis simpul. Nilai pada parameter ini tidak dapat diubah.


Jenis simpul	<code>max_cache_memory</code> (dalam megabyte)	<code>num_threads</code>
<code>cache.t1.micro</code>	213	1
<code>cache.t2.micro</code>	555	1
<code>cache.t2.small</code>	1588	1
<code>cache.t2.medium</code>	3301	2
<code>cache.t3.micro</code>	512	2
<code>cache.t3.small</code>	1402	2
<code>cache.t3.medium</code>	3364	2
<code>cache.t4g.micro</code>	512	2
<code>cache.t4g.small</code>	1402	2
<code>cache.t4g.medium</code>	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).

Penskalaan ElastiCache untuk Memcached

Penskalaan ElastiCache untuk Memcached

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 waktu henti apa pun 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 ECPU/detik untuk cache Anda untuk mengontrol biaya cache. Melakukannya akan memastikan bahwa penggunaan cache Anda tidak pernah melebihi maksimum yang dikonfigurasi.

Jika Anda menetapkan skala maksimum, 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 menggunakan logika LRU. Ketika Anda menetapkan maksimum ECPU/detik dan penggunaan komputasi beban kerja Anda melebihi nilai ini, ElastiCache akan mulai membatasi permintaan Memcached.

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 (ECPU) 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 ECPU/detik yang tersedia pada cache dalam waktu 60 menit, dan mengirimkan pemberitahuan acara ketika pembaruan batas minimum selesai.

Cara kerja pra-penskalaan

Ketika batas minimum untuk ECPU/detik atau penyimpanan data diperbarui melalui konsol, CLI, atau API, batas baru itu tersedia dalam 1 jam. ElastiCache Tanpa server mendukung 30K ECPU/detik pada cache kosong, dan hingga 90K ECPU/detik saat menggunakan fitur Baca dari Replika. ElastiCache dapat menggandakan ECPU/detik setiap 10-12 menit. Kecepatan penskalaan ini cukup untuk sebagian besar beban kerja. Jika Anda mengantisipasi bahwa peristiwa penskalaan yang akan datang mungkin melebihi tingkat ini, maka sebaiknya atur ECPU/detik minimum ke ECPU/detik puncak yang Anda harapkan setidaknya 60 menit sebelum peristiwa puncak. Jika tidak, aplikasi mungkin mengalami peningkatan latensi dan pembatasan permintaan.

Setelah pembaruan batas minimum selesai, ElastiCache Tanpa Server akan mulai mengukur Anda untuk ECPU minimum baru 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 akan 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 ECPU/detik 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 ECPU/detik per slot (90K ECPU/detik saat menggunakan Read from Replica menggunakan koneksi

READONLY). Sebagai praktik terbaik, aplikasi Anda harus memastikan bahwa distribusi kunci di seluruh slot Redis dan lalu 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. Pada panel navigasi, pilih mesin yang berjalan pada cache yang ingin diubah itu.
3. Muncul daftar cache yang menjalankan mesin yang dipilih.
4. Pilih cache untuk dimodifikasi dengan memilih tombol radio di sebelah kiri nama cache.
5. Pilih Tindakan, dan kemudian pilih Ubah.
6. Di bawah batas Penggunaan, tetapkan batas Memori atau Komputasi yang sesuai.
7. Klik Pratinjau perubahan dan kemudian 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 ElastiCache untuk cluster yang dirancang sendiri Memcached

Jumlah data aplikasi Anda yang perlu diproses jarang statis. Hal ini meningkat dan menurun saat bisnis Anda tumbuh atau mengalami fluktuasi permintaan yang normal. Jika cache Anda dikelola sendiri, Anda perlu menyediakan perangkat keras yang memadai untuk puncak permintaan Anda, yang bisa jadi mahal. 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.

Berikut ini akan membantu Anda menemukan topik yang benar untuk tindakan penskalaan yang ingin Anda lakukan.

Menskalakan Klaster Memcached

Tindakan	Topik
Menskalakan ke luar	Menambahkan simpul ke klaster
Menskalakan ke dalam	Menghapus simpul dari klaster
Mengubah tipe simpul	Menskalakan Memcached secara vertikal

Cluster memcached terdiri dari 1 hingga 60 node. Menskalakan klaster Memcached ke luar dan ke dalam semudah menambahkan atau menghapus simpul dari klaster.

Jika Anda membutuhkan lebih dari 60 node dalam cluster Memcached, atau lebih dari 300 node total di AWS Region, isi formulir Permintaan Peningkatan ElastiCache Batas di <https://aws.amazon.com/contact-us/elasticache-node-limit-request/>.

Karena Anda dapat mempartisi data di semua simpul dalam sebuah kluster Memcached, menaikkan skala ke tipe simpul dengan memori yang lebih besar jarang diperlukan. Namun, karena mesin Memcached tidak mempertahankan data, jika Anda menskalakan ke jenis simpul yang berbeda, kluster baru akan dimulai kosong kecuali aplikasi Anda mengisinya.

Topik

- [Menskalakan Memcached secara Horizontal](#)
- [Menskalakan Memcached secara vertikal](#)

Menskalakan Memcached secara Horizontal

Mesin Memcached mendukung partisi data Anda di beberapa simpul. Karena itu, kluster Memcached dapat diskalakan secara horizontal dengan mudah. Cluster Memcached dapat memiliki 1 hingga 60 node. Untuk menskalakan kluster Memcached Anda secara horizontal, cukup tambahkan atau hapus simpul.

Jika Anda membutuhkan lebih dari 60 node dalam cluster Memcached, atau lebih dari 300 node total di AWS Region, isi formulir Permintaan Peningkatan ElastiCache Batas di <https://aws.amazon.com/contact-us/elasticache-node-limit-request/>.

Topik berikut memberikan detail cara menskalakan kluster Memcached Anda ke luar atau ke dalam dengan menambahkan atau menghapus simpul.

- [Menambahkan simpul ke kluster](#)
- [Menghapus simpul dari kluster](#)

Setiap kali Anda mengubah jumlah simpul dalam kluster Memcached, Anda harus memetakan ulang setidaknya beberapa keyspace agar dipetakan ke simpul yang benar. Untuk informasi lebih detail tentang penyeimbangan beban kluster Memcached Anda, lihat [Buat konfigurasi klien ElastiCache Anda untuk penyeimbangan beban yang efisien](#).

Jika Anda menggunakan penemuan otomatis pada kluster Memcached, Anda tidak perlu mengubah titik akhir dalam aplikasi saat Anda menambahkan atau menghapus simpul. Untuk informasi selengkapnya tentang penemuan otomatis, lihat [Identifikasi simpul di kluster Anda secara otomatis](#). Jika Anda tidak menggunakan penemuan otomatis, setiap kali Anda mengubah jumlah simpul dalam kluster Memcached Anda harus memperbarui titik akhir dalam aplikasi.

Menskalakan Memcached secara vertikal

Ketika Anda menskalakan kluster Memcached ke atas atau bawah, Anda harus membuat kluster baru. Kluster Memcached selalu dimulai kosong kecuali aplikasi Anda mengisinya.

Important

Jika Anda menurunkan skala ke tipe simpul yang lebih kecil, pastikan bahwa tipe simpul yang lebih kecil memadai untuk data dan overhead Anda. Untuk informasi selengkapnya, lihat [Pilih ukuran simpul cache](#).

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 kluster Memcached secara vertikal (konsol)

1. Buat kluster baru dengan tipe simpul baru. Untuk informasi selengkapnya, lihat [Membuat kluster Memcached \(konsol\)](#).
2. Dalam aplikasi Anda, perbarui titik akhir ke titik akhir kluster baru. Untuk informasi selengkapnya, lihat [Menemukan Titik Akhir Kluster \(Konsol\)](#).
3. Hapus kluster lama. Untuk informasi selengkapnya, lihat [Menghapus simpul baru di Memcached](#).

Menskalakan Memcached secara vertikal (AWS CLI)

Prosedur berikut memandu Anda dalam menskalakan kluster 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 dengan 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](#).

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 tipe simpul baru. Lihat informasi yang lebih lengkap di [Membuat cluster \(ElastiCache API\)](#)
2. Dalam aplikasi Anda, perbarui titik akhir ke titik akhir klaster cache baru. Untuk informasi selengkapnya, lihat [Menemukan Titik Akhir \(API ElastiCache\)](#).
3. Hapus klaster cache lama. Lihat informasi yang lebih lengkap di [Menggunakan API ElastiCache](#).

Menandai sumber daya ElastiCache Anda

Untuk membantu Anda mengelola kluster dan sumber daya ElastiCache lainnya, Anda dapat menetapkan metadata Anda sendiri ke setiap sumber daya dalam bentuk tanda. Dengan tanda, Anda dapat mengategorikan sumber daya AWS Anda dengan berbagai cara, misalnya, berdasarkan tujuan, pemilik, atau lingkungan. Hal ini berguna ketika Anda memiliki banyak sumber daya dengan tipe yang sama—Anda dapat dengan cepat mengidentifikasi sumber daya tertentu berdasarkan tanda yang telah Anda tetapkan. Topik ini memberikan penjelasan tentang tanda dan menunjukkan cara membuatnya.

Warning

Sebagai praktik terbaik, sebaiknya Anda tidak menyertakan data sensitif ke dalam tanda.

Dasar-dasar tanda

Tanda merupakan label yang Anda tetapkan ke sumber daya AWS. Setiap tanda terdiri dari kunci dan nilai opsional, yang keduanya Anda tentukan. Tanda memungkinkan Anda mengategorikan sumber daya AWS Anda dengan berbagai cara, misalnya, berdasarkan tujuan atau pemilik. Misalnya, Anda dapat menentukan serangkaian tanda untuk kluster ElastiCache akun Anda yang dapat membantu melacak setiap pemilik dan grup pengguna dari setiap instans.

Sebaiknya rancang serangkaian kunci tanda yang memenuhi kebutuhan setiap tipe sumber daya. Penggunaan set kunci tanda yang konsisten akan memudahkan pengelolaan sumber daya Anda. Anda dapat mencari dan memfilter sumber daya berdasarkan tanda yang Anda tambahkan. Untuk informasi selengkapnya tentang cara mengimplementasikan strategi penandaan sumber daya yang efektif, lihat [Laporan resmi AWS Praktik Terbaik Penandaan](#).

Tanda tidak memiliki makna semantik pada ElastiCache dan ditafsirkan sebagai string karakter. Selain itu, tanda tidak secara otomatis ditetapkan ke sumber daya Anda. Anda dapat mengedit kunci dan nilai tanda, serta menghapus tanda dari sumber daya kapan saja. Anda dapat menetapkan nilai tanda ke null. Jika Anda menambahkan tanda yang memiliki kunci yang sama dengan tanda yang telah ada di sumber daya tersebut, nilai yang baru akan menimpa nilai yang lama. Jika Anda menghapus sumber daya, semua tanda untuk sumber daya tersebut juga akan dihapus. Selain itu, jika Anda menambahkan atau menghapus tanda di grup replikasi, semua simpul di dalam grup replikasi itu juga akan mendapat penambahan atau penghapusan tanda yang sama.

Anda dapat bekerja dengan tanda menggunakan AWS Management Console, AWS CLI, dan API ElastiCache.

Jika menggunakan IAM, Anda dapat mengontrol pengguna mana di akun AWS Anda yang memiliki izin untuk membuat, mengedit, atau menghapus tanda. Untuk informasi selengkapnya, lihat [Izin tingkat sumber daya](#).

Sumber daya yang dapat Anda tandai

Anda dapat menandai sebagian besar sumber daya ElastiCache yang sudah ada dalam akun Anda. Tabel di bawah ini mencantumkan sumber daya yang mendukung penandaan. Jika Anda menggunakan AWS Management Console, Anda dapat menerapkan tanda ke sumber daya dengan menggunakan [Editor Tanda](#). Beberapa layar sumber daya memungkinkan Anda menentukan tanda untuk sebuah sumber daya saat sumber daya tersebut dibuat; misalnya, tanda dengan kunci Nama dan nilai yang Anda tentukan. Dalam kebanyakan kasus, konsol menerapkan tanda segera setelah sumber daya dibuat (alih-alih selama pembuatan sumber daya). Konsol dapat mengatur sumber daya sesuai dengan tanda Nama, tetapi tanda ini tidak memiliki makna semantik pada layanan ElastiCache.

Selain itu, beberapa tindakan pembuatan sumber daya memungkinkan Anda menentukan tanda untuk sumber daya saat sumber daya tersebut dibuat. Jika tanda tidak dapat diterapkan selama pembuatan sumber daya, kami akan mengembalikan proses pembuatan sumber daya. Hal ini untuk memastikan bahwa sumber daya dibuat dengan tanda atau tidak akan dibuat sama sekali, dan tidak akan ada sumber daya yang dibiarkan tidak bertanda. Dengan menandai sumber daya saat pembuatan, Anda tidak perlu menjalankan skrip penandaan kustom setelah pembuatan sumber daya.

Jika Anda menggunakan API Amazon ElastiCache, CLI AWS, atau SDK AWS, Anda dapat menggunakan parameter Tags pada tindakan ElastiCache API yang relevan untuk menerapkan tanda. File tersebut adalah:

- `CreateServerlessCache`
- `CreateCacheCluster`
- `CreateCacheParameterGroup`
- `CreateCacheSecurityGroup`
- `CreateCacheSubnetGroup`
- `PurchaseReservedCacheNodesOffering`

Tabel berikut menjelaskan sumber daya ElastiCache yang dapat ditandai, dan sumber daya yang dapat ditandai saat pembuatan dengan menggunakan API ElastiCache, CLI AWS, atau SDK AWS.

Dukungan penandaan untuk sumber daya ElastiCache

Mendukung tanda	Mendukung penandaan saat pembuatan
Ya	Ya
Ya	Ya
Ya	Ya
Ya	Ya
Ya	Ya
Ya	Ya

Anda dapat menerapkan izin tingkat sumber daya berbasis tanda dalam kebijakan IAM pada tindakan API ElastiCache yang mendukung penandaan saat pembuatan guna mengimplementasikan kontrol terperinci atas pengguna dan grup yang dapat menandai sumber daya saat pembuatan. Sumber daya Anda diamankan secara tepat sejak pembuatan—tanda yang diterapkan segera ke sumber daya Anda. Oleh karena itu, izin tingkat sumber daya berbasis tanda apa pun yang mengontrol penggunaan sumber daya akan langsung diterapkan. Sumber daya Anda dapat dilacak dan dilaporkan dengan lebih akurat. Anda dapat menerapkan penggunaan penandaan pada sumber daya baru serta mengontrol kunci dan nilai tanda mana yang ditetapkan pada sumber daya Anda.

Untuk informasi selengkapnya, lihat [Contoh penandaan sumber daya](#).

Untuk informasi selanjutnya tentang penandaan sumber daya Anda untuk penagihan, lihat [Memantau biaya dengan tanda alokasi biaya](#).

Batasan tanda

Batasan dasar berikut berlaku untuk tanda:

- Jumlah maksimum tanda per sumber daya – 50
- Untuk setiap sumber daya, setiap kunci tanda harus unik, dan setiap kunci tanda 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 tandanya, layanan lain mungkin lebih terbatas. Karakter yang diizinkan di semua layanan adalah huruf, angka, dan spasi yang dapat direpresentasikan dalam UTF-8, serta karakter berikut: + - = . _ : / @
- Kunci dan nilai tanda peka huruf besar dan kecil.
- Prefiks `aws :` disimpan untuk penggunaan AWS. Jika tanda memiliki kunci tanda dengan prefiks ini, Anda tidak dapat mengedit atau menghapus kunci atau nilai tanda tersebut. Tanda dengan prefiks `aws :` tidak dihitung terhadap tanda 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 tandai dengan tanda kunci yang disebut `DeleteMe`, Anda harus menggunakan tindakan `DeleteSnapshot` dengan pengidentifikasi sumber daya snapshot tersebut, seperti `snap-1234567890abcdef0`.

Untuk informasi selengkapnya tentang sumber daya ElastiCache yang dapat ditandai, lihat [Sumber daya yang dapat Anda tandai](#).

Contoh penandaan sumber daya

- Membuat cache nirserver menggunakan tanda

```
aws elasticache create-serverless-cache \  
  --serverless-cache-name CacheName \  
  --engine memcached \  
  --tags Key="Cost Center", Value="1110001" Key="project",Value="XYZ"
```

- Menambahkan tanda ke cache nirserver

```
aws elasticache add-tags-to-resource \  
  --resource-id <resource-id>
```

```
--resource-name arn:aws:elasticache:us-east-1:111111222233:serverlesscache:my-cache \
--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 memcached \
--tags Key="project",Value="XYZ" Key="Elasticache",Value="Service"
```

Contoh kebijakan kontrol akses Berbasis Tanda

1. Mengizinkan tindakan `AddTagsToResource` untuk klaster hanya jika klaster memiliki tanda `Project=XYZ`.

```
{
  "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 tanda `Project` dan `Service` serta kunci yang berbeda dari `Project` dan `Service`.

```
{
```

```

"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 tanda berbeda dari `Project` dan `Service`.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "elasticache:AddTagsToResource",
      "Resource": [
        "arn:aws:elasticache:*:*:*:*"
      ],
      "Condition": {
        "ForAnyValue:StringNotEqualsIgnoreCase": {
          "aws:TagKeys": [
            "Service",
            "Project"
          ]
        }
      }
    }
  ]
}

```



```

    }
  }
]
}

```

4. Menolak CreateCacheCluster tindakan jika tag permintaan Project hilang atau tidak sama dengan Dev, QA atau Prod.

```

{
  "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 syarat ini, lihat [Menggunakan kunci syarat](#).

Memantau biaya dengan tanda alokasi biaya

Ketika Anda menambahkan tanda alokasi biaya ke sumber daya Anda di Amazon ElastiCache, Anda dapat melacak biaya dengan mengelompokkan pengeluaran pada faktur Anda berdasarkan nilai tanda sumber daya.

Tanda alokasi biaya ElastiCache adalah pasangan kunci dan nilai yang Anda tentukan dan kaitkan dengan sumber daya ElastiCache. Kunci dan nilai peka terhadap huruf besar dan kecil. Anda dapat menggunakan kunci tanda untuk menentukan kategori, dan nilai tanda dapat berupa item dalam kategori tersebut. Misalnya, Anda dapat menentukan kunci tanda `CostCenter` dan nilai tanda `10010`, yang menunjukkan bahwa sumber daya tersebut ditetapkan ke pusat pembiayaan `10010`. Anda juga dapat menggunakan tanda untuk menunjukkan bahwa sumber daya sedang digunakan untuk pengujian atau produksi dengan menggunakan kunci seperti `Environment` dan nilai seperti `test` atau `production`. Sebaiknya gunakan kumpulan kunci tanda yang konsisten untuk mempermudah pelacakan biaya yang terkait dengan sumber daya Anda.

Gunakan tanda alokasi biaya untuk mengatur tagihan AWS guna mencerminkan struktur biaya Anda sendiri. Untuk melakukannya, daftar untuk mendapatkan tagihan akun AWS Anda dengan menyertakan nilai kunci tanda. Kemudian, untuk melihat biaya sumber daya gabungan, atur informasi penagihan Anda sesuai dengan sumber daya Anda dengan nilai kunci tanda yang sama. Misalnya, Anda dapat menandai beberapa sumber daya dengan nama aplikasi tertentu, lalu mengatur informasi penagihan untuk melihat biaya total aplikasi tersebut pada beberapa layanan.

Anda juga dapat menggabungkan tanda untuk melacak biaya dengan tingkat detail yang lebih besar. Misalnya, untuk melacak biaya layanan Anda menurut wilayah, Anda dapat menggunakan kunci tanda 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 biaya ElastiCache Anda yang dikelompokkan menurut wilayah. Untuk informasi selengkapnya, lihat [Menggunakan Tanda Alokasi Biaya](#) di Panduan Pengguna AWS Billing.

Anda dapat menambahkan tanda alokasi biaya ElastiCache ke klaster Memcached. Saat Anda menambahkan, menampilkan daftar, mengubah, menyalin, atau menghapus tanda, operasi tersebut hanya akan diterapkan ke klaster yang ditentukan.

Karakteristik tanda alokasi biaya ElastiCache

- Tanda alokasi biaya diterapkan ke sumber daya ElastiCache 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>`

Memcached: Tanda hanya diterapkan ke klaster.

Contoh arn: `arn:aws:elasticache:us-west-2:1234567890:cluster:my-cluster`

- Kunci tag adalah nama tanda yang wajib diisi. Nilai string tanda 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 aroba (`@`).
- Nilai tanda adalah nilai tanda 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 aroba (`@`).
- Sumber daya ElastiCache dapat memiliki maksimum 50 tanda.

- Nilai tidak harus unik di dalam kumpulan tanda. Misalnya, Anda dapat memiliki kumpulan tanda dengan kunci Service dan Application yang memiliki nilai ElastiCache.

AWS tidak menerapkan makna semantik pada tanda Anda. Tanda ditafsirkan hanya sebagai string karakter. AWS tidak menetapkan secara otomatis tanda apa pun pada sumber daya ElastiCache.

Mengelola tanda alokasi biaya menggunakan AWS CLI

Anda dapat menggunakan AWS CLI untuk menambah, mengubah, atau menghapus tanda alokasi biaya.

Tanda alokasi biaya diterapkan ke klaster ElastiCache for Memcached. Klaster yang akan ditandai ditentukan menggunakan ARN (Amazon Resource Name).

Contoh arn: `arn:aws:elasticache:us-west-2:1234567890:cluster:my-cluster`

Contoh arn: `arn:aws:elasticache:us-west-2:1234567890:cluster:my-cluster`

Topik

- [Menampilkan daftar tanda menggunakan AWS CLI](#)
- [Menambah tanda menggunakan AWS CLI](#)
- [Mengubah tanda menggunakan AWS CLI](#)
- [Menghapus tanda menggunakan AWS CLI](#)

Menampilkan daftar tanda menggunakan AWS CLI

Anda dapat menggunakan AWS CLI untuk menampilkan daftar tanda di sumber daya ElastiCache yang ada dengan menggunakan operasi [list-tags-for-resource](#).

Kode berikut menggunakan AWS CLI untuk menampilkan daftar tanda pada klaster Memcached `my-cluster` 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
```

Untuk Windows:

```
aws elasticache list-tags-for-resource ^  
  --resource-name arn:aws:elasticache:us-west-2:0123456789:cluster:my-cluster
```

Output dari operasi ini akan terlihat seperti berikut ini, daftar dari semua tanda pada sumber daya.

```
{  
  "TagList": [  
    {  
      "Value": "10110",  
      "Key": "CostCenter"  
    },  
    {  
      "Value": "EC2",  
      "Key": "Service"  
    }  
  ]  
}
```

Jika tidak ada tanda pada sumber daya, output akan menjadi TagList kosong.

```
{  
  "TagList": []  
}
```

Untuk informasi selengkapnya, lihat AWS CLI untuk ElastiCache [list-tags-for-resource](#).

Menambah tanda menggunakan AWS CLI

Anda dapat menggunakan AWS CLI untuk menampilkan daftar tanda di sumber daya ElastiCache yang ada dengan menggunakan operasi CLI [add-tags-to-resource](#). Jika kunci tanda 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 elasticache dan us-west-2 masing-masing ke kluster my-cluster di wilayah us-west-2.

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=EC2,Key=Region,Value=us-west-2
```

```
--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
```

Output dari operasi ini akan terlihat seperti berikut ini, daftar dari semua tanda pada sumber daya mengikuti operasi tersebut.

```
{  
  "TagList": [  
    {  
      "Value": "elasticache",  
      "Key": "Service"  
    },  
    {  
      "Value": "us-west-2",  
      "Key": "Region"  
    }  
  ]  
}
```

Untuk informasi selengkapnya, lihat AWS CLI untuk ElastiCache [add-tags-to-resource](#).

Anda juga dapat menggunakan AWS CLI untuk menambahkan tanda ke kluster ketika Anda membuat sebuah kluster baru dengan menggunakan operasi [create-cache-cluster](#). Anda tidak dapat menambahkan tanda saat membuat kluster menggunakan konsol manajemen ElastiCache. Setelah kluster dibuat, Anda kemudian dapat menggunakan konsol untuk menambahkan tanda pada kluster tersebut.

Mengubah tanda menggunakan AWS CLI

Anda dapat menggunakan AWS CLI untuk mengubah tanda pada kluster ElastiCache for Memcached.

Untuk mengubah tanda:

- Gunakan [add-tags-to-resource](#) untuk menambahkan tanda dan nilai baru atau untuk mengubah nilai yang terkait dengan tanda yang ada.
- Gunakan [remove-tags-from-resource](#) untuk menghapus tanda tertentu dari sumber daya.

Output dari kedua operasi tersebut akan berupa daftar tanda dan nilai-nilainya di kluster yang ditentukan.

Menghapus tanda menggunakan AWS CLI

Anda dapat menggunakan tanda dari kluster AWS CLI ElastiCache for Memcached yang ada dengan menggunakan operasi [remove-tags-from-resource](#).

Kode berikut menggunakan AWS CLI untuk menghapus tanda dengan kunci Service dan Region dari kluster `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
```

Output dari operasi ini akan terlihat seperti berikut ini, daftar dari semua tanda pada sumber daya mengikuti operasi tersebut.

```
{  
  "TagList": []  
}
```

Untuk informasi selengkapnya, lihat AWS CLI untuk ElastiCache [remove-tags-from-resource](#).

Mengelola tanda alokasi biaya Anda menggunakan API ElastiCache

Anda dapat menggunakan API ElastiCache untuk menambah, mengubah, atau menghapus tanda alokasi biaya.

Tanda alokasi biaya diterapkan ke klaster ElastiCache for Memcached. Klaster yang akan ditandai ditentukan menggunakan ARN (Amazon Resource Name).

Contoh arn: `arn:aws:elasticache:us-west-2:1234567890:cluster:my-cluster`

Topik

- [Menampilkan daftar tanda menggunakan API ElastiCache](#)
- [Menambahkan tanda menggunakan API ElastiCache](#)
- [Mengubah tanda menggunakan API ElastiCache](#)
- [Menghapus tanda menggunakan API ElastiCache](#)

Menampilkan daftar tanda menggunakan API ElastiCache

Anda dapat menggunakan API ElastiCache untuk menampilkan daftar tanda di sumber daya yang ada dengan menggunakan operasi [ListTagsForResource](#).

Kode berikut menggunakan API ElastiCache untuk menampilkan daftar tanda di 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>
```

Menambahkan tanda menggunakan API ElastiCache

Anda dapat menggunakan API ElastiCache untuk menambahkan tanda ke klaster ElastiCache yang ada dengan menggunakan operasi [AddTagsToResource](#). Jika kunci tanda 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 API ElastiCache untuk menambahkan kunci `Service` dan `Region` dengan nilai `elasticache` dan `us-west-2` masing-masing ke sumber daya `my-cluster` di wilayah `us-west-2`.


```
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>
```

Untuk informasi selengkapnya, lihat [AddTagsToResource](#) di Referensi API Amazon ElastiCache.

Mengubah tanda menggunakan API ElastiCache

Anda dapat menggunakan API ElastiCache untuk mengubah tanda di kluster ElastiCache.

Untuk mengubah nilai tanda:

- Gunakan operasi [AddTagsToResource](#) untuk menambahkan tanda dan nilai baru atau untuk mengubah nilai tanda yang ada.
- Gunakan [RemoveTagsFromResource](#) untuk menghapus tanda dari sumber daya.

Hasil dari kedua operasi tersebut akan berupa daftar tanda dan nilai-nilainya di sumber daya yang ditentukan.

Gunakan [RemoveTagsFromResource](#) untuk menghapus tanda dari sumber daya.

Menghapus tanda menggunakan API ElastiCache

Anda dapat menggunakan API ElastiCache untuk menghapus tanda dari kluster ElastiCache for Memcached yang ada dengan menggunakan operasi [RemoveTagsFromResource](#).

Kode berikut menggunakan API ElastiCache untuk menghapus tag dengan kunci Service dan Region dari kluster 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
```

```
&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 Lensa Amazon ElastiCache Well-Architected, kumpulan prinsip desain dan panduan untuk merancang beban kerja ElastiCache well-architected.

- Lensa ElastiCache adalah aditif untuk [Kerangka Kerja AWS Well-Architected](#).
- Setiap Pilar memiliki serangkaian pertanyaan untuk membantu memulai diskusi seputar Arsitektur ElastiCache.
 - Setiap pertanyaan memiliki sejumlah praktik terkemuka bersama dengan skor mereka untuk pelaporan.
 - Wajib - Diperlukan sebelum mulai ke prod (tidak memenuhinya menimbulkan risiko tinggi)
 - Terbaik - Kemungkinan status terbaik yang bisa dilakukan pelanggan
 - Bagus - Apa yang kami rekomendasikan untuk dimiliki pelanggan (tidak memenuhinya menimbulkan risiko sedang)
- Terminology Well-Architected
 - [Komponen](#) — Kode, konfigurasi, dan Sumber Daya AWS yang bersama-sama memenuhi persyaratan. Komponen berinteraksi dengan komponen lain, dan sering disamakan dengan layanan dalam arsitektur microservice.
 - [Beban Kerja](#) Seperangkat 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.

Topik

- [Pilar Keunggulan Operasional Lensa Well-Architected Amazon ElastiCache](#)
- [Pilar Keamanan Lensa Amazon ElastiCache Well-Architected](#)
- [Pilar Keandalan Lensa Amazon ElastiCache yang Dirancang dengan Baik](#)
- [Pilar Efisiensi Kinerja Lensa Amazon ElastiCache yang Dirancang dengan Baik](#)

- [Pilar Optimisasi Biaya Lensa Well-Architected Amazon ElastiCache](#)

Pilar Keunggulan Operasional Lensa Well-Architected Amazon ElastiCache

Pilar keunggulan operasional berfokus untuk menjalankan dan memantau sistem guna memberikan nilai bisnis, dan terus meningkatkan proses dan prosedur. Topik utama yang dibahas meliputi mengotomatiskan perubahan, merespons peristiwa, dan mendefinisikan standar untuk mengelola operasi harian.

Topik

- [OE 1: Bagaimana cara memahami dan menanggapi peringatan dan peristiwa yang dipicu oleh kluster ElastiCache Anda?](#)
- [OE 2: Kapan dan bagaimana Anda menskalakan kluster ElastiCache yang ada?](#)
- [OE 3: Bagaimana cara Anda mengelola sumber daya kluster ElastiCache dan menjaga kluster Anda tetap terbaru?](#)
- [OE 4: Bagaimana cara mengelola koneksi klien ke kluster ElastiCache Anda?](#)
- [OE 5: Bagaimana cara men-deploy Komponen ElastiCache untuk Beban Kerja?](#)
- [OE 6: Bagaimana cara melakukan perencanaan dan mitigasi terhadap kegagalan?](#)
- [OE 7: Bagaimana cara memecahkan masalah peristiwa mesin Redis?](#)

OE 1: Bagaimana cara memahami dan menanggapi peringatan dan peristiwa yang dipicu oleh kluster ElastiCache Anda?

Pengenalan tingkat pertanyaan: Saat Anda mengoperasikan kluster ElastiCache, Anda dapat menerima notifikasi dan peringatan secara opsional saat peristiwa tertentu terjadi. ElastiCache, secara default, mencatat log [peristiwa](#) yang berhubungan dengan sumber daya Anda, seperti failover, penggantian simpul, operasi penskalaan, pemeliharaan terjadwal, dan banyak lagi. Setiap peristiwa menyertakan tanggal dan waktu, nama sumber dan jenis sumber, serta deskripsi.

Manfaat tingkat pertanyaan: Dengan memahami dan mengelola alasan mendasar di balik peristiwa yang memicu peringatan yang dihasilkan oleh kluster, Anda dapat beroperasi secara lebih efektif dan merespons peristiwa dengan tepat.

- [Wajib] Tinjau peristiwa yang dihasilkan oleh ElastiCache pada konsol ElastiCache (setelah memilih wilayah Anda) atau menggunakan perintah [describe-events](#) [Amazon Command Line](#)

[Interface](#) (AWS CLI) dan [API ElastiCache](#). Anda dapat mengonfigurasi ElastiCache untuk mengirim notifikasi untuk peristiwa klaster penting menggunakan Amazon Simple Notification Service (Amazon SNS). Penggunaan Amazon SNS dengan klaster Anda memungkinkan Anda mengambil tindakan pada peristiwa ElastiCache secara programatis.

- Ada dua kategori besar peristiwa: peristiwa terkini dan terjadwal. Daftar peristiwa terkini meliputi: pembuatan dan penghapusan sumber daya, operasi penskalaan, failover, boot ulang simpul, pembuatan snapshot, modifikasi 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 untuk penggantian selama periode pemeliharaan dan penggantian simpul yang dijadwalkan ulang.
- Meskipun Anda mungkin tidak perlu segera bereaksi terhadap beberapa peristiwa ini, penting untuk melihat semua peristiwa kegagalan terlebih dahulu:
 - ElastiCache:AddCacheNodeFailed
 - ElastiCache:CacheClusterProvisioningFailed
 - ElastiCache:CacheClusterScalingFailed
 - ElastiCache:CacheNodesRebooted
 - ElastiCache:SnapshotFailed (hanya Redis)
- [Sumber Daya]:
 - [Mengelola notifikasi ElastiCache Amazon SNS](#)
 - [Notifikasi Peristiwa dan Amazon SNS](#)
- [Terbaik] Untuk mengotomatiskan respons terhadap peristiwa, manfaatkan kemampuan produk dan layanan AWS seperti Fungsi Lambda dan SNS. 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 metrik Amazon CloudWatch untuk memantau klaster.

[Sumber Daya]: [Pantau titik akhir replika baca Amazon ElastiCache for Redis \(mode klaster dinonaktifkan\) menggunakan Lambda AWS, Amazon Route 53, dan Amazon SNS](#) untuk kasus penggunaan yang menggunakan Lambda dan SNS.

OE 2: Kapan dan bagaimana Anda menskalakan klaster ElastiCache yang ada?

Pengenalan tingkat pertanyaan: Rightsizing klaster ElastiCache 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 mengatasi pemanfaatan sumber daya Anda yang kurang atau berlebihan, ElastiCache dapat menskalakan dua dimensi ke dalam. 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 mengalihkan operasi baca ke simpul replika. Gunakan simpul replika untuk operasi baca guna mengurangi pemanfaatan simpul primer. Hal ini dapat dikonfigurasi di pustaka klien Redis Anda dengan menghubungkan ke titik akhir pembaca ElastiCache untuk mode kluster dinonaktifkan, atau dengan menggunakan perintah Redis READONLY untuk mode kluster diaktifkan.

[Sumber Daya]:

- [Menemukan titik akhir koneksi](#)
- [Rightsizing Kluster](#)
- [Perintah Redis READONLY](#)
- [Wajib] Pantau pemanfaatan sumber daya kluster penting seperti CPU, memori, dan jaringan. Pemanfaatan sumber daya kluster yang spesifik ini perlu dilacak untuk menentukan keputusan Anda dalam penskalaan, dan jenis operasi penskalaan. Untuk kluster ElastiCache for Redis mode dinonaktifkan, simpul primer dan replika dapat diskalakan secara vertikal. Simpul replika juga dapat diskalakan secara horizontal dari 0 ke 5 simpul. Untuk mode kluster diaktifkan, hal yang sama berlaku dalam setiap serpihan kluster. Selain itu, Anda dapat menambah atau mengurangi jumlah serpihan.

[Sumber Daya]:

- [Praktik terbaik pemantauan dengan Amazon ElastiCache for Redis menggunakan Amazon CloudWatch](#)
- [Menskalakan Kluster ElastiCache for Redis](#)
- [Menskalakan kluster ElastiCache for 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 metrik CloudWatch untuk memindai rentang waktu yang lebih lama.

Pembelajaran dari pengamatan metrik CloudWatch dalam jangka waktu lama akan membentuk perkiraan Anda seputar pemanfaatan sumber daya kluster. Titik data dan metrik CloudWatch tersedia untuk rentang waktu hingga 455 hari.

[Sumber Daya]:

- [Memantau ElastiCache for Redis dengan Metrik CloudWatch](#)
- [Memantau Memcached dengan Metrik CloudWatch](#)
- [Praktik terbaik pemantauan dengan Amazon ElastiCache for Redis menggunakan Amazon CloudWatch](#)
- [Terbaik] Jika sumber daya ElastiCache Anda dibuat dengan CloudFormation, praktik terbaiknya adalah melakukan perubahan menggunakan templat CloudFormation untuk menjaga konsistensi operasional dan menghindari perubahan konfigurasi dan penyimpangan tumpukan yang tidak terkelola.

[Sumber Daya]:

- [Referensi jenis sumber daya ElastiCache untuk CloudFormation](#)
- [Terbaik] Otomatiskan operasi penskalaan Anda menggunakan data operasional kluster dan tentukan ambang batas di CloudWatch untuk menyiapkan alarm. Gunakan CloudWatch Events dan Simple Notification Service (SNS) untuk memicu fungsi Lambda dan menjalankan API ElastiCache untuk menskalakan kluster Anda secara otomatis. Contohnya adalah menambahkan serpihan ke kluster Anda ketika metrik EngineCPUUtilization mencapai 80% untuk jangka waktu yang lama. Pilihan lain adalah menggunakan DatabaseMemoryUsedPercentages untuk ambang batas berbasis memori.

[Sumber Daya]:

- [Menggunakan Alarm Amazon CloudWatch](#)
- [Apa itu peristiwa Amazon CloudWatch?](#)
- [Menggunakan AWS Lambda dengan Amazon Simple Notification Service](#)
- [Referensi API ElastiCache](#)

OE 3: Bagaimana cara Anda mengelola sumber daya kluster ElastiCache dan menjaga kluster Anda tetap terbaru?

Pendahuluan tingkat pertanyaan: Saat beroperasi dalam skala besar, penting bagi Anda untuk dapat menentukan dan mengidentifikasi semua sumber daya ElastiCache Anda. Saat meluncurkan fitur

aplikasi baru, Anda perlu menyamakan versi klaster di semua jenis lingkungan ElastiCache Anda: pengembangan, pengujian, dan produksi. 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 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. Dengan memanfaatkan fitur ElastiCache native, tim rekayasa Anda dapat fokus pada pemenuhan tujuan bisnis dan bukan pada pemeliharaan ElastiCache.

- [Terbaik] Jalankan klaster pada versi mesin terbaru yang tersedia dan terapkan Pembaruan Layanan Mandiri segera saat tersedia. ElastiCache secara otomatis memperbarui infrastruktur dasarnya selama periode pemeliharaan klaster 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 Terkelola dan Pembaruan Layanan Amazon ElastiCache](#)
- [Terbaik] Atur sumber daya ElastiCache Anda menggunakan tag. Gunakan tag pada grup replikasi dan bukan pada simpul individu. Anda dapat mengonfigurasi tag agar ditampilkan saat Anda mengkueri 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](#)
- [Referensi jenis sumber daya ElastiCache untuk CloudFormation](#)
- [Grup Parameter](#)

OE 4: Bagaimana cara mengelola koneksi klien ke kluster ElastiCache Anda?

Pengenalan tingkat pertanyaan: Saat beroperasi dalam skala besar, Anda perlu memahami bagaimana klien Anda terhubung dengan kluster ElastiCache untuk mengelola aspek operasional aplikasi Anda (seperti waktu respons).

Manfaat tingkat pertanyaan: Dengan memilih mekanisme koneksi yang paling tepat, aplikasi Anda dipastikan tidak akan terputus karena kesalahan konektivitas, seperti batas waktu habis.

- [Wajib] Pisahkan operasi baca dari tulis dan hubungkan ke simpul replika untuk menjalankan operasi baca. Namun, ketahuilah ketika Anda memisahkan operasi baca dan tulis, Anda akan kehilangan kemampuan untuk membaca kunci segera setelah menulisnya karena sifat replikasi Redis yang tidak sinkron. Perintah WAIT dapat dimanfaatkan untuk meningkatkan keamanan data dunia nyata dan memaksa replika mengonfirmasi penulisan sebelum merespons klien, dengan mengorbankan performa secara keseluruhan. Penggunaan simpul replika untuk operasi baca dapat dikonfigurasi di pustaka klien ElastiCache for Redis Anda menggunakan titik akhir pembaca ElastiCache untuk mode kluster dinonaktifkan. Untuk mode kluster diaktifkan, gunakan perintah READONLY ElastiCache for Redis. Untuk banyak pustaka klien ElastiCache for Redis, READONLY ElastiCache for Redis diimplementasikan secara default atau melalui pengaturan konfigurasi.

[Sumber Daya]:

- [Menemukan titik akhir koneksi](#)
- [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 mengimplementasikan pooling koneksi melalui pustaka klien ElastiCache for Redis (jika didukung), dengan Framework yang tersedia untuk lingkungan aplikasi Anda, atau membuatnya dari awal.

- [Terbaik] Pastikan batas waktu soket klien diatur ke setidaknya satu detik (vs. pengaturan default "tidak ada" di beberapa klien).
 - Pengaturan nilai batas waktu yang terlalu rendah dapat menyebabkan kemungkinan batas 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. Hal ini akan mengurangi latensi dan pemanfaatan CPU yang diperlukan untuk membuka dan menutup koneksi, serta melakukan handshake TLS jika TLS diaktifkan di klaster.

[Sumber Daya]: [Konfigurasi Amazon ElastiCache for Redis 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 cara men-deploy Komponen ElastiCache untuk Beban Kerja?

Pengenalan tingkat pertanyaan: Lingkungan ElastiCache dapat di-deploy secara manual melalui Konsol AWS, atau secara programatis melalui API, CLI, toolkit, dll. Praktik terbaik Keunggulan Operasional menyarankan untuk mengotomatiskan deployment melalui kode jika memungkinkan. Selain itu, klaster ElastiCache dapat diisolasi berdasarkan beban kerja atau digabungkan untuk tujuan optimisasi biaya.

Manfaat tingkat pertanyaan: Dengan memilih mekanisme deployment yang paling tepat untuk lingkungan ElastiCache Anda, Keunggulan Operasional dapat meningkat 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 lingkungan ElastiCache khusus per beban kerja atau menggabungkan beberapa beban kerja menjadi satu klaster, atau kombinasinya. Memahami kompromi dapat membantu mencapai keseimbangan antara Keunggulan Operasional dan Optimisasi Biaya

- [Wajib] Pahami opsi deployment yang tersedia untuk ElastiCache, dan otomatisasi prosedur ini jika memungkinkan. Kemungkinan cara otomatisasi termasuk CloudFormation, AWS CLI/SDK, dan API.

[Sumber Daya]:

- [Referensi jenis sumber daya Amazon ElastiCache](#)
 - [elasticache](#)
 - [Referensi API Amazon ElastiCache](#)
- [Wajib] Untuk semua beban kerja, tentukan tingkat isolasi kluster yang diperlukan.
 - [Terbaik]: Isolasi Tinggi – pemetaan beban kerja ke kluster 1:1. Memungkinkan kontrol paling terperinci atas akses, ukuran, penskalaan, dan pengelolaan sumber daya ElastiCache berdasarkan per beban kerja.
 - [Lebih Baik]: Isolasi Sedang – M:1 diisolasi berdasarkan tujuan, tetapi mungkin dibagi di beberapa beban kerja (misalnya kluster 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 melakukan perencanaan dan mitigasi terhadap kegagalan?

Pengenalan tingkat pertanyaan: Keunggulan Operasional mencakup antisipasi kegagalan dengan melakukan latihan "pre-mortem" reguler untuk mengidentifikasi sumber kegagalan potensial sehingga dapat diatasi atau dimitigasi. ElastiCache menawarkan API Failover yang memungkinkan simulasi peristiwa kegagalan simpul, demi tujuan pengujian.

Manfaat tingkat pertanyaan: Dengan menguji skenario kegagalan terlebih dahulu, Anda dapat mempelajari bagaimana pengaruhnya terhadap beban kerja Anda. Hal ini memungkinkan pengujian prosedur respons yang aman dan efektivitasnya, serta membuat tim Anda terbiasa dengan eksekusinya.

[Wajib] Lakukan pengujian failover secara berkala di akun dev/tes. [TestFailover](#)

OE 7: Bagaimana cara memecahkan masalah peristiwa mesin Redis?

Pengenalan tingkat pertanyaan: Keunggulan Operasional membutuhkan kemampuan untuk menyelidiki informasi tingkat layanan dan tingkat mesin untuk menganalisis kondisi dan status kluster

Anda. Amazon ElastiCache for Redis dapat mengirimkan log mesin Redis ke Amazon CloudWatch dan Amazon Kinesis Data Firehose.

Manfaat tingkat pertanyaan: Pengaktifan log mesin Redis di kluster Amazon ElastiCache for Redis akan memberikan wawasan tentang peristiwa yang memengaruhi kondisi dan performa kluster. Log mesin Redis menyediakan data langsung dari mesin Redis yang tidak tersedia melalui mekanisme peristiwa ElastiCache. Melalui pengamatan yang cermat terhadap peristiwa ElastiCache (lihat OE-1 di atas) dan log mesin Redis, urutan peristiwa dapat ditentukan saat memecahkan masalah dari perspektif layanan ElastiCache dan perspektif mesin Redis.

- [Wajib] Pastikan fungsionalitas logging mesin Redis diaktifkan, yang tersedia pada ElastiCache for Redis 6.2 dan yang lebih baru. Hal ini dapat dilakukan selama pembuatan kluster atau dengan memodifikasi kluster setelah pembuatan.
- Tentukan apakah Log Amazon CloudWatch atau Amazon Kinesis Data Firehose adalah target yang tepat untuk log mesin Redis.
- Pilih log target yang sesuai dalam CloudWatch atau Kinesis Data Firehose untuk mempertahankan log. Jika Anda memiliki beberapa kluster, pertimbangkan log target yang berbeda untuk setiap kluster karena ini akan membantu mengisolasi data saat pemecahan masalah.

[Sumber Daya]:

- Pengiriman log: [Pengiriman log](#)
- Destinasi logging: [Log Amazon CloudWatch](#)
- Pengenalan Log Amazon CloudWatch: [Apa itu Log Amazon CloudWatch?](#)
- Pengenalan Amazon Kinesis Data Firehose: [Apa Itu Amazon Kinesis Data Firehose?](#)
- [Terbaik] Jika menggunakan Log Amazon CloudWatch, pertimbangkan untuk memanfaatkan Wawasan Log Amazon CloudWatch untuk mengkueri log mesin Redis untuk informasi penting.

Sebagai contoh, buat kueri terhadap grup Log CloudWatch yang berisi log mesin Redis yang akan menampilkan peristiwa dengan LogLevel 'WARNING', seperti:

```
fields @timestamp, LogLevel, Message
| sort @timestamp desc
| filter LogLevel = "WARNING"
```

[Sumber Daya]: [Menganalisis data log dengan Wawasan Log CloudWatch](#)

Pilar Keamanan Lensa Amazon ElastiCache Well-Architected

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 istimewa, melindungi sistem, dan menetapkan kontrol untuk mendeteksi peristiwa keamanan.

Topik

- [SEC 1: Langkah apa yang Anda ambil dalam mengontrol akses resmi ke data ElastiCache?](#)
- [SEC 2: Apakah aplikasi Anda memerlukan otorisasi tambahan untuk ElastiCache terhadap dan di atas kontrol berbasis jaringan?](#)
- [SEC 3: Apakah ada risiko bahwa perintah dapat dijalankan secara tidak sengaja yang menyebabkan kehilangan atau kegagalan data?](#)
- [SEC 4: Bagaimana Anda memastikan enkripsi data saat dalam penyimpanan dengan ElastiCache](#)
- [SEC 5: Bagaimana Anda mengenkripsi data saat dalam pengiriman dengan ElastiCache?](#)
- [SEC 6: Bagaimana Anda membatasi akses untuk bidang kontrol sumber daya?](#)
- [SEC 7: Bagaimana Anda mendeteksi dan menanggapi peristiwa keamanan?](#)

SEC 1: Langkah apa yang Anda ambil dalam mengontrol akses resmi ke data ElastiCache?

Pertanyaan pengenalan: Semua kluster ElastiCache dirancang untuk diakses dari instans Amazon Elastic Compute Cloud dalam VPC, fungsi nirserver (AWS Lambda), atau kontainer (Amazon Elastic Container Service). Skenario yang paling sering ditemui adalah mengakses kluster ElastiCache dari instans Amazon Elastic Compute Cloud di Amazon Virtual Private Cloud yang sama (Amazon Virtual Private Cloud). Sebelum Anda dapat menghubungkan ke kluster dari instans Amazon EC2, Anda harus memberikan otorisasi pada instans Amazon EC2 untuk mengakses kluster tersebut. Untuk mengakses kluster ElastiCache yang berjalan di VPC, Anda perlu memberikan izin masuk jaringan ke kluster tersebut.

Pertanyaan terkait manfaat: Masuknya jaringan ke dalam kluster dikendalikan melalui grup keamanan VPC. Grup keamanan bertindak sebagai firewall virtual untuk instans Amazon EC2 Anda untuk mengontrol lalu lintas masuk dan ke luar. Aturan ke dalam mengontrol lalu lintas yang masuk ke instans Anda, dan aturan ke luar mengontrol lalu lintas yang ke luar dari instans Anda. Dalam kasus ElastiCache, saat meluncurkan kluster, diperlukan pengaitan grup keamanan. Hal ini memastikan bahwa aturan lalu lintas masuk dan keluar berlaku untuk semua simpul yang membentuk kluster.

Selain itu, ElastiCache dikonfigurasi untuk melakukan deployment pada subnet pribadi secara eksklusif sehingga hanya dapat diakses melalui jaringan pribadi VPC.

- [Diperlukan] Grup keamanan yang dikaitkan dengan klaster Anda mengontrol jaringan masuk dan akses 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, konfigurasi aturan masuk pada grup keamanan yang menentukan rentang/alamat IP sumber, lalu lintas jenis TCP, dan port untuk klaster ElastiCache Anda (misalnya port default 6379 untuk ElastiCache for Redis). Meskipun mengizinkan serangkaian sumber masuk yang sangat luas mungkin dilakukan, seperti semua sumber daya dalam VPC (0.0.0.0/0), sebaiknya definisikan aturan masuk sedetail mungkin seperti hanya mengotorisasi akses masuk ke klien Redis yang berjalan di instans Amazon EC2 yang dikaitkan dengan grup keamanan tertentu.

[Referensi]:

- [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](#)
- [Diperlukan] Kebijakan AWS Identity and Access Management dapat ditetapkan ke fungsi AWS Lambda yang memungkinkannya mengakses data ElastiCache. Untuk mengaktifkan fitur ini, buat peran eksekusi IAM dengan izin `AWSLambdaVPCAccessExecutionRole`, lalu tetapkan peran ke fungsi AWS Lambda tersebut.

[Referensi]: Mengonfigurasi fungsi Lambda untuk mengakses Amazon ElastiCache di Amazon VPC: [Tutorial: Mengonfigurasi fungsi Lambda untuk mengakses Amazon ElastiCache di VPC Amazon](#)

SEC 2: Apakah aplikasi Anda memerlukan otorisasi tambahan untuk ElastiCache terhadap dan di atas kontrol berbasis jaringan?

Pertanyaan pengenalan: Dalam skenario di mana perlu untuk membatasi atau mengontrol akses ke klaster ElastiCache for Redis pada tingkat klien individu, sebaiknya autentikasi melalui perintah ElastiCache for Redis AUTH. Token autentikasi ElastiCache for Redis, dengan manajemen pengguna dan grup pengguna opsional, memungkinkan ElastiCache for Redis untuk mensyaratkan kata sandi sebelum mengizinkan klien menjalankan perintah dan kunci akses, sehingga meningkatkan keamanan bidang data.

Pertanyaan terkait manfaat: Untuk membantu menjaga keamanan data Anda, ElastiCache for Redis menyediakan mekanisme untuk mencegah akses yang tidak sah ke data Anda. Hal ini termasuk menegakkan ROLE-Based Access Control (RBAC) AUTH, atau token AUTH (kata sandi) yang digunakan oleh klien untuk terhubung ke ElastiCache sebelum melakukan perintah yang diizinkan.

- [Terbaik] Untuk ElastiCache for Redis 6.x dan yang lebih tinggi, tentukan kontrol autentikasi 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 saat dalam pengiriman harus diaktifkan. Pastikan Anda menggunakan klien Redis yang mendukung TLS untuk dapat memanfaatkan RBAC.

[Referensi]:

- [Menerapkan RBAC ke Grup Replikasi untuk ElastiCache for Redis](#)
 - [Menentukan Izin Menggunakan String Akses](#)
 - [ACL](#)
 - [Versi ElastiCache for Redis yang Didukung](#)
- [Terbaik] Untuk ElastiCache for Redis versi sebelum 6.x, selain mengatur token/kata sandi yang kuat dan mempertahankan kebijakan kata sandi yang ketat untuk ElastiCache for Redis AUTH, praktik terbaiknya adalah merotasi kata sandi/token. ElastiCache dapat mengelola hingga dua (2) token autentikasi pada waktu tertentu. Anda juga dapat mengubah klaster agar secara eksplisit memerlukan penggunaan token autentikasi.

[Referensi]: [Mengubah token AUTH pada klaster ElastiCache for Redis](#)

SEC 3: Apakah ada risiko bahwa perintah dapat dijalankan secara tidak sengaja yang menyebabkan kehilangan atau kegagalan data?

Pertanyaan pengenalan: Ada sejumlah perintah Redis yang dapat berdampak buruk pada operasi jika dijalankan secara tidak sengaja atau oleh pihak yang tidak bertanggung jawab. Perintah ini dapat memiliki konsekuensi yang tidak diinginkan dari perspektif kinerja dan keamanan data. Misalnya pengembang 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.

Pertanyaan terkait manfaat: Dimulai dengan ElastiCache for Redis 5.0.3 di ElastiCache, 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.

- [Diperlukan]

[Referensi]:

- [ElastiCache for Redis versi 5.0.3 \(usang, gunakan versi 5.0.6\)](#)
- [Perubahan parameter Redis 5.0.3](#)
- [Keamanan Redis](#)

SEC 4: Bagaimana Anda memastikan enkripsi data saat dalam penyimpanan dengan ElastiCache

Pertanyaan pengenalan: Meskipun ElastiCache for Redis adalah penyimpanan data dalam memori, dimungkinkan untuk mengenkripsi data apa pun yang mungkin disimpan (pada penyimpanan) sebagai bagian dari operasi standar klaster. 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.

Pertanyaan terkait manfaat: ElastiCache for Redis menyediakan enkripsi saat dalam penyimpanan opsional untuk meningkatkan keamanan data.

- [Diperlukan] Enkripsi saat dalam penyimpanan dapat diaktifkan pada klaster ElastiCache (grup replikasi) hanya pada saat pembuatannya. Klaster yang sudah ada tidak dapat diubah agar mulai mengenkripsi data saat dalam penyimpanan. Secara default, ElastiCache akan menyediakan dan mengelola kunci yang digunakan dalam enkripsi saat dalam penyimpanan.

[Referensi]:

- [Persyaratan Enkripsi Saat dalam Penyimpanan](#)
- [Mengaktifkan Enkripsi Saat dalam Penyimpanan](#)
- [Terbaik] Manfaatkan jenis instans Amazon EC2 yang mengenkripsi data saat berada di memori (seperti M6g atau R6g). Jika memungkinkan, pertimbangkan untuk mengelola kunci Anda sendiri untuk enkripsi saat dalam penyimpanan. Untuk lingkungan keamanan data yang lebih ketat, AWS Key Management Service (KMS) dapat digunakan untuk mengelola sendiri Customer Master Keys (CMK). Melalui integrasi ElastiCache dengan AWS Key Management Service, Anda

dapat membuat, memiliki, dan mengelola kunci yang digunakan untuk enkripsi data saat dalam penyimpanan untuk kluster ElastiCache for Redis Anda.

[Referensi]:

- [Mengggunakan kunci yang dikelola pelanggan dari AWS Key Management Service](#)
- [Layanan Pengelolaan Kunci AWS](#)
- [Konsep KMS AWS](#)

SEC 5: Bagaimana Anda mengenkripsi data saat dalam pengiriman dengan ElastiCache?

Pertanyaan pengenalan: Adalah persyaratan umum untuk memitigasi data disusupi saat dalam pengiriman. Hal ini mewakili data dalam komponen sistem terdistribusi, serta antara klien aplikasi dan simpul kluster. ElastiCache for Redis mendukung persyaratan ini dengan memungkinkan enkripsi data saat dalam pengiriman antara klien dan kluster, dan antara simpul kluster itu sendiri. Jenis instans dalam keluarga M6g dan R6g juga memiliki fitur enkripsi dalam memori yang selalu aktif.

Pertanyaan terkait manfaat: Enkripsi saat dalam pengiriman Amazon ElastiCache adalah fitur opsional yang memungkinkan Anda untuk meningkatkan keamanan data Anda pada titik yang paling rentan, saat data sedang dikirim dari satu lokasi ke lokasi lain.

- [Diperlukan] Enkripsi saat dalam pengiriman hanya dapat diaktifkan pada kluster ElastiCache for Redis (grup replikasi) pada pembuatan. Perlu diketahui bahwa karena pemrosesan tambahan yang diperlukan untuk mengenkripsi/mendekripsi data, penerapan enkripsi saat dalam pengiriman akan memiliki beberapa dampak terhadap kinerja. Untuk memahami dampaknya, sebaiknya bandingkan beban kerja Anda sebelum dan sesudah mengaktifkan enkripsi saat dalam pengiriman.

[Referensi]:

- [Gambaran umum enkripsi saat dalam pengiriman](#)

SEC 6: Bagaimana Anda membatasi akses untuk bidang kontrol sumber daya?

Pertanyaan pengenalan: Kebijakan IAM dan ARN memungkinkan kontrol akses terperinci untuk ElastiCache for Redis, memungkinkan kontrol yang lebih ketat untuk mengelola pembuatan, modifikasi, dan penghapusan kluster ElastiCache for Redis.

Pertanyaan terkait manfaat: Pengelolaan sumber daya Amazon ElastiCache, seperti grup replikasi, simpul, dan lainnya dapat dibatasi ke akun AWS yang memiliki izin khusus berdasarkan kebijakan IAM, sehingga meningkatkan keamanan dan keandalan sumber daya.

- [Diperlukan] Kelola akses ke sumber daya Amazon ElastiCache dengan menetapkan kebijakan AWS Identity and Access Management khusus kepada pengguna AWS. Hal ini memungkinkan kontrol yang lebih baik atas akun dan tindakan yang dapat dilakukannya pada kluster.

[Referensi]:

- [Gambaran umum cara mengelola izin akses untuk sumber daya ElastiCache](#)
- [Mengggunakan kebijakan berbasis identitas \(kebijakan IAM\) untuk Amazon ElastiCache](#)

SEC 7: Bagaimana Anda mendeteksi dan menanggapi peristiwa keamanan?

Pertanyaan pengenalan: ElastiCache, saat di-deploy dengan mengaktifkan RBAC, mengekspor metrik CloudWatch untuk memberi tahu pengguna tentang peristiwa keamanan. Metrik ini membantu mengidentifikasi upaya autentikasi yang gagal, kunci akses, atau menjalankan perintah yang tidak diizinkan untuk menghubungkan pengguna RBAC.

Selain itu, sumber daya produk dan layanan AWS membantu mengamankan beban kerja Anda secara keseluruhan dengan mengotomatiskan deployment dan mencatat log semua tindakan dan modifikasi untuk peninjauan/audit nanti.

Pertanyaan terkait manfaat: 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.

- [Diperlukan] Pelajari Metrik CloudWatch yang diterbitkan yang berkaitan dengan kegagalan autentikasi dan otorisasi RBAC.
 - AuthenticationFailures = Upaya untuk mengautentikasi ke Redis gagal
 - KeyAuthorizationFailures = Upaya gagal oleh pengguna untuk mengakses kunci tanpa izin
 - CommandAuthorizationFailures = Upaya gagal oleh pengguna untuk menjalankan perintah tanpa izin

[Referensi]:

- [Metrik untuk Redis](#)

- [Terbaik] Sebaiknya siapkan peringatan dan notifikasi pada metrik ini dan respons sesuai kebutuhan.

[Referensi]:

- [Menggunakan alarm Amazon CloudWatch](#)

- [Terbaik] Gunakan perintah Redis ACL LOG untuk mengumpulkan detail lebih lanjut

[Referensi]:

- [ACL LOG](#)

- [Terbaik] Pelajari kemampuan produk dan layanan AWS yang berkaitan dengan pemantauan, pencatatan log, dan analisis deployment serta peristiwa ElastiCache

[Referensi]:

- [Membuat log panggilan API Amazon ElastiCache dengan CloudTrail AWS](#)
- [ElastiCache-redis-cluster-automatic-backup-check](#)
- [Memantau Penggunaan dengan Metrik CloudWatch](#)

Pilar Keandalan Lensa Amazon ElastiCache yang Dirancang dengan Baik

Topik

- [REL 1: Bagaimana Anda mendukung deployment arsitektur ketersediaan tinggi \(HA\)?](#)
- [REL 2: Bagaimana Anda memenuhi Sasaran Titik Pemulihan \(RPO\) dengan ElastiCache?](#)
- [REL 3: Bagaimana Anda mendukung persyaratan pemulihan bencana \(DR\)?](#)
- [REL 4: Bagaimana Anda merencanakan failover secara efektif?](#)
- [REL 5: Apakah komponen ElastiCache Anda dirancang untuk melakukan penskalaan?](#)

REL 1: Bagaimana Anda mendukung deployment arsitektur ketersediaan tinggi (HA)?

Pertanyaan pengenalan: Memahami arsitektur ketersediaan tinggi Amazon ElastiCache akan memungkinkan Anda beroperasi dalam kondisi yang andal selama peristiwa ketersediaan.

Pertanyaan terkait manfaat: Merancang kluster ElastiCache Anda agar tahan terhadap kegagalan memastikan ketersediaan yang lebih tinggi untuk deployment ElastiCache Anda.

- [Diperlukan] Tentukan tingkat keandalan yang Anda butuhkan untuk kluster ElastiCache Anda. Beban kerja yang berbeda memiliki standar ketahanan yang berbeda, dari beban kerja sementara

hingga beban kerja penting. Tentukan kebutuhan untuk setiap jenis lingkungan tempat Anda beroperasi seperti dev/tes dan produksi.

Mesin caching: Memcached vs ElastiCache for Redis

1. Memcached tidak menyediakan mekanisme replikasi apa pun dan digunakan terutama untuk beban kerja sementara.
 2. ElastiCache for Redis menawarkan fitur HA yang dibahas di bawah ini
- [Terbaik] Untuk beban kerja yang membutuhkan HA, gunakan ElastiCache for Redis dalam mode kluster dengan minimal dua replika per serpihan, bahkan untuk beban kerja dengan persyaratan throughput kecil yang hanya membutuhkan satu serpihan.
1. Untuk mode kluster aktif, Multi-AZ diaktifkan secara otomatis.

Multi-AZ meminimalkan periode nonaktif 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 dengan serpihan, memiliki minimal tiga serpihan memberikan pemulihan yang lebih cepat selama peristiwa failover karena Protokol Kluster Redis memerlukan ketersediaan sebagian besar simpul primer untuk mencapai kuorum.
3. Siapkan dua replika atau lebih di seluruh Ketersediaan.

Memiliki dua replika memberikan peningkatan skalabilitas baca dan juga ketersediaan baca dalam skenario di mana satu replika sedang menjalani pemeliharaan.

4. Gunakan tipe simpul berbasis Graviton2 (simpul default di sebagian besar wilayah).

Amazon ElastiCache for Redis telah menambahkan kinerja yang dioptimalkan pada simpul ini. Hasilnya, Anda mendapatkan kinerja replikasi dan sinkronisasi yang lebih baik, sehingga meningkatkan ketersediaan secara keseluruhan.

5. Pemantauan dan ukuran yang tepat untuk menangani puncak lalu lintas yang diantisipasi: saat menangani beban berat, mesin ElastiCache for Redis mungkin menjadi tidak responsif, yang memengaruhi ketersediaan. `BytesUsedForCache` dan `DatabaseMemoryUsagePercentage` merupakan indikator yang baik dari penggunaan memori Anda, sedangkan `ReplicationLag` merupakan indikator kesehatan replikasi berdasarkan tingkat operasi tulis. Anda dapat menggunakan metrik ini untuk memicu penskalaan kluster.
6. Pastikan ketahanan sisi klien dengan melakukan pengujian dengan [API Failover sebelum peristiwa failover di lingkungan produksi](#).

[Referensi]:

- [Mengonfigurasi Amazon ElastiCache for Redis untuk ketersediaan yang lebih tinggi](#)
- [Ketersediaan tinggi menggunakan grup replikasi](#)

REL 2: Bagaimana Anda memenuhi Sasaran Titik Pemulihan (RPO) dengan ElastiCache?

Pertanyaan pengenalan: Memahami RPO beban kerja untuk membantu menentukan keputusan tentang strategi pencadangan dan pemulihan ElastiCache.

Pertanyaann terkait manfaat: Dengan menerapkan strategi RPO, Anda dapat meningkatkan kelangsungan bisnis jika terjadi skenario pemulihan bencana. Merancang kebijakan pencadangan dan pemulihan dapat membantu Anda memenuhi Sasaran Titik Pemulihan (RPO) untuk data ElastiCache Anda. ElastiCache for Redis menawarkan kemampuan snapshot yang disimpan di Amazon S3, bersama dengan kebijakan retensi yang dapat dikonfigurasi. Snapshot ini diambil selama masa 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.

- [Diperlukan] Memahami dan mendokumentasikan RPO deployment ElastiCache Anda.
 - Perlu diketahui bahwa Memcached tidak menawarkan proses pencadangan apa pun.
 - Tinjau kemampuan fitur Pencadangan dan Pemulihan ElastiCache.
- [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 kinerja klaster.
- [Baik] Manfaatkan fitur pencadangan terjadwal ElastiCache untuk mencadangkan data Anda secara berkala selama periode yang ditentukan.
 - Tes secara berkala pemulihan dari cadangan Anda.
- [Referensi]:

- [Redis](#)
- [Pencadangan dan Pemulihan untuk ElastiCache for Redis](#)
- [Membuat pencadangan manual](#)
- [Menjadwalkan pencadangan otomatis](#)
- [Pencadangan dan Pemulihan Kluster Redis ElastiCache](#)

REL 3: Bagaimana Anda mendukung persyaratan pemulihan bencana (DR)?

Pertanyaan pengenalan: Pemulihan bencana merupakan aspek penting dari setiap tindakan perencanaan beban kerja. ElastiCache for Redis menawarkan beberapa opsi untuk menerapkan pemulihan bencana berdasarkan persyaratan ketahanan beban kerja. Dengan Penyimpanan Data Global Amazon ElastiCache for Redis, Anda dapat menulis ke kluster ElastiCache for Redis di satu wilayah dan memiliki data yang tersedia untuk dibaca dari dua kluster replika lintas wilayah lainnya. Hal ini memungkinkan pembacaan latensi rendah dan pemulihan bencana di seluruh wilayah.

Pertanyaan terkait manfaat: Memahami dan merencanakan berbagai skenario bencana dapat memastikan kelangsungan bisnis. Strategi DR harus seimbang dengan biaya, dampak kinerja, dan potensi kehilangan data.

- [Diperlukan] Kembangkan dan dokumentasikan strategi DR untuk semua komponen ElastiCache Anda berdasarkan persyaratan beban kerja. ElastiCache unik karena beberapa kasus penggunaan bersifat sepenuhnya sementara dan tidak memerlukan strategi DR apa pun. Namun, beberapa kasus penggunaan lain berada di ujung spektrum yang berlawanan dan memerlukan strategi DR yang sangat solid. 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 melakukan deployment dengan Mode Kluster diaktifkan dalam arsitektur Multi-AZ, dengan minimal 3 AZ yang tersedia.
- 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.
- [Referensi]:
 - [Memitigasi Kegagalan](#)
 - [Replikasi di seluruh Wilayah AWS menggunakan penyimpanan data global](#)
 - [Memulihkan dari cadangan dengan opsi perubahan ukuran klaster](#)
 - [Meminimalkan periode nonaktif di ElastiCache for Redis dengan Multi-AZ](#)

REL 4: Bagaimana Anda merencanakan failover secara efektif?

Pertanyaan pengenalan: Mengaktifkan Multi-AZ dengan failover otomatis adalah praktik terbaik ElastiCache. Dalam kasus tertentu, ElastiCache for Redis mengganti simpul primer sebagai bagian dari operasi layanan. Contohnya termasuk peristiwa pemeliharaan yang direncanakan dan peristiwa yang tidak diharapkan seperti kasus kegagalan simpul atau masalah zona ketersediaan. Failover yang berhasil bergantung pada ElastiCache dan konfigurasi pustaka klien Anda.

Pertanyaan terkait manfaat: Mengikuti praktik terbaik untuk failover ElastiCache bersama dengan pustaka klien ElastiCache for Redis terkait membantu Anda meminimalkan potensi periode nonaktif selama peristiwa failover.

- [Diperlukan] Untuk mode klaster nonaktif, gunakan waktu tunggu sehingga klien dapat mendeteksi jika perlu memutuskan koneksi dari simpul primer lama dan terhubung kembali ke simpul utama baru, menggunakan alamat IP titik akhir primer yang diperbarui. Untuk mode klaster aktif, pustaka klien bertanggung jawab untuk mendeteksi perubahan dalam topologi klaster yang mendasarinya. Tindakan ini paling sering dilakukan dengan pengaturan konfigurasi di pustaka klien ElastiCache for Redis, yang juga memungkinkan Anda untuk mengonfigurasi frekuensi dan metode penyegaran. Setiap pustaka klien menawarkan pengaturannya sendiri dan detail lebih lanjut tersedia dalam dokumentasi yang sesuai.

[Referensi]:

- [Meminimalkan periode nonaktif di ElastiCache for Redis dengan Multi-AZ](#)
- Tinjau praktik terbaik pustaka klien ElastiCache for Redis.
- [Diperlukan] Failover yang berhasil bergantung pada lingkungan replikasi yang sehat antara simpul primer dan replika. Tinjau dan pahami sifat asinkron replikasi Redis, serta metrik CloudWatch yang tersedia untuk melaporkan jeda replikasi antara simpul primer dan replika. Untuk kasus penggunaan yang membutuhkan keamanan data yang lebih kuat, manfaatkan perintah Redis

WAIT untuk memaksa replika mengonfirmasi operasi tulis sebelum merespons klien yang terhubung.

[Referensi]:

- [Metrik untuk Redis](#)
- [Praktik terbaik pemantauan dengan Amazon ElastiCache for Redis menggunakan Amazon CloudWatch](#)
- [Terbaik] Validasi respons aplikasi Anda selama failover secara berkala menggunakan ElastiCache Test Failover API.

[Referensi]:

- [Menguji Failover Otomatis ke Replika Baca di Amazon ElastiCache for Redis](#)
- [Menguji failover otomatis](#)

REL 5: Apakah komponen ElastiCache Anda dirancang untuk melakukan penskalaan?

Pertanyaan pengenalan: Dengan memahami kemampuan penskalaan dan topologi deployment yang tersedia, komponen ElastiCache Anda dapat menyesuaikan dari waktu ke waktu untuk memenuhi persyaratan beban kerja yang berubah. ElastiCache menawarkan 4 cara penskalaan: ke dalam/ke luar (horizontal) serta naik/turun (vertikal).

pertanyaan terkait manfaat: Mengikuti praktik terbaik untuk deployment ElastiCache memberikan fleksibilitas penskalaan terbesar, serta memenuhi prinsip Well Architected dalam penskalaan secara horizontal untuk meminimalkan dampak kegagalan.

- [Diperlukan] Memahami perbedaan antara topologi Mode Kluster Aktif dan Mode Kluster Nonaktif. Dalam hampir semua kasus, disarankan untuk men-deploy dengan mode kluster aktif karena memungkinkan skalabilitas yang lebih besar dari waktu ke waktu. Komponen mode kluster nonaktif memiliki kemampuan terbatas untuk menskalakan secara horizontal dengan menambahkan replika baca.
- [Diperlukan] 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 (meningkatkan skala)

- [Terbaik] Deploy komponen ElastiCache Anda dengan mode klaster aktif, dengan bias terhadap lebih banyak simpul yang lebih kecil daripada lebih sedikit simpul yang lebih besar. Opsi ini secara efektif membatasi radius ledakan kegagalan simpul.
- [Terbaik] Sertakan replika di klaster Anda untuk meningkatkan respons selama peristiwa penskalaan
- [Baik] Untuk mode klaster nonaktif, manfaatkan replika baca untuk meningkatkan kapasitas baca secara keseluruhan. ElastiCache memiliki dukungan hingga 5 replika baca dalam mode klaster nonaktif, serta penskalaan vertikal.
- [Referensi]:
 - [Menskalakan klaster ElastiCache for Redis](#)
 - [Menaikkan skala secara online](#)
 - [Menskalakan klaster ElastiCache for Memcached](#)

Pilar Efisiensi Kinerja Lensa Amazon ElastiCache yang Dirancang dengan Baik

Pilar efisiensi kinerja 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 kinerja, dan pembuatan keputusan berdasarkan informasi untuk mempertahankan efisiensi seiring dengan berkembangnya kebutuhan bisnis.

Topik

- [PE 1: Bagaimana cara memantau kinerja klaster Amazon ElastiCache Anda?](#)
- [PE 2: Bagaimana cara mendistribusikan pekerjaan di seluruh Simpul klaster ElastiCache Anda?](#)
- [PE 3: Untuk beban kerja caching, bagaimana cara melacak dan melaporkan efektivitas dan kinerja cache Anda?](#)
- [PE 4: Bagaimana beban kerja Anda mengoptimalkan penggunaan sumber daya dan koneksi jaringan?](#)
- [PE 5: Bagaimana cara mengelola penghapusan kunci dan/atau pengosongan?](#)
- [PE 6: Bagaimana Anda memodelkan dan berinteraksi dengan data di ElastiCache?](#)
- [PE 7: Bagaimana cara log perintah yang berjalan lambat di klaster Amazon ElastiCache Anda?](#)
- [PE8: Bagaimana Penskalaan Otomatis membantu meningkatkan kinerja klaster ElastiCache?](#)

PE 1: Bagaimana cara memantau kinerja kluster Amazon ElastiCache Anda?

Pengenalan 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 kluster Anda.

Manfaat tingkat pertanyaan: Memahami metrik yang terkait dengan kluster Anda dapat membantu memandu teknik pengoptimalan yang dapat mengurangi latensi dan peningkatan throughput.

- [Wajib] Pengujian kinerja dasar menggunakan subset beban kerja Anda.
 - Anda harus memantau kinerja beban kerja aktual menggunakan mekanisme seperti pengujian beban.
 - Pantau metrik CloudWatch saat menjalankan pengujian ini untuk mendapatkan pemahaman tentang metrik yang tersedia, dan untuk menetapkan garis dasar kinerja.
- [Terbaik] Untuk beban kerja ElastiCache for Redis, ganti nama perintah yang mahal secara komputasi, seperti KEYS, untuk membatasi kemampuan pengguna menjalankan perintah pemblokiran pada kluster produksi.
 - Beban kerja ElastiCache for Redis yang menjalankan mesin 6.x, dapat memanfaatkan kontrol akses berbasis peran untuk membatasi perintah tertentu. Akses ke perintah dapat dikontrol dengan membuat Pengguna dan Grup Pengguna dengan Konsol atau CLI AWS, dan mengaitkan Grup Pengguna ke kluster ElastiCache for Redis. Di Redis 6, ketika RBAC diaktifkan, kita dapat menggunakan “- @dangerous” dan itu akan menolak perintah mahal seperti KEYS, MONITOR, SORT, dll. untuk pengguna tersebut.
 - Untuk mesin versi 5.x, ganti nama perintah menggunakan parameter `rename-commands` pada grup parameter kluster Amazon ElastiCache for Redis.
- [Lebih baik] Analisis kueri lambat dan cari teknik pengoptimalan.
 - Untuk beban kerja ElastiCache for Redis, pelajari lebih lanjut kueri Anda dengan menganalisis Log Lambat. Misalnya, Anda dapat menggunakan perintah berikut, `redis-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 struktur data ElastiCache for Redis. Sebagai contoh, untuk pencarian rentang gaya numerik, aplikasi dapat mengimplementasikan indeks numerik sederhana dengan Set Berurut. Mengelola indeks ini dapat mengurangi pemindaian yang dilakukan pada set data, dan mengembalikan data dengan efisiensi kinerja yang lebih besar.

- Untuk beban kerja ElastiCache for Redis, `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]:
 - [Memantau penggunaan dengan Metrik CloudWatch](#)
 - [Memantau penggunaan dengan Metrik CloudWatch](#)
 - [Menggunakan alarm Amazon CloudWatch](#)
 - [Parameter khusus Redis](#)
 - [SLOWLOG](#)
 - [Tolok ukur Redis](#)

PE 2: Bagaimana cara mendistribusikan pekerjaan di seluruh Simpul kluster ElastiCache Anda?

Pengenalan tingkat pertanyaan: Cara aplikasi Anda tersambung ke simpul Amazon ElastiCache dapat memengaruhi kinerja dan skalabilitas kluster.

Manfaat tingkat pertanyaan: Memanfaatkan simpul yang tersedia di kluster dengan tepat akan memastikan bahwa pekerjaan terdistribusi ke seluruh sumber daya yang tersedia. Teknik-teknik berikut juga membantu menghindari sumber daya yang menganggur.

- [Wajib] Minta klien tersambung ke titik akhir ElastiCache yang tepat.
 - Amazon ElastiCache for Redis mengimplementasikan titik akhir yang berbeda berdasarkan mode kluster yang digunakan. Untuk mode kluster diaktifkan, ElastiCache akan menyediakan titik akhir konfigurasi. Untuk mode cluster dinonaktifkan, ElastiCache menyediakan titik akhir primer, biasanya digunakan untuk menulis, dan titik akhir pembaca untuk menyeimbangkan pembacaan di seluruh replika. Menerapkan titik akhir ini dengan benar akan menghasilkan kinerja yang lebih baik, dan operasi penskalaan yang lebih mudah. Hindari menghubungkan ke titik akhir simpul individu kecuali ada persyaratan khusus untuk melakukannya.
 - Untuk kluster Memcached multisimpul, ElastiCache menyediakan titik akhir konfigurasi yang memungkinkan Penemuan Otomatis. Disarankan untuk menggunakan 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 mendukung hashing yang konsisten dan cara menerapkannya. Anda dapat menemukan informasi selengkapnya tentang penerapan fitur-fitur ini [di sini](#).

- [Lebih baik] Terapkan strategi untuk mengidentifikasi dan memulihkan kunci sibuk dalam beban kerja Anda.
- Pertimbangkan dampak struktur data Redis multi-dimensi seperti daftar, aliran, set, dll. Struktur data tersebut disimpan dalam Kunci Redis tunggal, yang berada pada satu simpul. 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.
- kunci sibuk dalam beban kerja dapat memengaruhi kinerja simpul yang digunakan. Untuk beban kerja ElastiCache for Redis, Anda dapat mendeteksi kunci sibuk menggunakan `redis-cli --hotkeys` jika kebijakan LFU max-memory diberlakukan.
- Pertimbangkan untuk mereplikasi kunci sibuk di beberapa simpul untuk mendistribusikan akses ke simpul secara lebih merata. Pendekatan ini mengharuskan klien untuk menulis ke beberapa simpul primer (simpul Redis itu sendiri tidak akan menyediakan fungsi ini) dan untuk mempertahankan daftar nama kunci yang dibaca, selain nama kunci asli.
- ElastiCache for Redis versi 6 mendukung [caching sisi klien](#) yang dibantu server. Hal ini memungkinkan aplikasi untuk menunggu perubahan pada kunci sebelum membuat jaringan memanggil kembali ke ElastiCache.
- [Sumber Daya]:
 - [Mengonfigurasi Amazon ElastiCache for Redis untuk ketersediaan yang lebih tinggi](#)
 - [Menemukan titik akhir koneksi](#)
 - [Praktik terbaik penyeimbangan beban](#)
 - [Caching sisi klien di Redis](#)

PE 3: Untuk beban kerja caching, bagaimana cara melacak dan melaporkan efektivitas dan kinerja cache Anda?

Pengenalan tingkat pertanyaan: Caching adalah beban kerja yang umum ditemui di ElastiCache dan penting bagi Anda untuk memahami cara mengelola efektivitas dan kinerja cache Anda.

Manfaat tingkat pertanyaan: Aplikasi Anda mungkin menunjukkan tanda-tanda kinerja yang lambat. Kemampuan Anda untuk menggunakan metrik khusus cache untuk menginformasikan keputusan Anda tentang cara meningkatkan kinerja 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 hit cache'. Rasio hit cache ditentukan oleh total hit kunci dibagi dengan total hit dan gagal. Semakin dekat ke 1 rasionya, semakin efektif cache Anda. Rasio hit cache yang rendah disebabkan oleh volume cache yang gagal. Kegagalan cache terjadi ketika kunci yang diminta tidak ditemukan di cache. Kunci tidak ada dalam cache karena telah dikosongkan atau dihapus, telah kedaluwarsa, atau tidak pernah ada. Pahami mengapa kunci tidak ada dalam cache dan kembangkan strategi yang tepat untuk menyimpan kunci dalam cache.

[Sumber Daya]:

- [Wajib] Ukur dan kumpulkan kinerja cache aplikasi Anda bersamaan dengan nilai latensi dan pemanfaatan CPU untuk memahami apakah Anda perlu melakukan penyesuaian pada waktu tayang atau komponen aplikasi lainnya. ElastiCache menyediakan satu set metrik CloudWatch untuk latensi agregat untuk setiap struktur data. Metrik latensi ini dihitung menggunakan statistik `commandstats` dari perintah `INFO ElastiCache for Redis` dan tidak menyertakan jaringan dan waktu I/O. Hanya ini waktu yang dihabiskan oleh ElastiCache for Redis untuk memproses operasi.

[Sumber Daya]:

- [Praktik terbaik pemantauan dengan Amazon ElastiCache for Redis menggunakan Amazon CloudWatch](#)
- [Terbaik] Pilih strategi caching yang tepat untuk kebutuhan Anda. Rasio hit cache yang rendah disebabkan oleh volume cache yang gagal. Jika beban kerja Anda dirancang untuk memiliki volume kegagalan cache yang rendah (seperti komunikasi waktu nyata), tindakan terbaik adalah melakukan tinjauan strategi caching dan menerapkan resolusi yang paling tepat untuk beban kerja Anda, seperti instrumentasi kueri untuk mengukur memori dan kinerja. Strategi aktual yang Anda gunakan 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 Pembuatan Cache](#)
- [Praktik Terbaik Caching](#)
- [Kinerja pada Skala dengan Whitepaper Amazon ElastiCache](#)

PE 4: Bagaimana beban kerja Anda mengoptimalkan penggunaan sumber daya dan koneksi jaringan?

Pengenalan tingkat pertanyaan: ElastiCache for Redis dan Memcached didukung oleh banyak klien aplikasi, dan implementasinya dapat bervariasi. Anda perlu memahami jaringan dan manajemen koneksi di tempat untuk menganalisis potensi dampak kinerja.

Manfaat tingkat pertanyaan: Penggunaan sumber daya jaringan yang efisien dapat meningkatkan efisiensi kinerja kluster Anda. Rekomendasi berikut dapat mengurangi tuntutan jaringan, dan meningkatkan latensi dan throughput kluster.

- [Wajib] Kelola koneksi secara aktif ke kluster ElastiCache Anda.
 - Pengumpulan koneksi dalam aplikasi mengurangi jumlah overhead pada kluster yang dibuat dengan membuka dan menutup koneksi. Pantau perilaku koneksi di Amazon CloudWatch menggunakan `CurConnections` dan `NewConnections`.
 - Hindari kebocoran koneksi dengan menutup koneksi klien dengan benar sesuai keperluan. Strategi manajemen koneksi termasuk menutup koneksi yang tidak digunakan dengan benar, dan pengaturan batas waktu koneksi.
 - Untuk beban kerja Memcached, ada jumlah memori yang dapat dikonfigurasi yang disediakan 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 mendekompres data.
 - Kompresi juga mengurangi jumlah memori yang dikonsumsi oleh kunci
 - Berdasarkan kebutuhan aplikasi Anda, pertimbangkan pertukaran antara rasio kompresi dan kecepatan kompresi.
- [Sumber Daya]:
 - [Amazon ElastiCache for Redis - Penyimpanan Data Global](#)
 - [Parameter spesifik Memcached](#)
 - [Amazon ElastiCache for Redis 5.0.3 meningkatkan penanganan I/O untuk meningkatkan kinerja](#)
 - [Mengonfigurasi Amazon ElastiCache for Redis untuk ketersediaan yang lebih tinggi](#)

PE 5: Bagaimana cara mengelola penghapusan kunci dan/atau pengosongan?

Pengenalan tingkat pertanyaan: Beban kerja memiliki persyaratan yang berbeda, dan perilaku yang diharapkan saat simpul kluster mendekati batas konsumsi memori. Amazon ElastiCache for Redis memiliki kebijakan yang berbeda untuk menangani situasi tersebut.

Manfaat tingkat pertanyaan: Manajemen memori yang tersedia yang tepat, dan pemahaman tentang kebijakan pengosongan akan membantu memastikan kesadaran perilaku kluster ketika batas memori instans terlampaui.

- [Wajib] Instrumen akses data untuk mengevaluasi kebijakan mana yang akan diterapkan. Identifikasi kebijakan `max-memory` yang sesuai untuk mengontrol apakah dan bagaimana pengosongan dilakukan di kluster.
 - Pengosongan terjadi ketika memori maksimum pada kluster dikonsumsi dan kebijakan diberlakukan untuk memungkinkan pengosongan. Perilaku kluster dalam situasi ini tergantung pada kebijakan pengosongan yang ditentukan. Kebijakan ini dapat dikelola menggunakan `maxmemory-policy` pada grup parameter kluster ElastiCache for Redis.
 - Kebijakan default `volatile-lru` membebaskan memori dengan mengosongkan kunci dengan set waktu kedaluwarsa (nilai TTL). Kebijakan yang paling jarang digunakan (LFU) dan yang paling lama tidak digunakan (LRU) menghapus kunci berdasarkan penggunaan.
 - Untuk beban kerja Memcached, ada kebijakan LRU default yang mengendalikan pengosongan di setiap simpul. Jumlah pengosongan di kluster Amazon ElastiCache Anda dapat dipantau menggunakan metrik Pengosongan di Amazon CloudWatch.
- [Lebih Baik] Lakukan standarisasi perilaku penghapusan untuk mengontrol dampak kinerja pada kluster Anda untuk menghindari kemacetan kinerja yang tidak terduga.
 - Untuk beban kerja ElastiCache for Redis, saat secara eksplisit menghapus kunci dari kluster, `UNLINK` seperti `DEL`: itu menghapus kunci yang ditentukan. Namun, perintah melakukan klaim ulang memori aktual di thread yang berbeda, sehingga tidak memblokir, sementara `DEL` memblokir. Penghapusan yang sebenarnya akan terjadi nanti secara asinkron.
 - Untuk beban kerja ElastiCache for Redis 6.x, perilaku perintah `DEL` dapat dimodifikasi dalam grup parameter menggunakan parameter `lazyfree-lazy-user-del`.
- [Sumber Daya]:
 - [Mengonfigurasi parameter mesin menggunakan grup parameter](#)
 - [UNLINK](#)
 - [Manajemen Keuangan Cloud dengan AWS](#)

PE 6: Bagaimana Anda memodelkan dan berinteraksi dengan data di 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 ElastiCache for Redis yang tersedia dan pastikan Anda menggunakan struktur data yang paling tepat untuk kebutuhan Anda.

Manfaat tingkat pertanyaan: Pemodelan data di ElastiCache memiliki beberapa lapisan, termasuk kasus penggunaan aplikasi, jenis data, dan hubungan antar elemen data. Selain itu, setiap jenis data dan perintah ElastiCache for Redis memiliki kinerja khusus mereka sendiri yang terdokumentasi dengan baik.

- [Terbaik] Praktik terbaiknya adalah mengurangi penempatan data yang tidak disengaja. Gunakan konvensi penamaan yang meminimalkan tumpang tindihnya nama kunci. Penamaan konvensional struktur data Anda menggunakan metode hierarkis seperti: APPNAME : CONTEXT : ID, dan juga ORDER-APP : CUSTOMER : 123.

[Sumber Daya]:

- [Penamaan kunci](#)
- [Terbaik] Perintah ElastiCache for Redis memiliki kompleksitas waktu yang ditentukan oleh notasi “Big O”. Kompleksitas waktu perintah ini adalah representasi algoritmik/matematis 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 thread tunggal ElastiCache for Redis, volume besar operasi kompleksitas waktu yang tinggi akan menghasilkan kinerja yang lebih rendah dan potensi waktu habis saat operasi.

[Sumber Daya]:

- [Perintah](#)
- [Terbaik] Gunakan API untuk mendapatkan visibilitas GUI ke dalam model data di kluster Anda.

[Sumber Daya]:

- [Redis Commander](#)
- [Redis Browser](#)
- [Redsmin](#)

PE 7: Bagaimana cara log perintah yang berjalan lambat di kluster Amazon ElastiCache Anda?

Pengenalan tingkat pertanyaan: Manfaat penyetelan kinerja 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 kinerja optimal mesin. Amazon ElastiCache for Redis 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 menyediakan peristiwa notifikasi untuk perintah lambat dapat membantu analisis kinerja terperinci dan dapat digunakan untuk memicu peristiwa otomatis.

- [Wajib] Amazon ElastiCache for Redis menjalankan mesin versi 6.0 atau yang lebih baru, grup parameter yang dikonfigurasi dengan benar dan pencatatan SLOWLOG diaktifkan di kluster.
 - Parameter yang diperlukan hanya tersedia ketika kompatibilitas versi mesin diatur ke Redis versi 6.0 atau yang lebih tinggi.
 - Pencatatan log SLOWLOG terjadi ketika waktu eksekusi server dari suatu perintah membutuhkan waktu lebih lama dari nilai yang ditentukan. Perilaku kluster tergantung pada parameter Grup Parameter terkait yaitu `slowlog-log-slower-than` dan `slowlog-max-len`.
 - Perubahan akan diterapkan segera.
- [Terbaik] Manfaatkan kemampuan CloudWatch atau Kinesis Data Firehose.
 - Gunakan kemampuan pemfilteran dan alarm CloudWatch, Wawasan Log CloudWatch, dan Amazon Simple Notification Services untuk mencapai pemantauan performa dan notifikasi peristiwa.
 - Gunakan kemampuan streaming Kinesis Data Firehose untuk mengarsipkan log SLOWLOG ke penyimpanan permanen atau untuk memicu penyetelan parameter kluster otomatis.
 - Tentukan apakah JSON atau format TEXT biasa yang paling sesuai dengan kebutuhan Anda.
 - Berikan izin IAM untuk menerbitkan ke CloudWatch atau Kinesis Data Firehose.
- [Lebih baik] Konfigurasi `slowlog-log-slower-than` ke nilai selain default.
 - Parameter ini menentukan berapa lama perintah dapat dijalankan dalam mesin Redis sebelum dicatat sebagai perintah berjalan lambat. Nilai defaultnya 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 untuk jumlah perintah yang berjalan lambat yang ditangkap dalam memori Redis pada waktu tertentu. Nilai 0 secara efektif menonaktifkan penangkapan. Semakin tinggi nilainya, semakin banyak entri yang akan disimpan dalam memori, mengurangi kemungkinan informasi penting dihapus 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 `redis-cli` melalui perintah `SLOWLOG`, pertimbangkan untuk meningkatkan nilai ini. Itu memungkinkan lebih banyak perintah untuk tetap disimpan di memori Redis.

Jika Anda memancarkan data `SLOWLOG` ke CloudWatch Logs atau Kinesis Data Firehose, data akan dipertahankan dan dapat dianalisis di luar sistem ElastiCache, mengurangi kebutuhan untuk menyimpan sejumlah besar perintah yang berjalan lambat di memori Redis.

- [Sumber Daya]:
 - [Bagaimana cara mengaktifkan log Slow Redis di kluster cache ElastiCache for Redis?](#)
 - [Pengiriman log](#)
 - [Parameter khusus Redis](#)
 - <https://aws.amazon.com/cloudwatch/> Amazon CloudWatch
 - [Amazon Kinesis Data Firehose](#)

PE8: Bagaimana Penskalaan Otomatis membantu meningkatkan kinerja kluster ElastiCache?

Pengenalan tingkat pertanyaan: Dengan menerapkan fitur penskalaan otomatis Redis, komponen ElastiCache Anda dapat menyesuaikan dari waktu ke waktu untuk menambah atau mengurangi serpihan 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. Penskalaan Otomatis ElastiCache for Redis terus memantau pemanfaatan CPU/memori Anda untuk memastikan kluster Anda beroperasi pada tingkat kinerja yang Anda inginkan.

- [Wajib] Saat meluncurkan kluster untuk ElastiCache for Redis:
 1. Pastikan mode Kluster diaktifkan
 2. Pastikan instans berasal dari keluarga jenis dan ukuran tertentu yang mendukung penskalaan otomatis
 3. Pastikan kluster tidak berjalan di penyimpanan data Global, Outposts atau Zona Lokal

[Sumber Daya]:

- [Penskalaan kluster di Redis \(Mode Cluster Diaktifkan\)](#)
- [Menggunakan Penskalaan Otomatis dengan serpihan](#)
- [Menggunakan Penskalaan Otomatis dengan replika](#)
- [Terbaik] Identifikasi apakah beban kerja Anda berat baca atau berat tulis untuk menentukan kebijakan penskalaan. Untuk kinerja terbaik, gunakan hanya satu metrik pelacakan. Disarankan untuk menghindari beberapa kebijakan untuk setiap dimensi, karena kebijakan penskalaan otomatis akan menskalakan keluar saat target tercapai, namun menskalakan masuk hanya jika semua kebijakan pelacakan target siap untuk diskalakan.

[Sumber Daya]:

- [Kebijakan Penskalaan Otomatis](#)
- [Menetapkan kebijakan penskalaan](#)
- [Terbaik] Memantau kinerja dari waktu ke waktu dapat membantu Anda mendeteksi perubahan beban kerja tidak disadari jika dipantau pada titik waktu tertentu. Anda dapat menganalisis metrik CloudWatch yang sesuai untuk pemanfaatan kluster 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]:

- [Memantau penggunaan dengan Metrik CloudWatch](#)
- [Lebih baik] Kami menyarankan untuk menguji aplikasi Anda dengan beban kerja minimum dan maksimum yang diharapkan, guna mengidentifikasi jumlah pasti serpihan/replika yang diperlukan kluster untuk mengembangkan kebijakan penskalaan dan mengurangi masalah ketersediaan.

[Sumber Daya]:

- [Mendaftarkan Target yang Dapat Diskalakan](#)
- [Mendaftarkan Target yang Dapat Diskalakan](#)

Pilar Optimisasi Biaya Lensa Well-Architected Amazon ElastiCache

Pilar optimisasi biaya berfokus 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 memantau biaya yang terkait dengan sumber daya ElastiCache Anda? Bagaimana Anda mengembangkan mekanisme untuk memungkinkan pengguna membuat, mengelola, dan menghapus sumber daya yang dibuat?](#)
- [BIAYA 2: Bagaimana cara Anda menggunakan alat pemantauan berkelanjutan untuk membantu Anda mengoptimalkan biaya yang terkait dengan sumber daya ElastiCache Anda?](#)
- [BIAYA 3: Haruskah Anda menggunakan tipe instans yang mendukung tingkatan data? Apa keuntungan dari tingkatan data? Kapan sebaiknya tidak menggunakan instans tingkatan data?](#)

BIAYA 1: Bagaimana Anda mengidentifikasi dan memantau biaya yang terkait dengan sumber daya ElastiCache Anda? Bagaimana Anda mengembangkan mekanisme untuk memungkinkan pengguna membuat, mengelola, dan menghapus sumber daya yang dibuat?

Pengenalan 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. Dengan memastikan Anda memiliki proses dan alat untuk memantau sumber daya yang dibuat dari pengembangan hingga tahap produksi dan pemensiunan sumber daya, Anda akan dapat mengelola biaya yang terkait dengan ElastiCache.

Manfaat tingkat pertanyaan: Pemantauan berkelanjutan pada semua biaya yang terkait dengan beban kerja Anda memerlukan pemahaman mendalam tentang arsitektur yang menyertakan ElastiCache sebagai salah satu komponennya. Selain itu, Anda harus memiliki rencana manajemen biaya untuk mengumpulkan dan membandingkan data penggunaan dengan anggaran Anda.

- [Wajib] Jalankan Pusat Keunggulan Cloud (CCoE) dengan salah satu piagam pendiriannya untuk menentukan, memantau, dan mengambil tindakan pada metrik seputar penggunaan ElastiCache di organisasi Anda. Jika CCoE ada dan berfungsi, pastikan CCoE mengetahui cara membaca dan melacak biaya yang terkait dengan 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 menerbitkan metrik biaya yang diperbarui secara rutin per bulan seputar penggunaan ElastiCache utama di seluruh data kategori 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 di dalam dan lintas wilayah
 - e. Instans yang berjalan di Amazon Outposts
 2. CCoE terdiri dari berbagai 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](#)
- [Harga Amazon ElastiCache](#)
- [Wajib] Gunakan tag alokasi biaya untuk memantau biaya pada tingkat granularitas yang rendah. Gunakan Manajemen Biaya AWS untuk memvisualisasikan, memahami, dan mengelola biaya AWS dan penggunaan Anda dari waktu ke waktu.
 1. Gunakan tag untuk mengatur sumber daya Anda, dan tag alokasi biaya untuk melacak biaya AWS Anda secara mendetail. Setelah Anda mengaktifkan tag alokasi biaya, AWS menggunakan tag alokasi biaya untuk mengatur biaya sumber daya Anda di laporan alokasi biaya Anda untuk memudahkan Anda mengategorikan dan melacak biaya AWS Anda. AWS menyediakan dua jenis tag alokasi biaya, tag yang dihasilkan AWS dan tag yang ditentukan pengguna. AWS mendefinisikan, membuat, dan menerapkan tag yang dihasilkan AWS 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 tagihan AWS guna mencerminkan struktur biaya Anda sendiri. Ketika Anda menambahkan tag alokasi biaya ke sumber daya Anda di Amazon ElastiCache, Anda 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 tag alokasi biaya AWS](#)
 - [Memantau biaya dengan tag alokasi biaya](#)
 - [AWS Cost Explorer](#)
- [Terbaik] Hubungkan biaya ElastiCache ke metrik yang mencakup 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 serta rata-rata per operasi dan pengguna, skor keterlibatan pengguna, tingkat pengembalian pengguna/minggu, durasi/ pengguna sesi, tingkat pengabaian, laju hit cache, dan kunci yang dilacak

[Sumber Daya]:

- [Memantau penggunaan dengan Metrik CloudWatch](#)
- [Baik]Pertahankan visibilitas arsitektur dan operasional terkini pada metrik dan biaya di seluruh beban kerja yang menggunakan ElastiCache.
 1. Pahami seluruh ekosistem solusi Anda. ElastiCache cenderung menjadi bagian dari ekosistem lengkap layanan AWS 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 serta memiliki kebutuhan dan kemampuan sendiri yang menambah dan/atau membantu Anda mengelola biaya keseluruhan.
 3. Diagram Anda harus mencakup penggunaan komputasi, jaringan, penyimpanan, kebijakan siklus hidup, pengumpulan metrik, serta elemen ElastiCache operasional dan fungsional 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 agar Anda dapat memiliki strategi manajemen biaya yang efektif untuk ElastiCache Anda.

BIAYA 2: Bagaimana cara Anda menggunakan alat pemantauan berkelanjutan untuk membantu Anda mengoptimalkan biaya yang terkait dengan sumber daya ElastiCache Anda?

Pengenalan tingkat pertanyaan: Anda perlu menargetkan keseimbangan yang tepat antara biaya ElastiCache dan metrik performa aplikasi Anda. Amazon CloudWatch menyediakan visibilitas ke dalam metrik operasional utama yang dapat membantu Anda menilai apakah sumber daya ElastiCache Anda terlalu banyak atau kurang dimanfaatkan, dibandingkan dengan kebutuhan Anda. Dari perspektif optimisasi biaya, Anda perlu memahami kapan penyediaan Anda berlebih dan dapat mengembangkan mekanisme yang tepat untuk mengubah ukuran sumber daya ElastiCache Anda sambil memenuhi kebutuhan operasional, ketersediaan, ketahanan, dan performa 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 sumber daya ElastiCache yang terlalu besar untuk jangka waktu yang lama.

- [Wajib] Gunakan CloudWatch untuk memantau kluster ElastiCache 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 get dan cache miss). Metrik ini diukur dan diterbitkan untuk setiap simpul cache dalam interval 60 detik.
 2. Metrik performa ElastiCache (CPUUtilization, EngineUtilization, SwapUsage, CurrConnections, dan Evictions) dapat menunjukkan bahwa Anda perlu menaikkan/menurunkan skala (menggunakan tipe simpul cache yang lebih besar/lebih kecil) atau melakukan penskalaan ke dalam/ke luar (menambahkan lebih banyak/lebih sedikit serpihan). 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]:

- [Memantau penggunaan dengan Metrik CloudWatch](#)
 - [Metrik Apa yang Harus Saya Pantau?](#)
 - [Harga Amazon ElastiCache](#)
- [Wajib] Pahami dan dokumentasikan strategi pencadangan dan implikasi biaya Anda.
 1. Dengan ElastiCache, cadangan disimpan di Amazon S3, yang menyediakan penyimpanan yang durabel. 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 kluster ElastiCache menggunakan simpul sesuai permintaan hingga Anda mengumpulkan data yang cukup untuk memperkirakan kebutuhan instans terpesan. 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 tipe simpul cache baru

BIAYA 3: Haruskah Anda menggunakan tipe instans yang mendukung tingkatan data? Apa keuntungan dari tingkatan data? Kapan sebaiknya tidak menggunakan instans tingkatan data?

Pengenalan 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. Untuk memastikan bahwa jenis instans yang dipilih sudah benar, diperlukan pemeriksaan berkala terhadap waktu idle objek ElastiCache.

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 untuk kluster ElastiCache for Redis.
 2. Hanya tipe instans tertentu yang mendukung tingkatan data.
 3. Hanya ElastiCache for Redis versi 6.2 dan yang lebih baru yang didukung
 4. Item besar tidak ditukar ke SSD. Objek di atas 128 MiB dipertahankan dalam memori.

[Sumber Daya]:

- [Tingkatan data](#)
- [Harga Amazon ElastiCache](#)
- [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 hot terhadap data warm adalah sekitar 20:80.
 2. Kembangkan pelacakan tingkat kluster 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 kluster ElastiCache menggunakan instans non-tingkatan data hingga Anda memiliki pemahaman yang lebih baik tentang kebutuhan Anda.
 2. Analisis pola penggunaan data kluster ElastiCache 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 kluster 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 tipe simpul cache baru.

[Sumber Daya]:

- [OBJECT IDLETIME](#)
- [Harga Amazon ElastiCache](#)

Langkah-langkah pemecahan masalah umum dan praktik terbaik

Topik

- [Masalah koneksi](#)
- [Kesalahan klien Redis](#)
- [Memecahkan masalah latensi tinggi di Tanpa Server ElastiCache](#)
- [Memecahkan masalah pembatasan di Tanpa Server ElastiCache](#)
- [Topik-Topik Terkait](#)

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 [di sini](#).
2. VPC: ElastiCache cache hanya dapat diakses dari dalam VPC. Pastikan bahwa instans EC2 dari mana Anda mengakses cache dan cache dibuat dalam VPC yang sama. ElastiCache Atau, Anda harus mengaktifkan [peering VPC antara](#) VPC tempat instans EC2 Anda 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 instans EC2 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 Redis. Pelajari lebih lanjut tentang cara mengatur akses port [di sini](#).

Kesalahan klien Redis

ElastiCache Tanpa server hanya dapat diakses menggunakan klien Redis yang mendukung protokol mode cluster Redis. Cluster yang dirancang sendiri dapat diakses dari klien Redis dalam mode mana pun, tergantung pada konfigurasi cluster.

Jika Anda mengalami kesalahan Redis di klien Anda, pertimbangkan hal berikut:

1. Mode cluster: Jika Anda mengalami kesalahan atau kesalahan CROSSLOT dengan perintah [SELECT](#) Redis, Anda mungkin mencoba mengakses cache Cluster Mode Enabled dengan klien Redis yang tidak mendukung protokol Redis Cluster. ElastiCache Tanpa server hanya mendukung klien Redis yang mendukung protokol cluster Redis. Jika Anda ingin menggunakan Redis 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 tidak termasuk dalam 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 Redis, 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.

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 Personal Health Anda untuk informasi lebih lanjut. Jika perlu, pertimbangkan untuk membuka kasus dukungan dengan AWS Support.

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 Redis 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 dalam AZ yang sama dengan cache Anda:** Anda dapat mengamati latensi sisi klien yang lebih tinggi jika aplikasi Anda tidak di-deploy di AZ yang 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 disebar di AZ yang sama. 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 Redis 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 Redis, 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:** Saat 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 Redis, distribusi kunci atau permintaan per slot yang tidak merata dapat menghasilkan slot panas yang dapat mengakibatkan latensi tinggi. ElastiCache Tanpa server mendukung hingga 30.000 ECPUS/detik (90.000 ECPU/detik saat menggunakan Baca dari Replika) pada satu slot, dalam beban kerja yang menjalankan perintah SET/GET sederhana. 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 Redis 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 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 Redis, 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 ECPU/detik, dalam beban kerja yang mengeksekusi perintah SET/GET sederhana.
- Baca dari Replika: Jika aplikasi Anda mengizinkannya, pertimbangkan untuk menggunakan fitur “Baca dari Replika”. Sebagian besar klien Redis 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 ECPU/detik pada satu slot, untuk beban kerja dengan perintah SET/GET sederhana.

Topik-Topik Terkait

- [Langkah pemecahan masalah tambahan](#)
- [the section called “Praktik terbaik dan strategi caching”](#)

Langkah pemecahan masalah tambahan

Item berikut harus diverifikasi saat memecahkan masalah konektivitas persisten dengan: ElastiCache

Topik

- [Grup keamanan](#)
- [ACL jaringan](#)

- [Tabel rute](#)
- [Resolusi DNS](#)
- [Mengidentifikasi masalah dengan diagnostik sisi server](#)
- [Validasi konektivitas jaringan](#)
- [Batas terkait jaringan](#)
- [Penggunaan CPU](#)
- [Koneksi yang dihentikan dari sisi server](#)
- [Pemecahan masalah sisi klien untuk instans Amazon EC2](#)
- [Membedah waktu yang dibutuhkan untuk menyelesaikan satu permintaan tunggal](#)

Grup keamanan

Grup Keamanan adalah firewall virtual yang melindungi ElastiCache klien Anda (instans EC2, AWS Lambda fungsi, wadah Amazon ECS, dll.) dan cache. ElastiCache Grup keamanan bersifat stateful, artinya bahwa setelah lalu lintas masuk atau keluar diizinkan, maka tanggapan untuk lalu lintas itu akan secara otomatis mendapat 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 pada saat yang sama 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 ke. ElastiCache Port default adalah 11211 untuk Memcached dan 6379 untuk Redis. 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](#).

ACL jaringan

Daftar Kontrol Akses (ACL) jaringan adalah aturan yang stateless. Lalu lintas harus diizinkan di kedua arah (masuk dan keluar) agar koneksi berhasil. ACL jaringan ditempatkan ke subnet, bukan untuk

sumber daya tertentu. Dimungkinkan untuk memiliki ACL yang sama yang ditugaskan ke ElastiCache dan sumber daya klien, terutama jika mereka berada di subnet yang sama.

Secara default, ACL jaringan mengizinkan semua lalu lintas. 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 mengizinkan lalu lintas Redis adalah:

ACL Jaringan sisi klien:

- Aturan Masuk:
- Nomor aturan: sebaiknya lebih rendah dari semua aturan menolak;
- Jenis: Aturan TCP Khusus;
- 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 Khusus;
- Protokol: TCP
- Rentang Port: 6379
- Sumber: 0.0.0.0/0 (atau subnet cluster. ElastiCache Ingatlah bahwa menggunakan IP tertentu 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 Khusus;
- 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 Khusus;
- Protokol: TCP
- Rentang Port: 1024-65535
- Sumber: 0.0.0.0/0 (atau subnet cluster. ElastiCache Ingatlah bahwa menggunakan IP tertentu dapat menimbulkan masalah jika terjadi failover atau penskalaan cluster)
- Izinkan/Tolak: Izinkan

Untuk informasi selengkapnya, silakan lihat [ACL Jaringan](#).

Tabel rute

Sama seperti ACL Jaringan, 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 VPC, perutean dinamis, atau firewall jaringan, mungkin akan menyulitkan pemecahan masalah. 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, silakan 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 `nslookup`.


```
$ 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:** Redis adalah aplikasi multi-thread. Akan tetapi, 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 Redis, dan penggunaan `CPUUtilization` di semua vCPU. Simpul dengan lebih dari satu vCPU biasanya memiliki nilai yang berbeda untuk `CPUUtilization` dan `EngineCPUUtilization`, di mana nilai yang kedua umumnya 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 di 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, silakan 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 asli. Dengan tanda yang tepat, pustaka akan dapat secara otomatis menemukan topologi klaster, simpul replika, mengaktifkan operasi baca melalui perintah Redis [READONLY](#), dan mengirimkan permintaan baca ke replika.

- Jumlah koneksi yang meningkat:
- `CurConnections` dan `NewConnections`: `CurConnection` 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 jabat tangan 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 pengumpulan koneksi atau koneksi yang persisten. Dengan pengumpulan koneksi, angka `curConnections` tidak akan memiliki variasi besar, dan `NewConnections` seharusnya menjadi serendah mungkin. Redis memberikan kinerja yang optimal dengan `curConnections` yang kecil. Menjaga `curConnection` 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 penggunaan 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 mengenai batasan.

Mensimulasi beban dari aplikasi akan memberikan hasil yang lebih akurat. Akan tetapi, alat tolok ukur dapat memberikan gambaran yang baik tentang batasan.

- Untuk kasus di mana permintaan didominasi oleh proses baca, menggunakan replika untuk operasi baca akan mengurangi beban pada simpul primer. Jika kasus penggunaan didominasi proses 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 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 dilayani pada satu thread tunggal, yang berarti bahwa permintaan dilayani 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 outputnya.

Contoh yang terkenal buruk adalah perintah KEYS. Perintah ini menyapu seluruh keyspace untuk mencari pola tertentu dan memblokir pelaksanaan dari perintah lain selama pelaksanaannya. Redis menggunakan notasi “Big O” untuk menggambarkan kompleksitas dari 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 keyspace untuk mencarinya, dan waktu untuk menyelesaikan perintah akan sama.

Alternatif untuk KEYS adalah perintah SCAN. Perintah ini melakukan iterasi atas keypace 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 defaultnya 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 bolak-balik 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 100,000, iterasi tunggal akan diperlukan dan hanya 0,5 milidetik yang akan dihabiskan untuk mengirimnya. Akan tetapi, mesin akan sepenuhnya terblokir untuk operasi lain sampai perintah itu selesai menyapu semua keypace.

Selain KEYS, beberapa perintah lain berpotensi membahayakan jika tidak digunakan dengan benar. Untuk melihat daftar semua perintah dan kompleksitas waktunya masing-masing, kunjungi <https://redis.io/commands>.

Contoh potensi masalah:

- Skrip Lua: Redis menyediakan interpreter Lua tertanam, yang memungkinkan pelaksanaan skrip di sisi server. Skrip Lua pada Redis dijalankan di tingkat mesin dan bersifat atom menurut definisi, yang berarti bahwa tidak ada perintah atau skrip lain akan diizinkan untuk bekerja sementara skrip itu dijalankan. Skrip Lua menyediakan kemungkinan untuk menjalankan beberapa perintah, algoritma pengambilan keputusan, penguraian data, dan lain-lain secara langsung pada mesin Redis. 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 keypace 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 proses tulis apapun selama pelaksanaan

skrip. Perintah [transactions](#) adalah alternatif untuk menjamin konsistensi dari perubahan beberapa kunci yang terkait di Redis. Perintah transaction memungkinkan eksekusi dari suatu blok perintah, memperhatikan perubahan pada kunci yang telah 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, set berurutan, 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. Alternatif untuk DEL adalah UNLINK, yang merupakan perintah asinkron yang tersedia sejak Redis 4. UNLINK seharusnya lebih dipilih daripada DEL jika dimungkinkan. Dimulai ElastiCache untuk Redis 6x, `lazyfree-lazy-user-del` parameter membuat DEL perintah berperilaku seperti UNLINK saat diaktifkan. Untuk informasi selengkapnya, lihat [Perubahan Parameter Redis 6.0](#).
- 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 menyediakan lebih banyak perintah yang bekerja dengan cara yang 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. Akan tetapi, 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 mempengaruhi 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 juga menyediakan perintah yang bekerja pada satu atau beberapa kunci, terlepas dari tipe datanya. ElastiCache untuk

Redis 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 menjadi mahal. 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. Sets, Lists, Sorted Sets, dan Hyperloglogs juga dapat memakan waktu yang lama untuk dimanipulasi tergantung pada ukuran dan perintah yang digunakan. Untuk informasi selengkapnya tentang perintah tersebut, lihat <https://redis.io/commands>.

Validasi konektivitas jaringan

Setelah meninjau konfigurasi jaringan yang berkaitan dengan resolusi DNS, grup keamanan, ACL jaringan, 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. Kunjungi <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/home?#ReachabilityAnalyzer> dan pilih opsi berikut:
 - Jenis Sumber: Pilih instance jika ElastiCache klien Anda berjalan pada instans Amazon EC2 atau Antarmuka Jaringan jika menggunakan layanan lain, seperti AWS Fargate Amazon ECS dengan jaringan awsvpc, dll) AWS Lambda, dan ID sumber daya masing-masing (instans EC2 atau ID ENI);
 - Jenis Tujuan: Pilih Antarmuka Jaringan dan pilih ElastiCache ENI dari daftar.

- Port tujuan: tentukan 6379 untuk Redis atau 11211 ElastiCache 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 pembatasan lain yang dimungkinkan di tingkat sistem operasi.

Di ElastiCache konsol, periksa apakah Enkripsi dalam transit diaktifkan di detail ElastiCache kluster 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 menghasilkan sesuatu yang berbeda, periksa alasan untuk kesalahan itu 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 tanda konfigurasi atau otoritas sertifikasi tepercaya yang tepat;

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 `CurrConnections` metrik pada 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 untuk memproses permintaan koneksi. Sebaiknya mempertimbangkan menerapkan teknik seperti pengumpulan koneksi atau koneksi persisten untuk menangani koneksi itu dengan lebih baik. Jika memungkinkan, lakukan konfigurasi kumpulan 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](#) untuk mengidentifikasi kemungkinan batas jaringan yang terkena pada ElastiCache node:
 - `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 tulis intensif mungkin tidak mampu mengatasi backlog replikasi. Untuk kasus seperti itu, praktik terbaik adalah menaikkan skala (mengubah jenis simpul), menskalakan ke luar (menambahkan serpihan dalam klaster dengan mode klaster diaktifkan), mengurangi jumlah replika, atau meminimalkan jumlah proses 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.
- `NetworkPacketsPerSecondAllowanceExceeded`: 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 jeda replikasi, peningkatan byte yang Digunakan untuk cache, penurunan memori yang dapat dikosongkan, swap tinggi, dan penggunaan CPU. Instans Amazon EC2 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 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` di dalam [grup parameter khusus](#). 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 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`, `hset`, `hsetnx`;
 - **HyperLogLogBasedCmdsLatency:** `pfselftest`, `pfcount`, `pfdebug`, `pfadd`, `pfmerge`;
 - **KeyBasedCmdsLatency:** Perintah yang dapat bertindak atas tipe data yang berbeda: `dump`, `existskeys`, `object`, `pttl`, `randomkey`, `ttdel`, `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:
 - info commandstats: Menyediakan daftar perintah yang dilaksanakan sejak mesin Redis dimulai, jumlah pelaksanaan kumulatifnya, jumlah waktu pelaksanaan, dan rata-rata waktu pelaksanaan per perintah;
 - client list: Menyediakan daftar dari klien yang saat ini tersambung dan informasi yang relevan seperti penggunaan buffer, perintah yang dilaksanakan terakhir, dll;
- Backup dan replikasi: ElastiCache untuk versi Redis 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 proses tulis intensif.

Dimulai dengan ElastiCache Redis 2.8.22, AWS memperkenalkan metode pencadangan dan replikasi tanpa garpu. Metode yang baru dapat menunda proses 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 jendela pencadangan atau metrik SaveInProgress bernilai 1 pada periode tersebut. Sebaiknya menjadwalkan jendela 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 membuat koneksi klien dibuat tanpa batas waktu. Akan tetapi, dalam beberapa kasus penghentian koneksi mungkin diinginkan. Misalnya:

- Bug dalam aplikasi klien dapat menyebabkan koneksi dilupakan dan tetap tersambung dengan keadaan diam. Ini disebut “kebocoran koneksi “ dan konsekuensinya adalah peningkatan yang tetap pada jumlah koneksi tersambung yang diamati pada metrik `CurConnections`. 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 untuk koneksi diam untuk bertahan. Jika klien tidak mengirimkan permintaan apapun dalam periode itu, mesin Redis akan mengakhiri koneksi itu segera setelah koneksi itu mencapai nilai waktu habis. Nilai waktu habis yang kecil dapat mengakibatkan pemutusan koneksi yang tidak perlu dan klien akan perlu menangani ini dengan tepat dan menyambungkan 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 tidak membatasi ukuran buffer keluaran klien biasa. Jika batas `maxmemory` tercapai, mesin akan mencoba untuk melakukan pengosongan item untuk memenuhi penggunaan buffer. Dalam kondisi memori yang sangat rendah, ElastiCache untuk Redis 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 khusus 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 biasa, pilihan berikut mengontrol buffer untuk simpul replika dan klien Pub/Sub (Penerbitan/Berlangganan):

- `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 Batas dari instans EC2 dan sistem operasi harus ditinjau dengan hati-hati sambil melakukan pemecahan masalah konektivitas yang putus-putus atau masalah waktu habis. Beberapa poin penting untuk diamati:

- CPU
 - Penggunaan CPU instans EC2: 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 [instans EC2 burstable](#), pastikan bahwa saldo kredit CPU belum habis. Informasi ini tersedia pada `CPUcreditBalance` CloudWatch metrik.
 - Periode singkat penggunaan CPU yang tinggi dapat menyebabkan batas waktu tanpa mencerminkan pemanfaatan 100 persen. 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 [Tipe Instans Amazon EC2](#).
 - Pada instans dengan ena penggerak Jaringan yang Ditingkatkan, periksa [statistik ena](#) untuk batas waktu 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 PPS dapat dicapai ketika lalu lintas jaringan terdiri dari ribuan atau jutaan permintaan yang sangat kecil per detik. ElastiCache Lalu lintas dapat dioptimalkan untuk memanfaatkan paket jaringan dengan lebih baik melalui saluran pipa 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,sack0K,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,sack0K,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 jabat tangan 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 bekerja seperti yang diharapkan, jika tidak, penyelidikan lebih lanjut mungkin diperlukan. Menggunakan Redis yang sama, perintah PING 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"], 0x7ffffede7cc38 /* 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\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\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 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 socket TCP (745659 hingga 747858), dan waktu 0,4 ms (747858 hingga 748330) dibutuhkan untuk mengirimkan dan menerima respons untuk permintaan.

Keamanan di Amazon ElastiCache

Keamanan cloud di AWS menjadi prioritas tertinggi. Sebagai seorang pelanggan AWS, Anda mendapatkan manfaat dari arsitektur pusat data dan jaringan yang dibangun untuk memenuhi persyaratan keamanan organisasi yang paling sensitif.

Keamanan menjadi tanggung jawab bersama antara 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 layanan–layanan AWS di dalam AWS Cloud. AWS juga memberi Anda layanan yang dapat digunakan dengan aman. Auditor pihak ketiga secara berkala menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari [Program kepatuhan AWS](#). Untuk mempelajari program kepatuhan yang berlaku pada Amazon ElastiCache, lihat [Layanan AWS dalam Cakupan Program Kepatuhan](#).
- Keamanan di cloud – Tanggung jawab Anda ditentukan menurut layanan AWS 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 akan membantu Anda memahami cara menerapkan model tanggung jawab bersama saat Anda menggunakan Amazon ElastiCache. Topik berikut menunjukkan kepada Anda cara mengonfigurasi Amazon ElastiCache untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga akan mempelajari cara menggunakan layanan AWS lain yang dapat membantu Anda memantau dan mengamankan sumber daya Amazon ElastiCache Anda.

Topik

- [Perlindungan data di Amazon ElastiCache](#)
- [Privasi lalu lintas kerja internet](#)
- [Manajemen Identitas dan Akses untuk Amazon ElastiCache](#)
- [Validasi kepatuhan untuk Amazon ElastiCache](#)
- [Ketahanan di Amazon ElastiCache](#)
- [Keamanan infrastruktur di AWS ElastiCache](#)
- [Pembaruan layanan di ElastiCache](#)

Perlindungan data di Amazon ElastiCache

[Model tanggung jawab bersama](#) AWS diterapkan ke perlindungan data di AWS ElastiCache (ElastiCache). Sebagaimana diuraikan dalam model ini, AWS bertanggung jawab untuk melindungi infrastruktur global yang menjalankan AWS Cloud secara keseluruhan. Anda harus bertanggung jawab untuk memelihara kendali atas konten yang di-hosting di infrastruktur ini. Konten ini meliputi konfigurasi keamanan dan tugas-tugas pengelolaan untuk berbagai layanan AWS yang Anda gunakan. Untuk informasi selengkapnya tentang privasi data, lihat [FAQ privasi data](#).

Untuk tujuan perlindungan data, sebaiknya Anda melindungi kredensial akun AWS 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 melakukan komunikasi dengan sumber daya AWS.
- Siapkan API dan logging aktivitas pengguna dengan AWS CloudTrail.
- Gunakan solusi enkripsi AWS, bersama dengan semua kontrol keamanan default dalam layanan AWS.
- 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. Hal ini termasuk ketika Anda menangani ElastiCache atau layanan AWS lainnya menggunakan konsol, API, AWS CLI, atau AWS SDK. Setiap data yang Anda masukkan ke dalam ElastiCache atau layanan lain mungkin akan diambil untuk disertakan 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 yang tidak sah terhadap data Anda di server.

Amazon ElastiCache for Memcached juga menyediakan fitur enkripsi untuk data pada cache yang menjalankan Memcached versi 1.6.12 atau yang lebih baru.

- Enkripsi data bergerak mengenkripsi data Anda setiap kali data bergerak dari satu tempat ke tempat lain, misalnya antara simpul di kluster atau antara cache dan aplikasi Anda.
- Enkripsi data diam mengenkripsi data pada disk Anda selama operasi sinkronisasi dan pencadangan.

Topik

- [Enkripsi bergerak ElastiCache \(TLS\)](#)
- [Enkripsi Diam di ElastiCache](#)

Enkripsi bergerak ElastiCache (TLS)

Untuk membantu menjaga keamanan data Anda, Amazon ElastiCache dan Amazon EC2 menyediakan mekanisme untuk mencegah akses yang tidak sah terhadap data Anda di server. Dengan menyediakan kemampuan enkripsi bergerak, ElastiCache menyediakan alat untuk membantu Anda melindungi data saat bergerak dari satu lokasi ke lokasi lain.

Semua cache nirserver memiliki enkripsi bergerak yang aktif. Untuk kluster yang dirancang sendiri, Anda dapat mengaktifkan enkripsi bergerak pada kluster cache dengan menyetel parameter `TransitEncryptionEnabled` ke `true` (CLI: `--transit-encryption-enabled`) saat Anda membuat kluster cache menggunakan operasi `CreateCacheCluster` (CLI: `create-cache-cluster`).

Topik

- [Gambaran umum enkripsi bergerak](#)
- [Kondisi enkripsi bergerak](#)
- [Praktik terbaik enkripsi bergerak](#)
- [Mengaktifkan enkripsi bergerak](#)
- [Menghubungkan ke simpul dengan enkripsi bergerak yang aktif menggunakan Openssl](#)
- [Membuat klien Memcached TLS menggunakan Java](#)
- [Membuat klien Memcached TLS menggunakan PHP](#)

Gambaran umum enkripsi bergerak

Enkripsi bergerak Amazon ElastiCache adalah fitur opsional yang memungkinkan Anda meningkatkan keamanan data Anda pada titik yang paling rentan—saat data dalam keadaan bergerak 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 melakukan tolok ukur terhadap data Anda dengan dan tanpa enkripsi bergerak untuk menentukan dampak terhadap performa pada kasus penggunaan Anda.

Enkripsi bergerak ElastiCache menerapkan fitur berikut:

- Koneksi klien terenkripsi—koneksi klien ke simpul cache dienkripsi TLS.
- Koneksi server terenkripsi—data yang bergerak antarsimpul dalam kluster dienkripsi.
- Autentikasi server—Klien dapat mengautentikasi bahwa koneksinya dilakukan ke server yang benar.

Kondisi enkripsi bergerak

Batasan berikut pada enkripsi bergerak Amazon ElastiCache harus Anda ingat saat merencanakan implementasi kluster yang dirancang sendiri:

- Enkripsi bergerak didukung pada kluster 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 kluster yang berjalan di Amazon VPC.
- Enkripsi bergerak hanya didukung untuk kluster yang menjalankan jenis simpul berikut.
 - R6g, R5, R4
 - M6g, M5, M4
 - T4g, T3

Untuk informasi selengkapnya, lihat [Tipe simpul yang didukung](#).

- Enkripsi bergerak diaktifkan dengan menetapkan parameter `TransitEncryptionEnabled` ke `true` secara eksplisit.
- Anda dapat mengaktifkan enkripsi bergerak pada kluster hanya saat pembuatan kluster. Anda tidak dapat mengaktifkan dan menonaktifkan enkripsi data bergerak dengan memodifikasi kluster.

- 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, penerapan 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 berbiaya mahal, Anda dapat mengurangi dampak enkripsi bergerak pada performa dengan mempersistensi koneksi TLS Anda.

Mengaktifkan enkripsi bergerak

Untuk mengaktifkan enkripsi bergerak saat membuat kluster Memcached menggunakan Konsol Manajemen AWS, buat pilihan berikut:

- Pilih Memcached sebagai mesin Anda.
- Pilih versi mesin 1.6.12 atau yang lebih baru.
- Di bawah Enkripsi saat transit, pilih Aktifkan.

Untuk proses langkah demi langkah, lihat [Membuat kluster Memcached \(konsol\)](#).

Menghubungkan ke simpul dengan enkripsi bergerak yang aktif menggunakan Openssl

Untuk mengakses data dari simpul ElastiCache for Memcached dengan enkripsi bergerak diaktifkan, Anda perlu menggunakan klien yang berfungsi dengan Lapisan Soket Aman (SSL). Anda juga dapat menggunakan `s_client` Openssl di Amazon Linux dan Amazon Linux 2.

Untuk menggunakan `s_client` Openssl untuk terhubung ke kluster Memcached dengan enkripsi bergerak yang aktif 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 `SSLContext` yang sesuai:

```
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 SSLContext yang sesuai:

```
<?php

/**
 * Sample PHP code to show how to create a TLS Memcached client. In this example we
 * will use the Amazon ElastiCache Auto Discovery 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/mem-ug/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->getLastError(), "\n";
    exit(1);
}
```

Untuk informasi selengkapnya tentang menggunakan klien PHP, lihat [Menginstal ElastiCache Cluster Client untuk PHP](#).

Enkripsi Diam di ElastiCache

Untuk membantu menjaga keamanan data Anda, Amazon ElastiCache dan Amazon S3 menyediakan beberapa cara untuk membatasi akses ke data di dalam cache Anda. Untuk informasi selengkapnya, lihat [Amazon VPC dan keamanan ElastiCache](#) dan [Manajemen Identitas dan Akses untuk Amazon ElastiCache](#).

- Disk selama operasi sinkronisasi dan swap

ElastiCache menawarkan enkripsi diam secara default (dikelola layanan), dan juga kemampuan untuk menggunakan kunci AWS KMS Anda sendiri yang simetris dan dikelola pelanggan di [AWS Key Management Service \(KMS\)](#). 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](#).

Note

Enkripsi default (dikelola layanan) adalah satu-satunya pilihan yang tersedia di Wilayah GovCloud (US).

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

Batasan berikut pada enkripsi diam ElastiCache harus diperhatikan saat Anda merencanakan penerapan enkripsi diam ElastiCache:

- Enkripsi diam didukung hanya pada cache nirserver.
- Opsi untuk menggunakan kunci yang dikelola pelanggan untuk enkripsi diam tidak tersedia di Wilayah AWS GovCloud (us-gov-east-1 dan us-gov-west-1).

Menggunakan kunci yang dikelola pelanggan dari AWS KMS

ElastiCache mendukung kunci AWS KMS (kunci KMS) yang simetris dan dikelola pelanggan untuk enkripsi diam. Kunci KMS yang dikelola pelanggan adalah kunci enkripsi yang Anda buat, miliki, dan kelola di akun AWS Anda. Untuk informasi selengkapnya, lihat [Kunci AWS KMS](#) di Panduan Developer untuk AWS Key Management Service. Kunci harus dibuat di AWS KMS sebelum dapat digunakan dengan ElastiCache.

Untuk mempelajari cara membuat kunci root AWS KMS, lihat [Buat Kunci](#) di Panduan Developer AWS Key Management Service.

ElastiCache memungkinkan Anda berintegrasi dengan AWS KMS. Untuk informasi selengkapnya, lihat [Menggunakan Grant](#) di Panduan Developer AWS Key Management Service. Tidak diperlukan tindakan pelanggan untuk mengaktifkan integrasi Amazon ElastiCache dengan AWS KMS.

Kunci kondisi `kms:ViaService` membatasi penggunaan dari kunci AWS KMS (kunci KMS) untuk meminta dari layanan AWS yang ditentukan. Untuk menggunakan `kms:ViaService` dengan ElastiCache, masukkan kedua nama `ViaService` ke dalam nilai kunci kondisi: `elasticache.AWS_region.amazonaws.com` dan `dax.AWS_region.amazonaws.com`. Untuk informasi selengkapnya, lihat [kms:ViaService](#).

Anda dapat menggunakan [AWS CloudTrail](#) untuk melacak permintaan yang dikirim Amazon ElastiCache ke AWS Key Management Service atas nama Anda. Semua panggilan API ke AWS Key Management Service yang terkait dengan kunci yang dikelola pelanggan memiliki log CloudTrail yang sesuai. Anda juga dapat melihat grant yang dibuat oleh ElastiCache dengan memanggil API KMS [ListGrants](#).

- 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, kunci tidak dapat diubah 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 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 ElastiCache. Cache Amazon ElastiCache terenkripsi tidak mendukung rotasi kunci manual, yang melibatkan pembuatan kunci root baru dan pembaruan referensi apa pun ke kunci lama. Untuk mempelajari lebih lanjut, lihat [Merotasikan kunci AWS KMS](#) di Panduan Developer AWS Key Management Service.
- Mengenkripsi cache ElastiCache menggunakan kunci KMS memerlukan satu grant per cache. Grant ini digunakan sepanjang masa pakai cache.
- Untuk informasi selengkapnya tentang grant dan batas AWS KMS, lihat bagian [Batas](#) di Panduan Developer AWS Key Management Service.

Mengaktifkan Enkripsi Diam

Semua cache nirserver mengaktifkan enkripsi diam.

Anda dapat mengaktifkan enkripsi diam saat Anda membuat cache ElastiCache. Anda dapat melakukannya menggunakan AWS Management Console, AWS CLI, atau API ElastiCache.

Saat membuat cache, Anda dapat memilih salah satu opsi berikut:

- Default – Opsi ini menggunakan enkripsi diam yang dikelola layanan.
- Kunci yang dikelola pelanggan – Opsi ini memungkinkan Anda menyediakan Kunci ID/ARN dari AWS KMS untuk enkripsi diam.

Untuk mempelajari cara membuat kunci root AWS KMS, lihat [Membuat Kunci](#) di Panduan Developer AWS Key Management Service

Daftar Isi

- [Mengaktifkan Enkripsi Diam Menggunakan AWS Management Console](#)

Mengaktifkan Enkripsi Diam Menggunakan AWS Management Console

Mengaktifkan Enkripsi Diam di Kluster Nirserver (Konsol)

Semua cache nirserver mengaktifkan enkripsi diam. Secara default, kunci KMS yang dimiliki AWS digunakan untuk mengenkripsi data. Untuk memilih kunci AWS KMS 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 terkelola pelanggan di bagian Pengaturan kunci enkripsi.
- Pilih kunci di bagian pengaturan kunci AWS KMS.

Lihat Juga

- [Amazon VPC dan keamanan ElastiCache](#)
- [Manajemen Identitas dan Akses untuk Amazon ElastiCache](#)

Privasi lalu lintas kerja internet

Amazon ElastiCache menggunakan teknik berikut untuk mengamankan data cache Anda dan melindunginya dari akses yang tidak sah:

- [Amazon VPC dan keamanan ElastiCache](#) menjelaskan jenis grup keamanan yang Anda butuhkan untuk instalasi Anda.
- [Manajemen Identitas dan Akses untuk Amazon ElastiCache](#) untuk memberikan dan membatasi tindakan pengguna, grup, dan peran.

Amazon VPC dan keamanan ElastiCache

Karena keamanan data itu penting, ElastiCache menyediakan sarana bagi Anda untuk mengontrol siapa yang memiliki akses ke data Anda. Cara Anda mengontrol akses ke data Anda bergantung pada iya atau tidaknya Anda meluncurkan kluster di Amazon Virtual Private Cloud (Amazon VPC) atau Amazon EC2-Classic.

Important

Kami tidak lagi menggunakan Amazon EC2-Classic untuk meluncurkan kluster 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 tradisional. 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 kluster cache ke jaringan virtual, dan mengontrol akses ke kluster cache dengan menggunakan grup keamanan Amazon VPC.

Bagian ini menjelaskan cara mengonfigurasi kluster ElastiCache secara manual di Amazon VPC. Informasi ini ditujukan bagi pengguna yang menginginkan pemahaman yang lebih mendalam tentang cara kerja ElastiCache dan Amazon VPC.

Topik

- [Memahami ElastiCache dan Amazon VPC](#)
- [Pola Akses untuk Mengakses Cache ElastiCache di Amazon VPC](#)
- [Membuat Cloud Privat Virtual \(VPC\)](#)
- [Menghubungkan ke cache yang berjalan di Amazon VPC](#)

Memahami ElastiCache dan Amazon VPC

ElastiCache terintegrasi penuh dengan Amazon Virtual Private Cloud (Amazon VPC). Untuk pengguna ElastiCache, hal ini berarti sebagai berikut:

- Jika akun AWS Anda hanya mendukung platform EC2-VPC, ElastiCache selalu meluncurkan klaster Anda di Amazon VPC.
- Jika Anda baru menggunakan AWS, klaster Anda akan di-deploy ke Amazon VPC. VPC default akan dibuat untuk Anda secara otomatis.
- Jika Anda memiliki VPC default dan tidak menentukan subnet saat meluncurkan sebuah klaster, klaster tersebut akan diluncurkan ke Amazon VPC default Anda.

Untuk informasi selengkapnya, lihat [Mendeteksi Platform Anda yang Didukung dan Apakah Anda Memiliki VPC Default](#).

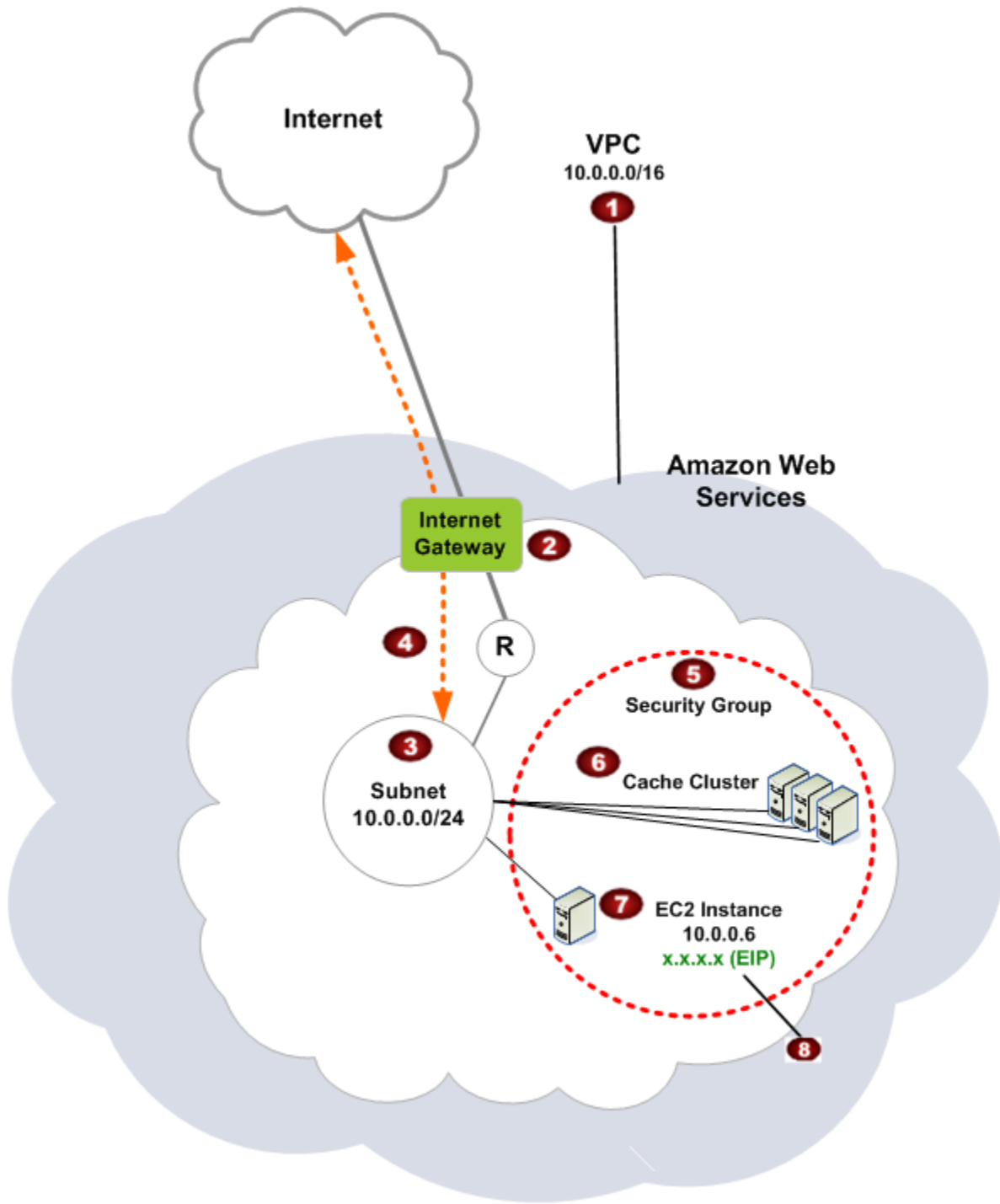
Dengan Amazon Virtual Private Cloud, Anda dapat membuat jaringan virtual di cloud AWS yang sangat menyerupai pusat data tradisional. Anda dapat mengonfigurasi Amazon VPC Anda, termasuk memilih rentang alamat IP, membuat subnet, dan mengonfigurasi tabel rute, gateway jaringan, dan pengaturan keamanan.

Fungsionalitas dasar ElastiCache sama di cloud privat virtual; ElastiCache mengelola peningkatan perangkat lunak, patching, deteksi kegagalan, dan pemulihan baik klaster Anda di-deploy di dalam atau di luar Amazon VPC.

Simpul cache ElastiCache yang di-deploy di luar Amazon VPC ditugaskan alamat IP yang diselesaikan oleh nama titik akhir/DNS. Ini menyediakan konektivitas dari instans Amazon Elastic Compute Cloud (Amazon EC2). Saat Anda meluncurkan klaster ElastiCache ke dalam subnet privat Amazon VPC, setiap simpul cache ditugaskan alamat IP privat dalam subnet tersebut.

Gambaran umum ElastiCache di Amazon VPC

Diagram dan tabel berikut menjelaskan lingkungan Amazon VPC, beserta klaster ElastiCache dan instans Amazon EC2 yang diluncurkan di Amazon VPC.



1

Amazon VPC adalah bagian terisolasi dari Cloud AWS yang ditugaskan blok alamat IP sendiri.

2

Gateway Internet menghubungkan Amazon VPC Anda secara langsung ke Internet dan menyediakan akses ke sumber daya AWS lainnya seperti Amazon Simple Storage Service (Amazon S3) yang berjalan di luar Amazon VPC Anda.

3

Subnet Amazon VPC adalah segmen rentang alamat IP dari Amazon VPC tempat Anda dapat mengisolasi sumber daya AWS 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 mengendalikan lalu lintas masuk dan keluar untuk kluster ElastiCache dan instans Amazon EC2.

6

Anda dapat meluncurkan kluster ElastiCache di subnet. Simpul cache memiliki alamat IP privat dari rentang alamat subnet.

7

Anda juga dapat meluncurkan instans Amazon EC2 dalam subnet. Setiap instans Amazon EC2 memiliki alamat IP privat dari rentang alamat subnet. Instans Amazon EC2 dapat tersambung ke simpul cache apa pun di dalam subnet yang sama.

8

Agar instans Amazon EC2 di Amazon VPC dapat dijangkau dari Internet, Anda perlu menugaskan alamat publik statis yang disebut alamat IP Elastic ke instans tersebut.

Prasyarat

Untuk membuat kluster ElastiCache dalam Amazon VPC, Amazon VPC Anda harus memenuhi persyaratan berikut:

- Amazon VPC harus mengizinkan instans Amazon EC2 tidak terdedikasi. Anda tidak dapat menggunakan ElastiCache di Amazon VPC yang dikonfigurasi untuk penghunian instans khusus.

- Grup subnet cache harus ditentukan untuk Amazon VPC Anda. ElastiCache menggunakan grup subnet cache tersebut untuk memilih subnet dan alamat IP dalam subnet tersebut untuk mengasosiasikannya dengan titik akhir VPC dan simpul cache Anda.
- Blok CIDR untuk setiap subnet harus cukup besar untuk menyediakan alamat IP cadangan untuk ElastiCache untuk digunakan selama aktivitas 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, Amazon VPC memiliki akses langsung ke sumber daya AWS lainnya yang tidak berjalan di Amazon VPC 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 kasus ini, Anda dikenakan biaya bandwidth tambahan jika Anda mengakses sumber daya AWS melalui Internet.

Anda dapat menggunakan grup keamanan Amazon VPC untuk membantu mengamankan kluster ElastiCache dan instans Amazon EC2 di Amazon VPC 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 Perintah Amazon EC2 (perintah Amazon VPC ditemukan di dalam referensi Amazon EC2)
Deskripsi lengkap dari operasi API Amazon VPC, tipe data, dan kesalahan	Referensi API Amazon EC2 (operasi API Amazon VPC ditemukan dalam referensi Amazon EC2)
Informasi untuk administrator jaringan yang perlu membuat konfigurasi gateway di ujung koneksi VPN IPsec opsional Anda	Apa itu Site-to-Site VPN AWS?

Untuk informasi lebih detail tentang Amazon Virtual Private Cloud, lihat [Amazon Virtual Private Cloud](#).

Pola Akses untuk Mengakses Cache ElastiCache di Amazon VPC

Amazon ElastiCache mendukung skenario berikut untuk mengakses cache di Amazon VPC:

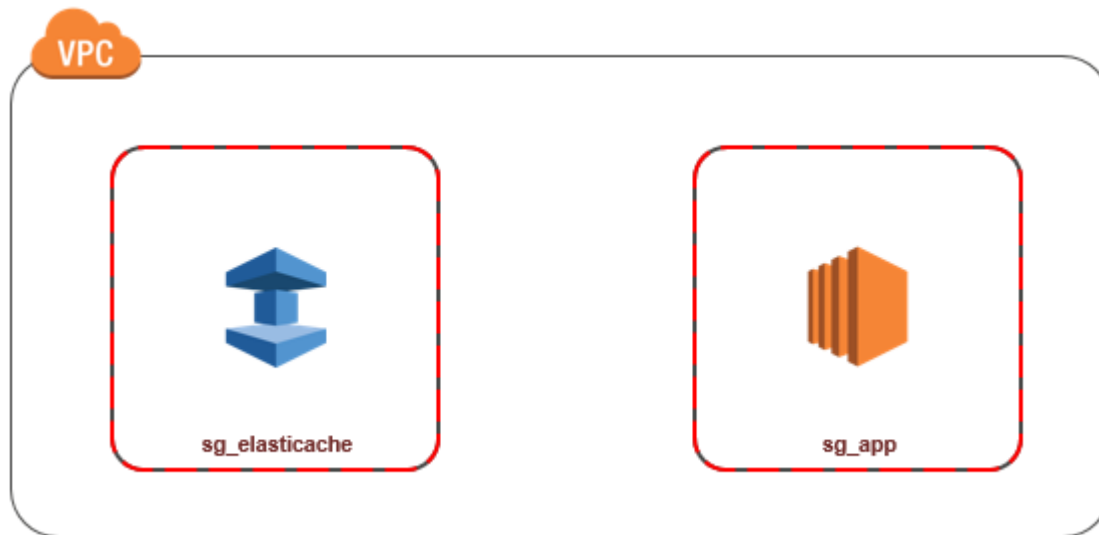
Daftar Isi

- [Mengakses Cache ElastiCache saat berada di Amazon VPC yang sama dengan Instans Amazon EC2](#)
- [Mengakses Cache ElastiCache dan Instans Amazon EC2 yang berada di Amazon VPC yang Berbeda](#)
 - [Mengakses Cache ElastiCache dan Instans Amazon EC2 yang berada di Amazon VPC yang Berbeda di Wilayah yang Sama](#)
 - [Menggunakan Gateway Transit](#)
 - [Mengakses Cache ElastiCache dan Instans Amazon EC2 yang berada di Amazon VPC yang Berbeda di Wilayah yang Berbeda](#)
 - [Menggunakan VPC Transit](#)
- [Mengakses Cache ElastiCache dari Aplikasi yang Berjalan di Pusat Data Pelanggan](#)
 - [Mengakses Cache ElastiCache dari Aplikasi yang Berjalan di Pusat Data Pelanggan Menggunakan Konektivitas VPN](#)
 - [Mengakses Cache ElastiCache dari Aplikasi yang Berjalan di Pusat Data Pelanggan Menggunakan Direct Connect](#)

Mengakses Cache ElastiCache saat berada di Amazon VPC yang sama dengan Instans Amazon EC2

Kasus penggunaan yang paling umum adalah saat aplikasi yang disebarakan pada instans EC2 perlu terhubung ke klaster di VPC yang sama.

Diagram berikut menggambarkan skenario ini



Cara paling sederhana untuk mengelola akses antara instans EC2 dan cache di VPC yang sama adalah dengan melakukan tindakan berikut:

1. Buat grup keamanan VPC untuk cache Anda. Grup keamanan ini dapat digunakan untuk membatasi akses ke cache. Contohnya, Anda dapat membuat aturan khusus 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.

2. Buat grup keamanan VPC untuk instans EC2 Anda (server web dan aplikasi). Jika diperlukan, grup keamanan ini dapat mengizinkan akses ke instans EC2 dari Internet dengan menggunakan tabel perutean VPC. Sebagai contoh, Anda dapat menetapkan aturan pada grup keamanan ini untuk mengizinkan akses TCP ke instans EC2 melalui port 22.
3. Buat aturan kustom di grup keamanan untuk cache Anda yang memungkinkan koneksi dari grup keamanan yang Anda buat untuk instans EC2 Anda. Hal ini akan mengizinkan semua anggota grup keamanan untuk mengakses cache.

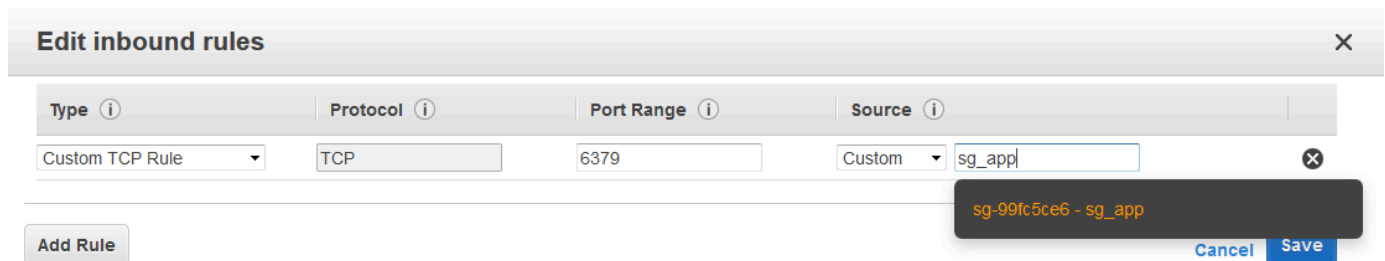
Note

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 Manajemen AWS dan buka konsol Amazon VPC 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. Di bawah 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 Khusus.
 - a. Untuk Rentang Port, tentukan port yang Anda gunakan saat membuat cache.

Port default untuk cache Memcached adalah 11211.
 - b. Pada kotak Sumber, masukkan ID dari grup keamanan. Dari daftar, pilih grup keamanan yang akan Anda gunakan untuk instans Amazon EC2 Anda.
5. Pilih Simpan jika selesai.



Mengakses Cache ElastiCache dan Instans Amazon EC2 yang berada di Amazon VPC yang Berbeda

Jika cache Anda berada dalam VPC yang berbeda dengan instans EC2 yang Anda gunakan untuk mengaksesnya, ada beberapa cara untuk mengakses cache. Jika instans EC2 ada di VPC yang berbeda tetapi di wilayah yang sama, Anda dapat menggunakan peering VPC. Jika cache dan instans EC2 berada di wilayah yang berbeda, Anda dapat membuat konektivitas VPN di antara wilayah.

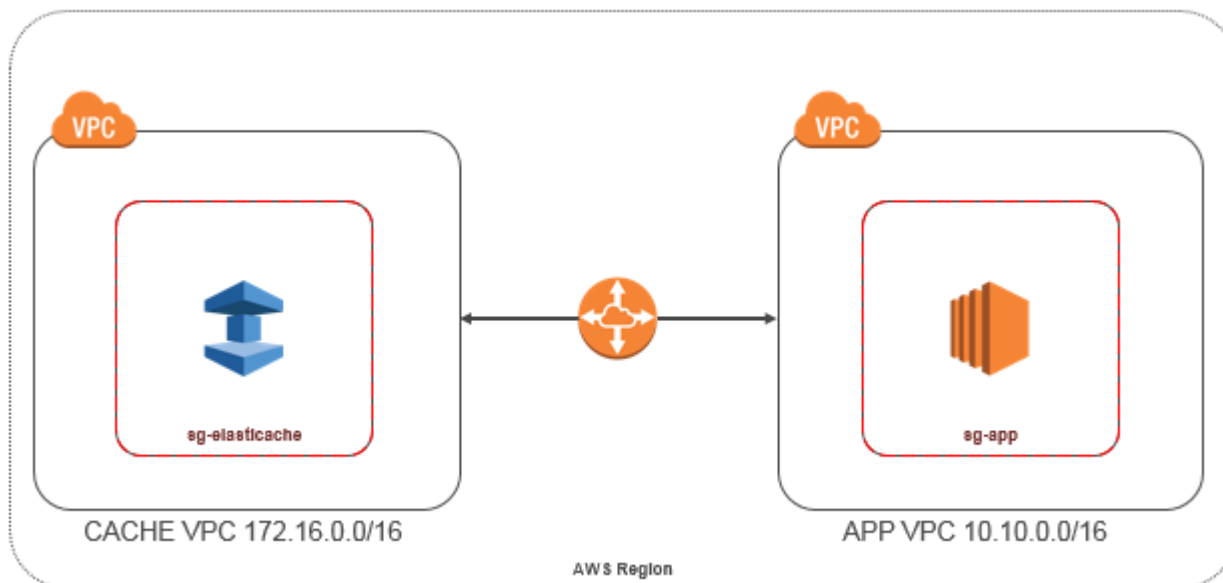
Topik

- [Mengakses Cache ElastiCache dan Instans Amazon EC2 yang berada di Amazon VPC yang Berbeda di Wilayah yang Sama](#)

- [Mengakses Cache ElastiCache dan Instans Amazon EC2 yang berada di Amazon VPC yang Berbeda di Wilayah yang Berbeda](#)

Mengakses Cache ElastiCache dan Instans Amazon EC2 yang berada di Amazon VPC yang Berbeda di Wilayah yang Sama

Diagram berikut menunjukkan pengaksesan cache oleh instans Amazon EC2 di Amazon VPC yang berbeda di wilayah yang sama menggunakan koneksi peering VPC.



Cache yang diakses oleh instans Amazon EC2 di Amazon VPC yang berbeda di Wilayah yang sama - Koneksi Peering VPC

Koneksi peering VPC adalah koneksi jaringan antara dua VPC yang memungkinkan Anda merutekan lalu lintas di antara keduanya menggunakan alamat IP privat. Instans di kedua VPC tersebut dapat berkomunikasi satu sama lain seolah berada di jaringan yang sama. Anda dapat membuat koneksi peering VPC antara Amazon VPC Anda sendiri, atau dengan Amazon VPC di akun AWS lain dalam satu wilayah. Untuk mempelajari selengkapnya peering Amazon VPC, lihat [Dokumentasi VPC](#).

Note

Resolusi nama DNS mungkin gagal untuk VPC yang telah di-peering, tergantung pada konfigurasi yang diterapkan ke VPC ElastiCache. Untuk mengatasi hal ini, kedua VPC

harus mengaktifkan 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 bahwa kedua VPC tidak memiliki rentang IP yang tumpang tindih atau Anda tidak akan melakukan peering.
2. Buat koneksi peering di antara kedua VPC. 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	Destination	Target
172.16.0.0/16	local	10.10.0.0/16	local
10.10.0.0/16	pcx-a894f1c1	0.0.0.0/0	igw-bfdcccd8
		172.16.0.0/16	pcx-a894f1c1

Tabel Perutean VPC

4. Ubah Grup Keamanan cache ElastiCache Anda agar mengizinkan koneksi masuk dari grup keamanan Aplikasi di dalam VPC yang tersambung peering. Untuk informasi selengkapnya, lihat [Grup Keamanan VPC Peer Referensi](#).

Mengakses cache melalui koneksi peering akan dikenakan biaya transfer data tambahan.

Menggunakan Gateway Transit

Gateway transit memungkinkan Anda untuk menyematkan koneksi VPC dan VPN di Wilayah AWS yang sama dan mengarahkan rute lalu lintas di antara keduanya. Gateway transit bekerja di seluruh akun AWS, dan Anda dapat menggunakan AWS Resource Access Manager untuk berbagi transit gateway Anda dengan akun lain. Setelah Anda berbagi gateway transit dengan akun AWS lain, pemilik akun dapat menyematkan VPC mereka ke gateway transit Anda. Pengguna dari kedua akun dapat menghapus lampuran tersebut kapan saja.

Anda dapat mengaktifkan multicast pada gateway transit, lalu membuat domain multicast gateway transit yang mengizinkan lalu lintas multicast dikirim dari sumber multicast Anda untuk anggota grup multicast melalui lampiran VPC yang Anda kaitkan dengan domain.

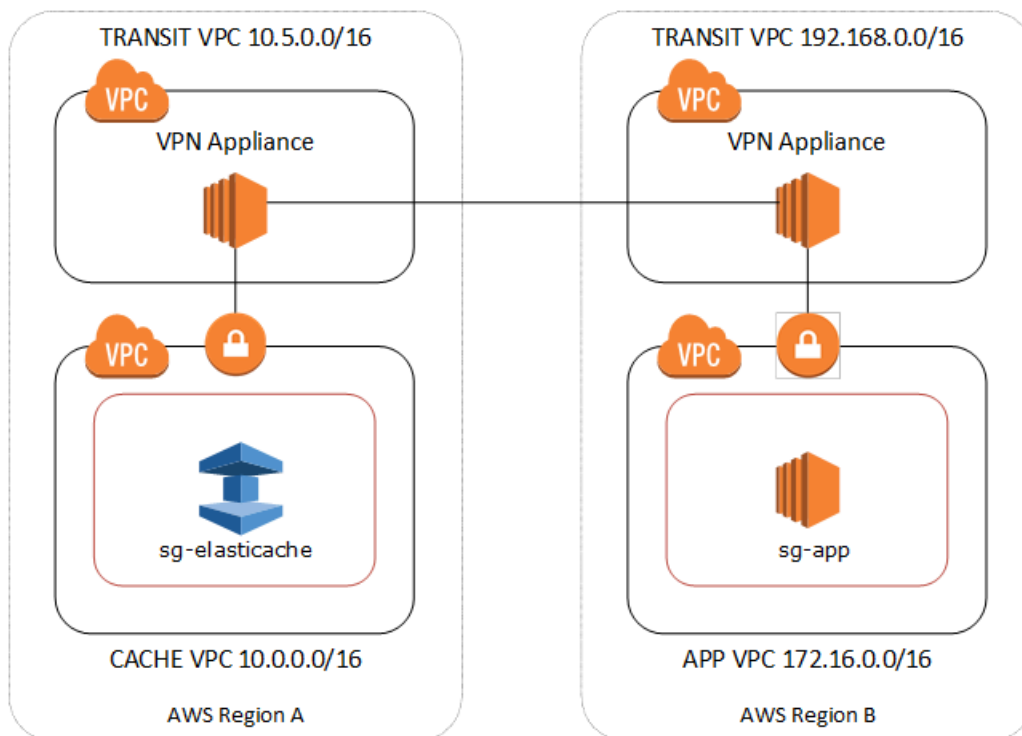
Anda juga dapat membuat lampiran koneksi peering antara gateway transi di Wilayah AWS yang berbeda. Hal ini memungkinkan Anda untuk mengatur rute lalu lintas antara beberapa lampiran gateway transit di Wilayah yang berbeda.

Untuk informasi selengkapnya, silakan lihat [Gateway transit](#).

Mengakses Cache ElastiCache dan Instans Amazon EC2 yang berada di Amazon VPC yang Berbeda di Wilayah yang Berbeda

Menggunakan VPC Transit

Selain menggunakan peering VPC, strategi umum lain untuk menghubungkan beberapa VPC dan jaringan jauh yang berjauhan secara geografis adalah membuat VPC transit yang berfungsi sebagai pusat transit jaringan global. VPC transit menyederhanakan manajemen jaringan dan meminimalkan jumlah koneksi yang diperlukan untuk menghubungkan beberapa VPC dan jaringan jarak jauh. Rancangan ini dapat menghemat waktu dan tenaga serta mengurangi biaya, karena diimplementasikan secara virtual tanpa biaya tradisional untuk membangun kehadiran fisik di hub transit kolokasi atau menyebarkan peralatan jaringan fisik.



Menghubungkan seluruh VPC yang berbeda di wilayah yang berbeda

Setelah Amazon VPC Transit dibuat, aplikasi yang disebarakan di sebuah VPC “kisi” di satu wilayah dapat terhubung ke cache ElastiCache di VPC “kisi” di wilayah lain.

Mengakses cache di VPC yang berbeda di Wilayah AWS yang berbeda

1. Menyebarkan Solusi VPC Transit. Untuk informasi selengkapnya, lihat [AWS Transit Gateway](#).
2. Perbarui tabel perutean VPC di VPC Cache dan Aplikasi untuk merutekan lalu lintas melalui VGW (Gateway Privat Virtual) dan Perangkat VPN. Dalam kasus Perutean Dinamis dengan Protokol Gateway Batas (BGP), rute Anda dapat disebarakan secara otomatis.
3. Ubah Grup Keamanan cache ElastiCache Anda agar mengizinkan koneksi masuk dari rentang IP instans Aplikasi. Perhatikan bahwa Anda tidak akan dapat mereferensikan Grup Keamanan server aplikasi dalam skenario ini.

Mengakses cache di seluruh wilayah akan menimbulkan latensi jaringan dan biaya transfer data lintas wilayah tambahan.

Mengakses Cache ElastiCache dari Aplikasi yang Berjalan di Pusat Data Pelanggan

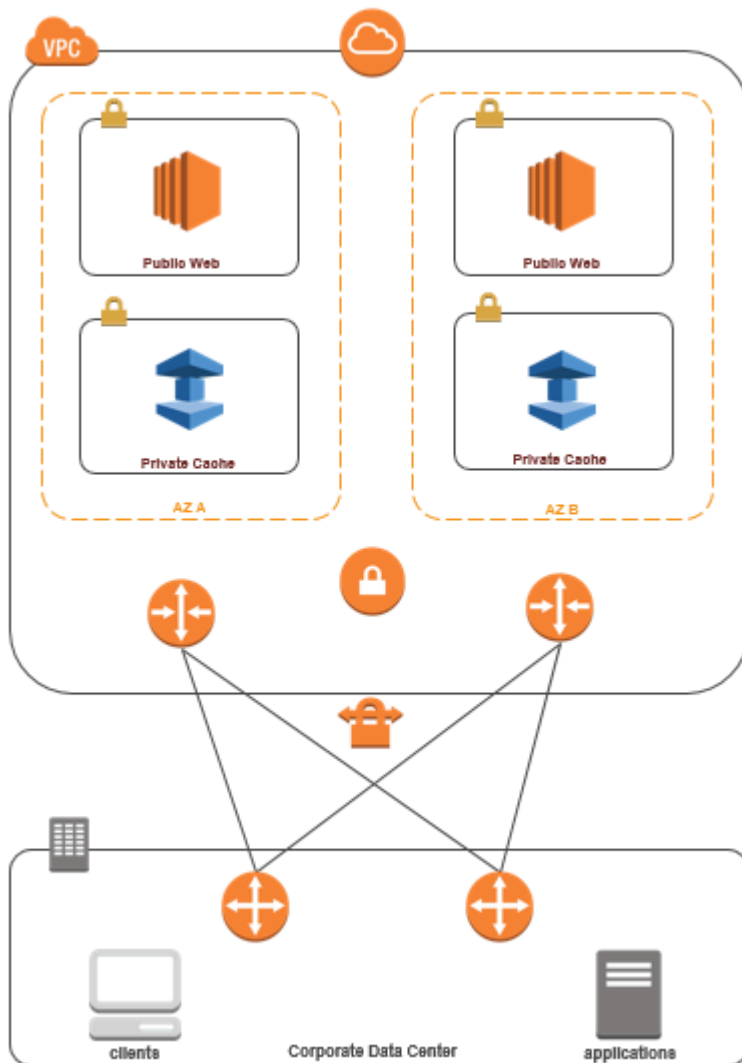
Skenario lain yang mungkin adalah arsitektur Hibrida di mana klien atau aplikasi di pusat data pelanggan mungkin perlu mengakses cache ElastiCache di VPC. Skenario ini juga didukung jika ada konektivitas antara VPC pelanggan dan pusat data baik melalui VPN atau Direct Connect.

Topik

- [Mengakses Cache ElastiCache dari Aplikasi yang Berjalan di Pusat Data Pelanggan Menggunakan Konektivitas VPN](#)
- [Mengakses Cache ElastiCache dari Aplikasi yang Berjalan di Pusat Data Pelanggan Menggunakan Direct Connect](#)

Mengakses Cache ElastiCache dari Aplikasi yang Berjalan di Pusat Data Pelanggan Menggunakan Konektivitas VPN

Diagram berikut menunjukkan pengaksesan cache ElastiCache dari aplikasi yang berjalan di jaringan perusahaan Anda menggunakan koneksi VPN.



Menyambungkan ke ElastiCache dari pusat data Anda melalui VPN

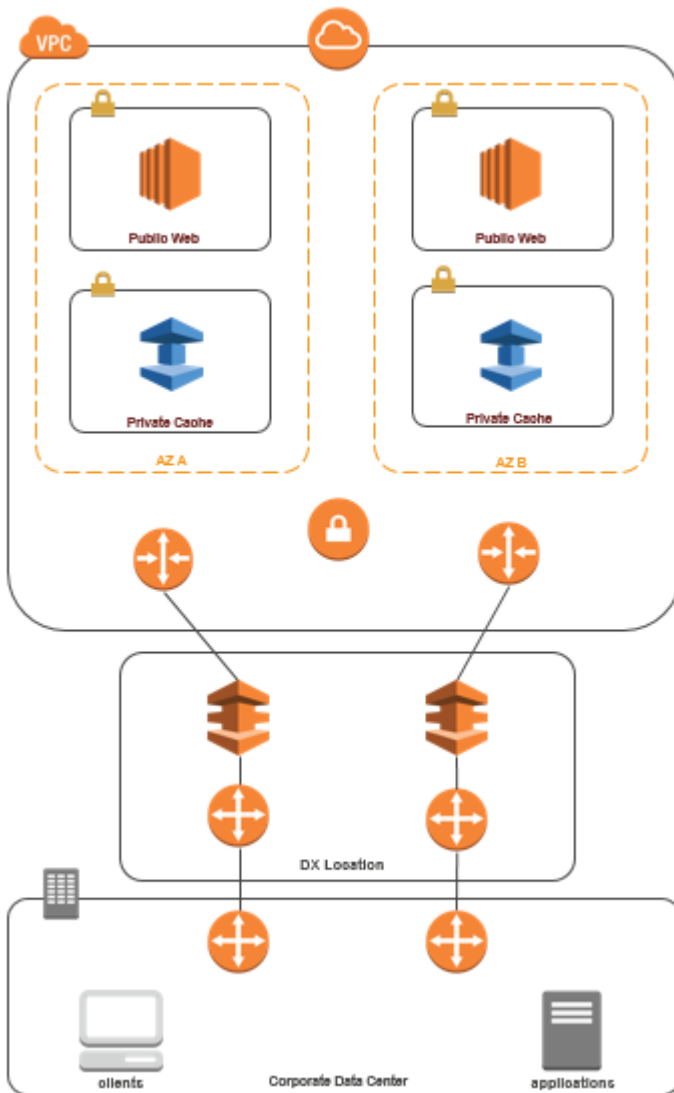
Untuk mengakses cache di VPC dari aplikasi on-premise melalui koneksi VPN

1. Buat Konektivitas VPN dengan menambahkan Gateway Privat Virtual perangkat keras ke VPC Anda. Untuk informasi selengkapnya, lihat [Menambahkan Gateway Privat Virtual ke VPC Anda](#).
2. Perbarui tabel perutean VPC untuk subnet tempat cache ElastiCache Anda disebarkan guna memungkinkan lalu lintas dari server aplikasi on-premise Anda. Dalam kasus Perutean Dinamis dengan BGP, rute Anda dapat disebarkan secara otomatis.
3. Ubah Grup Keamanan cache ElastiCache Anda agar mengizinkan koneksi masuk dari server aplikasi on-premise.

Mengakses cache melalui koneksi VPN akan menimbulkan latensi jaringan dan biaya transfer data tambahan.

Mengakses Cache ElastiCache dari Aplikasi yang Berjalan di Pusat Data Pelanggan Menggunakan Direct Connect

Diagram berikut menunjukkan pengaksesan cache ElastiCache dari aplikasi yang berjalan di jaringan perusahaan Anda menggunakan Direct Connect.



Menyambungkan ke ElastiCache dari pusat data Anda melalui Direct Connect

Mengakses cache ElastiCache 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 cache ElastiCache Anda agar mengizinkan koneksi masuk dari server aplikasi on-premise.

Mengakses 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.

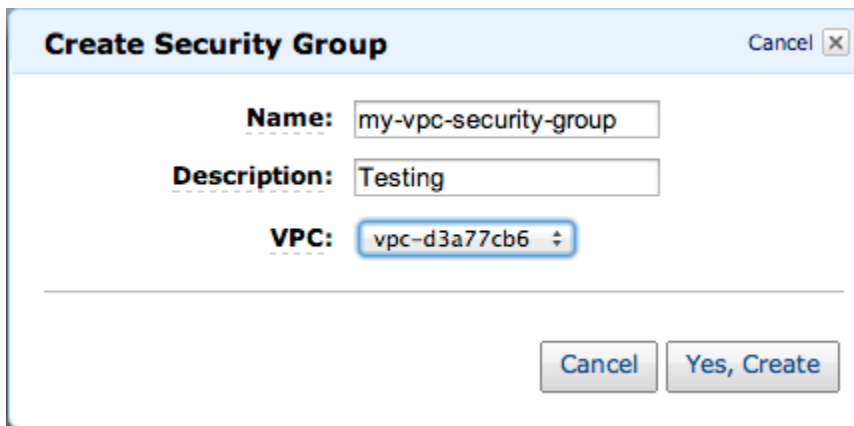
Membuat Amazon VPC (Konsol)

1. Login ke Konsol Manajemen AWS dan buka konsol Amazon VPC 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 bagian Jumlah Zona Ketersediaan (AZ), pilih jumlah Zona Ketersediaan yang ingin Anda luncurkan subnet ke sana.
5. Di bagian Jumlah subnet publik, pilih jumlah subnet publik yang ingin Anda tambahkan ke VPC Anda.
6. Di bagian Jumlah subnet pribadi, pilih jumlah subnet pribadi yang ingin Anda tambahkan ke VPC Anda.

Tip

Perhatikan pengidentifikasi subnet Anda, serta identifikasi mana yang publik dan privat. Anda akan membutuhkan informasi ini nanti saat meluncurkan kluster dan menambahkan instans Amazon EC2 ke Amazon VPC Anda.

7. Buat grup keamanan Amazon VPC. Anda akan menggunakan grup ini untuk kluster cache dan instans Amazon EC2 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.



Create Security Group Cancel

Name: my-vpc-security-group

Description: Testing

VPC: vpc-d3a77cb6

Cancel Yes, Create

- d. Jika pengaturan sudah sesuai keinginan Anda, pilih Ya, Buat.
8. Tentukan aturan masuk jaringan untuk grup keamanan Anda. Aturan ini akan mengizinkan Anda terhubung ke instans Amazon EC2 dengan menggunakan Secure Shell (SSH).
- a. Di daftar navigasi, pilih Grup Keamanan.
 - b. Temukan grup keamanan Anda di 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 \(konsol\)](#).

Menghubungkan ke cache yang berjalan di Amazon VPC

Contoh ini menunjukkan cara meluncurkan instans Amazon EC2 di Amazon VPC Anda. Anda kemudian dapat log masuk ke instans ini dan mengakses cache ElastiCache yang berjalan di Amazon VPC.

Menghubungkan ke cache yang berjalan di Amazon VPC (Konsol)

Dalam contoh ini, Anda membuat instans Amazon EC2 di Amazon VPC. Anda dapat menggunakan instans Amazon EC2 ini untuk terhubung ke simpul cache yang berjalan di Amazon VPC.

Note

Untuk informasi tentang menggunakan Amazon EC2, lihat [Panduan Memulai Amazon EC2](#) di [Dokumentasi Amazon EC2](#).

Untuk membuat instans Amazon EC2 di Amazon VPC dengan menggunakan konsol Amazon EC2

1. Masuk ke AWS Management Console lalu buka konsol Amazon EC2 di <https://console.aws.amazon.com/ec2/>.
2. Pada konsol, pilih Luncurkan Instans dan ikuti langkah-langkah ini:
3. Pada halaman Pilih Amazon Machine Image (AMI), pilih AMI Amazon Linux 64-bit, lalu pilihlah 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.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot Instances to take advantage pricing, assign an access management role to the instance, and more.

Number of instances ⓘ

Purchasing option ⓘ Request Spot Instances

Network ⓘ

Subnet ⓘ
250 IP Addresses available

Public IP ⓘ Automatically assign a public IP address to your instances

Jika pengaturan sudah sesuai keinginan Anda, pilih 4. Tambahkan Penyimpanan.

6. Pada halaman Tambahkan Penyimpanan, pilih 5. Tandai Instans.
7. Pada halaman Tandai Instans, ketikkan nama untuk instans Amazon EC2 Anda, dan kemudian pilih 6. Konfigurasi Grup Keamanan.
8. Pada halaman Konfigurasi Grup Keamanan, pilih Pilih grup keamanan yang sudah ada. Untuk informasi lebih lanjut tentang grup keamanan, silakan lihat [Grup keamanan Amazon EC2 untuk instans Linux](#).

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: Create a new security group
 Select an existing security group

Security Group ID	Name	Description
<input type="checkbox"/> sg-1a3d2178	default	default VPC security group
<input checked="" type="checkbox"/> sg-f13d2193	my-vpc-security-group	Testing

Pilih nama grup keamanan Amazon VPC Anda, dan kemudian 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 Memulai Amazon EC2](#).

11. Ketika Anda siap untuk meluncurkan instans Amazon EC2 Anda, pilih Luncurkan instans.

Sekarang Anda dapat menetapkan alamat IP Elastis untuk instans Amazon EC2 yang baru saja Anda buat. Anda harus menggunakan alamat IP ini untuk terhubung ke instans Amazon EC2.

Untuk menetapkan alamat IP elastis (Konsol)

1. Buka konsol Amazon VPC di <https://console.aws.amazon.com/vpc/>.
2. Di daftar navigasi, pilih IP Elastis.
3. Pilih Alokasi 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 Kaitkan Alamat, di kotak Instans, pilih ID instans Amazon EC2 yang Anda luncurkan.

Di kotak Alamat IP privat, pilih kotak untuk mendapatkan alamat IP privat dan kemudian pilih Kaitkan.

Sekarang Anda dapat menggunakan SSH untuk terhubung ke instans Amazon EC2 menggunakan alamat IP Elastis yang Anda buat.

Hubungkan ke instans Amazon EC2 Anda

- Buka jendela perintah. Pada prompt perintah, keluarkan perintah berikut, menggantikan `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 dulu log keluar dari instans Amazon EC2 Anda.

Sekarang Anda siap untuk berinteraksi dengan kluster ElastiCache Anda. Sebelum Anda dapat melakukannya, Anda perlu menginstal utilitas telnet jika Anda belum melakukannya.

Untuk menginstal telnet dan berinteraksi dengan kluster cache Anda (AWS CLI)

1. Buka jendela perintah. Di prompt perintah, keluarkan perintah berikut. Pada 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!
```

2. Buka konsol ElastiCache di <https://console.aws.amazon.com/elasticache/> dan dapatkan titik akhir untuk salah satu simpul dalam kluster cache Anda. Untuk informasi selengkapnya, lihat [Mendapatkan temuan titik akhir koneksi](#) untuk Memcached.
3. Gunakan telnet untuk tersambung 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 dan kemudian mendapatkannya segera sesudahnya. Akhirnya, Anda akan terputus sambungan 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 sambungan dari mesin cache, ketik yang berikut ini:

```
quit
```

Important

Agar tidak dikenakan biaya tambahan pada akun AWS Anda, pastikan untuk menghapus segala sumber daya AWS yang tidak lagi Anda inginkan setelah mencoba contoh-contoh ini.

API Amazon ElastiCache dan titik akhir VPC antarmuka (AWS PrivateLink)

Anda dapat membuat koneksi privat antara VPC Anda dan titik akhir API Amazon ElastiCache dengan membuat titik akhir VPC antarmuka. Titik akhir antarmuka didukung oleh [AWS PrivateLink](#). AWS PrivateLink memungkinkan Anda mengakses operasi API Amazon ElastiCache tanpa gateway internet, perangkat NAT, koneksi VPN, atau koneksi AWS Direct Connect.

Instans dalam VPC Anda tidak memerlukan alamat IP publik untuk berkomunikasi dengan titik akhir API Amazon ElastiCache. Instans Anda juga tidak memerlukan alamat IP publik untuk menggunakan operasi API ElastiCache yang tersedia. Lalu lintas antara VPC dan Amazon ElastiCache tidak keluar dari 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](#) dalam Panduan Pengguna Amazon EC2.

- Untuk informasi selengkapnya tentang titik akhir VPC, lihat [Titik Akhir VPC Antarmuka \(AWS PrivateLink\)](#) dalam Panduan Pengguna Amazon VPC.
- Untuk informasi selengkapnya tentang operasi API ElastiCache, lihat [Operasi API ElastiCache](#).

Setelah membuat titik akhir VPC antarmuka, jika Anda mengaktifkan nama host [DNS privat](#) untuk titik akhir, maka titik akhir ElastiCache default (<https://elasticache.Region.amazonaws.com>) akan diresolusi 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 \(AWS PrivateLink\)](#) dalam Panduan Pengguna Amazon VPC. ElastiCache mendukung pembuatan panggilan ke semua [Tindakan API](#)-nya 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 privat harus dinonaktifkan.

Pertimbangan untuk titik akhir VPC

Sebelum Anda menyiapkan titik akhir VPC antarmuka untuk titik akhir API Amazon ElastiCache, pastikan bahwa Anda meninjau [Properti dan batasan titik akhir antarmuka](#) dalam Panduan Pengguna Amazon VPC. Semua operasi API ElastiCache yang relevan dengan pengelolaan sumber daya Amazon ElastiCache tersedia dari VPC Anda menggunakan AWS PrivateLink.

Kebijakan titik akhir VPC didukung untuk titik akhir API ElastiCache. Secara default, akses penuh ke operasi API ElastiCache diizinkan melalui titik akhir. Untuk informasi selengkapnya, lihat [Mengendalikan 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 API Amazon ElastiCache menggunakan konsol Amazon VPC atau AWS CLI. Untuk informasi selengkapnya, lihat [Membuat sebuah 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, titik akhir Amazon ElastiCache default (<https://elasticache.Region.amazonaws.com>) akan diresolusi ke titik akhir VPC Anda. Untuk Wilayah AWS Tiongkok (Beijing) dan Tiongkok (Ningxia), Anda dapat membuat permintaan API dengan titik akhir VPC menggunakan elasticache.cn-north-1.amazonaws.com.cn untuk Beijing dan elasticache.cn-northwest-1.amazonaws.com.cn untuk Ningxia. Untuk informasi selengkapnya, lihat [Mengakses layanan melalui titik akhir antarmuka](#) dalam Panduan Pengguna Amazon VPC.

Membuat kebijakan titik akhir VPC untuk API Amazon ElastiCache

Anda dapat melampirkan kebijakan titik akhir ke titik akhir VPC Anda yang mengontrol akses ke API ElastiCache. Kebijakan menentukan hal berikut:

- Prinsipal yang dapat melakukan tindakan.
- Tindakan-tindakan yang dapat dilakukan.
- Sumber daya yang menjadi target tindakan.

Untuk informasi selengkapnya, lihat [Mengendalikan Akses ke Layanan dengan titik akhir VPC](#) dalam Panduan Pengguna Amazon VPC.

Example Kebijakan titik akhir VPC untuk tindakan API ElastiCache

Berikut adalah contoh kebijakan titik akhir untuk API ElastiCache. Saat dilampirkan ke sebuah titik akhir, kebijakan ini memberikan akses ke tindakan API ElastiCache yang tercantum untuk semua prinsipal pada semua sumber daya.

```
{
  "Statement": [{
    "Principal": "*",
    "Effect": "Allow",
    "Action": [
      "elasticache:CreateCacheCluster",
      "elasticache:ModifyCacheCluster"
    ],
    "Resource": "*"
  }]
}
```

Example Kebijakan titik akhir VPC yang menolak semua akses dari akun AWS tertentu

Kebijakan titik akhir VPC berikut menolak semua akses dari akun AWS **123456789012** ke sumber daya yang menggunakan titik akhir yang ditentukan. 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 (Amazon VPC).

Jika Anda membuat klaster yang dirancang sendiri di Amazon VPC, Anda harus menggunakan grup subnet. ElastiCache menggunakan grup subnet tersebut untuk memilih subnet dan alamat IP dalam subnet tersebut untuk mengasosiasikannya dengan simpul Anda.

ElastiCache menyediakan grup subnet IPv4 default atau Anda dapat memilih untuk membuat yang baru. Untuk IPv6, Anda perlu membuat grup subnet dengan blok CIDR IPv6. Jika Anda memilih dual-stack, Anda harus memilih jenis IP Discovery, baik IPv6 atau IPv4.

ElastiCache Nirserver tidak menggunakan sumber daya grup subnet, dan sebagai gantinya mengambil daftar subnet secara langsung selama pembuatan.

Bagian ini membahas cara membuat dan memanfaatkan subnet dan grup subnet untuk mengelola akses ke sumber daya ElastiCache Anda.

Untuk informasi selengkapnya tentang penggunaan grup subnet di lingkungan Amazon VPC, lihat [Mengakses klaster Anda](#).

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 di dalam VPC. Saat meluncurkan cache di VPC, Anda harus memilih grup subnet cache. Kemudian ElastiCache menggunakan grup subnet cache tersebut untuk memberikan alamat IP di dalam subnet tersebut ke setiap simpul cache di dalam cache.

Saat Anda membuat grup subnet baru, perhatikan jumlah alamat IP yang tersedia. Jika subnet memiliki sangat sedikit alamat IP yang tersedia, 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 di 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 tersebut untuk memilih subnet dan alamat IP dalam subnet tersebut yang akan dikaitkan dengan simpul Anda. Jika Anda memilih dual-stack atau IPV6, Anda akan diarahkan untuk membuat subnet dual-stack atau IPV6. Untuk informasi selengkapnya tentang jenis jaringan, lihat [Jenis jaringan](#). Untuk informasi selengkapnya, lihat [Membuat subnet di VPC Anda](#).

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

Membuat grup subnet (Konsol)

Prosedur berikut menunjukkan cara membuat grup subnet (konsol).

Untuk membuat grup subnet (Konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol ElastiCache 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 Zona Ketersediaan atau [Zona Lokal](#) dan ID subnet pribadi Anda, lalu pilihlah Pilih.
5. Pada pesan konfirmasi yang muncul, pilih Tutup.

Grup subnet baru Anda muncul pada daftar Grup Subnet dari 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 keluaran yang serupa dengan yang berikut:

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

```
    }
  ],
  "CacheSubnetGroupName": "mysubnetgroup"
}
```

Untuk informasi selengkapnya, lihat AWS CLI topik [create-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.

Mengubah grup subnet

Anda dapat mengubah deskripsi grup subnet, atau mengubah daftar ID subnet yang terhubung 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 konsol ElastiCache 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 keluaran 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 selengkapnya, lihat AWS CLI topik [modify-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 konsol ElastiCache 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, dan kemudian pilih Hapus.
4. Saat Anda diminta untuk mengonfirmasi operasi ini, ketik nama grup subnet di kolom input teks dan pilih Hapus.

Menghapus grup subnet (AWS CLI)

Menggunakan AWS CLI, memanggil 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 selengkapnya, lihat AWS CLI topik [delete-cache-subnet-group](#).

Manajemen Identitas dan Akses untuk Amazon ElastiCache

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengendalikan akses ke sumber daya AWS secara aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diotorisasi (memiliki izin) untuk menggunakan sumber daya ElastiCache. IAM adalah Layanan AWS yang dapat Anda gunakan tanpa dikenakan biaya tambahan.

Topik

- [Audiens](#)
- [Mengautentikasi dengan identitas](#)
- [Mengelola kebijakan menggunakan akses](#)
- [Cara Amazon ElastiCache bekerja dengan IAM](#)
- [Contoh kebijakan berbasis identitas untuk Amazon ElastiCache](#)
- [Memecahkan masalah identitas dan akses Amazon ElastiCache](#)
- [Kontrol akses](#)
- [Gambaran umum pengelolaan izin akses untuk sumber daya ElastiCache Anda](#)

Audiens

Cara Anda menggunakan (IAM) AWS Identity and Access Management berbeda, tergantung pada pekerjaan yang Anda lakukan di ElastiCache.

Pengguna layanan – Jika Anda menggunakan layanan ElastiCache untuk melakukan tugas Anda, administrator Anda akan memberikan kredensial dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak fitur ElastiCache untuk melakukan pekerjaan, Anda mungkin memerlukan izin tambahan. Memahami cara akses dikelola dapat membantu Anda meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses suatu fitur di ElastiCache, silakan lihat [Memecahkan masalah identitas dan akses Amazon ElastiCache](#).

Administrator layanan – Jika Anda bertanggung jawab atas sumber daya ElastiCache di perusahaan Anda, Anda mungkin memiliki akses penuh ke ElastiCache. Merupakan tugas Anda untuk menentukan fitur dan sumber daya ElastiCache mana yang dapat diakses oleh 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 selengkapnya cara perusahaan Anda dapat menggunakan IAM dengan ElastiCache, silakan lihat [Cara Amazon ElastiCache bekerja dengan IAM](#).

Administrator IAM – Jika Anda seorang administrator IAM, Anda mungkin ingin belajar dengan lebih detail cara Anda dapat menulis kebijakan untuk mengelola akses ke ElastiCache. Untuk melihat contoh kebijakan berbasis identitas ElastiCache yang dapat Anda gunakan di IAM, silakan lihat [Contoh kebijakan berbasis identitas untuk Amazon ElastiCache](#).

Mengautentikasi dengan identitas

Autentikasi merupakan cara Anda untuk masuk ke AWS menggunakan kredensial identitas Anda. Anda harus terautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengambil peran IAM.

Anda dapat masuk ke AWS sebagai identitas terfederasi dengan menggunakan kredensial yang disediakan melalui sumber identitas. Pengguna AWS IAM Identity Center (Pusat Identitas IAM), autentikasi masuk tunggal perusahaan Anda, dan kredensial Google atau Facebook Anda merupakan contoh identitas terfederasi. Saat Anda masuk sebagai identitas terfederasi, administrator Anda sebelumnya menyiapkan federasi identitas dengan menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil suatu peran.

Tergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal akses AWS. Untuk informasi selengkapnya tentang masuk ke AWS, lihat [Cara masuk ke Akun AWS Anda](#) dalam Panduan Pengguna AWS Sign-In.

Jika Anda mengakses AWS secara programatis, AWS menyediakan kit pengembangan perangkat lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensial Anda. Jika Anda tidak menggunakan alat AWS, maka Anda harus menandatangani sendiri permintaan tersebut. Untuk informasi selengkapnya tentang penggunaan metode yang disarankan untuk menandatangani permintaan sendiri, lihat [Menandatangani permintaan API AWS](#) dalam Panduan Pengguna IAM.

Terlepas dari metode autentikasi yang Anda gunakan, Anda mungkin juga diminta untuk menyediakan informasi keamanan tambahan. Misalnya, AWS menyarankan agar Anda menggunakan autentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, lihat [Autentikasi multi-faktor](#) dalam Panduan Pengguna AWS IAM Identity Center dan [Menggunakan autentikasi multi-faktor \(MFA\) dalam AWS](#) dalam Panduan Pengguna IAM.

Pengguna root Akun AWS

Ketika Anda membuat Akun AWS, Anda memulai dengan satu identitas masuk yang memiliki akses ke semua Layanan AWS dan sumber daya di akun tersebut. Identitas ini disebut pengguna root Akun AWS dan diakses dengan cara masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan 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 terfederasi

Sebagai praktik terbaik, wajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan kredensial sementara.

Identitas terfederasi adalah pengguna dari direktori pengguna korporasi Anda, penyedia identitas web, AWS Directory Service, direktori Pusat Identitas, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensial yang disediakan melalui sumber identitas. Ketika identitas terfederasi mengakses Akun AWS, identitas tersebut mengambil peran, dan peran ini memberikan kredensial sementara.

Untuk pengelolaan 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 Akun AWS Anda 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 Akun AWS Anda yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, kami merekomendasikan untuk mengandalkan kredensial sementara alih-alih 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 pengelolaan izin untuk sejumlah besar pengguna sekaligus. Misalnya, Anda dapat memiliki grup yang bernama IAMAdmins dan memberikan izin ke 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, silakan lihat [Kapan harus membuat pengguna IAM \(bukan peran\)](#) dalam Panduan Pengguna IAM.

Peran IAM

[Peran IAM](#) merupakan identitas dalam Akun AWS Anda yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Anda dapat mengambil peran IAM untuk sementara di AWS Management Console dengan [beralih peran](#). Anda dapat mengambil peran dengan cara memanggil operasi API AWS CLI atau AWS atau menggunakan URL kustom. Untuk informasi selengkapnya tentang cara menggunakan peran, lihat [Menggunakan peran IAM](#) 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 mengotentikasi, identitas tersebut terhubung dengan peran dan diberi izin yang ditentukan oleh peran. Untuk informasi tentang peran-peran untuk federasi, lihat [Membuat 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, pada beberapa Layanan AWS, Anda dapat melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan suatu peran sebagai perantara). Untuk mempelajari perbedaan antara kebijakan berbasis peran dan berbasis

sumber daya untuk akses lintas akun, lihat [Bagaimana peran IAM berbeda dari kebijakan berbasis sumber daya](#) dalam Panduan Pengguna IAM.

- Akses lintas layanan – Sebagian Layanan AWS menggunakan fitur di Layanan AWS lainnya. Sebagai contoh, ketika Anda memanggil 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 maju (FAS) – Ketika Anda menggunakan pengguna IAM atau peran IAM untuk melakukan tindakan-tindakan di AWS, Anda akan 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 prinsipal yang memanggil Layanan AWS, yang dikombinasikan dengan Layanan AWS, untuk mengajukan permintaan ke layanan hilir. Permintaan FAS hanya diajukan ketika sebuah layanan menerima permintaan yang memerlukan interaksi dengan Layanan AWS lain atau sumber daya lain untuk diselesaikan. 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 – Sebuah peran layanan adalah sebuah [peran IAM](#) yang dijalankan oleh suatu layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, memodifikasi, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat [Membuat sebuah peran untuk mendelegasikan izin ke Layanan AWS](#) dalam Panduan pengguna IAM.
- Peran terkait layanan – Peran terkait layanan adalah tipe peran layanan yang tertaut dengan Layanan AWS. Layanan tersebut dapat menjalankan peran untuk melakukan sebuah tindakan atas nama Anda. Peran terkait layanan akan muncul di Akun AWS Anda dan dimiliki oleh layanan tersebut. 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 kredensial sementara untuk aplikasi yang berjalan di instans EC2 dan mengajukan permintaan API AWS CLI atau AWS. Cara ini lebih baik daripada menyimpan kunci akses dalam instans EC2. Untuk memberikan peran AWS ke instans EC2 dan menyediakannya di semua aplikasinya, Anda dapat membuat profil instans yang dilampirkan ke instans tersebut. Profil instans berisi peran dan memungkinkan program yang berjalan di instans EC2 mendapatkan kredensial sementara. Untuk informasi selengkapnya, lihat [Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan dalam instans Amazon EC2](#) dalam Panduan Pengguna IAM.

Untuk mempelajari apakah kita harus menggunakan peran IAM atau pengguna IAM, lihat [Kapan harus membuat peran IAM \(bukan pengguna\)](#) dalam Panduan Pengguna IAM.

Mengelola kebijakan menggunakan akses

Anda mengendalikan akses di AWS dengan membuat kebijakan dan melampirkannya ke identitas atau sumber daya AWS. Kebijakan adalah objek di AWS yang, ketika terkait dengan identitas atau sumber daya, akan menentukan izinnya. AWS mengevaluasi kebijakan-kebijakan tersebut ketika prinsipal (pengguna, pengguna root, atau sesi peran) mengajukan permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan di 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 JSON AWS untuk menentukan secara spesifik siapa yang memiliki akses pada 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 mengabulkan 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 menjalankan 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 dapat memperoleh informasi peran dari AWS Management Console, AWS CLI, atau API AWS.

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, misalnya pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol tindakan apa yang dapat dilakukan oleh pengguna dan peran, pada sumber daya mana, dan dalam kondisi apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat [Membuat kebijakan IAM](#) dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan terkelola. 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 di Akun AWS Anda. Kebijakan terkelola meliputi kebijakan yang dikelola AWS dan kebijakan

yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan terkelola atau kebijakan inline, lihat [Memilih antara kebijakan terkelola 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-kebijakan berbasis sumber daya adalah kebijakan tepercaya 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 principal](#) dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna terfederasi, atau Layanan AWS.

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan yang dikelola AWS dari IAM dalam kebijakan berbasis sumber daya.

Daftar kontrol akses (ACL)

Daftar kontrol akses (ACL) mengendalikan prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun kebijakan tersebut tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh-contoh layanan yang mendukung ACL. Untuk mempelajari ACL selengkapnya, lihat [Gambaran umum daftar kontrol akses \(ACL\)](#) dalam Panduan Developer Amazon Simple Storage Service.

Tipe-tipe kebijakan lain

AWS mendukung tipe kebijakan tambahan, yang kurang umum. Tipe-tipe kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda oleh tipe 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 batas 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 (SCP) – SCP adalah kebijakan JSON yang menentukan izin maksimum untuk sebuah organisasi atau unit organisasi (OU) di AWS Organizations. AWS Organizations adalah layanan untuk mengelompokkan dan secara terpusat mengelola beberapa Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur di suatu organisasi, maka Anda dapat menerapkan kebijakan kontrol layanan (SCP) ke salah satu atau ke semua akun Anda. SCP membatasi izin untuk entitas dalam akun anggota, termasuk setiap Pengguna root akun AWS. Untuk informasi selengkapnya tentang Organisasi dan SCP, lihat [Cara kerja SCP](#) dalam Panduan Pengguna AWS Organizations.
- 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 dapat berasal dari kebijakan berbasis sumber daya. Penolakan eksplisit dalam salah satu kebijakan ini menindahi izin. Untuk informasi selengkapnya, lihat [Kebijakan sesi](#) dalam Panduan Pengguna IAM.

Berbagai tipe kebijakan

Ketika beberapa tipe kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan ketika beberapa tipe kebijakan digunakan, lihat [Logika evaluasi kebijakan](#) dalam Panduan Pengguna IAM.

Cara Amazon ElastiCache bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ke ElastiCache, pelajari fitur IAM apakah yang tersedia untuk digunakan pada ElastiCache.

Fitur-fitur IAM yang dapat Anda gunakan dengan Amazon ElastiCache

Fitur IAM	Dukungan ElastiCache
Kebijakan berbasis identitas	Ya
Kebijakan berbasis sumber daya	Tidak
Tindakan kebijakan	Ya

Fitur IAM	Dukungan ElastiCache
Sumber daya kebijakan	Ya
Kunci-kunci persyaratan kebijakan	Ya
ACL	Ya
ABAC (tag dalam kebijakan)	Ya
Kredensial sementara	Ya
Izin pengguna utama	Ya
Peran layanan	Ya
Peran tertaut layanan	Ya

Untuk mendapatkan tampilan tingkat tinggi tentang cara ElastiCache dan layanan AWS lain bekerja dengan sebagian besar fitur IAM, silakan lihat [Layanan AWS 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, misalnya pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol tindakan apa yang dapat dilakukan oleh pengguna dan peran, pada sumber daya mana, dan dalam kondisi apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat [Membuat kebijakan IAM](#) dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan secara spesifik apakah tindakan dan sumber daya diizinkan atau ditolak, serta persyaratan yang menjadi dasar dikabulkan atau ditolaknya tindakan tersebut. Anda tidak dapat menentukan secara spesifik pengguna utama dalam sebuah kebijakan berbasis identitas karena pengguna utama berlaku bagi pengguna atau peran yang melekat kepadanya. Untuk mempelajari semua elemen yang dapat Anda gunakan dalam kebijakan JSON, lihat [Referensi elemen kebijakan JSON IAM](#) dalam Panduan Pengguna IAM.

Contoh kebijakan berbasis identitas untuk ElastiCache

Untuk melihat contoh kebijakan-kebijakan berbasis identitas ElastiCache, silakan 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-kebijakan berbasis sumber daya adalah kebijakan tepercaya 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 principal](#) dalam kebijakan berbasis sumber daya. Pengguna utama dapat mencakup akun, pengguna, peran, pengguna gabungan, atau Layanan AWS.

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan secara spesifik seluruh akun atau entitas IAM di akun lain sebagai pengguna utama dalam kebijakan berbasis sumber daya. Menambahkan pengguna utama akun silang ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika pengguna utama dan sumber daya berada dalam Akun AWS yang berbeda, Administrator IAM di akun tepercaya juga harus memberikan izin kepada entitas pengguna utama (pengguna atau peran) untuk mengakses sumber daya. Mereka memberikan izin melampirkan kebijakan berbasis identitas kepada entitas. Namun, jika kebijakan berbasis sumber daya memberikan akses kepada pengguna utama dalam akun yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk informasi selengkapnya, lihat [Perbedaan antara peran IAM dan kebijakan berbasis sumber daya](#) di Panduan Pengguna IAM.

Tindakan kebijakan untuk ElastiCache

Mendukung tindakan kebijakan	Ya
------------------------------	----

Administrator dapat menggunakan kebijakan JSON AWS untuk menentukan secara spesifik siapa yang memiliki akses pada apa. Yaitu, pengguna utama manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan syarat apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan-tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama sebagaimana operasi API AWS yang dikaitkan padanya. Ada beberapa pengecualian, misalnya tindakan hanya izin yang tidak memiliki operasi API yang cocok. Ada juga beberapa operasi yang memerlukan beberapa tindakan dalam sebuah kebijakan. Tindakan-tindakan tambahan ini disebut tindakan dependen.

Menyertakan tindakan dalam suatu kebijakan untuk memberikan izin guna melakukan operasi yang terkait.

Untuk melihat daftar tindakan ElastiCache, silakan lihat [Tindakan yang ditentukan oleh Amazon ElastiCache](#) di Rujukan Otorisasi Layanan.

Tindakan kebijakan di ElastiCache menggunakan awalan berikut sebelum tindakan:

```
elasticache
```

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan-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-kebijakan berbasis identitas ElastiCache, silakan lihat [Contoh kebijakan berbasis identitas untuk Amazon ElastiCache](#).

Sumber daya kebijakan untuk ElastiCache

Mendukung sumber daya kebijakan

Ya

Administrator dapat menggunakan kebijakan JSON AWS untuk menentukan secara spesifik siapa yang memiliki akses pada apa. Yaitu, pengguna utama manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan syarat apa.

Elemen kebijakan JSON `Resource` menentukan objek atau objek-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-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 mengindikasikan bahwa pernyataan tersebut berlaku bagi semua sumber daya.

```
"Resource": "*" 
```

Untuk melihat daftar jenis sumber daya ElastiCache dan ARN-nya, lihat [Sumber Daya yang Ditentukan oleh Amazon ElastiCache](#) di Referensi Otorisasi Layanan. Untuk mempelajari jenis tindakan yang dapat Anda tentukan dengan ARN di tiap sumber daya, lihat [Tindakan yang Ditentukan oleh Amazon ElastiCache](#).

Untuk melihat contoh kebijakan-kebijakan berbasis identitas ElastiCache, silakan lihat [Contoh kebijakan berbasis identitas untuk Amazon ElastiCache](#).

Kunci syarat kebijakan untuk ElastiCache

Mendukung kunci kondisi kebijakan khusus layanan	Ya
--	----

Administrator dapat menggunakan kebijakan JSON AWS untuk menentukan secara spesifik siapa yang memiliki akses pada apa. Yaitu, pengguna utama manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan syarat apa.

Elemen `Condition` (atau blok `Condition`) akan memungkinkan Anda menentukan syarat yang menjadi dasar suatu pernyataan berlaku. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi bersyarat yang menggunakan [operator syarat](#), misalnya sama dengan atau kurang dari, untuk mencocokkan syarat 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 dengan menggunakan operasi AND logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS akan mengevaluasi syarat tersebut menggunakan operasi OR logis. Semua persyaratan harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan syarat. Sebagai contoh, Anda dapat memberikan izin kepada pengguna IAM untuk mengakses sumber daya hanya jika izin tersebut mempunyai tag yang sesuai dengan nama pengguna IAM terkait. Untuk informasi selengkapnya, lihat [Elemen kebijakan IAM: variabel dan tag](#) di Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi spesifik layanan. Untuk melihat semua kunci kondisi global AWS, lihat [kunci konteks syarat global AWS](#) di Panduan Pengguna IAM.

Untuk melihat daftar kunci-kunci persyaratan ElastiCache, silakan lihat [kunci persyaratan untuk Amazon ElastiCache](#) di Rujukan Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat Anda gunakan dengan kunci syarat, lihat [Tindakan yang Ditentukan oleh Amazon ElastiCache](#).

Untuk melihat contoh kebijakan-kebijakan berbasis identitas ElastiCache, silakan lihat [Contoh kebijakan berbasis identitas untuk Amazon ElastiCache](#).

Daftar kontrol akses (ACL) di ElastiCache

Mendukung ACL	Ya
---------------	----

Daftar kontrol akses (ACL) mengendalikan prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun kebijakan tersebut tidak menggunakan format dokumen kebijakan JSON.

Kontrol akses berbasis atribut (ABAC) dengan ElastiCache

Mendukung ABAC (tag dalam kebijakan)	Ya
--------------------------------------	----

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang menentukan izin berdasarkan atribut. Di AWS, atribut-atribut ini disebut tag. Anda dapat melampirkan tag ke entitas IAM (pengguna atau peran) dan ke banyak sumber daya AWS. Pemberian tag ke entitas dan sumber daya adalah

langkah pertama dari ABAC. Kemudian rancanglah kebijakan ABAC untuk mengizinkan operasi ketika tag milik pengguna utama cocok dengan tag yang ada di sumber daya yang ingin diakses.

ABAC sangat berguna di lingkungan yang berkembang dengan cepat dan berguna di situasi dimana pengelolaan kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tag, berikan informasi tentang tag di [elemen syarat](#) dari kebijakan dengan 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 [Apa itu ABAC?](#) di Panduan Pengguna IAM. Untuk melihat tutorial yang menguraikan langkah-langkah pengaturan ABAC, lihat [Menggunakan kontrol akses berbasis atribut \(ABAC\)](#) di Panduan Pengguna IAM.

Menggunakan kredensial Temporer dengan ElastiCache

Mendukung penggunaan kredensial sementara Ya

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensial sementara. Sebagai informasi tambahan, termasuk tentang Layanan AWS mana saja yang berfungsi dengan kredensial sementara, lihat [Layanan AWS yang berfungsi dengan IAM](#) di Panduan Pengguna IAM.

Anda menggunakan kredensial sementara jika Anda masuk ke AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Sebagai contoh, ketika Anda mengakses AWS dengan menggunakan tautan masuk tunggal (SSO) milik perusahaan Anda, proses itu secara otomatis akan membuat kredensial sementara. Anda juga akan secara otomatis membuat kredensial sementara ketika Anda masuk ke konsol sebagai seorang pengguna dan kemudian beralih peran. Untuk informasi selengkapnya tentang peralihan peran, lihat [Peralihan peran \(konsol\)](#) di Panduan Pengguna IAM.

Anda dapat membuat kredensial sementara secara manual menggunakan API AWS CLI atau AWS. Anda kemudian dapat menggunakan kredensial sementara tersebut untuk mengakses AWS. AWS menyarankan agar Anda membuat kredensial sementara secara dinamis alih-alih menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, silakan lihat [Kredensial keamanan sementara di IAM](#).

Izin pengguna utama lintas layanan untuk ElastiCache

Mendukung sesi akses maju (FAS) Ya

Saat Anda menggunakan pengguna IAM atau peran IAM untuk melakukan tindakan di AWS, Anda akan dianggap sebagai pengguna utama. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian dilanjutkan oleh tindakan lain pada layanan yang berbeda. FAS menggunakan izin dari prinsipal yang memanggil Layanan AWS, yang dikombinasikan dengan Layanan AWS, untuk mengajukan permintaan ke layanan hilir. Permintaan FAS hanya diajukan ketika sebuah layanan menerima permintaan yang memerlukan interaksi dengan Layanan AWS lain atau sumber daya lain untuk diselesaikan. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat [Meneruskan sesi akses](#).

Peran layanan untuk ElastiCache

Mendukung peran layanan Ya

Peran layanan adalah sebuah [peran IAM](#) yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, memodifikasi, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat [Membuat sebuah peran untuk mendelegasikan izin ke Layanan AWS](#) dalam Panduan pengguna IAM.

Warning

Mengubah izin untuk sebuah peran layanan dapat merusak fungsionalitas ElastiCache. Melakukan edit peran layanan hanya jika ElastiCache memberikan panduan untuk melakukannya.

Peran tertaut layanan untuk ElastiCache

Mendukung peran yang tertaut layanan Ya

Peran yang tertaut layanan adalah jenis peran layanan yang tertaut dengan Layanan AWS. Layanan tersebut dapat menjalankan peran untuk melakukan sebuah tindakan atas nama Anda. Peran terkait layanan akan muncul di Akun AWS Anda dan dimiliki oleh layanan tersebut. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Untuk detail tentang pembuatan atau pengelolaan peran tertaut layanan, silakan lihat [Layanan AWS yang berfungsi dengan IAM](#). Cari layanan dalam tabel yang memiliki Yes di kolom Peran tertaut layanan. Pilih tautan Ya untuk melihat dokumentasi peran tertaut layanan untuk layanan tersebut.

Contoh kebijakan berbasis identitas untuk Amazon ElastiCache

Secara bawaan, pengguna dan peran tidak memiliki izin untuk membuat atau melakukan modifikasi atas sumber daya ElastiCache. Pengguna dan peran tersebut juga tidak dapat melakukan tugas dengan menggunakan API AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS. 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 menggunakan contoh dokumen kebijakan JSON ini, lihat [Membuat kebijakan IAM](#) di Panduan Pengguna IAM.

Untuk mengetahui hal detail tentang tindakan dan jenis sumber daya yang ditentukan oleh ElastiCache, termasuk format ARN untuk setiap jenis sumber daya, silakan lihat [Tindakan, Sumber Daya, dan kunci kondisi untuk Amazon ElastiCache](#) di Rujukan Otorisasi Layanan.

Topik

- [Praktik terbaik kebijakan](#)
- [Menggunakan Konsol ElastiCache](#)
- [Izinkan pengguna melihat izin mereka sendiri](#)

Praktik terbaik kebijakan

Kebijakan-kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus sumber daya ElastiCache yang ada di akun Anda. Tindakan ini mengenakan biaya kepada Akun AWS Anda. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Memulai kebijakan terkelola AWS dan beralih ke izin dengan hak akses paling rendah – Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan terkelola AWS yang memberikan izin untuk banyak kasus penggunaan umum. Kebijakan tersedia di Akun AWS Anda. Sebaiknya kurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan AWS yang bersifat khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat [Kebijakan terkelola AWS](#) atau [Kebijakan terkelola AWS untuk fungsi tugas](#) di 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 di IAM](#) di Panduan Pengguna IAM.
- Gunakan syarat dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu syarat ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis syarat kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan syarat untuk memberi akses ke tindakan layanan jika digunakan melalui Layanan AWS tertentu, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat [Elemen kebijakan JSON IAM: Syarat](#) di 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 IAM Access Analyzer](#) di Panduan Pengguna IAM.
- Memerlukan autentikasi multi-faktor (MFA) – Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Akun AWS Anda, aktifkan MFA untuk keamanan tambahan. Untuk meminta MFA ketika operasi API dipanggil, tambahkan syarat MFA pada kebijakan Anda. Untuk informasi selengkapnya, silakan lihat [Mengonfigurasi akses API yang dilindungi MFA](#) di Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik di IAM, lihat [Praktik terbaik keamanan di IAM](#) di Panduan Pengguna IAM.

Menggunakan Konsol ElastiCache

Untuk mengakses konsol Amazon ElastiCache, Anda harus memiliki serangkaian izin minimum. Izin tersebut harus memperbolehkan Anda untuk membuat daftar dan melihat detail tentang sumber daya ElastiCache di akun AWS Anda. 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 meloloskan izin konsol minimum bagi pengguna yang hanya melakukan panggilan ke API AWS CLI atau AWS. 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 konsol ElastiCache, lampirkan juga ElastiCache ConsoleAccess atau kebijakan terkelola ReadOnly AWS ke entitas. Untuk informasi selengkapnya, lihat [Menambah izin untuk pengguna](#) di Panduan Pengguna IAM.

Izinkan pengguna melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan para pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan pada konsol atau menggunakan API AWS CLI atau AWS secara terprogram.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupsWithUser",
        "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 identitas dan akses Amazon ElastiCache

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temukan saat menggunakan ElastiCache dan IAM.

Topik

- [Saya tidak diizinkan untuk melakukan tindakan di ElastiCache](#)
- [Saya tidak diizinkan untuk melakukan iam:PassRole](#)
- [Saya ingin mengizinkan orang di luar akun AWS saya untuk mengakses sumber daya ElastiCache](#)

Saya tidak diizinkan untuk melakukan tindakan di ElastiCache

Jika AWS Management Console memberi tahu bahwa Anda tidak diizinkan untuk melakukan tindakan, Anda harus menghubungi administrator untuk mendapatkan bantuan. Administrator adalah orang yang memberikan nama pengguna dan kata sandi kepada 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 agar dia dapat mengakses sumber daya *my-example-widget* dengan menggunakan tindakan `elasticache:GetWidget`.

Saya tidak diizinkan untuk melakukan `iam:PassRole`

Jika Anda menerima pesan kesalahan bahwa Anda tidak diizinkan untuk melakukan tindakan `iam:PassRole`, kebijakan Anda harus diperbarui agar Anda memiliki izin untuk meneruskan peran ke ElastiCache.

Sebagian Layanan AWS mengizinkan Anda untuk memberikan peran yang sudah ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran tertaut-layanan. Untuk melakukan tindakan tersebut, Anda harus memiliki izin untuk memberikan peran pada layanan tersebut.

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 membutuhkan bantuan, hubungi administrator AWS Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya ingin mengizinkan orang di luar akun AWS saya untuk mengakses sumber daya ElastiCache

Anda dapat membuat peran yang dapat digunakan para pengguna di akun lain atau orang-orang di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa yang dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis sumber daya atau daftar kontrol akses (ACL), 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, lihat [Cara Amazon ElastiCache bekerja dengan IAM](#).
- Untuk mempelajari cara memberikan akses ke sumber daya di seluruh Akun AWS yang Anda miliki, lihat [Menyediakan akses ke pengguna IAM di Akun AWS lainnya yang Anda miliki](#) di Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses ke sumber daya Anda ke pihak ketiga Akun AWS, lihat [Menyediakan akses ke akun Akun AWS yang dimiliki oleh pihak ketiga](#) dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat [Menyediakan akses ke pengguna terautentikasi eksternal \(federasi identitas\)](#) di Panduan Pengguna IAM .
- Untuk mempelajari perbedaan antara penggunaan kebijakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Perbedaan antara kebijakan peran IAM dan kebijakan berbasis sumber daya](#) di Panduan Pengguna IAM.

Kontrol akses

Anda dapat memiliki kredensial yang valid untuk mengautentikasi permintaan, tetapi kecuali jika Anda memiliki izin, Anda tidak dapat membuat atau mengakses sumber daya ElastiCache. Misalnya, Anda harus memiliki izin untuk membuat klaster ElastiCache.

Bagian berikut menjelaskan cara mengelola izin untuk ElastiCache. Anda disarankan untuk membaca gambaran umum terlebih dahulu.

- [Gambaran umum pengelolaan izin akses untuk sumber daya ElastiCache Anda](#)
- [Menggunakan kebijakan berbasis identitas \(kebijakan IAM\) untuk Amazon ElastiCache](#)

Gambaran umum pengelolaan izin akses untuk sumber daya ElastiCache Anda

Setiap sumber daya AWS dimiliki oleh akun AWS, 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 pelampiran kebijakan izin pada sumber daya.

Note

Administrator akun (atau pengguna administrator) adalah pengguna dengan hak akses administrator. Untuk informasi selengkapnya tentang administrator, 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 petunjuk dalam [Buat set izin](#) dalam Panduan Pengguna AWS IAM Identity Center.

- Pengguna yang dikelola di IAM melalui penyedia identitas:

Buat peran untuk federasi identitas. Ikuti petunjuk dalam [Membuat peran untuk penyedia identitas pihak ketiga \(federasi\)](#) dalam Panduan Pengguna IAM.

- Pengguna IAM:

- Buat peran yang dapat diambil pengguna Anda. Ikuti petunjuk dalam [Membuat peran untuk pengguna IAM](#) dalam Panduan Pengguna IAM.
- (Tidak disarankan) Pasang kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti petunjuk dalam [Menambahkan izin ke pengguna \(konsol\)](#) dalam Panduan Pengguna IAM.

Topik

- [Sumber daya dan operasi Amazon ElastiCache](#)
- [Memahami kepemilikan sumber daya](#)
- [Mengelola akses ke sumber daya](#)

- [Kebijakan terkelola AWS untuk Amazon ElastiCache](#)
- [Menggunakan kebijakan berbasis identitas \(kebijakan IAM\) untuk Amazon ElastiCache](#)
- [Izin tingkat sumber daya](#)
- [Menggunakan kunci syarat](#)
- [Menggunakan Peran Tertaut Layanan untuk Amazon ElastiCache](#)
- [Izin API ElastiCache: Referensi tindakan, sumber daya, dan kondisi](#)

Sumber daya dan operasi Amazon ElastiCache

Untuk melihat daftar jenis sumber daya ElastiCache dan ARN-nya, lihat [Sumber Daya yang Ditentukan oleh Amazon ElastiCache](#) dalam Referensi Otorisasi Layanan. Untuk mempelajari jenis tindakan yang dapat Anda gunakan untuk menentukan ARN dari setiap sumber daya, lihat [Tindakan yang Ditentukan oleh Amazon ElastiCache](#).

Memahami kepemilikan sumber daya

Pemilik sumber daya adalah akun AWS yang membuat sumber daya. Artinya, pemilik sumber daya adalah akun AWS dari entitas prinsipal yang melakukan autentikasi permintaan yang membuat sumber daya. Entitas prinsipal dapat berupa akun root, pengguna IAM, atau peran IAM). Contoh berikut menggambarkan cara kerjanya:

- Misalkan Anda menggunakan kredensial akun root dari akun AWS Anda untuk membuat sebuah kluster cache. Dalam hal ini, akun AWS Anda adalah pemilik sumber daya. Di ElastiCache, sumber dayanya adalah kluster cache.
- Misalkan Anda membuat pengguna IAM dalam akun AWS Anda dan memberikan izin untuk membuat kluster cache kepada pengguna tersebut. Dalam hal ini, pengguna tersebut dapat membuat kluster cache. Namun, akun AWS Anda, yang berisi pengguna tersebut, adalah pemilik sumber daya kluster cache.
- Misalkan Anda membuat pengguna IAM dalam akun AWS Anda dengan izin untuk membuat kluster cache. Dalam hal ini, siapa pun yang dapat mengambil peran tersebut akan dapat membuat kluster cache. Akun AWS Anda, yang berisi peran tersebut, adalah pemilik sumber daya kluster 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 diberikan agar pengguna tersebut dapat membuat sumber daya ElastiCache, seperti klaster 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 akun AWS lain (misalnya, Akun B) atau layanan AWS 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 bagi semua pengguna di Akun B. Dengan demikian, pengguna di akun B dapat membuat atau mengakses sumber daya di akun A. Dalam beberapa kasus, Anda mungkin ingin memberi layanan AWS izin 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 adalah contoh kebijakan yang mengizinkan pengguna melakukan tindakan `DescribeCacheClusters` untuk akun AWS Anda. ElastiCache juga mendukung identifikasi sumber daya tertentu menggunakan ARN sumber daya untuk tindakan API. (Pendekatan ini juga disebut sebagai izin tingkat sumber daya).

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "DescribeCacheClusters",
    "Effect": "Allow",
    "Action": [
      "elasticache:DescribeCacheClusters"],
    "Resource": resource-arn
  ]
}
```

Untuk informasi selengkapnya tentang penggunaan 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 sumber daya Amazon ElastiCache (lihat [Sumber daya dan operasi Amazon ElastiCache](#)), layanan menentukan kumpulan operasi API (lihat [Tindakan](#)). Untuk memberikan

izin bagi operasi API ini, ElastiCache menentukan kumpulan tindakan yang dapat Anda tetapkan dalam kebijakan. Misalnya, untuk sumber daya kluster ElastiCache, tindakan berikut ditentukan: `CreateCacheCluster`, `DeleteCacheCluster`, and `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 [Sumber daya dan operasi Amazon ElastiCache](#).
- Tindakan – Anda menggunakan kata kunci tindakan untuk mengidentifikasi operasi sumber daya yang ingin Anda izinkan atau tolak. Misalnya, tergantung pada `Effect` yang ditentukan, izin `elasticache:CreateCacheCluster` mengizinkan atau menolak pengguna untuk melakukan operasi `CreateCacheCluster` Amazon ElastiCache.
- Efek – Anda menentukan efek ketika pengguna meminta tindakan tertentu—baik mengizinkan maupun menolak. 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 menunjukkan semua tindakan API Amazon ElastiCache, lihat [Izin API ElastiCache: 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 kunci kondisi khusus ElastiCache, lihat [Menggunakan kunci syarat](#). Terdapat kunci kondisi yang berlaku di seluruh AWS yang dapat Anda gunakan sesuai kebutuhan. Untuk daftar lengkap kunci di seluruh AWS, lihat [Kunci yang Tersedia untuk Syarat](#) dalam Panduan Pengguna IAM.

Kebijakan terkelola AWS untuk Amazon ElastiCache

Kebijakan terkelola AWS adalah kebijakan mandiri yang dibuat dan dikelola AWS. Kebijakan terkelola AWS dirancang untuk memberikan izin bagi banyak kasus penggunaan umum agar Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan yang dikelola AWS mungkin tidak memberikan izin hak akses paling rendah untuk kasus penggunaan spesifik Anda karena kebijakan ini 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 yang dikelola AWS. Jika AWS memperbarui izin yang ditentukan dalam sebuah kebijakan yang dikelola AWS, maka pembaruan tersebut akan memengaruhi semua identitas prinsipal (pengguna, grup, dan peran) yang terkait dengan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan yang dikelola AWS saat sebuah Layanan AWS baru diluncurkan atau operasi API baru tersedia untuk layanan yang sudah ada.

Untuk informasi selengkapnya, lihat [Kebijakan yang dikelola AWS](#) dalam Panduan Pengguna IAM.

Kebijakan terkelola AWS: ElastiCacheServiceRolePolicy

Anda tidak dapat melampirkan ElastiCacheServiceRolePolicy ke entitas IAM Anda. Kebijakan ini dilampirkan ke peran tertaut layanan yang memungkinkan ElastiCache melakukan mewakili Anda.

Kebijakan ini memungkinkan ElastiCache mengelola sumber daya AWS mewakili Anda seperti yang diperlukan untuk mengelola cache Anda:

- `ec2` - Kelola sumber daya jaringan EC2 untuk dilampirkan ke simpul cache, termasuk titik akhir VPC (untuk cache nirserver), Antarmuka Jaringan Elastis (ENI) (untuk klaster yang dirancang sendiri), dan grup keamanan.
- `cloudwatch` - Memancarkan data metrik dari layanan ke CloudWatch.
- `outposts` - Mengizinkan pembuatan simpul cache pada AWS Outpost.

Anda dapat menemukan kebijakan [ElastiCacheServiceRolePolicy](#) pada konsol IAM dan [ElastiCacheServiceRolePolicy](#) pada Panduan Referensi Kebijakan Terkelola AWS.

Kebijakan terkelola AWS: AmazonElastiCacheFullAccess

Anda dapat melampirkan kebijakan AmazonElastiCacheFullAccess ke identitas IAM Anda.

Kebijakan ini memungkinkan pengguna utama memiliki akses penuh ke ElastiCache menggunakan Konsol Manajemen AWS:

- `elasticache` - Mengakses semua API.
- `iam` - Membuat peran terkait layanan yang diperlukan untuk operasi layanan.
- `ec2` - Menjelaskan sumber daya EC2 dependen yang diperlukan untuk pembuatan cache (VPC, subnet, grup keamanan) dan mengizinkan pembuatan titik akhir VPC (untuk cache nirserver).
- `kms` - Mengizinkan penggunaan CMK yang dikelola pelanggan untuk enkripsi data diam.
- `cloudwatch` - Mengizinkan akses ke metrik untuk menampilkan metrik ElastiCache di konsol.
- `application-autoscaling` - Mengizinkan akses untuk menjelaskan kebijakan penskalaan otomatis untuk cache.
- `logs` - Digunakan untuk mengisi log stream untuk fungsionalitas pengiriman log di konsol.
- `firehose` - Digunakan untuk mengisi pengiriman stream 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 kebijakan [AmazonElastiCacheFullAccess](#) pada konsol IAM dan [AmazonElastiCacheFullAccess](#) pada Panduan Referensi Kebijakan Terkelola AWS.

Kebijakan terkelola AWS: AmazonElastiCacheReadOnlyAccess

Anda dapat melampirkan kebijakan AmazonElastiCacheReadOnlyAccess ke identitas IAM Anda.

Kebijakan ini memungkinkan pengguna utama memiliki akses baca saja ke ElastiCache menggunakan Konsol Manajemen AWS:

- `elasticache` - Mengakses API Describe baca saja.

Anda dapat menemukan kebijakan [AmazonElastiCacheReadOnlyAccess](#) pada konsol IAM dan [AmazonElastiCacheReadOnlyAccess](#) pada Panduan Referensi Kebijakan Terkelola AWS.

ElastiCache memperbarui kebijakan terkelola AWS

Tampilkan detail tentang pembaruan untuk kebijakan terkelola AWS untuk ElastiCache sejak layanan ini mulai melacak perubahan-perubahan tersebut. Untuk peringatan otomatis tentang perubahan pada halaman ini, silakan berlangganan ke umpan RSS di halaman Riwayat dokumen ElastiCache.

Perubahan	Deskripsi	Tanggal
AmazonElastiCacheFullAccess — Pembaruan ke kebijakan yang sudah ada	ElastiCache menambahkan izin baru untuk memungkinkan pengelolaan cache nirserver, dan untuk mengaktifkan penggunaan semua fitur layanan melalui konsol.	27 November 2023
ElastiCacheServiceRolePolicy — Pembaruan ke kebijakan yang sudah ada	ElastiCache menambahkan izin baru untuk memungkinkan pengelolaan titik akhir VPC untuk sumber daya cache nirserver.	27 November 2023
ElastiCache mulai melacak perubahan	ElastiCache mulai melacak perubahan untuk kebijakan terkelola AWS.	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

Sebaiknya Anda terlebih dahulu membaca topik yang menjelaskan konsep dasar dan opsi untuk mengelola akses ke sumber daya Amazon ElastiCache. Untuk informasi selengkapnya, lihat [Gambaran umum pengelolaan izin akses untuk sumber daya ElastiCache Anda](#).

Bagian dalam topik ini mencakup hal berikut:

- [Kebijakan terkelola AWS untuk Amazon ElastiCache](#)
- [Contoh kebijakan yang dikelola pelanggan](#)

Berikut adalah contoh kebijakan izin.

```
{
  "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 tindakan Amazon ElastiCache (`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 pengguna utama yang mendapatkan izin. Saat Anda menyematkan kebijakan kepada pengguna, pengguna tersebut menjadi pengguna utama secara implisit. Saat Anda menyematkan kebijakan izin pada peran IAM, pengguna utama yang diidentifikasi dalam kebijakan kepercayaan peran tersebut akan mendapatkan izin.

Untuk tabel yang menampilkan semua tindakan API Amazon ElastiCache dan sumber daya yang diterapkan, lihat [Izin API ElastiCache: Referensi tindakan, sumber daya, dan kondisi](#).

Contoh kebijakan yang dikelola pelanggan

Jika Anda tidak menggunakan kebijakan default dan memilih untuk menggunakan kebijakan yang dikelola khusus, pastikan salah satu dari dua hal berikut. Apakah Anda harus memiliki izin untuk memanggil `iam:createServiceLinkedRole` (untuk informasi selengkapnya, lihat [Contoh 4: Mengizinkan pengguna untuk memanggil API CreateServiceLinkedRole IAM](#)). Atau Anda perlu membuat peran tertaut layanan ElastiCache.

Saat digabungkan dengan izin minimum yang diperlukan untuk menggunakan konsol Amazon ElastiCache, contoh kebijakan di bagian ini memberikan izin tambahan. Contoh ini juga relevan untuk SDK AWS dan AWS CLI.

Untuk instruksi 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 tindakan ElastiCache yang tampak sederhana mungkin memerlukan dukungan tindakan lain saat Anda menggunakan konsol ElastiCache. Misalnya, `elasticache:CreateCacheCluster` memberikan izin untuk membuat klaster cache ElastiCache. Namun, untuk melakukan operasi ini, konsol ElastiCache menggunakan sejumlah tindakan `Describe` dan `List` untuk mengisi daftar konsol.

Contoh

- [Contoh 1: Mengizinkan akses baca-saja kepada pengguna ke sumber daya ElastiCache](#)
- [Contoh 2: Mengizinkan pengguna untuk melakukan tugas umum administrator sistem ElastiCache](#)
- [Contoh 3: Mengizinkan pengguna untuk mengakses semua tindakan API ElastiCache](#)
- [Contoh 4: Mengizinkan pengguna untuk memanggil API CreateServiceLinkedRole IAM](#)

Contoh 1: Mengizinkan akses baca-saja kepada pengguna ke sumber daya ElastiCache

Kebijakan berikut memberikan izin tindakan ElastiCache yang mengizinkan pengguna untuk menampilkan daftar sumber daya. Biasanya, Anda menyematkan jenis kebijakan izin ini untuk grup manajer.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "ECReadOnly",
    "Effect": "Allow",
    "Action": [
      "elasticache:Describe*",
      "elasticache:List*"
    ],
    "Resource": "*"
  }
]
```

Contoh 2: Mengizinkan pengguna untuk melakukan tugas umum administrator sistem ElastiCache

Tugas umum administrator sistem termasuk mengubah sumber daya. Administrator sistem mungkin juga ingin mendapatkan informasi tentang peristiwa ElastiCache. Kebijakan berikut memberikan izin kepada pengguna untuk melakukan tindakan ElastiCache untuk tugas umum administrator sistem tersebut. Biasanya, Anda menyematkan jenis kebijakan izin ini untuk grup administrator sistem.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "ECAAllowMutations",
    "Effect": "Allow",
    "Action": [
      "elasticache:Modify*",
      "elasticache:Describe*",

```

```

        "elasticache:ResetCacheParameterGroup"
    ],
    "Resource": "*"
}
]
}

```

Contoh 3: Mengizinkan pengguna untuk mengakses semua tindakan API ElastiCache

Kebijakan berikut mengizinkan pengguna mengakses semua tindakan ElastiCache. Sebaiknya Anda memberikan jenis kebijakan izin ini hanya untuk pengguna administrator.

```

{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "ECAAllowAll",
    "Effect": "Allow",
    "Action": [
      "elasticache:*"
    ],
    "Resource": "*"
  }
]
}

```

Contoh 4: Mengizinkan pengguna untuk memanggil API CreateServiceLinkedRole IAM

Kebijakan berikut mengizinkan pengguna untuk memanggil API CreateServiceLinkedRole IAM. Sebaiknya Anda memberikan jenis kebijakan izin ini kepada pengguna yang menginvokasi operasi ElastiCache mutatif.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CreateSLRAllows",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": "*",
      "Condition": {

```

```
    "StringLike":{
      "iam:AWSServiceName":"elasticache.amazonaws.com"
    }
  }
}
```

Izin tingkat sumber daya

Anda dapat membatasi cakupan izin dengan menentukan sumber daya dalam kebijakan IAM. Banyak tindakan API ElastiCache yang mendukung jenis sumber daya yang bervariasi tergantung pada perilaku tindakan itu. 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 ElastiCache dan ARN-nya, lihat [Sumber Daya yang Ditetapkan oleh Amazon ElastiCache](#) di Referensi Otorisasi Layanan. Untuk mempelajari jenis tindakan yang dapat Anda gunakan menentukan ARN dari setiap sumber daya, lihat [Tindakan yang Ditetapkan oleh Amazon ElastiCache](#).

Contoh

- [Contoh 1: Memberikan akses penuh pada pengguna ke jenis sumber daya ElastiCache tertentu](#)
- [Contoh 2: Menolak akses pengguna ke cache nirserver.](#)

Contoh 1: Memberikan akses penuh pada pengguna ke jenis sumber daya ElastiCache 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 syarat

Anda dapat menentukan kondisi yang menentukan penerapan kebijakan IAM. Pada ElastiCache, Anda dapat menggunakan elemen `Condition` kebijakan JSON untuk membandingkan kunci dalam konteks permintaan dengan nilai kunci yang Anda tentukan di dalam kebijakan Anda. Untuk informasi selengkapnya, lihat [Elemen kebijakan JSON IAM: Syarat](#).

Untuk melihat daftar kunci-kunci persyaratan ElastiCache, lihat [Kunci persyaratan untuk Amazon ElastiCache](#) di Rujukan Otorisasi Layanan.

Untuk melihat daftar kunci syarat global, lihat [Kunci konteks syarat global AWS](#).

Menentukan Syarat: Menggunakan Kunci Syarat

Untuk mengimplementasikan kontrol yang lebih spesifik, Anda menuliskan kebijakan izin IAM yang menentukan syarat untuk mengontrol set parameter individu pada permintaan tertentu. Kemudian Anda menerapkan kebijakan ke pengguna, grup, atau peran IAM yang Anda buat menggunakan konsol IAM.

Untuk menerapkan persyaratan tersebut, Anda menambahkan informasi persyaratan pada pernyataan kebijakan IAM. Pada contoh berikut, Anda menentukan syarat bahwa setiap kluster cache yang dirancang sendiri akan menjadi jenis simpul `cache.r5.large`.

```
{  
  "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: [Contoh kebijakan kontrol akses Berbasis Tag](#).

Untuk informasi selengkapnya penggunaan operator syarat kebijakan, lihat [Izin API ElastiCache: Referensi tindakan, sumber daya, dan kondisi](#).

Kebijakan Contoh: Menggunakan Syarat untuk Kontrol Parameter Terperinci

Bagian ini menunjukkan contoh kebijakan untuk mengimplementasikan kontrol akses yang terperinci pada parameter ElastiCache yang dicantumkan sebelumnya.

1. `elasticache:MaximumDataStorage`: Menentukan penyimpanan data maksimum cache nirserver. Dengan menggunakan syarat yang disediakan, pelanggan tidak dapat membuat cache yang dapat menyimpan lebih dari jumlah data ditentukan.

```

{
  "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:MaximumECPUPerSecond`: Menentukan nilai ECPU maksimum per detik dari cache nirsriver. Dengan menggunakan syarat yang disediakan, pelanggan tidak dapat membuat cache yang dapat menjalankan ECPU per detik lebih dari dari jumlah yang ditentukan.

```

{
  "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`: Menentukan `NodeType` yang dapat dibuat pengguna. Dengan syarat yang disediakan, pelanggan dapat menentukan nilai tunggal atau rentang untuk jenis simpul.

```

{
  "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"
            ]
        }
    }
}
]
}

```

4. elasticache:EngineVersion: Menentukan penggunaan mesin versi 1.6.6.

```

{
    "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"
        }
    }
}
]
}

```

5. `elasticache:KmsKeyId`: Menentukan penggunaan kunci KMS AWS yang dikelola pelanggan.

```

{
  "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"
        }
      }
    }
  ]
}

```

```
]
}
```

6. `elasticache:CacheParameterGroupName`: Menentukan grup parameter non default dengan parameter tertentu dari organisasi pada kluster Anda. Anda juga dapat menentukan pola penamaan untuk grup parameter Anda atau memblokir dan menghapus nama grup parameter tertentu. Berikut ini adalah contoh pembatasan penggunaan “my-org-param-group” saja.

```
{
  "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"
        }
      }
    }
  ]
}
```

7. `elasticache:CreateCacheCluster`: Menolak tindakan `CreateCacheCluster` jika tag permintaan Project hilang atau tidak sama dengan Dev, QA atau Prod.

```
{
  "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"
          ]
        }
      }
    }
  ]
}
```

```

    }
  }
}
]
}

```

8. `elasticache:CacheNodeType`: Mengizinkan `CreateCacheCluster` dengan `cacheNodeType` `cache.r5.large` atau `cache.r6g.4xlarge` dan tag `Project=XYZ`.

```

{
  "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 menegakkan tag dan kunci syarat lainnya secara bersama, syarat `IfExists` mungkin diperlukan pada elemen kunci syarat karena persyaratan kebijakan tambahan `eLasticache:AddTagsToResource` untuk permintaan pembuatan dengan parameter `--tags`.

Menggunakan Peran Tertaut Layanan untuk Amazon ElastiCache

Amazon ElastiCache menggunakan [peran tertaut layanan](#) AWS Identity and Access Management (IAM). Peran tertaut layanan adalah tipe peran IAM unik yang tertaut langsung ke layanan AWS, seperti Amazon ElastiCache. Peran tertaut layanan Amazon ElastiCache telah ditetapkan sebelumnya oleh Amazon ElastiCache. Mereka menyertakan semua izin yang diperlukan layanan untuk memanggil layanan AWS atas nama kluster Anda.

Peran tertaut layanan mempermudah pengaturan Amazon ElastiCache karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Peran sudah ada di dalam akun AWS Anda, tetapi tertaut ke kasus penggunaan Amazon ElastiCache dan memiliki izin yang telah ditetapkan. Hanya Amazon ElastiCache yang dapat mengambil peran ini, dan hanya peran ini yang dapat menggunakan kebijakan izin yang telah ditetapkan. Anda dapat menghapus peran tersebut hanya setelah pertama kali menghapus sumber dayanya yang terkait. Hal ini melindungi sumber daya Amazon ElastiCache karena Anda tidak dapat menghapus izin untuk mengakses sumber daya secara tidak sengaja.

Untuk informasi tentang layanan lain yang mendukung peran tertaut layanan, lihat [Layanan AWS yang bisa digunakan dengan IAM](#) dan cari layanan yang memiliki opsi Ya di kolom Peran Tertaut Layanan. Pilih Yes (Ya) bersama tautan untuk melihat dokumentasi peran tertaut layanan untuk layanan tersebut.

Daftar Isi

- [Izin Peran Tertaut Layanan untuk Amazon ElastiCache](#)
 - [Izin untuk membuat peran tertaut layanan](#)
- [Membuat Peran Tertaut Layanan \(IAM\)](#)
 - [Membuat Peran Tertaut Layanan \(Konsol IAM\)](#)

- [Membuat Peran Tertaut Layanan \(CLI IAM\)](#)
- [Membuat Peran Tertaut Layanan \(API IAM\)](#)
- [Mengedit Deskripsi Peran Tertaut Layanan untuk Amazon ElastiCache](#)
 - [Mengedit Deskripsi Peran Tertaut Layanan \(Kebijakan IAM\)](#)
 - [Mengedit Deskripsi Peran Tertaut Layanan \(CLI IAM\)](#)
 - [Mengedit Deskripsi Peran Tertaut Layanan \(API IAM\)](#)
- [Menghapus Peran Tertaut Layanan untuk Amazon ElastiCache](#)
 - [Membersihkan Peran Tertaut Layanan](#)
 - [Menghapus Peran Tertaut Layanan \(Konsol IAM\)](#)
 - [Menghapus Peran Tertaut Layanan \(IAM CLI\)](#)
 - [Menghapus Peran Terkait Layanan \(API IAM\)](#)

Izin Peran Tertaut Layanan untuk Amazon ElastiCache

Izin untuk membuat peran tertaut layanan

Untuk mengizinkan entitas IAM membuat AWSperan tertaut layanan ServiceRoleForElastiCache

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/AWSServiceRoleForElastiCache*",
  "Condition": {"StringLike": {"iam:AWSserviceName": "elasticache.amazonaws.com"}}
}
```

Untuk mengizinkan entitas IAM menghapus peran tertaut layanan AWSServiceRoleForElastiCache

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/
AWSServiceRoleForElastiCache*",
    "Condition": {"StringLike": {"iam:AWSServiceName": "elasticache.amazonaws.com"}}
}
```

Anda juga dapat menggunakan kebijakan AWS terkelola untuk memberikan akses penuh ke Amazon ElastiCache.

Membuat Peran Tertaut Layanan (IAM)

Anda dapat membuat peran tertaut layanan menggunakan konsol IAM, CLI, atau API.

Membuat Peran Tertaut Layanan (Konsol IAM)

Anda dapat menggunakan konsol IAM untuk membuat peran tertaut layanan.

Untuk membuat peran tertaut 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 tipe entitas tepercaya, pilih Layanan AWS.
4. Di bagian Atau pilih layanan untuk melihat kasus penggunaannya, pilih ElastiCache.
5. Pilih Selanjutnya: Izin.
6. Di bagian Nama kebijakan, perhatikan bahwa `ElastiCacheServiceRolePolicy` diperlukan untuk peran ini. Pilih Selanjutnya: Tanda.
7. Perhatikan bahwa tanda tidak didukung untuk peran Tertaut-Layanan. Pilih Selanjutnya: Tinjau.
8. (Opsional) Untuk Deskripsi peran, edit deskripsi untuk peran tertaut layanan baru.
9. Tinjau peran, lalu pilih Buat peran.

Membuat Peran Tertaut Layanan (CLI IAM)

Anda dapat menggunakan operasi IAM dari AWS Command Line Interface untuk membuat peran tertaut layanan. Peran ini dapat mencakup kebijakan kepercayaan dan kebijakan terkait yang diperlukan oleh layanan untuk mengambil peran tersebut.

Untuk membuat peran tertaut layanan (CLI)

Gunakan operasi berikut:

```
$ aws iam create-service-linked-role --aws-service-name elasticache.amazonaws.com
```

Membuat Peran Tertaut Layanan (API IAM)

Anda dapat menggunakan API IAM untuk membuat peran tertaut layanan. Peran ini dapat berisi kebijakan kepercayaan dan kebijakan terkait yang diperlukan oleh layanan untuk mengambil peran tersebut.

Untuk membuat peran tertaut 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 untuk mengedit peran tertaut layanan `AWSServiceRoleForElastiCache`. Setelah Anda membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat mengedit penjelasan peran menggunakan IAM.

Mengedit Deskripsi Peran Tertaut Layanan (Kebijakan IAM)

Anda dapat menggunakan konsol IAM untuk mengedit deskripsi peran tertaut layanan.

Untuk mengedit deskripsi peran tertaut 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 Tertaut Layanan (CLI IAM)

Anda dapat menggunakan operasi IAM dari AWS Command Line Interface untuk mengedit deskripsi peran tertaut layanan.

Untuk mengubah deskripsi peran tertaut layanan (CLI)

1. (Opsional) Untuk melihat deskripsi peran saat ini, gunakan AWS CLI untuk operasi IAM [get-role](#).

Example

```
$ aws iam get-role --role-name AWSServiceRoleForElastiCache
```

Gunakan nama peran, bukan ARN, untuk merujuk ke peran 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 tertaut layanan, gunakan AWS CLI untuk operasi IAM [update-role-description](#).

Untuk Linux, macOS, atau Unix:

```
$ aws iam update-role-description \  
  --role-name AWSServiceRoleForElastiCache \  
  --description "new description"
```

Untuk Windows:

```
$ aws iam update-role-description ^  
  --role-name AWSServiceRoleForElastiCache ^  
  --description "new description"
```

Mengedit Deskripsi Peran Tertaut Layanan (API IAM)

Anda dapat menggunakan API IAM untuk mengedit deskripsi peran tertaut layanan.

Untuk mengubah deskripsi peran tertaut layanan (API)

1. (Opsional) Untuk melihat deskripsi peran saat ini, gunakan operasi API IAM [GetRole](#).

Example

```
https://iam.amazonaws.com/  
?Action=GetRole
```

```
&RoleName=AWSServiceRoleForElastiCache
&Version=2010-05-08
&AUTHPARAMS
```

2. Untuk memperbarui deskripsi peran, gunakan operasi API IAM [UpdateRoleDescription](#).

Example

```
https://iam.amazonaws.com/
?Action=UpdateRoleDescription
&RoleName=AWSServiceRoleForElastiCache
&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 tertaut layanan, sebaiknya hapus peran tersebut. Dengan begitu, Anda tidak perlu lagi memantau atau memelihara entitas yang tidak digunakan. Namun, Anda harus membersihkan peran tertaut layanan sebelum dapat menghapusnya.

Amazon ElastiCache tidak menghapus peran tertaut layanan untuk Anda.

Membersihkan Peran Tertaut Layanan

Sebelum Anda dapat menggunakan IAM untuk menghapus peran tertaut layanan, pastikan terlebih dahulu bahwa peran tersebut tidak memiliki sumber daya—klaster—yang terkait dengannya.

Untuk memastikan peran tertaut 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. Lalu pilih nama (bukan kotak centang) peran `AWSServiceRoleForElastiCache`.
3. Di halaman Ringkasan untuk peran yang dipilih, pilih tab Penasihat Akses.
4. Di tab Penasihat Akses, tinjau aktivitas terbaru untuk peran tertaut layanan tersebut.

Untuk menghapus sumber daya Amazon ElastiCache yang memerlukan `AWSServiceRoleForElasticache`

- Untuk menghapus kluster, lihat referensi berikut:
 - [Menggunakan AWS Management Console](#)
 - [Menggunakan AWS CLI](#)
 - [Menggunakan API ElastiCache](#)

Menghapus Peran Tertaut Layanan (Konsol IAM)

Anda dapat menggunakan konsol IAM untuk menghapus peran tertaut layanan.

Untuk menghapus peran tertaut 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 akses terakhir layanan, yang menunjukkan waktu terakhir setiap peran yang dipilih mengakses layanan AWS. Hal ini membantu Anda mengonfirmasi aktif tidaknya peran tersebut saat ini. Jika Anda ingin melanjutkan, pilih Ya, Hapus guna mengirimkan peran tertaut layanan untuk penghapusan.
5. Perhatikan notifikasi konsol IAM untuk memantau progres penghapusan peran tertaut layanan. Karena penghapusan peran tertaut 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 Tertaut Layanan (IAM CLI)

Anda dapat menggunakan operasi IAM dari AWS Command Line Interface untuk menghapus peran tertaut layanan.

Untuk menghapus peran yang terhubung dengan layanan (CLI)

1. Jika Anda tidak tahu nama peran tertaut layanan yang ingin dihapus, masukkan perintah berikut. Perintah ini menampilkan daftar peran dan Amazon Resource Names (ARN) 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 tertaut layanan tidak dapat dihapus jika sedang digunakan atau memiliki sumber daya terkait, Anda harus mengirimkan permintaan penghapusan. Permintaan tersebut dapat ditolak jika syarat ini tidak terpenuhi. Anda harus menangkap `deletion-task-id` dari tanggapan untuk memeriksa status tugas penghapusan. Masukkan perintah berikut untuk mengirimkan permintaan penghapusan peran tertaut 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
```

Kemungkinan 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 tertaut layanan.

Untuk menghapus peran tertaut layanan (API)

1. Untuk mengirimkan permintaan penghapusan untuk peran tertaut layanan, panggil [DeleteServiceLinkedRole](#). Di permintaan tersebut, tentukan nama peran.

Karena peran tertaut layanan tidak dapat dihapus jika sedang digunakan atau memiliki sumber daya terkait, Anda harus mengirimkan permintaan penghapusan. Permintaan tersebut dapat

ditolak jika syarat 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`.

Kemungkinan 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.

Izin API ElastiCache: Referensi tindakan, sumber daya, dan kondisi

Saat Anda menyiapkan [kontrol akses](#) dan menulis kebijakan izin untuk melampirkan kebijakan IAM (baik berbasis identitas atau berbasis sumber daya), gunakan tabel berikut sebagai referensi. Tabel berikut ini mencantumkan setiap operasi API Amazon ElastiCache dan tindakan-tindakan terkait yang dapat Anda berikan izin untuk melakukan tindakan tersebut. 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 baik sumber daya yang diperlukan maupun sumber daya opsional. Jika tidak terdapat ARN sumber daya, sumber daya dalam kebijakan adalah wildcard (*).

Anda dapat menggunakan kunci syarat di seluruh di kebijakan ElastiCache untuk menyatakan syarat. Untuk melihat daftar berisi kunci kondisi khusus ElastiCache bersama dengan tindakan dan jenis sumber daya tujuan penerapannya, lihat [Menggunakan kunci syarat](#). Untuk daftar lengkap kunci seluruh AWS, lihat [Kunci konteks syarat global AWS AWS](#) dalam Panduan Pengguna IAM.

Note

Untuk menentukan tindakan, gunakan awalan `elasticache:` diikuti dengan nama operasi API (misalnya, `elasticache:DescribeCacheClusters`).

Untuk melihat daftar tindakan ElastiCache, lihat [Tindakan yang ditentukan oleh Amazon ElastiCache](#) di Rujukan Otorisasi Layanan.

Validasi kepatuhan untuk Amazon ElastiCache


Auditor pihak ketiga menilai keamanan dan kepatuhan layanan AWS sebagai bagian dari beberapa program kepatuhan AWS, seperti SOC, PCI, FedRAMP, dan HIPAA.

Untuk mempelajari apakah Layanan AWS berada dalam lingkup program kepatuhan khusus, lihat [Layanan AWS di Scope oleh Program](#) Program Kepatuhan yang Anda minati. Untuk informasi umum, lihat [Program Kepatuhan 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 berdasarkan sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, serta hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- [Panduan Mulai Cepat Keamanan dan Kepatuhan](#) – Panduan deployment ini membahas pertimbangan arsitektur dan menyediakan langkah-langkah untuk melakukan deployment lingkungan dasar di AWS yang menjadi fokus keamanan dan kepatuhan.
- [Merancang Keamanan dan Kepatuhan HIPAA di Amazon Web Services](#) – Laporan resmi ini menjelaskan cara perusahaan dapat menggunakan AWS untuk membuat aplikasi yang memenuhi syarat HIPAA.

 Note

Tidak semua Layanan AWS memenuhi syarat HIPAA. Untuk informasi selengkapnya, lihat [Referensi Layanan yang Memenuhi Syarat HIPAA](#).

- [Sumber Daya Kepatuhan AWS](#) – Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- [Panduan Kepatuhan Pelanggan AWS](#) – Pahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan ini merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan kontrol keamanan di banyak kerangka kerja (termasuk National Institute of Standards and Technology (NIST), Payment Card Industry Security Standards Council (PCI), dan International Organization for Standardization (ISO)).
- [Mengevaluasi Sumber Daya dengan Aturan](#) dalam Panduan Developer AWS Config – Layanan AWS Config menilai seberapa baik konfigurasi sumber daya Anda dalam mematuhi praktik-praktik internal, pedoman industri, dan regulasi internal.
- [AWS Security Hub](#) – Layanan AWS ini memberikan gambaran komprehensif tentang status keamanan Anda di dalam 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](#).
- [AWS Audit Manager](#) – Layanan AWS ini akan membantu Anda untuk terus-menerus mengaudit penggunaan AWS untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap regulasi dan standar industri.

Informasi selengkapnya

Untuk informasi umum tentang kepatuhan AWS Cloud, lihat hal berikut:

- [Titik Akhir FIPS berdasarkan Layanan](#)
- [Pembaruan layanan di ElastiCache](#)
- [Kepatuhan AWS Cloud](#)
- [Model Tanggung Jawab Bersama](#)
- [Program Kepatuhan AWS PCI DSS](#)

Ketahanan di Amazon ElastiCache

Infrastruktur global AWS dibangun berdasarkan Wilayah dan Zona Ketersediaan AWS. Wilayah AWS menyediakan beberapa Zona Ketersediaan yang terpisah dan terisolasi secara fisik yang terhubung dengan jaringan yang memiliki latensi rendah, throughput tinggi, dan redundansi tinggi. Dengan Zona Ketersediaan, Anda dapat merancang serta 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 dengan satu atau beberapa infrastruktur pusat data biasa.

Untuk informasi selengkapnya tentang Wilayah AWS dan Zona Ketersediaan, lihat [Infrastruktur Global AWS](#).

Selain infrastruktur global AWS, Amazon ElastiCache memberikan beberapa fitur untuk membantu mendukung kebutuhan ketahanan dan pencadangan data 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 mencakup pendekatan yang dapat Anda ambil untuk melindungi aplikasi dan data Anda dari kegagalan.

Topik

- [Mitigasi Kegagalan saat Menjalankan Memcached](#)

- [Rekomendasi](#)

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 (AZ).

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 kluster yang dirancang sendiri, sebarkan data cache Anda ke lebih banyak simpul. Karena kluster yang dirancang sendiri tidak mendukung replikasi, kegagalan simpul akan selalu mengakibatkan hilangnya beberapa data dari kluster 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 pembuatan dan provisi simpul pengganti. Jika data yang sama disimpan pada cache di 3 simpul yang besar, maka kegagalan satu simpul akan menghilangkan sekitar 33% dari data cache Anda.

Jika Anda membutuhkan lebih dari 60 node dalam cluster Memcached, atau lebih dari 300 node total di AWS Region, isi formulir Permintaan Peningkatan ElastiCache Batas di <https://aws.amazon.com/contact-us/elasticache-node-limit-request/>.

Untuk informasi tentang menentukan jumlah simpul dalam sebuah kluster Memcached, lihat [Membuat kluster 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 kluster yang dirancang sendiri, tempatkan simpul Anda di sebanyak mungkin Zona Ketersediaan. Jika sampai terjadi kegagalan AZ, Anda akan kehilangan data cache pada AZ tersebut, bukan data cache di AZ yang lain.

Mengapa perlu banyak simpul?

Jika wilayah saya hanya memiliki 3 Zona Tersedia, 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 yang diupayakan adalah 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 kluster Anda. Anda harus menimbang dampak kehilangan data vs kemungkinan kegagalan vs biaya, dan mendapatkan kesimpulan Anda sendiri.

Untuk informasi mengenai penentuan jumlah simpul dalam sebuah kluster Memcached, lihat [Membuat kluster Memcached \(konsol\)](#).

Untuk informasi lain mengenai wilayah dan Zona Ketersediaan, lihat [Wilayah dan Zona Ketersediaan](#).

Rekomendasi

Sebaiknya buat cache nirserver alih-alih kluster yang dirancang sendiri, karena Anda secara otomatis mendapatkan toleransi kesalahan yang lebih baik tanpa konfigurasi tambahan. Saat membuat kluster yang dirancang sendiri, ada dua jenis kegagalan yang perlu direncanakan: kegagalan simpul tunggal dan kegagalan Zona Ketersediaan yang luas. Rencana mitigasi kegagalan terbaik akan mengatasi kedua jenis kegagalan tersebut.

Meminimalkan Dampak Kegagalan Simpul

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 Availability Zone yang tersedia. Menyebarkan simpul Anda secara merata di seluruh AZ akan meminimalkan dampak jika terjadi kegagalan AZ. Ini dilakukan secara otomatis untuk cache nirserver.

Keamanan infrastruktur di AWS ElastiCache

Sebagai layanan terkelola, AWS ElastiCache dilindungi oleh prosedur keamanan jaringan global AWS yang dijelaskan pada bagian Keamanan dan Kepatuhan di [Pusat Arsitektur AWS](#).

Gunakan panggilan API AWS yang 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-sistem modern seperti Java 7 dan versi yang lebih baru mendukung mode-mode ini.

Selain itu, permintaan harus ditandatangani menggunakan ID kunci akses dan kunci akses rahasia yang dikaitkan dengan prinsipal IAM. Atau Anda dapat menggunakan [AWS Security Token Service](#) (AWS STS) untuk membuat kredensial keamanan sementara untuk 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.

Mengelola pembaruan layanan

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.

Topik

- [Menerapkan pembaruan layanan](#)
- [Memverifikasi bahwa Anda memiliki Pembaruan Layanan terbaru yang Diterapkan menggunakan AWS konsol](#)
- [Menghentikan pembaruan layanan](#)

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 telah mengaktifkan pembaruan otomatis, Anda dapat memilih untuk tidak mengambil tindakan apa pun saat pembaruan tersebut 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 kluster Anda.
 - Direkomendasikan Terapkan Berdasarkan Tanggal: tanggal ElastiCache panduan untuk menerapkan pembaruan oleh.
 - Status: Status pembaruan, yang merupakan salah satu dari berikut ini:
 - tersedia: Pembaruan tersedia untuk kluster 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

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 lebih lanjut, 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
```

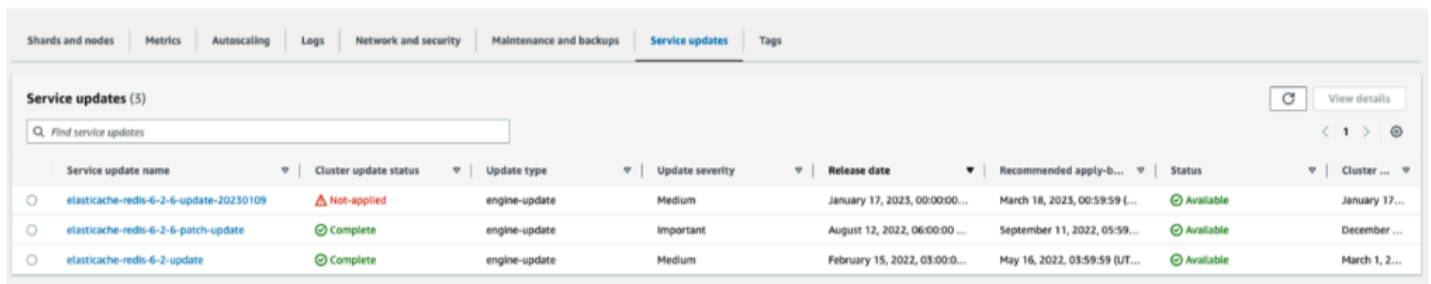
Untuk informasi lebih lanjut, lihat [batch-apply-update-action](#).

Memverifikasi bahwa Anda memiliki Pembaruan Layanan terbaru yang Diterapkan menggunakan AWS konsol

Anda dapat ElastiCache memverifikasi bahwa kluster Redis menjalankan pembaruan layanan terbaru dengan mengikuti langkah-langkah berikut:

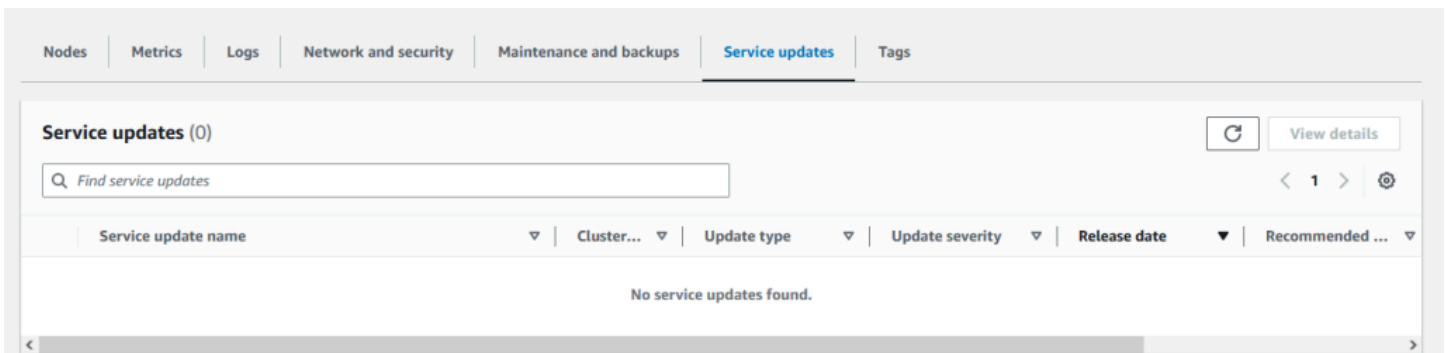
1. Pilih kluster yang berlaku di halaman Redis Clusters
2. Pilih Pembaruan layanan di panel navigasi untuk melihat pembaruan layanan yang berlaku untuk kluster tersebut, jika ada.

Jika konsol menampilkan daftar pembaruan layanan, Anda dapat memilih pembaruan layanan dan memilih Terapkan sekarang.



Service update name	Cluster update status	Update type	Update severity	Release date	Recommended apply-b...	Status	Cluster ...
elasticsearch-redis-6-2-6-update-20230109	Not-applied	engine-update	Medium	January 17, 2023, 00:00:00...	March 18, 2023, 00:59:59 (...)	Available	January 17...
elasticsearch-redis-6-2-6-patch-update	Complete	engine-update	Important	August 12, 2022, 06:00:00 ...	September 11, 2022, 05:59...	Available	December ...
elasticsearch-redis-6-2-update	Complete	engine-update	Medium	February 15, 2022, 03:00:0...	May 16, 2022, 03:59:59 (UT...	Available	March 1, 2...

Jika konsol menampilkan “Tidak ada pembaruan layanan yang ditemukan”, itu berarti kluster ElastiCache for Redis sudah menerapkan pembaruan layanan terbaru.



Service update name	Cluster...	Update type	Update severity	Release date	Recommended ...
No service updates found.					

Menghentikan pembaruan layanan

Anda dapat menghentikan pembaruan pada kluster jika diperlukan. Misalnya, Anda mungkin ingin menghentikan pembaruan jika Anda mengalami lonjakan tak terduga pada kluster yang sedang diperbarui. Atau Anda mungkin ingin 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 cluster 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
```


Untuk informasi lebih lanjut, lihat [BatchStopUpdateAction](#).

Logging dan pemantauan di Amazon ElastiCache

Untuk mengelola cache Anda, penting bagi Anda untuk mengetahui performa cache Anda. ElastiCache menghasilkan metrik yang diterbitkan ke Log Amazon CloudWatch untuk memantau performa cache Anda. Selain itu, ElastiCache menghasilkan peristiwa ketika perubahan signifikan terjadi pada sumber daya cache Anda (misalnya cache baru dibuat, atau cache dihapus).

Topik

- [Metrik dan peristiwa nirserver](#)
- [Metrik dan peristiwa klaster yang dirancang sendiri](#)
- [Logging panggilan API Amazon ElastiCache dengan AWS CloudTrail](#)
- [Logging panggilan API Amazon ElastiCache dengan AWS CloudTrail](#)

Metrik dan peristiwa nirserver

Bagian ini menjelaskan metrik dan peristiwa yang dapat Anda pantau saat bekerja dengan cache nirserver.

Topik

- [Metrik cache nirserver](#)
- [Peristiwa cache nirserver](#)

Metrik cache nirserver

Namespace AWS/ElastiCache menyertakan metrik CloudWatch berikut untuk cache nirserver Redis.

Metrik	Deskripsi	Unit
BytesUsedForCache	Jumlah total byte yang digunakan oleh data yang disimpan dalam cache Anda.	Bytes

Metrik	Deskripsi	Unit
ElastiCacheProcessingUnits	Jumlah total ElastiCacheProcessingUnits (ECPUs) yang dikonsumsi oleh permintaan yang dijalankan di cache Anda	Jumlah
SuccessfulReadRequestLatency	Latensi permintaan baca yang berhasil.	Mikrodetik
SuccessfulWriteRequestLatency	Latensi permintaan tulis yang berhasil	Mikrodetik
TotalCmdsCount	Jumlah total semua perintah yang dijalankan pada cache Anda	Jumlah
CurrConnections	Jumlah koneksi klien ke cache Anda.	Jumlah
ThrottledCmds	Jumlah permintaan yang mendapat throttling oleh ElastiCache karena beban kerja menskalakan lebih cepat daripada yang dapat diskalakan oleh ElastiCache.	Jumlah
NewConnections	Jumlah seluruh koneksi yang telah diterima oleh server selama periode ini.	Jumlah
CurrItems	Jumlah item dalam cache.	Jumlah
NetworkBytesIn	Total byte yang ditransfer ke cache	Bytes
NetworkBytesOut	Total byte yang ditransfer ke cache	Bytes

Metrik	Deskripsi	Unit
Pengosongan	Hitungan kunci yang dilakukan pengosongan oleh cache	Jumlah
Diklaim kembali	Jumlah kunci yang sudah tidak berlaku menurut cache	Jumlah

Metrik tingkat perintah

ElastiCache juga mengeluarkan metrik tingkat perintah Memcached berikut

Metrik	Deskripsi	Unit
CmdGet	Jumlah perintah get yang telah diterima oleh cache.	Jumlah
CmdSet	Jumlah perintah set yang telah diterima oleh cache.	Jumlah
CmdTouch	Jumlah perintah touch yang telah diterima oleh cache.	Jumlah
GetHits	Jumlah permintaan get yang telah diterima oleh cache di mana kunci yang diminta ditemukan.	Jumlah
GetMisses	Jumlah permintaan get yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Jumlah
IncrHits	Jumlah permintaan increment yang telah diterima oleh cache di mana kunci yang diminta ditemukan.	Jumlah

Metrik	Deskripsi	Unit
IncrMisses	Jumlah permintaan increment yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Jumlah
DecrHits	Jumlah permintaan decrement yang telah diterima oleh cache di mana kunci yang diminta ditemukan.	Jumlah
DecrMisses	Jumlah permintaan decrement yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Jumlah
DeleteHits	Jumlah permintaan delete yang telah diterima oleh cache di mana kunci yang diminta ditemukan.	Jumlah
DeleteMisses	Jumlah permintaan delete yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Jumlah
TouchHits	Jumlah kunci yang telah di-touch dan diberikan waktu habis masa berlaku yang baru.	Jumlah
TouchMisses	Jumlah kunci yang telah di-touch, tetapi tidak ditemukan.	Jumlah
CasHits	Jumlah permintaan cas yang telah diterima oleh cache di mana kunci yang diminta ditemukan dan nilai cas cocok.	Jumlah

Metrik	Deskripsi	Unit
CasMisses	Jumlah permintaan cas yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Jumlah
CasBadval	Jumlah permintaan cas yang telah diterima oleh cache di mana nilai cas tidak cocok dengan nilai cas yang tersimpan.	Jumlah
CmdFlush	Jumlah perintah flush yang telah diterima oleh cache.	Jumlah

Peristiwa cache nirserver

ElastiCache mencatat log peristiwa yang berhubungan dengan cache nirserver Anda. Informasi ini mencakup tanggal dan waktu peristiwa, nama sumber dan jenis sumber peristiwa, serta deskripsi dari peristiwa itu. Anda dapat dengan mudah mengambil peristiwa dari log menggunakan konsol ElastiCache, perintah `describe-events` AWS CLI, atau tindakan `DescribeEvents` API ElastiCache.

Anda dapat memilih untuk memantau, menyerap, mengubah, dan bertindak pada peristiwa ElastiCache menggunakan Amazon EventBridge. Pelajari lebih lanjut di Amazon EventBridge <https://docs.aws.amazon.com/eventbridge/latest/userguide/>.

Melihat peristiwa ElastiCache (Konsol)

Untuk melihat peristiwa menggunakan konsol ElastiCache:

1. Masuk ke AWS Management Console dan buka konsol ElastiCache di <https://console.aws.amazon.com/elasticache/>
2. Untuk melihat daftar dari semua peristiwa yang tersedia, pada panel navigasi, pilih Peristiwa.
3. Pada layar Peristiwa setiap baris dari daftar mewakili satu peristiwa dan menampilkan sumber peristiwa, jenis peristiwa, waktu GMT peristiwa, serta deskripsi dari peristiwa itu. Dengan

menggunakan Filter Anda dapat menentukan apakah Anda ingin melihat semua peristiwa, atau hanya peristiwa dari jenis tertentu dalam daftar peristiwa.

Melihat peristiwa ElastiCache (AWS CLI)

Untuk menghasilkan daftar peristiwa ElastiCache menggunakan AWS CLI, gunakan perintah `describe-events`. Anda dapat menggunakan parameter opsional untuk mengontrol jenis peristiwa yang tercantum, kerangka waktu peristiwa yang tercantum, jumlah maksimum peristiwa untuk dicantumkan, dan banyak lagi.

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 gratis tidak cukup untuk

Jenis-Detail	Deskripsi	Unit	Sumber	Pesan
				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-name>. Anda tidak memiliki izin untuk membuat titik akhir VPC.

Peristiwa Pembaruan Cache Nirserver

Jenis-Detail	Daftar sumber daya	Kategori	Sumber	Pesan
Cache diperbarui	Cache arn	perubahan konfigurasi	cache-nirserver	SecurityGroups diperbarui untuk <cache-name>.

Jenis-Detail	Daftar sumber daya	Kategori	Sumber	Pesan
Cache diperbarui	Cache arn	perubahan konfigurasi	cache-nirserver	Tag diperbarui untuk cache <cache-name>.
Pembaruan cache gagal	Cache arn	perubahan konfigurasi	cache-nirserver	Pembaruan ke cache <cache-name> gagal. Pembaruan GrupKeamanan gagal.
Pembaruan cache gagal	Cache arn	perubahan konfigurasi	cache-nirserver	Pembaruan ke cache <cache-name> gagal. Pembaruan GrupKeamanan gagal karena izin yang tidak mencukupi.
Pembaruan cache gagal	Cache arn	perubahan konfigurasi	cache-nirserver	Pembaruan ke cache <cache-name> gagal. Pembaruan GrupKeamanan gagal karena GrupKeamanan tidak valid.

Peristiwa Penghapusan Cache Nirserver

Jenis-Detail	Daftar sumber daya	Kategori	Sumber	Pesan
Cache dihapus	Cache arn	penghapusan	cache-nirserver	Cache <cache-name> telah dihapus.

Peristiwa Batas Penggunaan Cache Nirserver

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 cache telah dihapus.
Pembaruan cache gagal	Cache arn	kegagalan	cache-nirserver	Pembaruan batas ke cache <cache-name> gagal karena konfigurasi tidak valid.

Metrik dan peristiwa klaster yang dirancang sendiri

Bagian ini menjelaskan metrik, peristiwa, dan log yang dapat Anda lihat saat Anda menangani klaster yang dirancang sendiri.

Topik

- [Metrik dan peristiwa klaster yang dirancang sendiri](#)
- [Peristiwa untuk klaster yang dirancang sendiri](#)
- [Memantau penggunaan dengan Metrik CloudWatch](#)
- [Pemantauan Amazon SNS dari peristiwa ElastiCache](#)

Metrik dan peristiwa klaster yang dirancang sendiri

Saat Anda merancang klaster sendiri, ElastiCache menerbitkan metrik di setiap tingkat simpul, termasuk metrik tingkat host dan metrik cache.

Untuk informasi selengkapnya tentang metrik tingkat host untuk Memcached, lihat [Metrik Tingkat Host](#).

Untuk informasi selengkapnya tentang metrik Memcached tingkat simpul, lihat [Metrik untuk Memcached](#).

Peristiwa untuk klaster yang dirancang sendiri

Peristiwa log ElastiCache yang berkaitan dengan cache yang Anda rancang sendiri. Saat menangani klaster yang dirancang sendiri, Anda dapat melihat peristiwa klaster di konsol ElastiCache, menggunakan AWS CLI, atau menggunakan Amazon Simple Notification Service (SNS). Peristiwa klaster yang dirancang sendiri tidak diterbitkan 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 konsol ElastiCache, perintah describe-events AWS CLI, atau tindakan DescribeEvents API ElastiCache.

Melihat peristiwa ElastiCache (Konsol)

Prosedur berikut menampilkan peristiwa menggunakan konsol ElastiCache.

Untuk melihat peristiwa menggunakan konsol ElastiCache:

1. Masuk ke AWS Management Console dan buka konsol ElastiCache 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 daftar merepresentasikan satu peristiwa dan menampilkan sumber peristiwa, jenis peristiwa, waktu GMT peristiwa, serta deskripsi peristiwa tersebut.

Dengan menggunakan Filter, Anda dapat menentukan apakah Anda ingin melihat semua peristiwa, atau hanya peristiwa dari jenis tertentu dalam daftar peristiwa.

Melihat peristiwa ElastiCache (AWS CLI)

Untuk menghasilkan daftar peristiwa ElastiCache menggunakan AWS CLI, gunakan perintah `describe-events`. 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 nirserver 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.


Peristiwa ElastiCache berikut memicu notifikasi Amazon SNS. Untuk informasi tentang detail peristiwa, lihat [Melihat peristiwa ElastiCache](#).

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:AddCacheNodeComplete	ElastiCache:AddCacheNodeComplete : <i>cache-cluster</i>	Simpul cache telah ditambahkan ke klaster cache dan siap untuk digunakan.
ElastiCache:AddCacheNodeFailed karena alamat IP yang bebas tidak mencukupi	ElastiCache:AddCacheNodeFailed : <i>cluster-name</i>	Simpul cache tidak dapat ditambahkan karena alamat IP yang tersedia tidak cukup.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CacheClusterParametersChanged	ElastiCache:CacheClusterParametersChanged : <i>cluster-name</i>	Satu atau beberapa parameter klaster cache telah berubah.
ElastiCache:CacheClusterProvisioningComplete	ElastiCache:CacheClusterProvisioningComplete <i>cluster-name-0001-005</i>	Penyediaan klaster cache selesai, dan simpul cache dalam klaster cache siap untuk digunakan.
ElastiCache:CacheClusterProvisioningFailed karena status jaringan yang tidak kompatibel	ElastiCache:CacheClusterProvisioningFailed : <i>cluster-name</i>	Percobaan dilakukan untuk meluncurkan klaster cache baru ke cloud privat virtual (VPC) yang tidak ada.
ElastiCache:CacheClusterScalingComplete	CacheClusterScalingComplete : <i>cluster-name</i>	Penskalaan untuk klaster cache berhasil diselesaikan.
ElastiCache:CacheClusterScalingFailed	ElastiCache:CacheClusterScalingFailed : <i>nama klaster</i>	Operasi peningkatan skala pada klaster cache gagal.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CacheClusterSecurityGroupModified	ElastiCache:CacheClusterSecurityGroupModified : <i>cluster-name</i>	<p>Salah satu peristiwa berikut telah terjadi:</p> <ul style="list-style-type: none">• Daftar grup keamanan cache yang diotorisasi untuk klaster cache telah berubah.• Satu atau beberapa grup keamanan EC2 telah diotorisasi pada grup keamanan cache yang terkait dengan klaster cache.• Satu atau beberapa grup keamanan EC2 baru telah dicabut otorisasinya dari salah satu grup keamanan cache yang terkait dengan klaster cache.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CacheNodeReplaceStarted	ElastiCache:CacheNodeReplaceStarted : <i>cluster-name</i>	<p>ElastiCache telah mendeteksi bahwa host yang menjalankan simpul cache mengalami penurunan performa atau tidak dapat dijangkau dan telah mulai mengganti simpul cache.</p> <div data-bbox="1068 590 1507 856" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px;"><p> Note Entri DNS untuk simpul cache yang diganti tidak berubah.</p></div> <p>Pada kebanyakan kasus, Anda tidak perlu menyegarkan daftar server untuk klien Anda ketika peristiwa ini terjadi. Namun, beberapa pustaka klien cache mungkin berhenti menggunakan simpul cache bahkan setelah ElastiCache mengganti simpul cache; dalam hal ini, aplikasi harus menyegarkan daftar server ketika peristiwa ini terjadi.</p>

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CacheNodeReplaceComplete	ElastiCache:CacheNodeReplaceComplete : <i>cluster-name</i>	<p>ElastiCache mendeteksi bahwa host yang menjalankan simpul cache terdegradasi atau tidak terjangkau dan telah selesai mengganti simpul cache.</p> <div data-bbox="1068 541 1507 808" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; margin: 10px 0;"> <p> Note</p> <p>Entri DNS untuk simpul cache yang diganti tidak berubah.</p> </div> <p>Pada kebanyakan kasus, Anda tidak perlu menyegarkan daftar server untuk klien Anda ketika peristiwa ini terjadi. Namun, beberapa pustaka klien cache mungkin berhenti menggunakan simpul cache bahkan setelah ElastiCache mengganti simpul cache; dalam hal ini, aplikasi harus menyegarkan daftar server ketika peristiwa ini terjadi.</p>
ElastiCache:CacheNodesRebooted	ElastiCache:CacheNodesRebooted : <i>cluster-name</i>	<p>Satu atau beberapa simpul cache telah di-boot ulang.</p> <p>Pesan (Memcached): "Cache node %s shutdown" Kemudian pesan kedua: "Cache node %s restarted"</p>

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CertificateRenewalComplete (hanya Redis)	ElastiCache:CertificateRenewalComplete	Sertifikat Amazon CA berhasil diperbarui.
ElastiCache:CreateReplicationGroupComplete	ElastiCache:CreateReplicationGroupComplete : <i>cluster-name</i>	Grup replikasi berhasil dibuat.
ElastiCache>DeleteCacheClusterComplete	ElastiCache>DeleteCacheClusterComplete : <i>cluster-name</i>	Penghapusan klaster cache dan semua simpul cache terkait telah selesai.
ElastiCache:FailoverComplete (hanya Redis)	ElastiCache:FailoverComplete : <i>mycluster</i>	Failover ke simpul replika telah berhasil.
ElastiCache:ReplicationGroupIncreaseReplicaCountFinished	ElastiCache:ReplicationGroupIncreaseReplicaCountFinished : <i>cluster-name-0001-005</i>	Jumlah replika dalam klaster telah meningkat.
ElastiCache:ReplicationGroupIncreaseReplicaCountStarted	ElastiCache:ReplicationGroupIncreaseReplicaCountStarted : <i>cluster-name-0003-004</i>	Proses penambahan replika ke klaster Anda telah dimulai.
ElastiCache:NodeReplacementCanceled	ElastiCache:NodeReplacementCanceled : <i>cluster-name</i>	Sebuah simpul di klaster Anda yang dijadwalkan akan diganti tidak lagi dijadwalkan akan diganti.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:NodeReplacementRescheduled	ElastiCache:NodeReplacementRescheduled : <i>cluster-name</i>	<p>Sebuah simpul di kluster Anda yang sebelumnya dijadwalkan akan diganti telah dijadwalkan ulang untuk diganti selama periode baru yang dijelaskan dalam notifikasi.</p> <p>Untuk informasi tentang tindakan yang dapat Anda lakukan, lihat Mengganti simpul cache untuk Memcached.</p>
ElastiCache:NodeReplacementScheduled	ElastiCache:NodeReplacementScheduled : <i>cluster-name</i>	<p>Sebuah simpul di kluster Anda dijadwalkan akan diganti selama periode yang dijelaskan dalam notifikasi.</p> <p>Untuk informasi tentang tindakan yang dapat Anda lakukan, lihat Mengganti simpul cache untuk Memcached.</p>
ElastiCache:RemoveCacheNodeComplete	ElastiCache:RemoveCacheNodeComplete : <i>cluster-name</i>	Sebuah simpul cache telah dihapus dari kluster cache.
ElastiCache:ReplicationGroupScalingComplete	ElastiCache:ReplicationGroupScalingComplete : <i>cluster-name</i>	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 (hanya Redis)	ElastiCache:SnapshotComplete : <i>cluster-name</i>	Sebuah snapshot cache telah berhasil diselesaikan.
ElastiCache:SnapshotFailed (hanya Redis)	SnapshotFailed : <i>cluster-name</i>	Sebuah snapshot cache telah gagal. Lihat peristiwa kluster cache untuk penyebab yang lebih terperinci. Jika Anda mendeskripsikan snapshot, lihat DescribeSnapshots , statusnya adalah failed.

Memantau penggunaan dengan Metrik CloudWatch

ElastiCache menyediakan metrik yang memungkinkan Anda memantau kluster Anda. Anda dapat mengakses metrik ini melalui CloudWatch. Untuk informasi selengkapnya tentang CloudWatch, lihat [dokumentasi CloudWatch](#).

ElastiCache menyediakan metrik tingkat host (misalnya, penggunaan CPU) dan metrik yang khusus untuk perangkat lunak mesin cache (misalnya, cache get dan cache miss). Metrik ini diukur dan diterbitkan untuk setiap simpul Cache dalam interval 60 detik.

Important

Sebaiknya atur alarm CloudWatch untuk beberapa metrik penting tertentu, sehingga Anda akan diberi tahu jika performa kluster cache Anda mulai menurun. Untuk informasi selengkapnya, lihat [Metrik apa yang harus saya pantau?](#) dalam panduan ini.

Topik

- [Metrik Tingkat Host](#)
- [Metrik untuk Memcached](#)
- [Metrik apa yang harus saya pantau?](#)
- [Memantau Metrik Klaster dan Simpul CloudWatch](#)

Metrik Tingkat Host

Ruang nama AWS/ElastiCache mencakup metrik tingkat host berikut untuk simpul cache individual.

Lihat Juga

- [Metrik untuk Memcached](#)

Metrik	Deskripsi	Unit
CPUUtilization	Persentase pemanfaatan CPU untuk keseluruhan host.	Persen
CPUCreditBalance	<p>Jumlah kredit CPU yang diperoleh yang diakumulasi oleh instans sejak diluncurkan atau dimulai. Untuk T2 Standar, CPUCreditBalance juga mencakup jumlah kredit peluncuran yang telah diakumulasi.</p> <p>Kredit diakumulasikan 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.</p> <p>Kredit dalam CPUCreditBalance tersedia untuk instans untuk digunakan hingga melonjak melebihi pemanfaatan CPU acuan.</p>	Kredit (Menit vCPU)

Metrik	Deskripsi	Unit
	Metrik kredit CPU tersedia pada frekuensi lima menit saja.	
CPUCreditUsage	<p>Jumlah kredit CPU yang digunakan oleh instans untuk pemanfaatan CPU. Satu kredit CPU sama dengan satu vCPU yang berjalan dengan pemanfaatan 100% selama satu menit atau kombinasi yang setara dari vCPU, pemanfaatan, dan waktu (misalnya, satu vCPU yang berjalan dengan pemanfaatan 50% selama dua menit atau dua vCPU yang berjalan dengan pemanfaatan 25% selama dua menit).</p> <p>Metrik kredit CPU tersedia pada frekuensi lima menit saja. Jika Anda menentukan periode lebih dari lima menit, gunakan statistik Sum, bukan statistik Average.</p>	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
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 yang masuk dari segi jumlah paket pada instans tunggal.	Jumlah

Metrik	Deskripsi	Unit
NetworkPacketsOut	Jumlah paket yang dikirimkan di semua antarmuka jaringan oleh instans. Metrik ini mengidentifikasi volume lalu lintas yang keluar dari segi jumlah paket pada instans tunggal.	Jumlah
NetworkBandwidthIn AllowanceExceeded	Jumlah paket yang di-shaping karena bandwidth agregat masuk melebihi nilai maksimum untuk instans.	Jumlah
NetworkConntrackAllowanceExceeded	Jumlah paket yang di-shaping karena pelacakan koneksi melebihi nilai maksimum untuk instans dan koneksi baru tidak dapat dibuat. Hal ini dapat mengakibatkan hilangnya paket untuk lalu lintas ke atau dari instans.	Jumlah
NetworkBandwidthOut AllowanceExceeded	Jumlah paket yang di-shaping karena bandwidth agregat keluar melebihi nilai maksimum untuk instans.	Jumlah
Network Packets Per Second Allowance Exceeded	Jumlah paket yang di-shaping karena paket dua arah per detik melebihi nilai maksimum untuk instans.	Jumlah
NetworkMaxBytesIn	Semburan maksimum byte yang diterima dalam setiap menit.	Byte
NetworkMaxBytesOut	Semburan maksimum byte yang ditransmisikan dalam setiap menit.	Byte
NetworkMaxPacketsIn	Semburan maksimum paket yang diterima dalam setiap menit.	Jumlah
NetworkMaxPacketsOut	Semburan maksimum paket yang ditransmisikan dalam setiap menit.	Jumlah
SwapUsage	Jumlah swap yang digunakan oleh host.	Byte

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
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

Metrik	Deskripsi	Unit
<code>CmdSet</code>	Jumlah perintah set yang telah diterima oleh cache.	Hitungan
<code>CurrConnections</code>	<p>Hitungan jumlah koneksi yang terhubung ke cache pada satu waktu. ElastiCache menggunakan dua hingga tiga koneksi untuk memantau cluster.</p> <p>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 <code>Nodetype Specific Parameters</code> dari Grup Parameter yang berlaku.</p> <p>Koneksi total adalah jumlah dari koneksi klien, koneksi untuk pemantauan, dan koneksi internal yang disebutkan di atas.</p>	Hitungan
<code>CurrItems</code>	Hitungan jumlah item yang saat ini disimpan di dalam cache.	Hitungan
<code>DecrHits</code>	Jumlah permintaan decrement yang telah diterima oleh cache di mana kunci yang diminta ditemukan.	Hitungan
<code>DecrMisses</code>	Jumlah permintaan decrement yang telah diterima oleh cache di mana kunci yang diminta tidak ditemukan.	Hitungan
<code>DeleteHits</code>	Jumlah permintaan delete yang telah diterima oleh cache di mana kunci yang diminta ditemukan.	Hitungan

Metrik	Deskripsi	Unit
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
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

Metrik	Deskripsi	Unit
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

Metrik	Deskripsi	Unit
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	<p>Jumlah memori yang tidak digunakan oleh data. Hal ini berasal dari statistik limit_max bytes dan bytes Memcached dengan mengurangi bytes dari limit_maxbytes.</p> <p>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.</p> <p>Untuk informasi yang lebih mendetail, lihat Memcached item memory usage.</p>	Byte

Metrik apa yang harus saya pantau?

Metrik CloudWatch berikut menawarkan wawasan yang bermanfaat tentang performa ElastiCache. Pada umumnya, kami merekomendasikan Anda untuk mengatur alarm CloudWatch untuk metrik ini agar Anda dapat mengambil tindakan korektif sebelum masalah performa terjadi.

Metrik yang Perlu Dipantau

- [CPUUtilization](#)
- [SwapUsage](#)
- [Evictions](#)
- [CurrConnections](#)

CPUUtilization

Ini adalah metrik tingkat host yang dilaporkan sebagai persentase. Untuk informasi selengkapnya, lihat [Metrik Tingkat Host](#).

Karena Memcached bersifat multi-thread, metrik ini dapat mencapai 90%. Jika Anda melebihi ambang batas yang dipilih, naikkan skala kluster cache Anda dengan menggunakan jenis simpul yang lebih besar, atau skalakan ke luar dengan menambahkan lebih banyak simpul cache.

SwapUsage

Ini adalah metrik tingkat host yang dilaporkan dalam byte. Untuk informasi selengkapnya, lihat [Metrik Tingkat Host](#).

Metrik `FreeableMemory` CloudWatch yang mendekati 0 (yaitu, di bawah 100MB) atau metrik `SwapUsage` yang lebih besar dari metrik `FreeableMemory` menunjukkan bahwa simpul berada dalam tekanan memori. Jika ini terjadi, sebaiknya Anda meningkatkan nilai parameter `ConnectionOverhead`.

Evictions

Ini adalah metrik mesin cache. Sebaiknya tentukan ambang batas alarm Anda sendiri untuk metrik ini berdasarkan kebutuhan aplikasi Anda.

Jika Anda melebihi ambang batas yang dipilih, naikkan skala kluster cache Anda dengan menggunakan jenis simpul yang lebih besar, atau skalakan ke luar dengan menambahkan lebih banyak simpul.

CurrConnections

Ini adalah metrik mesin cache. Sebaiknya tentukan ambang batas alarm Anda sendiri untuk metrik ini berdasarkan kebutuhan aplikasi Anda.

Peningkatan jumlah CurrConnections mungkin menunjukkan masalah di aplikasi Anda; Anda perlu menyelidiki perilaku aplikasi untuk mengatasi masalah ini.

Memantau Metrik Klaster dan Simpul CloudWatch

ElastiCache dan CloudWatch terintegrasi agar Anda dapat mengumpulkan berbagai metrik. Anda dapat memantau metrik ini menggunakan CloudWatch.

Note

Contoh berikut memerlukan alat baris perintah CloudWatch. Untuk informasi selengkapnya tentang CloudWatch dan untuk mengunduh alat developer, lihat [halaman produk CloudWatch](#).

Prosedur berikut menunjukkan cara menggunakan CloudWatch guna mengumpulkan statistik ruang penyimpanan untuk klaster 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 batas ElastiCache, lihat [Kuota Layanan AWS](#) untuk ElastiCache.

Memantau Metrik Klaster dan Simpul CloudWatch (Konsol)

Untuk mengumpulkan statistik pemanfaatan CPU untuk klaster cache

1. Masuk ke AWS Management Console dan buka konsol ElastiCache 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. Di halaman Klaster Cache pada Konsol Manajemen AWS, klik nama satu atau beberapa klaster cache.

Halaman detail untuk kluster 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 Metrik CloudWatch yang tersedia muncul di bagian bawah jendela konsol.

- d. Klik metrik Pemanfaatan CPU.

Konsol CloudWatch akan terbuka, yang menampilkan metrik pilihan Anda. Anda dapat menggunakan kotak daftar drop-down Statistik dan Periode, serta tab Rentang Waktu untuk mengubah metrik yang ditampilkan.

Memantau Metrik Kluster dan Simpul CloudWatch menggunakan CLI CloudWatch

Untuk mengumpulkan statistik pemanfaatan CPU untuk kluster 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 Klaster dan Simpul CloudWatch menggunakan API CloudWatch

Untuk mengumpulkan statistik pemanfaatan CPU untuk klaster cache

- Panggil API CloudWatch `GetMetricStatistics` dengan parameter berikut (perhatikan bahwa waktu mulai dan akhir ditampilkan sebagai contoh saja; Anda perlu mengganti waktu awal dan akhir yang sesuai dengan waktu Anda sendiri):
 - `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>
```


Pemantauan Amazon SNS dari peristiwa ElastiCache

Saat peristiwa penting terjadi untuk klaster, ElastiCache mengirimkan notifikasi 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 terbaru Anda dan dapat mengambil tindakan korektif sesuai peristiwa tersebut.

Topik

- [Mengelola notifikasi ElastiCache Amazon SNS](#)
- [Melihat peristiwa ElastiCache](#)
- [Notifikasi Peristiwa dan Amazon SNS](#)

Mengelola notifikasi ElastiCache Amazon SNS

Anda dapat mengkonfigurasi ElastiCache untuk mengirim notifikasi untuk peristiwa klaster 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](#).

Menambahkan topik Amazon SNS

Bagian berikut menunjukkan cara menambahkan topik Amazon SNS menggunakan konsol AWS, AWS CLI, atau API ElastiCache.

Menambahkan topik Amazon SNS (Konsol)

Prosedur berikut menunjukkan cara menambahkan topik Amazon SNS untuk klaster.

 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 konsol ElastiCache 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 Masukkan ARN manual dan ketik ARN topik Amazon SNS.
5. Pilih Ubah.

Menambahkan topik Amazon SNS (AWS CLI)

Untuk menambahkan atau mengubah 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 menambahkan 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.amazonaws.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 konsol ElastiCache di <https://console.aws.amazon.com/elasticache/>.

2. Untuk melihat daftar klaster Anda yang menjalankan Memcached, pada panel navigasi, pilih Memcached.
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
```

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 peristiwa ElastiCache

ElastiCache membuat log peristiwa yang berhubungan dengan instans 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 konsol ElastiCache, perintah AWS CLI `describe-events`, atau tindakan `DescribeEvents` API ElastiCache.

Prosedur berikut menunjukkan kepada Anda cara untuk melihat semua peristiwa ElastiCache dalam 24 jam terakhir (1440 menit).

Melihat peristiwa ElastiCache (Konsol)

Prosedur berikut menampilkan peristiwa menggunakan konsol ElastiCache.

Untuk melihat peristiwa menggunakan konsol ElastiCache:

1. Masuk ke AWS Management Console dan buka konsol ElastiCache di <https://console.aws.amazon.com/elasticache/>.
2. Untuk melihat daftar semua peristiwa yang tersedia, pada panel navigasi, pilih Peristiwa.

Di layar Peristiwa, setiap baris daftar merepresentasikan satu peristiwa dan menampilkan sumber peristiwa, jenis peristiwa (`cache-cluster`, `cache-parameter-group`, `cache-security-group`, atau `cache-subnet-group`), waktu GMT peristiwa, serta deskripsi dari peristiwa.

Dengan menggunakan Filter, Anda dapat menentukan apakah Anda ingin melihat semua peristiwa, atau hanya peristiwa dari jenis tertentu dalam daftar peristiwa.

Melihat peristiwa ElastiCache (AWS CLI)

Untuk menghasilkan daftar peristiwa ElastiCache menggunakan AWS CLI, gunakan perintah `describe-events`. 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 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 peristiwa ElastiCache (API ElastiCache)

Untuk menghasilkan daftar peristiwa ElastiCache menggunakan API ElastiCache, gunakan tindakan `DescribeEvents`. 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 seperti yang 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 olahpesan menggunakan Amazon Simple Notification Service (SNS) ketika peristiwa penting terjadi pada klaster 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 Amazon Simple Notification Service (SNS), termasuk informasi mengenai 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 notifikasi ElastiCache.
- Akun AWS yang memiliki topik Amazon SNS harus merupakan akun yang sama yang memiliki klaster cache yang notifikasinya diaktifkan.

- Topik Amazon SNS yang Anda publikasikan tidak dapat dienkripsi.

Note

Dimungkinkan untuk melampirkan topik Amazon SNS yang terenkripsi (diam) ke kluster. Namun, status topik dari konsol ElastiCache akan ditampilkan tidak aktif, yang secara efektif memisahkan topik itu dari kluster ketika ElastiCache mendorong pesan ke topik tersebut.


- Topik Amazon SNS harus berada di Wilayah yang sama dengan kluster ElastiCache.


Peristiwa ElastiCache

Peristiwa ElastiCache berikut memicu notifikasi Amazon SNS. Untuk informasi tentang detail peristiwa, lihat [Melihat peristiwa ElastiCache](#).

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:AddCacheNodeComplete	ElastiCache:AddCacheNodeComplete : <i>cache-cluster</i>	Simpul cache telah ditambahkan ke kluster cache dan siap untuk digunakan.
ElastiCache:AddCacheNodeFailed karena alamat IP yang bebas tidak mencukupi	ElastiCache:AddCacheNodeFailed : <i>cluster-name</i>	Simpul cache tidak dapat ditambahkan karena alamat IP yang tersedia tidak cukup.
ElastiCache:CacheClusterParametersChanged	ElastiCache:CacheClusterParametersChanged : <i>cluster-name</i>	Satu atau beberapa parameter kluster cache telah berubah.
ElastiCache:CacheClusterProvisioningComplete	ElastiCache:CacheClusterProvisioningComplete <i>cluster-name-0001-005</i>	Penyediaan kluster cache selesai, dan simpul cache dalam kluster cache siap untuk digunakan.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CacheClusterProvisioningFailed karena status jaringan yang tidak kompatibel	ElastiCache:CacheClusterProvisioningFailed : <i>cluster-name</i>	Percobaan dilakukan untuk meluncurkan kluster cache baru ke cloud privat virtual (VPC) yang tidak ada.
ElastiCache:CacheClusterScalingComplete	CacheClusterScalingComplete : <i>cluster-name</i>	Penskalaan untuk kluster cache berhasil diselesaikan.
ElastiCache:CacheClusterScalingFailed	ElastiCache:CacheClusterScalingFailed : <i>nama kluster</i>	Operasi peningkatan skala pada kluster cache gagal.
ElastiCache:CacheClusterSecurityGroupModified	ElastiCache:CacheClusterSecurityGroupModified : <i>cluster-name</i>	<p>Salah satu peristiwa berikut telah terjadi:</p> <ul style="list-style-type: none"> • Daftar grup keamanan cache yang diotorisasi untuk kluster cache telah berubah. • Satu atau beberapa grup keamanan EC2 telah diotorisasi pada grup keamanan cache yang terkait dengan kluster cache. • Satu atau beberapa grup keamanan EC2 baru telah dicabut otorisasinya dari salah satu grup keamanan cache yang terkait dengan kluster cache.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CacheNodeReplaceStarted	ElastiCache:CacheNodeReplaceStarted : <i>cluster-name</i>	<p>ElastiCache telah mendeteksi bahwa host yang menjalankan simpul cache mengalami penurunan performa atau tidak dapat dijangkau dan telah mulai mengganti simpul cache.</p> <div data-bbox="1068 590 1507 856"><p> Note Entri DNS untuk simpul cache yang diganti tidak berubah.</p></div> <p>Pada kebanyakan kasus, Anda tidak perlu menyegarkan daftar server untuk klien Anda ketika peristiwa ini terjadi. Namun, beberapa pustaka klien cache mungkin berhenti menggunakan simpul cache bahkan setelah ElastiCache mengganti simpul cache; dalam hal ini, aplikasi harus menyegarkan daftar server ketika peristiwa ini terjadi.</p>

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CacheNodeReplaceComplete	ElastiCache:CacheNodeReplaceComplete : <i>cluster-name</i>	<p>ElastiCache mendeteksi bahwa host yang menjalankan simpul cache terdegradasi atau tidak terjangkau dan telah selesai mengganti simpul cache.</p> <div data-bbox="1068 541 1507 808" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; margin: 10px 0;"> <p> Note</p> <p>Entri DNS untuk simpul cache yang diganti tidak berubah.</p> </div> <p>Pada kebanyakan kasus, Anda tidak perlu menyegarkan daftar server untuk klien Anda ketika peristiwa ini terjadi. Namun, beberapa pustaka klien cache mungkin berhenti menggunakan simpul cache bahkan setelah ElastiCache mengganti simpul cache; dalam hal ini, aplikasi harus menyegarkan daftar server ketika peristiwa ini terjadi.</p>
ElastiCache:CacheNodesRebooted	ElastiCache:CacheNodesRebooted : <i>cluster-name</i>	<p>Satu atau beberapa simpul cache telah di-boot ulang.</p> <p>Pesan (Memcached): "Cache node %s shutdown" Kemudian pesan kedua: "Cache node %s restarted"</p>

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:CertificateRenewalComplete (hanya Redis)	ElastiCache:CertificateRenewalComplete	Sertifikat Amazon CA berhasil diperbarui.
ElastiCache:CreateReplicationGroupComplete	ElastiCache:CreateReplicationGroupComplete : <i>cluster-name</i>	Grup replikasi berhasil dibuat.
ElastiCache>DeleteCacheClusterComplete	ElastiCache>DeleteCacheClusterComplete : <i>cluster-name</i>	Penghapusan klaster cache dan semua simpul cache terkait telah selesai.
ElastiCache:FailoverComplete (hanya Redis)	ElastiCache:FailoverComplete : <i>mycluster</i>	Failover ke simpul replika telah berhasil.
ElastiCache:ReplicationGroupIncreaseReplicaCountFinished	ElastiCache:ReplicationGroupIncreaseReplicaCountFinished : <i>cluster-name-0001-005</i>	Jumlah replika dalam klaster telah meningkat.
ElastiCache:ReplicationGroupIncreaseReplicaCountStarted	ElastiCache:ReplicationGroupIncreaseReplicaCountStarted : <i>cluster-name-0003-004</i>	Proses penambahan replika ke klaster Anda telah dimulai.
ElastiCache:NodeReplacementCanceled	ElastiCache:NodeReplacementCanceled : <i>cluster-name</i>	Sebuah simpul di klaster Anda yang dijadwalkan akan diganti tidak lagi dijadwalkan akan diganti.

Nama Peristiwa	Pesan	Deskripsi
ElastiCache:NodeReplacementRescheduled	ElastiCache:NodeReplacementRescheduled : <i>cluster-name</i>	<p>Sebuah simpul di kluster Anda yang sebelumnya dijadwalkan akan diganti telah dijadwalkan ulang untuk diganti selama periode baru yang dijelaskan dalam notifikasi.</p> <p>Untuk informasi tentang tindakan yang dapat Anda lakukan, lihat Mengganti simpul cache untuk Memcached.</p>
ElastiCache:NodeReplacementScheduled	ElastiCache:NodeReplacementScheduled : <i>cluster-name</i>	<p>Sebuah simpul di kluster Anda dijadwalkan akan diganti selama periode yang dijelaskan dalam notifikasi.</p> <p>Untuk informasi tentang tindakan yang dapat Anda lakukan, lihat Mengganti simpul cache untuk Memcached.</p>
ElastiCache:RemoveCacheNodeComplete	ElastiCache:RemoveCacheNodeComplete : <i>cluster-name</i>	Sebuah simpul cache telah dihapus dari kluster cache.
ElastiCache:ReplicationGroupScalingComplete	ElastiCache:ReplicationGroupScalingComplete : <i>cluster-name</i>	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 (hanya Redis)	ElastiCache:SnapshotComplete : <i>cluster-name</i>	Sebuah snapshot cache telah berhasil diselesaikan.
ElastiCache:SnapshotFailed (hanya Redis)	SnapshotFailed : <i>cluster-name</i>	Sebuah snapshot cache telah gagal. Lihat peristiwa kluster cache untuk penyebab yang lebih terperinci. Jika Anda menjelaskan snapshot, lihat DescribeSnapshots , statusnya adalah failed.

Topik terkait

- [Melihat peristiwa ElastiCache](#)

Logging panggilan API Amazon ElastiCache dengan AWS CloudTrail

Amazon ElastiCache terintegrasi dengan AWS CloudTrail, sebuah layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, atau layanan AWS di Amazon ElastiCache. CloudTrail mencatat semua panggilan API untuk Amazon ElastiCache sebagai peristiwa, termasuk panggilan dari konsol ElastiCache dan dari panggilan kode ke operasi API Amazon ElastiCache. Jika membuat jejak, Anda dapat memungkinkan pengiriman peristiwa CloudTrail berkelanjutan ke bucket Amazon S3, termasuk peristiwa untuk Amazon ElastiCache. Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa terbaru dalam konsol CloudTrail di Riwayat peristiwa. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat

ke Amazon ElastiCache, alamat IP asal permintaan, entitas yang membuat permintaan, waktu pembuatan permintaan, dan detail tambahan.

Untuk mempelajari selengkapnya tentang CloudTrail, lihat [Panduan Pengguna AWS CloudTrail](#).

Informasi Amazon ElastiCache di CloudTrail

CloudTrail diaktifkan pada akun AWS saat Anda membuat akun tersebut. Saat terjadi aktivitas di Amazon ElastiCache, aktivitas tersebut akan dicatat dalam peristiwa CloudTrail bersama peristiwa layanan AWS lainnya di Riwayat peristiwa. Anda dapat melihat, mencari, dan mengunduh peristiwa terbaru di akun AWS. Untuk informasi selengkapnya, lihat [Melihat Peristiwa dengan Riwayat Peristiwa CloudTrail](#).

Untuk catatan berkelanjutan terkait suatu peristiwa di akun AWS Anda, termasuk peristiwa untuk Amazon ElastiCache, buatlah jejak. Jejak memungkinkan CloudTrail mengirimkan file log ke bucket Amazon S3. Secara default, ketika Anda membuat jejak di konsol, jejak ini diterapkan ke semua Wilayah. Jejak mencatat log peristiwa dari semua wilayah di partisi AWS dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi layanan AWS lainnya untuk menganalisis lebih lanjut dan bertindak berdasarkan data peristiwa yang dikumpulkan di log CloudTrail. Untuk informasi selengkapnya, lihat berikut ini:

- [Gambaran umum untuk Membuat Jejak](#)
- [Layanan yang Didukung dan Integrasi CloudTrail](#)
- [Mengonfigurasi Notifikasi Amazon SNS untuk CloudTrail](#)
- [Menerima File Log CloudTrail dari Banyak Wilayah](#) dan [Menerima File Log CloudTrail dari Banyak Akun](#)

Semua tindakan Amazon ElastiCache dicatat oleh CloudTrail dan didokumentasikan dalam [Referensi API ElastiCache](#). Misalnya, panggilan ke tindakan `CreateCacheCluster`, `DescribeCacheCluster`, dan `ModifyCacheCluster` menghasilkan entri di file log CloudTrail.

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 peran atau pengguna terfederasi.
- Apakah permintaan tersebut dibuat oleh layanan AWS lainnya.

Untuk informasi selengkapnya, lihat [Elemen userIdentity CloudTrail](#).

Memahami entri file log Amazon ElastiCache

Jejak adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai file log ke bucket Amazon S3 yang telah Anda tentukan. File log CloudTrail berisi satu atau beberapa entri log. Peristiwa merepresentasikan satu permintaan dari sumber apa pun dan menyertakan informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. File log CloudTrail bukan merupakan jejak tumpukan panggilan API publik yang berurutan, sehingga file tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri log CloudTrail yang menunjukkan tindakan `CreateCacheCluster`.

```
{
  "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 log CloudTrail yang menunjukkan tindakan DescribeCacheCluster. Perhatikan bahwa untuk semua panggilan Describe Amazon ElastiCache (Describe*), bagian ResponseElements 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 log CloudTrail yang mencatat tindakan ModifyCacheCluster.

```

{
  "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"
}
```

Kuota untuk ElastiCache

Akun AWS Anda memiliki kuota default, yang sebelumnya disebut sebagai batas, untuk setiap layanan AWS. Kecuali dinyatakan sebaliknya, setiap kuota unik untuk suatu Wilayah. Anda dapat meminta penambahan untuk beberapa kuota, sementara kuota lainnya tidak dapat ditambah.

Untuk melihat kuota ElastiCache, buka [Konsol Kuota Layanan](#). Di panel navigasi, pilih Layanan AWS dan pilih ElastiCache.

Untuk meminta penambahan kuota, lihat [Meminta Penambahan Kuota](#) dalam Panduan Pengguna Kuota Layanan. Jika kuota belum tersedia dalam Kuota Layanan, gunakan [formulir penambahan batas](#).

Akun AWS Anda memiliki kuota berikut yang terkait dengan ElastiCache.

Sumber daya	Default
Cache nirserver per wilayah	40
Simpul per Wilayah	300*
Simpul per klaster	40
Grup parameter per Wilayah	300*
Grup keamanan per Wilayah	50
Grup subnet per Wilayah	300*
Subnet per grup subnet	20*

Referensi

Topik pada bagian ini membahas cara menggunakan API Amazon ElastiCache dan bagian ElastiCache untuk AWS CLI. Bagian ini juga menyertakan pesan kesalahan dan notifikasi layanan yang umum.

- [Menggunakan API ElastiCache](#)
- [Referensi API ElastiCache](#)
- [Bagian ElastiCache dalam Referensi AWS CLI](#)
- [Pesan kesalahan Amazon ElastiCache](#)
- [Notifikasi](#)

Menggunakan API ElastiCache

Bagian ini menyediakan deskripsi berorientasi tugas tentang cara menggunakan dan menerapkan operasi ElastiCache. Untuk mempelajari deskripsi lengkap dari operasi ini, lihat [Referensi API Amazon ElastiCache](#)

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. Tindakan ini mengasumsikan Anda terdaftar di 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 (pengidentifikasi kunci dari Kunci Akses Rahasia yang digunakan) untuk 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 dengan yang digunakan untuk menghitung tanda tangan pada permintaan.
6. Jika tanda tangan cocok, maka permintaan tersebut dianggap otentik. Jika perbandingan gagal, maka permintaan ditolak dan AWS menampilkan 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 Anda butuhkan nanti dalam prosedur ini:

- a. Urutkan komponen string kueri UTF-8 berdasarkan nama parameter dengan pengurutan byte alami. Parameter dapat berasal dari GET URI atau dari konten POST (jika Content-Type adalah `application/x-www-form-urlencoded`).
 - b. URL mengkode nama parameter dan nilainya sesuai dengan aturan berikut:
 - i. Jangan melakukan encode 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 encode persen pada semua karakter lain dengan %XY, di mana X dan Y adalah karakter hex 0-9 dan huruf besar A-F.
 - iii. Gunakan encode persen pada karakter UTF-8 yang diperluas dalam bentuk %XY%ZA...
 - iv. Gunakan encode persen pada karakter spasi sebagai %20 (dan bukan +, seperti skema pengkodean yang umum dilakukan).
 - c. Pisahkan nama parameter yang diencode dari nilai yang diencode dengan tanda sama dengan (=) (karakter ASCII 61), meskipun jika nilai parameter tersebut 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 HTTPRequestURI adalah komponen jalur absolut HTTP dari URI hingga, tapi tidak termasuk, string kueri. Jika HTTPRequestURI kosong, gunakan garis miring ke depan (/).

3. Hitung HMAC sesuai RFC 2104 dengan string yang baru saja Anda buat, Kunci Akses Rahasia sebagai kunci, dan SHA256 atau SHA1 sebagai algoritma hash.

Untuk informasi selengkapnya, lihat <https://www.ietf.org/rfc/rfc2104.txt>.

4. Konversi 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 cara menghitung tanda tangan permintaan, lihat topik [Proses Penandatanganan Tanda Tangan Versi 4](#) dan subtopiknya.

Pustaka yang tersedia

AWS menyediakan kit pengembangan perangkat lunak (SDK) untuk developer perangkat lunak yang lebih memilih membangun aplikasi menggunakan API bahasa tertentu, bukan API Kueri. SDK ini menyediakan fungsi dasar (tidak disertakan dalam API), seperti autentikasi permintaan, percobaan ulang permintaan, dan penanganan kesalahan sehingga lebih mudah untuk memulai. SDK dan sumber daya tambahan tersedia dalam bahasa pemrograman berikut:

- [Java](#)
- [Windows dan .NET](#)
- [PHP](#)
- [Python](#)
- [Ruby](#)

Untuk informasi tentang bahasa lain, lihat [Contoh Kode & Pustaka](#).

Memecahkan masalah aplikasi

ElastiCache memberikan penjelasan tentang kesalahan spesifik dan deskriptif untuk membantu Anda memecahkan masalah saat menangani 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 jika terjadi kesalahan adalah dengan mencari simpul `ERROR` di dalam respons dari API ElastiCache.

Sintaks XPath menyediakan cara sederhana untuk mencari keberadaan simpul `ERROR`, serta cara mudah untuk mengambil kode dan pesan kesalahan. Cuplikan kode berikut menggunakan modul Perl dan `XML::XPath` untuk menentukan jika kesalahan terjadi 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: ",
$xml->findvalue("//Error[1]/Code"), "\n", " ",
$xml->findvalue("//Error[1]/Message"), "\n\n"; }
```

Tips penyelesaian masalah

Dianjurkan lakukan proses berikut untuk mendiagnosis dan menyelesaikan masalah dengan API ElastiCache.

- Pastikan bahwa ElastiCache berjalan dengan benar.

Untuk melakukannya, cukup buka jendela browser dan kirimkan permintaan kueri ke layanan ElastiCache (seperti <https://ElastiCache.amazonaws.com>). `MissingAuthenticationTokenException` atau Kesalahan Server Internal 500 mengonfirmasi bahwa layanan tersedia dan menanggapi permintaan.

- Periksa struktur permintaan Anda.

Setiap operasi ElastiCache memiliki halaman referensi di Referensi API ElastiCache. 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 tempat Anda dapat mencari solusi untuk masalah yang dialami orang lain selama ini. Untuk melihat forum, kunjungi

<https://forums.aws.amazon.com/>.

Menyiapkan antarmuka baris perintah ElastiCache

Bagian ini menjelaskan prasyarat untuk menjalankan alat baris perintah, tempat mendapatkannya, cara mengatur lingkungannya, dan contoh umum penggunaan alat ini.

Ikuti petunjuk dalam topik ini hanya jika Anda akan menggunakan AWS CLI untuk ElastiCache.

Important

Antarmuka Baris Perintah (CLI) Amazon ElastiCache tidak mendukung peningkatan ElastiCache setelah API versi 2014-09-30. Untuk menggunakan fungsionalitas ElastiCache yang lebih baru dari baris perintah, gunakan [AWS Command Line Interface](#).

Topik

- [Prasyarat](#)
- [Mendapatkan alat baris perintah](#)
- [Menyiapkan alat](#)
- [Memberikan kredensial untuk alat](#)
- [Variabel lingkungan](#)

Prasyarat

Dokumen ini mengasumsikan bahwa Anda dapat menggunakan lingkungan Linux/UNIX atau Windows. Alat baris perintah Amazon ElastiCache juga berfungsi pada Mac OS X, yang merupakan lingkungan berbasis UNIX; namun, tidak ada petunjuk Mac OS X spesifik 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 khusus Linux/UNIX 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 JRE untuk berbagai platform, termasuk Linux/UNIX dan Windows, lihat [Java SE Downloads](#).

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 di [situs web Alat Developer ElastiCache](#). Alat ini ditulis dalam Java, dan menyertakan 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.nnn` (A, B dan n adalah nomor versi/rilis), serta berisi subdirektori bernama `bin` dan `lib`.

Untuk menetapkan variabel lingkungan `AWS_ELASTICACHE_HOME`

- Buka jendela baris perintah dan masukkan salah satu perintah berikut untuk menetapkan 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 mungkin ingin 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 membutuhkan Kunci Akses dan Kunci Akses Rahasia AWS yang disediakan dengan akun AWS Anda. Anda bisa mendapatkannya menggunakan baris perintah atau dari file kredensial yang terletak di sistem lokal Anda.

Deployment ini menyertakan file templat `${AWS_ELASTICACHE_HOME}/credential-file-path.template` perlu Anda edit dengan informasi Anda. Berikut adalah konten dari file templat:

```
AWSAccessKeyId=<Write your AWS access ID>  
AWSSecretKey=<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 menetapkan variabel lingkungan `AWS_CREDENTIAL_FILE` agar alat ElastiCache dapat menemukan informasi Anda.

Untuk menetapkan 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. Untuk memeriksa bahwa pengaturan Anda berfungsi 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 Perintah ElastiCache mendukung variabel berikut:

- `EC2_REGION` – Wilayah AWS yang digunakan.
- `AWS_ELASTICACHE_URL` – URL yang digunakan untuk panggilan layanan. Tidak perlu menentukan titik akhir wilayah yang berbeda jika `EC2_REGION` ditentukan atau parameter `--region` diteruskan.

Contoh berikut menunjukkan cara menetapkan 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 kesalahan Amazon ElastiCache

Pesan kesalahan berikut ditampilkan oleh Amazon ElastiCache. Anda mungkin menerima pesan kesalahan lain yang ditampilkan oleh ElastiCache, layanan AWS lain, atau oleh Memcached. Untuk deskripsi pesan kesalahan dari sumber selain ElastiCache, lihat dokumentasi dari sumber yang menghasilkan pesan kesalahan tersebut.

- [Cluster node quota exceeded](#)
- [Customer's node quota exceeded](#)
- [Insufficient cache cluster capacity](#)

Pesan Kesalahan: Kuota simpul klaster terlampaui. Setiap klaster dapat memiliki maksimal %n simpul di wilayah ini.

Penyebab: Anda mencoba membuat atau memodifikasi 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 simpul, buat permintaan Anda menggunakan [Formulir permintaan simpul Amazon ElastiCache](#).

Untuk informasi selengkapnya, lihat [Batas Amazon ElastiCache](#) dalam 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 memodifikasi 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 simpul, buat permintaan Anda menggunakan [Formulir permintaan simpul Amazon ElastiCache](#).

Untuk informasi selengkapnya, lihat [Batas Amazon ElastiCache](#) dalam Referensi Umum Amazon Web Services.

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 nanti). Untuk informasi selengkapnya, lihat [Penskalaan ElastiCache untuk Memcached](#).

Notifikasi

Topik berikut membahas notifikasi ElastiCache yang mungkin Anda minati. 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.

Notifikasi tidak mengumumkan fitur atau fungsionalitas ElastiCache yang baru atau ditingkatkan.

Notifikasi ElastiCache Umum

Saat ini tidak ada notifikasi ElastiCache yang belum ditangani yang bukan khusus untuk mesin tertentu.

Notifikasi ElastiCache for Memcached

Notifikasi ElastiCache berikut dikhususkan untuk mesin Memcached.

Notifikasi khusus ElastiCache for 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 di 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:

- Versi API: 2015-02-02
- Pembaruan dokumentasi terbaru: 27 November 2023

Tabel berikut menjelaskan perubahan penting dalam setiap rilis Panduan Pengguna ElastiCache untuk Memcached ElastiCache untuk Panduan Pengguna . Untuk notifikasi tentang pembaruan-pembaruan dokumentasi ini, Anda dapat berlangganan ke sebuah umpan RSS.

Terbaru ElastiCache untuk Memcached Update

Perubahan	Deskripsi	Tanggal
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 Memcached sekarang mendukung pembuatan cache tanpa server	Anda sekarang dapat membuat cache nirserver , yang menyederhanakan manajemen cache dan langsung melakukan penskalaan untuk mendukung aplikasi dengan tuntutan tinggi. Untuk informasi selengkapnya, lihat Memilih di antara opsi deployment . Sebagai bagian dari fitur ini, izin baru ditambahkan ke ElastiCacheService RolePolicy dan AmazonElastiCacheFullAccess untuk	27 November 2023

memungkinkan asosiasi cache nirsumber dengan titik akhir VPC terkelola. Selain itu, izin ditambahkan untuk mendukung pengalaman konsol yang direvisi menggunakan kebijakan AmazonElastiCacheFullAccess .

[Support ElastiCache untuk Memcached 1.6.17](#)

ElastiCache untuk Memcached sekarang mendukung Memcached 1.6.17. Ini mencakup perbaikan bug kumulatif dari versi [Memcached 1.6.12](#). Untuk informasi lebih lanjut, lihat [Memcached Versi 1.6.17](#).

18 Januari 2023

[ElastiCache untuk Memcached sekarang mendukung IPv6](#)

ElastiCache mendukung Internet Protocol versi 4 dan 6 (IPv4 dan IPv6), memungkinkan Anda untuk mengkonfigurasi cluster Anda untuk hanya menerima koneksi IPv4, hanya koneksi IPv6 atau koneksi IPv4 dan IPv6 (dual-stack). IPv6 didukung untuk beban kerja yang menggunakan mesin Memcached versi 1.6.6 dan seterusnya di semua instans yang dibangun di [sistem Nitro](#). Tidak ada biaya tambahan untuk mengakses ElastiCache lebih dari 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 memungkinkan Anda untuk 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 sekarang mendukung PrivateLink](#)

AWS PrivateLink memungkinkan 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 \(AWS PrivateLink\)](#) untuk Redis atau [ElastiCache Amazon API dan antarmuka VPC endpoint \(\)](#) untuk Memcached. AWS PrivateLink

24 Januari 2022

[Dukungan untuk penandaan 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 Memcached 1.6.6.](#)

ElastiCache untuk Memcached sekarang mendukung Memcached 1.6.6. Ini mencakup perbaikan bug kumulatif dari versi Memcached 1.5.16. Untuk informasi selengkapnya, lihat [Memcached Versi 1.6.6](#).

12 November 2020

[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](#) untuk Redis atau [Menggunakan Outposts](#) untuk Memcached.

8 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 menentukannya ke Local Zone. Untuk informasi selengkapnya, lihat [Menggunakan Zona Lokal](#).

25 September 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 level sumber daya](#).

12 Agustus 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 menggunakan jendela pemeliharaan Anda untuk menjadwalkan pembaruan otomatis cluster yang berlaku. Untuk informasi selengkapnya, lihat [Pembaruan secara mandiri](#).

13 Mei, 2020

[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 Standar T3 dari Amazon EC2 memberikan kinerja CPU di tingkat garis dasar dengan kemampuan untuk menangani lonjakan penggunaan CPU kapan pun hingga kredit terakumulasi habis. Untuk informasi selengkapnya, lihat [Jenis Simpul yang Didukung](#).

12 November 2019

[ElastiCache untuk Memcached 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 jendela pemeliharaan. Ini akan meminimalkan gangguan layanan, terutama selama arus bisnis puncak, dan membantu memastikan Anda tetap mematuhi standar pembaruan keamanan. Untuk informasi selengkapnya, lihat [Pembaruan Layanan Mandiri di Amazon ElastiCache](#).

9 Oktober 2019

[Support ElastiCache untuk Memcached 1.5.16](#)

ElastiCache untuk Memcached sekarang mendukung Memcached 1.5.16. Ini mencakup perbaikan bug kumulatif dari versi [Memcached 1.5.14](#) dan [Memcached 1.5.15](#). Untuk informasi selengkapnya, lihat [Memcached Versi 1.5.16](#).

6 September 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

[Support ElastiCache untuk Memcached 1.5.10](#)

ElastiCache untuk Memcached sekarang mendukung Memcached 1.5.10. Ini mencakup dukungan untuk parameter `no_modern` dan `inline_as_cii_resp`. Untuk informasi selengkapnya, lihat [Memcached Versi 1.5.10](#).

14 November 2018

[Perubahan struktur Panduan Pengguna](#)

Panduan ElastiCache Pengguna tunggal sekarang direstrukturisasi sehingga ada panduan pengguna terpisah untuk Redis (untuk Panduan Pengguna Redis) dan [ElastiCache untuk Memcached \(ElastiCache untuk Panduan Pengguna Memcached\)](#). Struktur dokumentasi di bagian [AWS CLI Command Reference : elasticache](#) dan [Referensi Amazon ElastiCache API](#) tetap tidak berubah.

20 April 2018

Tabel berikut menjelaskan perubahan penting pada Panduan Pengguna ElastiCache untuk Memcached ElastiCache untuk Panduan Pengguna .

Perubahan	Deskripsi	Tanggal Diubah
Dukungan untuk Wilayah Asia Pasifik (Osaka-lokal).	<p>ElastiCache Menambahkan dukungan untuk Wilayah Asia Pasifik (Osaka-lokal). Wilayah Asia Pasifik (Osaka) saat ini mendukung Zona Ketersediaan tunggal dan hanya berdasarkan undangan saja. Untuk informasi selengkapnya, lihat hal berikut:</p> <ul style="list-style-type: none"> • Wilayah yang didukung • Jenis simpul cache yang didukung 	12 Februari 2018
Dukungan untuk EU (Paris).	<p>ElastiCache menambahkan dukungan untuk Wilayah UE (Paris). Untuk informasi selengkapnya, lihat hal berikut:</p>	18 Desember 2017

Perubahan	Deskripsi	Tanggal Diubah
	<ul style="list-style-type: none">• Wilayah yang didukung• Jenis simpul cache yang didukung	
Dukungan untuk Wilayah Tiongkok (Ningxia)	<p>Amazon ElastiCache menambahkan dukungan untuk Wilayah China (Ningxia). Untuk informasi selengkapnya, lihat hal berikut:</p> <ul style="list-style-type: none">• Wilayah yang didukung• Jenis simpul cache yang didukung	11 Desember 2017
Dukungan untuk Peran Tertaut Layanan	<p>Rilis dukungan ElastiCache tambahan untuk Service Linked Roles (SLR) ini. Untuk informasi selengkapnya, lihat hal berikut:</p> <ul style="list-style-type: none">• Menggunakan Peran Tertaut Layanan untuk Amazon ElastiCache• Siapkan izin Anda (hanya ElastiCache pengguna baru)	7 Desember 2017
Dukungan untuk jenis simpul R4	<p>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 hal berikut:</p> <ul style="list-style-type: none">• Jenis simpul cache yang didukung• Parameter khusus jenis simpul Memcached	20 November 2017

Perubahan	Deskripsi	Tanggal Diubah
Topik pola koneksi	<p>ElastiCache dokumentasi menambahkan topik yang mencakup berbagai pola untuk mengakses ElastiCache cluster di VPC Amazon.</p> <p>Untuk informasi lain, lihat Pola Akses untuk Mengakses Cache ElastiCache di Amazon VPC dalam Panduan Pengguna ElastiCache .</p>	24 April 2017
Dukungan untuk Memcached 1.4.34	<p>ElastiCache menambahkan dukungan Memcached versi 1.4.34, yang menggabungkan sejumlah perbaikan ke versi Memcached sebelumnya.</p> <p>Untuk informasi lebih lanjut, lihat Memcached 1.4.34 Catatan Rilis di Memcached on. GitHub</p>	10 April 2017
Dukungan untuk Memcached 1.4.33	<p>ElastiCache menambahkan dukungan untuk Memcached versi 1.4.33. Untuk informasi selengkapnya, lihat hal berikut:</p> <ul style="list-style-type: none">• Memcached versi 1.4.33• Parameter yang ditambahkan dalam Memcached 1.4.33	20 Desember 2016
Dukungan untuk Wilayah EU West (London)	<p>ElastiCache menambahkan dukungan untuk Wilayah UE (London). Hanya jenis simpul T2 dan M4 yang didukung saat ini. Untuk informasi selengkapnya, lihat hal berikut:</p> <ul style="list-style-type: none">• Wilayah yang didukung• Jenis simpul cache yang didukung	13 Desember 2016

Perubahan	Deskripsi	Tanggal Diubah
Dukungan untuk Wilayah Kanada (Montreal)	<p>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 hal berikut:</p> <ul style="list-style-type: none">• Wilayah yang didukung• Jenis simpul cache yang didukung	8 Desember 2016
Dukungan untuk jenis simpul M4 dan R3	<p>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 hal berikut:</p> <ul style="list-style-type: none">• Wilayah yang didukung• Jenis simpul cache yang didukung	1 November 2016
Dukungan Wilayah AS Timur 2 (Ohio)	<p>ElastiCache menambahkan dukungan untuk Wilayah Timur AS (Ohio) (us-timur-2) dengan tipe node M4, T2, dan R3. Untuk informasi selengkapnya, lihat hal berikut:</p> <ul style="list-style-type: none">• Wilayah yang didukung• Jenis simpul cache yang didukung	17 Oktober 2016
Dukungan jenis simpul M4	<p>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 hal berikut:</p> <ul style="list-style-type: none">• Jenis simpul cache yang didukung• Parameter khusus jenis simpul Memcached	3 Agustus 2016

Perubahan	Deskripsi	Tanggal Diubah
Dukungan Wilayah Mumbai	ElastiCache Menambahkan dukungan untuk Wilayah Asia Pasifik (Mumbai). Untuk informasi selengkapnya, lihat hal berikut: <ul style="list-style-type: none">• Jenis simpul cache yang didukung• Parameter khusus jenis simpul Memcached	27 Juni 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
Mengakses ElastiCache menggunakan fungsi Lambda	Menambahkan tutorial tentang mengonfigurasi fungsi Lambda untuk ElastiCache mengakses di VPC Amazon. Untuk informasi selengkapnya, lihat Tutorial dan video ElastiCache .	12 Februari 2016
Mendukung Wilayah Asia Pasifik (Seoul)	ElastiCache menambahkan dukungan untuk Wilayah Asia Pasifik (Seoul) (ap-timur laut-2) dengan tipe node t2, m3, dan r3.	6 Januari 2016
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 yang paling tidak digunakan baru-baru ini (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 Catatan rilis Memcached .	27 Agustus 2015

Perubahan	Deskripsi	Tanggal Diubah
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 selengkapnya, lihat Mengompilasi kode sumber ElastiCache Cluster Client untuk PHP .	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 dari luar AWS .	9 Juli 2015
Pesan penggantian simpul ditambahkan	ElastiCache menambahkan tiga pesan yang berkaitan dengan penggantian node terjadwal, ElastiCache:NodeReplacementScheduled, ElastiCache:NodeReplacementRescheduled, dan ElastiCache:NodeReplacementCanceled. Untuk informasi dan tindakan selengkapnya yang dapat Anda lakukan saat node dijadwalkan untuk diganti, ElastiCache lihat Notifikasi Peristiwa dan Amazon SNS .	11 Juni 2015
Dukungan untuk tanda alokasi biaya	ElastiCache menambahkan dukungan untuk tag alokasi biaya. Untuk informasi selengkapnya, lihat Memantau biaya dengan tanda 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
Dukungan untuk Wilayah Eropa (Frankfurt)	ElastiCache menambahkan dukungan untuk Wilayah Eropa (Frankfurt) (eu-central-1).	19 Januari 2015

Perubahan	Deskripsi	Tanggal Diubah
AWS CloudTrail pencatatan panggilan API didukung	ElastiCache menambahkan dukungan untuk menggunakan AWS CloudTrail untuk mencatat semua panggilan ElastiCache API. Untuk informasi selengkapnya, lihat Logging panggilan API Amazon ElastiCache dengan AWS CloudTrail .	15 September 2014
Ukuran instans baru didukung	ElastiCache menambahkan dukungan untuk instance Tujuan Umum (T2) tambahan. Untuk informasi selengkapnya, lihat Mengonfigurasi parameter mesin menggunakan grup 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
Ukuran instans baru didukung	ElastiCache menambahkan dukungan untuk instans Tujuan Umum (M3) tambahan dan instans Memory Optimized (R3). Untuk informasi selengkapnya, lihat Mengonfigurasi parameter mesin menggunakan grup parameter .	1 Juli 2014
Penemuan otomatis PHP	Ditambahkan dukungan untuk penemuan otomatis PHP versi 5.5.	13 Mei 2014
Dukungan Cloud Privat Virtual (VPC) Amazon default	Dalam rilis ElastiCache ini, terintegrasi penuh dengan Amazon Virtual Private Cloud (VPC). Untuk pelanggan baru, kluster cache dibuat dalam Amazon VPC secara default. Untuk informasi selengkapnya, lihat Amazon VPC dan keamanan ElastiCache .	8 Januari 2013

Perubahan	Deskripsi	Tanggal Diubah
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
Dukungan Cloud Privat Virtual (VPC) Amazon	Dalam rilis ini, ElastiCache cluster dapat diluncurkan di Amazon Virtual Private Cloud (VPC). Secara default, kluster 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 VPC dan keamanan ElastiCache .	20 Desember 2012
Penemuan otomatis simpul cache dan versi mesin cache baru	<p>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.</p> <p>Rilis ini juga menawarkan versi mesin cache baru: Memcached versi 1.4.14. Mesin cache baru ini menyediakan kemampuan penyeimbangan slab yang ditingkatkan, peningkatan kinerja dan skalabilitas yang signifikan, serta beberapa perbaikan bug. Ada beberapa parameter baru cache yang dapat dikonfigurasi. Untuk informasi selengkapnya, lihat Mengonfigurasi parameter mesin menggunakan grup parameter.</p>	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.	5 April 2012

Perubahan	Deskripsi	Tanggal Diubah
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.