

CodeArtifact Panduan Pengguna

# CodeArtifact



# CodeArtifact: CodeArtifact Panduan Pengguna

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

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di antara para pelanggan, atau dengan cara apa pun yang menghina atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan properti dari masing-masing pemilik, yang mungkin berafiliasi, terkait dengan, atau disponsori oleh Amazon, atau tidak.

---

# Table of Contents

Apa AWS CodeArtifact? .....	1
Bagaimana cara CodeArtifact kerja? .....	1
Konsep .....	2
Aset .....	2
Domain .....	2
Repositori .....	3
Package .....	3
Namespace paket .....	3
Versi paket .....	4
Revisi versi paket .....	4
Repositori hulu .....	4
Bagaimana saya memulai CodeArtifact? .....	4
Menyiapkan .....	5
Daftar untuk AWS .....	5
Menginstal atau meningkatkan dan kemudian mengonfigurasi AWS CLI .....	6
Menyediakan pengguna IAM .....	7
Menginstal manajer paket atau alat pembangun .....	8
Langkah berikutnya .....	9
Mulai .....	10
Prasyarat .....	10
Memulai menggunakan konsol .....	11
Memulai menggunakan AWS CLI .....	13
Bekerja dengan repositori .....	21
Buatlah sebuah repositori .....	21
Membuat repositori (konsol) .....	22
Membuat repositori (AWS CLI) .....	23
Membuat repositori dengan repositori hulu .....	24
Terhubung ke repositori .....	25
Menggunakan klien manajer paket .....	25
Hapus sebuah repositori .....	26
Menghapus repositori (konsol) .....	26
Menghapus repositori (AWS CLI) .....	26
Mencantumkan repositori .....	27
Mencantumkan repositori dalam akun AWS .....	27

Mencantumkan repositori di domain .....	28
Melihat atau mengubah konfigurasi repositori .....	30
Melihat atau mengubah konfigurasi repositori (konsol) .....	30
Melihat atau mengubah konfigurasi repositori (AWS CLI) .....	32
Kebijakan repositori .....	34
Membuat kebijakan sumber daya untuk memberikan akses baca .....	34
Menetapkan kebijakan .....	36
Membaca kebijakan .....	37
Menghapus kebijakan .....	37
Memberikan akses baca ke prinsipal .....	38
Memberikan akses tulis ke paket .....	38
Memberikan akses tulis ke repositori .....	40
Menandai repositori .....	40
Menandai repositori (CLI) .....	41
Menandai repositori (konsol) .....	44
Bekerja dengan repositori hulu .....	48
Apa perbedaan antara repositori hulu dan koneksi eksternal? .....	48
Menambahkan atau menghapus repositori hulu .....	49
Menambahkan atau menghapus repositori hulu (konsol) .....	49
Menambahkan atau menghapus repositori hulu (AWS CLI) .....	50
Connect CodeArtifact repositori ke repositori publik .....	52
Connect ke repositori .....	53
Connect ke repositori .....	54
Repositori koneksi eksternal yang didukung .....	55
Menghapus koneksi eksternal (CLI) .....	56
Meminta versi paket dengan repositori hulu .....	56
Retensi paket dari repositori hulu .....	57
Mengambil paket melalui hubungan hulu .....	57
Retensi paket dalam repositori menengah .....	60
Meminta paket dari koneksi eksternal .....	61
Mengambil paket dari koneksi eksternal .....	61
Latensi koneksi lebih masensi koneksi lebih serensi .....	63
CodeArtifact perilaku ketika repositori eksternal tidak tersedia .....	63
Ketersediaan versi paket baru .....	64
Mengimpor versi paket dengan lebih dari satu aset .....	64
Urutan prioritas repositori hulu .....	64

Contoh urutan prioritas sederhana .....	65
Contoh urutan prioritas kompleks .....	66
Perilaku API dengan repositori hulu .....	67
Bekerja dengan paket .....	70
Gambaran umum paket .....	70
Format paket yang didukung .....	71
Publikasi paket .....	71
Status versi paket .....	73
Nama paket, versi paket, dan normalisasi nama aset .....	75
Mencantumkan nama paket .....	75
Mencantumkan nama paket npm .....	77
Mencantumkan nama paket Maven .....	78
Mencantumkan nama paket Python .....	79
Filter berdasarkan prefiks nama paket .....	79
Kombinasi opsi penelusuran yang didukung .....	80
Output format .....	80
Default dan opsi lainnya .....	81
Mencantumkan versi paket .....	82
Daftar versi paket npm .....	84
Daftar versi paket Maven .....	84
Mengurutkan versi .....	84
Versi tampilan default .....	85
Output format .....	86
Mencantumkan aset versi paket .....	86
Daftar aset dari paket npm .....	88
Daftar aset paket Maven .....	88
Mengunduh aset versi paket .....	88
Menyalin paket antar-repositori .....	89
Izin IAM yang diperlukan untuk menyalin paket .....	89
Menyalin versi paket .....	91
Menyalin paket dari repositori hulu .....	92
Menyalin paket npm dalam lingkup .....	92
Menyalin versi paket Maven .....	92
Versi yang tidak ada dalam repositori sumber .....	93
Versi yang sudah ada di repositori tujuan .....	94
Menentukan revisi versi paket .....	95

Menyalin paket npm .....	96
Menghapus paket .....	96
Menghapus paket ()AWS CLI .....	97
Menghapus versi paket ()AWS CLI .....	98
Menghapus paket (konsol) .....	99
Menghapus versi paket (konsol) .....	99
Menghapus paket npm .....	99
Menghapus paket Maven .....	100
Melihat dan memperbarui detail versi paket dan dependensi .....	100
Melihat detail versi paket .....	100
Lihat detail versi paket npm .....	101
Lihat detail versi paket Maven .....	102
Melihat dependensi versi paket .....	103
Melihat file readme versi paket .....	104
Memperbarui status versi paket .....	105
Memperbarui status versi paket .....	105
Izin IAM yang diperlukan untuk memperbarui status versi paket .....	107
Memperbarui status untuk paket npm cakupan .....	107
Memperbarui status untuk paket Maven .....	107
Menentukan revisi versi paket .....	108
Menggunakan parameter status yang diharapkan .....	109
Kesalahan dengan versi paket individual .....	110
Membuang versi paket .....	111
Mengedit kontrol asal paket .....	113
Skenario kontrol akses paket umum .....	113
Pengaturan kontrol asal paket .....	115
Mengedit kontrol asal paket .....	116
Publikasi dan repositori hulu .....	118
Bekerja dengan domain .....	119
Gambaran umum domain .....	119
Domain lintas akun .....	120
Jenis AWS KMS kunci yang didukung di CodeArtifact .....	121
Membuat domain .....	121
Membuat domain (konsol) .....	122
Membuat domain (AWS CLI) .....	122
Menghapus domain .....	124

Pembatasan penghapusan domain .....	124
Menghapus domain (konsol) .....	125
Menghapus domain (AWS CLI) .....	125
Kebijakan domain .....	125
Mengaktifkan akses lintas akun ke domain .....	126
Contoh kebijakan domain .....	128
Contoh kebijakan domain dengan AWS Organizations .....	128
Menetapkan kebijakan domain .....	129
Membaca kebijakan domain .....	130
Menghapus kebijakan domain .....	131
Menandai domain .....	131
Menandai domain (CLI) .....	131
Menandai domain (konsol) .....	134
Menggunakan npm .....	138
Konfigurasi dan gunakan npm .....	138
Mengkonfigurasi npm dengan perintah login .....	139
Mengonfigurasi npm tanpa menggunakan perintah login .....	139
Menjalankan perintah npm .....	142
Memverifikasi otentikasi dan otorisasi npm .....	142
Mengubah kembali ke registri npm default .....	143
Memecahkan masalah pemasangan lambat dengan npm 8.x atau lebih tinggi .....	143
Konfigurasi dan gunakan Yarn .....	143
Konfigurasi Yarn 1.X dengan <code>aws codeartifact login</code> .....	144
Konfigurasi Yarn 2.X dengan <code>yarn config set</code> .....	145
dukungan perintah npm .....	147
Perintah yang didukung yang berinteraksi dengan repositori .....	148
Perintah sisi klien yang didukung .....	149
Perintah tidak didukung .....	151
penanganan tanda npm .....	153
Edit tanda dengan klien npm .....	154
Tanda npm dan API <code>CopyPackageVersions</code> .....	154
tanda npm dan repositori hulu .....	154
Dukungan untuk manajer paket yang kompatibel dengan npm .....	156
Menggunakan Python .....	158
Konfigurasi dan gunakan pip dengan <code>CodeArtifact</code> .....	158
Konfigurasi pip dengan <code>login</code> .....	158

Mengonfigurasi pip tanpa perintah login .....	159
Menjalankan pip .....	160
Konfigurasi dan gunakan benang dengan CodeArtifact .....	161
Konfigurasi benang dengan login komando .....	161
Konfigurasi benang tanpa login komando .....	161
Menjalankan twine .....	162
Normalisasi nama paket Python .....	163
Kompatibilitas Python .....	163
dukungan perintah pip .....	163
Meminta paket Python dari upstream dan koneksi eksternal .....	165
Versi paket yang ditarik .....	165
Mengapa CodeArtifact tidak mengambil metadata atau aset terbaru yang ditarik untuk versi paket? .....	166
Menggunakan Maven .....	168
Gunakan CodeArtifact dengan Gradle .....	168
Mengambil dependensi .....	169
Mengambil plugin .....	170
Memublikasikan artefak .....	171
Menjalankan build Gradle di IntelliJ IDEA .....	172
Gunakan CodeArtifact dengan mvn .....	177
Mengambil dependensi .....	169
Memublikasikan artefak .....	171
Memublikasikan artefak pihak ketiga .....	181
Batasi unduhan ketergantungan Maven ke repositori CodeArtifact .....	182
Informasi Proyek Apache Maven .....	184
Gunakan CodeArtifact dengan deps.edn .....	184
Mengambil dependensi .....	185
Memublikasikan artefak .....	186
Publikasi dengan curl .....	187
Menggunakan checksum Maven .....	189
Penyimpanan checksum .....	189
Checksum tidak cocok selama penerbitan .....	191
Memulihkan dari checksum .....	192
Menggunakan snapshot Maven .....	192
Penerbitan snapshot di CodeArtifact .....	193
Mengkonsumsi versi snapshot .....	195



Menghapus hapus versi snapshot .....	195
Publilililikasi snapshot dengan curl .....	195
Snapshot dan koneksi eksternal .....	198
Snapshot dan repositori hulu hulu .....	198
Meminta paket Maven dari upstream dan koneksi eksternal .....	200
Mengimpor nama aset standar .....	200
Mengimpor nama aset non-standar .....	200
Memeriksa asal-usul aset .....	201
Mengimpor aset baru dan status versi paket di repositori hulu .....	201
Pemecahan masalah Maven .....	202
Nonaktifkan parallel puts untuk memperbaiki kesalahan 429: Terlalu Banyak Permintaan ....	202
MenggunakanNuGet .....	204
Menggunakan CodeArtifact dengan Visual Studio .....	204
Mengonfigurasi Visual Studio dengan CodeArtifact Credential Provider .....	205
Menggunakan konsol Visual Studio Package Manager .....	206
GunakanCodeArtifact dengan nuget atau dotnet .....	207
Mengonfigurasi nuget atau dotnet CLI .....	207
GunakanNuGet paket .....	212
MemublikasikanNuGet paket .....	214
CodeArtifactNuGetReferensi Penyedia Kredensil .....	214
CodeArtifactNuGetVersi Credential Provider .....	216
NuGetnama paket, versi, dan normalisasi nama aset .....	216
Kompatibilitas NuGet .....	217
Kompatibilitas NuGet umum .....	217
Dukungan baris perintah NuGet .....	218
Menggunakan Swift .....	219
Konfigurasikan Swift dengan CodeArtifact .....	219
Konfigurasikan Swift dengan perintah login .....	219
Konfigurasikan Swift tanpa perintah login .....	221
Mengonsumsi dan menerbitkan paket Swift .....	225
Mengonsumsi paket Swift .....	225
Mengonsumsi paket Swift di Xcode .....	226
Menerbitkan paket Swift .....	227
Mengambil paket Swift dari GitHub dan menerbitkan ulang ke CodeArtifact .....	229
Nama paket Swift dan normalisasi namespace .....	232
Pemecahan masalah cepat .....	232

Saya mendapatkan kesalahan 401 di Xcode bahkan setelah mengonfigurasi Swift Package Manager .....	232
Xcode hang pada mesin CI karena permintaan gantungan kunci untuk kata sandi .....	233
Menggunakan paket generik .....	236
Gambaran umum umum .....	236
Kendala .....	236
Perintah yang Didukung .....	237
Menerbitkan dan mengonsumsi paket generik .....	238
Menerbitkan paket generik .....	238
Mencantumkan aset Mencantumkan aset .....	240
Mengunduh aset Mengunduh aset Mengunduh aset .....	241
Menggunakan CodeArtifact dengan CodeBuild .....	243
Menggunakan paket npm diCodeBuild .....	243
Mengatur izin dengan IAM role .....	243
Masuk dan menggunakan npm .....	244
Menggunakan paket Python diCodeBuild .....	245
Mengatur izin dengan IAM role .....	245
Masuk dan gunakan pip atau twine .....	246
Menggunakan paket Maven diCodeBuild .....	248
Mengatur izin dengan IAM role .....	248
Menggunakan gradle atau mvn .....	249
MenggunakanNuGetpaket diCodeBuild .....	250
Mengatur izin dengan IAM role .....	250
MengonsumsiNuGetpaket .....	251
Bangun denganNuGetpaket .....	253
PublikasikanNuGetpaket .....	255
Caching Dependensi .....	257
Pemantauan CodeArtifact .....	258
Memantau CodeArtifact peristiwa .....	258
CodeArtifact format acara dan contoh .....	260
Gunakan acara untuk memulai CodePipeline eksekusi .....	264
Konfigurasi EventBridge izin .....	264
Buat EventBridge aturan .....	264
Buat target EventBridge aturan .....	264
Menggunakan peristiwa untuk menjalankan fungsi Lambda .....	265
Buat EventBridge aturan .....	265

Buat target EventBridge aturan .....	265
Konfigurasi EventBridge izin .....	266
Keamanan .....	267
Perlindungan data .....	268
Enkripsi data .....	269
Privasi lalu lintas .....	269
Memantau .....	269
Mencatat panggilan API CodeArtifact dengan AWS CloudTrail .....	270
Validasi kepatuhan .....	274
Autentikasi dan token .....	275
Token dibuat dengan perintah login .....	277
Token dibuat dengan API GetAuthorizationToken .....	278
Teruskan token auth menggunakan variabel lingkungan .....	279
Mencabut token otorisasi CodeArtifact .....	280
Ketahanan .....	281
Keamanan infrastruktur .....	281
Serangan substitusi ketergantungan .....	282
Manajemen Identitas dan Akses .....	282
Audiens .....	283
Mengautentikasi dengan identitas .....	284
Mengelola akses menggunakan kebijakan .....	287
Bagaimana AWS CodeArtifact bekerja dengan IAM .....	290
Contoh kebijakan berbasis identitas .....	298
Menggunakan tanda untuk mengontrol akses berbasis CodeArtifact peran .....	307
AWSCodeArtifactreferensi izin .....	312
Pemecahan Masalah .....	314
Bekerja dengan VPC endpoint .....	316
Buat VPC endpoint .....	317
Buat titik akhir gateway Amazon S3 .....	318
Izin bucket Amazon S3 minimum untuk AWS CodeArtifact .....	318
Gunakan CodeArtifact dari VPC .....	321
Konfigurasi AWS CLI untuk menggunakan titik <code>codeartifact.api</code> akhir .....	321
Gunakan <code>codeartifact.repositories</code> titik akhir tanpa DNS pribadi .....	322
Buat kebijakan titik akhir VPC .....	324
Sumber daya AWS CloudFormation .....	325
CodeArtifact dan AWS CloudFormation template .....	325

---

Mencegah penghapusan sumber daya CodeArtifact .....	325
Pelajari selengkapnya tentang AWS CloudFormation .....	326
Pemecahan Masalah .....	327
Saya tidak dapat melihat notifikasi .....	327
Penandaan pada sumber daya .....	328
CodeArtifact alokasi biaya dengan tag .....	329
Mengalokasikan biaya penyimpanan data di CodeArtifact .....	329
Mengalokasikan biaya permintaan di CodeArtifact .....	329
Kuota diAWSCodeArtifact .....	330
Riwayat dokumen .....	333
.....	cccxliv

# Apakah AWS CodeArtifact itu?

AWS CodeArtifact adalah layanan repositori artefak terkelola yang aman, sangat terukur, terkelola, membantu organisasi untuk menyimpan dan berbagi paket perangkat lunak untuk pengembangan aplikasi. Anda dapat menggunakan CodeArtifact dengan alat build populer dan manajer paket seperti NuGet CLI, Maven, Gradle, npm, yarn, pip, dan twine. CodeArtifact membantu mengurangi kebutuhan Anda untuk mengelola sistem penyimpanan artefak Anda sendiri atau khawatir tentang penskalaan infrastrukturnya. Tidak ada batasan jumlah atau ukuran total paket yang dapat Anda simpan di CodeArtifact repositori.

Anda dapat membuat koneksi antara CodeArtifact repositori pribadi Anda dan eksternal, repositori publik, seperti npmjs.com atau Maven Central. CodeArtifact kemudian akan mengambil dan menyimpan paket sesuai permintaan dari repositori publik ketika mereka diminta oleh manajer paket. Hal ini membuat lebih mudah untuk menggunakan dependensi sumber terbuka yang digunakan oleh aplikasi Anda dan membantu memastikan mereka selalu tersedia untuk membangun dan mengembangkan. Anda juga dapat mempublikasikan paket pribadi ke CodeArtifact repositori. Ini membantu Anda berbagi komponen perangkat lunak berpemilik antara beberapa aplikasi dan tim pengembangan di organisasi Anda.

Untuk informasi lebih lanjut, lihat [AWS CodeArtifact](#).

## Bagaimana cara CodeArtifact kerja?

CodeArtifact menyimpan paket perangkat lunak dalam repositori. Repositori bersifat polyglot—repositori tunggal dapat berisi paket dari jenis apa pun yang didukung. Setiap CodeArtifact repositori adalah anggota dari CodeArtifact domain tunggal. Sebaiknya gunakan satu domain produksi untuk organisasi Anda dengan satu atau beberapa repositori. Misalnya, Anda dapat menggunakan setiap repositori untuk tim pengembangan yang berbeda. Paket di repositori Anda kemudian dapat ditemukan dan dibagikan di seluruh tim pengembangan Anda.

Untuk menambahkan paket ke repositori, konfigurasi manajer paket seperti npm atau Maven untuk menggunakan titik akhir repositori (URL). Kemudian, Anda dapat menggunakan manajer paket untuk mempublikasikan paket ke repositori. Anda juga dapat mengimpor paket sumber terbuka ke repositori dengan mengonfigurasi koneksi eksternal ke repositori publik seperti npmjs, NuGet Galeri, Maven Central, atau PyPI. Untuk informasi selengkapnya, lihat [Connect CodeArtifact repositori ke repositori publik](#).

Anda dapat membuat paket dalam satu repositori tersedia untuk repositori lain di domain yang sama. Untuk melakukan ini, konfigurasi satu repositori sebagai hulu untuk yang lain. Semua versi paket yang tersedia untuk repositori hulu juga tersedia untuk repositori hilir. Selain itu, semua paket yang tersedia untuk repositori hulu melalui koneksi eksternal ke repositori publik tersedia untuk repositori hilir. Untuk informasi selengkapnya, lihat [Bekerja dengan repositori hulu di CodeArtifact](#).

## Konsep AWS CodeArtifact

Berikut adalah beberapa konsep dan syarat yang perlu diketahui ketika Anda menggunakan CodeArtifact.

Topik

- [Aset](#)
- [Domain](#)
- [Repositori](#)
- [Package](#)
- [Namespace paket](#)
- [Versi paket](#)
- [Revisi versi paket](#)
- [Repositori hulu](#)

### Aset

Aset adalah file individu yang disimpan dalam CodeArtifact yang terkait dengan versi paket, seperti file npm atau .tgz file Maven POM dan JAR.

### Domain

Repositori dikumpulkan menjadi entitas tingkat yang lebih tinggi yang dikenal sebagai domain. Semua aset dan metadata paket disimpan dalam domain, tetapi digunakan melalui repositori. Aset paket tertentu, seperti file JAR Maven, disimpan satu kali per domain, terlepas dari berapa banyak repositori tempat file tersebut berada. Semua aset dan metadata dalam domain dienkripsi dengan kunci KMS yang sama yang AWS KMS key disimpan di AWS Key Management Service (AWS KMS).

Setiap repositori adalah anggota dari satu domain dan tidak dapat dipindahkan ke domain lainnya.

Dengan menggunakan domain, Anda dapat menerapkan kebijakan organisasi di beberapa repositori. Dengan pendekatan ini, Anda menentukan akun mana yang dapat mengakses repositori di domain, dan repositori publik mana yang dapat digunakan sebagai sumber paket.

Meskipun sebuah organisasi dapat memiliki beberapa domain, kami merekomendasikan domain produksi tunggal yang berisi semua artefak yang dipublikasikan. Dengan begitu, tim dapat menemukan dan berbagi paket di seluruh organisasi Anda.

## Repositori

Sebuah CodeArtifact repositori berisi satu set [versi paket](#), yang masing-masing peta ke satu set [aset](#). Repositori bersifat polyglot—repositori tunggal dapat berisi paket dari jenis apa pun yang didukung. Setiap repositori mengekspos titik akhir untuk mengambil dan menerbitkan paket menggunakan alat seperti nuget CLI, npm CLI (mvn), dan pip. Anda dapat membuat hingga 1.000 repositori per domain.

## Package

Paket adalah kumpulan perangkat lunak dan metadata yang diperlukan untuk menyelesaikan dependensi dan menginstal perangkat lunak. Di CodeArtifact, paket terdiri dari nama paket, [namespace](#) opsional seperti `@types` di `@types/node`, satu set versi paket, dan metadata tingkat paket seperti tag npm.

AWS CodeArtifact mendukung [npm](#), [PyPI](#), [Maven](#), [NuGet](#), dan format paket [generik](#).

## Namespace paket

Beberapa format paket mendukung nama paket hierarkis untuk mengatur paket ke dalam kelompok logis dan membantu menghindari kesamaan nama. Misalnya, npm mendukung cakupan. Untuk informasi selengkapnya, lihat [dokumentasi cakupan npm](#). Paket npm `@types/node` memiliki cakupan `@types` dan nama `node`. Ada banyak nama paket lainnya dalam cakupan `@types`. Dalam CodeArtifact, cakupan (“jenis”) disebut sebagai namespace paket dan nama (“simpul”) disebut sebagai nama paket. Untuk paket Maven, namespace paket sesuai dengan groupID Maven. Paket Maven `org.apache.logging.log4j:log4j` memiliki groupID (namespace paket) `org.apache.logging.log4j` dan artifactID (nama paket) `log4j`. Untuk paket generik, sebuah [namespace](#) diperlukan. Beberapa format paket seperti PyPI tidak mendukung nama hierarkis dengan konsep yang mirip dengan cakupan npm atau groupID Maven. Tanpa mengelompokkan nama paket, menghindari kesamaan nama bisa jadi lebih sulit.

## Versi paket

Versi paket mengidentifikasi versi spesifik dari sebuah paket, seperti `@types/node 12.6.9`. Format nomor versi dan semantik bervariasi untuk format paket yang berbeda. Sebagai contoh, npm paket versi harus sesuai dengan [spesifikasi Versioning Semantik](#). Dalam CodeArtifact, versi paket terdiri dari pengidentifikasi versi, paket versi metadata tingkat, dan satu set aset.

## Revisi versi paket

Revisi versi paket adalah string yang mengidentifikasi satu set aset dan metadata tertentu untuk versi paket. Setiap kali versi paket diperbarui, revisi versi paket baru dibuat. Misalnya, Anda dapat memublikasikan arsip distribusi sumber (`sdist`) untuk versi paket Python, dan kemudian menambahkan sebuah wheel Python yang berisi kode yang dikompilasi ke versi yang sama. Ketika Anda memublikasikan wheel, revisi versi paket baru dibuat.

## Repositori hulu

Satu repositori merupakan hulu dari repositori lain ketika versi paket di dalamnya dapat diakses dari titik akhir repositori hilir. Pendekatan ini secara efektif menggabungkan isi dari dua repositori dari sudut pandang klien. Menggunakan CodeArtifact, Anda dapat membuat hubungan hulu antara dua repositori.

## Bagaimana saya memulai CodeArtifact?

Kami menyarankan agar Anda menyelesaikan langkah berikut:

1. Pelajari lebih lanjut tentang CodeArtifact dengan membaca [Konsep AWS CodeArtifact](#).
2. Siapkan pengguna Anda Akun AWS CLI, pengguna, dan IAM dengan mengikuti langkah-langkah di [Mengatur dengan AWS CodeArtifact](#).
3. Gunakan CodeArtifact dengan mengikuti petunjuk di [Memulai dengan CodeArtifact](#).



# Mengatur dengan AWS CodeArtifact

Jika Anda sudah mendaftar untuk Amazon Web Services (AWS), Anda dapat AWS CodeArtifact segera mulai menggunakannya. Anda dapat membuka CodeArtifact konsol, memilih Buat domain dan repositori, dan ikuti langkah-langkah di wizard peluncuran untuk membuat domain dan repositori pertama Anda.

Jika Anda belum mendaftar, atau memerlukan bantuan untuk AWS membuat domain dan repositori pertama Anda, selesaikan tugas-tugas berikut untuk menyiapkan penggunaan: CodeArtifact

## Topik

- [Daftar untuk AWS](#)
- [Menginstal atau meningkatkan dan kemudian mengonfigurasi AWS CLI](#)
- [Menyediakan pengguna IAM](#)
- [Menginstal manajer paket atau alat pembangun](#)

## Daftar untuk AWS

Saat mendaftar ke Amazon Web Services (AWS), Anda hanya dikenakan biaya untuk layanan dan sumber daya yang Anda gunakan, termasuk AWS CodeArtifact.

Jika Anda sudah memiliki Akun AWS, lompat ke tugas berikutnya, [Menginstal atau meningkatkan dan kemudian mengonfigurasi AWS CLI](#). Jika Anda belum memiliki Akun AWS, gunakan prosedur berikut untuk membuatnya.

### Untuk membuat 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 Akun AWS, 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](#).

# Menginstal atau meningkatkan dan kemudian mengonfigurasi AWS CLI

Untuk memanggil CodeArtifact perintah dari AWS Command Line Interface (AWS CLI) pada mesin pengembangan lokal, Anda harus menginstal AWS CLI.

Jika Anda memiliki versi yang lebih lama dari yang AWS CLI diinstal, Anda harus memutakhirkannya sehingga CodeArtifact perintah tersedia. CodeArtifact perintah tersedia dalam AWS CLI versi berikut:

1. AWS CLI 1: 1.18.77 dan lebih baru
2. AWS CLI 2: 2.0.21 dan lebih baru

Untuk memeriksa versi, gunakan perintah `aws --version`.

Untuk menginstal dan mengonfigurasi AWS CLI

1. Instal atau tingkatkan AWS CLI dengan petunjuk di [Installing the AWS Command Line Interface](#).
2. Konfigurasi AWS CLI, dengan perintah `configure`, seperti berikut.

```
aws configure
```

Saat diminta, tentukan kunci AWS akses dan kunci akses AWS rahasia pengguna IAM yang akan Anda gunakan. Saat diminta untuk Wilayah AWS nama default, tentukan Wilayah tempat Anda akan membuat pipeline, seperti `us-east-2`. Saat diminta untuk format output default, tentukan `json`.

## Important

Ketika Anda mengonfigurasi AWS CLI, Anda diminta untuk menentukan Wilayah AWS. Pilih salah satu Wilayah yang didukung yang tercantum di [Wilayah dan Titik Akhir](#) di Referensi Umum AWS.

Untuk informasi selengkapnya, lihat [Mengonfigurasi AWS Command Line Interface dan Mengelola kunci akses untuk pengguna IAM](#).

3. Untuk memverifikasi penginstalan atau peningkatan, panggil perintah berikut dari AWS CLI.

```
aws codeartifact help
```

Jika berhasil, perintah ini menampilkan daftar CodeArtifact perintah yang tersedia.

Selanjutnya, Anda dapat membuat pengguna IAM dan memberikan akses kepada CodeArtifact pengguna tersebut. Untuk informasi selengkapnya, lihat [Menyediakan pengguna IAM](#).

## Menyediakan pengguna IAM

Ikuti petunjuk ini untuk mempersiapkan pengguna IAM untuk digunakan CodeArtifact.

Untuk menyediakan pengguna AniAM

1. Buat pengguna IAM, atau gunakan yang terkait dengan pengguna AndaAkun AWS. Untuk informasi selengkapnya, lihat [Membuat pengguna IAM](#) dan [Ikhtisar kebijakan AWS IAM](#) di Panduan Pengguna IAM.
2. Berikan akses pengguna IAM ke CodeArtifact.
  - Opsi 1: Buat kebijakan IAM khusus. Dengan kebijakan IAM khusus, Anda dapat memberikan izin minimum yang diperlukan dan mengubah berapa lama autentikasi token berlaku. Untuk informasi selengkapnya dan kebijakan contoh, lihat [Contoh kebijakan berbasis identitas untuk AWS CodeArtifact](#).
  - Opsi 2: Gunakan kebijakan terkelola AWSCodeArtifactAdminAccess AWS. Cuplikan berikut menunjukkan isi kebijakan ini.

### Important

Kebijakan ini memberikan akses ke semua CodeArtifact API. Sebaiknya Anda selalu menggunakan izin minimum yang diperlukan untuk menyelesaikan tugas Anda. Untuk informasi selengkapnya, lihat [Praktik terbaik IAM](#) dalam Panduan Pengguna IAM.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
```

```
        "codeartifact:*"
    ],
    "Effect": "Allow",
    "Resource": "*"
  },
  {
    "Effect": "Allow",
    "Action": "sts:GetServiceBearerToken",
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "sts:AWSServiceName": "codeartifact.amazonaws.com"
      }
    }
  }
]
}
```

`sts:GetServiceBearerToken` izin diperlukan untuk memanggil CodeArtifact `GetAuthorizationToken` API. API ini mengembalikan token yang harus digunakan saat menggunakan manajer paket seperti `npm` atau `pip` dengan CodeArtifact. Untuk menggunakan manajer paket dengan CodeArtifact repositori, pengguna atau peran IAM Anda harus mengizinkan `sts:GetServiceBearerToken` seperti yang ditunjukkan pada contoh kebijakan sebelumnya.

Jika Anda belum menginstal manajer paket atau alat build yang Anda rencanakan untuk digunakan CodeArtifact, lihat [Menginstal manajer paket atau alat pembangun](#).

## Menginstal manajer paket atau alat pembangun

Untuk mempublikasikan atau menggunakan paket dari CodeArtifact, Anda harus menggunakan manajer paket. Ada manajer paket yang berbeda untuk setiap jenis paket. Daftar berikut berisi beberapa manajer paket yang dapat Anda gunakan CodeArtifact. Jika Anda belum melakukannya, instal manajer paket untuk jenis paket yang ingin Anda gunakan.

- [Untuk npm, gunakan npm CLI atau pnpm.](#)
- [Untuk Maven, gunakan Apache Maven \(\) atau Gradle. mvn](#)
- Untuk Python, gunakan [pip](#) untuk menginstal paket dan [benang](#) untuk menerbitkan paket.
- [Untuk NuGet, gunakan Toolkit for Visual Studio di Visual Studio atau CLI nuget atau dotnet.](#)

- Untuk paket [generik](#), gunakan [AWS CLI](#) atau SDK untuk mempublikasikan dan mengunduh konten paket.

## Langkah berikutnya

Langkah Anda selanjutnya akan tergantung pada jenis atau jenis paket yang Anda gunakan CodeArtifact, dan status sumber CodeArtifact daya Anda.

Jika Anda memulai untuk pertama kalinya CodeArtifact untuk diri sendiri, tim, atau organisasi Anda, lihat dokumentasi berikut untuk informasi umum memulai dan membantu menciptakan sumber daya yang Anda perlukan.

- [Memulai menggunakan konsol](#)
- [Memulai menggunakan AWS CLI](#)

Jika sumber daya Anda telah dibuat dan Anda siap untuk mengkonfigurasi manajer paket Anda untuk mendorong paket ke atau menginstal paket dari CodeArtifact repositori, lihat dokumentasi yang sesuai dengan jenis paket atau manajer paket Anda.

- [Menggunakan CodeArtifact dengan npm](#)
- [Menggunakan CodeArtifact dengan Python](#)
- [Menggunakan CodeArtifact dengan Maven](#)
- [Menggunakan CodeArtifact dengan NuGet](#)
- [Menggunakan CodeArtifact dengan paket generik](#)

# Memulai dengan CodeArtifact

Dalam tutorial memulai ini, Anda menggunakan CodeArtifact untuk membuat yang berikut:

- Domain yang disebut `my-domain`.
- Repositori yang disebut `my-repo` yang berada dalam `my-domain`.
- Repositori yang disebut `npm-store` yang berada dalam `my-domain`. `npm-store` memiliki koneksi eksternal ke repositori publik npm. Koneksi ini digunakan untuk menyerap paket npm ke dalam repositori `my-repo`.

Sebelum memulai tutorial ini, kami sarankan agar Anda meninjau CodeArtifact [Konsep AWS CodeArtifact](#).

## Note

Tutorial ini mengharuskan Anda untuk membuat sumber daya yang mungkin menimbulkan biaya untuk akun AWS Anda. Untuk informasi selengkapnya, lihat [CodeArtifact harga](#).

## Topik

- [Prasyarat](#)
- [Memulai menggunakan konsol](#)
- [Memulai menggunakan AWS CLI](#)

## Prasyarat

Anda dapat menyelesaikan tutorial ini menggunakan AWS Management Console atau AWS Command Line Interface (AWS CLI). Untuk mengikuti tutorial, Anda harus terlebih dahulu menyelesaikan prasyarat berikut:

- Selesaikan langkah-langkah dalam [Mengatur dengan AWS CodeArtifact](#).
- Instal CLI npm. Untuk informasi selengkapnya, lihat [Downloading and installing Node.js and npm](#) dalam dokumentasi npm.

# Memulai menggunakan konsol

Jalankan langkah-langkah berikut untuk memulai CodeArtifact menggunakan AWS Management Console. Panduan ini menggunakan manajer paket npm, jika Anda menggunakan manajer paket yang berbeda, Anda perlu memodifikasi beberapa langkah berikut.

1. Masuk ke AWS Management Console dan membuka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/start>. Untuk informasi selengkapnya, lihat [Mengatur dengan AWS CodeArtifact](#).
2. Pilih Buat repositori.
3. Di Repository name (Nama repositori), masukkan **my-repo**.
4. (Opsional) Dalam Repository Description (Deskripsi Repositori), masukkan deskripsi opsional untuk repositori Anda.
5. Di Public upstream repositories (Repositori hulu publik), pilih npm-store untuk membuat repositori yang terhubung ke npmjs yang merupakan hulu dari repositori my-repo Anda.

CodeArtifact memberikan nama npm-store ke repositori ini untuk Anda. Semua paket yang tersedia di repositori hulu npm-store juga tersedia untuk repositori hilirnya, my-repo.

6. Pilih Selanjutnya.
7. Di AWS account (Akun AWS), pilih This AWS account (Akun AWS ini).
8. Di Domain name (Nama domain), masukkan **my-domain**.
9. Perluas Additional configuration (Konfigurasi tambahan).
10. Anda harus menggunakan AWS KMS key (Kunci KMS) untuk mengenkripsi semua aset di domain Anda. Anda dapat menggunakan Kunci yang dikelola AWS atau kunci KMS yang Anda kelola:
  - Memiih Kunci terkelola AWS jika Anda ingin menggunakan default Kunci yang dikelola AWS.
  - Memiih Kunci yang dikelola pelanggan jika Anda ingin menggunakan kunci KMS yang Anda kelola. Untuk menggunakan kunci KMS yang Anda kelola, di Kunci yang dikelola pelanggan, cari dan pilih tombol KMS.

Untuk informasi selengkapnya, lihat [Kunci yang dikelola AWS](#) dan [Kunci yang dikelola pelanggan](#) di dalam AWS Key Management Service Panduan Pengembang.

11. Pilih Selanjutnya.
12. Masuk Memeriksa dan membuat, meninjau apa CodeArtifact menciptakan untuk Anda.

- Package flow (Alur paket) menunjukkan bagaimana `my-domain`, `my-repo`, dan `npm-store` terkait.
- Langkah 1: Membuat repositoriMenampilkan detail tentang`my-repo`dan`npm-store`.
- Langkah 2: Pilih domainMenampilkan detail tentang`my-domain`.

Saat Anda siap, pilih Create repository (Buat repositori).

13. Di halaman `my-repo`, pilih View connection instructions (Lihat petunjuk koneksi), lalu pilih `npm`.
14. GunakanAWS CLIuntuk menjalankanloginperintah ditampilkan di bawahKonfigurasi klien `npm` Anda menggunakan iniAWS CLI CodeArtifactkomando.

```
aws codeartifact login --tool npm --repository my-repo --domain my-domain --domain-owner 111122223333
```

Anda akan menerima output yang mengonfirmasi Anda berhasil masuk.

```
Successfully configured npm to use AWS CodeArtifact repository https://my-domain-111122223333.d.codeartifact.us-east-2.amazonaws.com/npm/my-repo/  
Login expires in 12 hours at 2020-10-08 02:45:33-04:00
```

Jika Anda menerima kesalahan`Could not connect to the endpoint URL`, pastikan bahwaAWS CLIdikonfigurasi dan bahwaNama wilayah defaultdiatur ke Wilayah yang sama di mana Anda membuat repositori Anda, lihat[Mengonfigurasi AWS Command Line Interface](#).

Untuk informasi selengkapnya, lihat[Konfigurasi dan gunakan npm dengan CodeArtifact](#)

15. Gunakan CLI `npm` untuk menginstal paket `npm`. Misalnya, untuk menginstal paket `npm` populer `lodash`, gunakan perintah berikut.

```
npm install lodash
```

16. Kembali ke CodeArtifact konsol. Jika repositori `my-repo` dibuka, segarkan halaman. Atau, di panel navigasi, pilih Repository (Repositori), lalu pilih `my-repo`.

Di bagian Packages (Paket), Anda akan melihat perpustakaan `npm` atau paket yang Anda instal. Anda dapat memilih nama paket untuk melihat versi dan statusnya. Anda dapat memilih versi terbarunya untuk melihat detail paket seperti dependensi, aset, dan lainnya.



**Note**

Mungkin ada penundaan antara ketika Anda menginstal paket dan ketika diserap ke dalam repositori Anda.

17. Untuk menghindari biaya AWS lebih lanjut, hapus sumber daya yang Anda gunakan selama tutorial ini:

**Note**

Anda tidak dapat menghapus domain yang berisi repositori, sehingga Anda harus menghapus `my-repo` dan `npm-store` sebelum Anda menghapus `my-domain`.

- a. Di panel navigasi, pilih Repository (Repositori).
- b. Pilih `npm-store`, pilih Delete (Hapus), dan kemudian ikuti langkah-langkah untuk menghapus repositori.
- c. Pilih `my-repo`, pilih Delete (Hapus), dan kemudian ikuti langkah-langkah untuk menghapus repositori.
- d. Dari panel navigasi, pilih Domains (Domain).
- e. Pilih `my-domain`, pilih Delete (Hapus), dan kemudian ikuti langkah-langkah untuk menghapus domain.

## Memulai menggunakan AWS CLI

Jalankan langkah-langkah berikut untuk memulai CodeArtifact menggunakan AWS Command Line Interface (AWS CLI). Untuk informasi selengkapnya, lihat [Menginstal atau meningkatkan dan kemudian mengonfigurasi AWS CLI](#). Panduan ini menggunakan manajer paket npm, jika Anda menggunakan manajer paket yang berbeda, Anda perlu memodifikasi beberapa langkah berikut.

1. Gunakan AWS CLI untuk menjalankan perintah `create-domain`.

```
aws codeartifact create-domain --domain my-domain
```

Data berformat JSON muncul dalam output dengan detail tentang domain baru Anda.

```
{
  "domain": {
    "name": "my-domain",
    "owner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:domain/my-domain",
    "status": "Active",
    "createdTime": "2020-10-07T15:36:35.194000-04:00",
    "encryptionKey": "arn:aws:kms:us-west-2:111122223333:key/your-kms-key",
    "repositoryCount": 0,
    "assetSizeBytes": 0
  }
}
```

Jika Anda menerima kesalahan `Could not connect to the endpoint URL`, pastikan bahwa AWS CLI dikonfigurasi dan bahwa nama wilayah default diatur ke Wilayah yang sama di mana Anda membuat repositori Anda, lihat [Mengonfigurasi AWS Command Line Interface](#).

- Gunakan perintah `create-repository` untuk membuat repositori di domain Anda.

```
aws codeartifact create-repository --domain my-domain --domain-owner 111122223333
--repository my-repo
```

Data berformat JSON muncul dalam output dengan detail tentang repositori baru Anda.

```
{
  "repository": {
    "name": "my-repo",
    "administratorAccount": "111122223333",
    "domainName": "my-domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/my-domain/my-repo",
    "upstreams": [],
    "externalConnections": []
  }
}
```

- Gunakan perintah `create-repository` untuk membuat repositori hulu untuk repositori `my-repo` Anda.

```
aws codeartifact create-repository --domain my-domain --domain-owner 111122223333
--repository npm-store
```

Data berformat JSON muncul dalam output dengan detail tentang repositori baru Anda.

```
{
  "repository": {
    "name": "npm-store",
    "administratorAccount": "111122223333",
    "domainName": "my-domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/my-domain/npm-store",
    "upstreams": [],
    "externalConnections": []
  }
}
```

4. Gunakan perintah `associate-external-connection` untuk menambahkan koneksi eksternal ke repositori publik npm ke repositori `npm-store`.

```
aws codeartifact associate-external-connection --domain my-domain --domain-owner 111122223333
--repository npm-store --external-connection "public:npmjs"
```

Data berformat JSON muncul dalam output dengan detail tentang repositori dan koneksi eksternalnya yang baru.

```
{
  "repository": {
    "name": "npm-store",
    "administratorAccount": "111122223333",
    "domainName": "my-domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/my-domain/npm-store",
    "upstreams": [],
    "externalConnections": [
      {
        "externalConnectionName": "public:npmjs",
        "packageFormat": "npm",
      }
    ]
  }
}
```

```
        "status": "AVAILABLE"
      }
    ]
  }
}
```

Untuk informasi selengkapnya, lihat [Connect CodeArtifact repositori ke repositori publik](#).

5. Gunakan perintah `update-repository` untuk mengaitkan repositori `npm-store` sebagai repositori hulu ke repositori `my-repo`.

```
aws codeartifact update-repository --repository my-repo --domain my-domain --
domain-owner 111122223333 --upstreams repositoryName=npm-store
```

Data berformat JSON muncul dalam output dengan detail tentang pembaruan repositori Anda, termasuk repositori hulu barunya.

```
{
  "repository": {
    "name": "my-repo",
    "administratorAccount": "111122223333",
    "domainName": "my-domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/my-
domain/my-repo",
    "upstreams": [
      {
        "repositoryName": "npm-store"
      }
    ],
    "externalConnections": []
  }
}
```

Untuk informasi selengkapnya, lihat [Menambahkan atau menghapus repositori hulu \(AWS CLI\)](#).

6. Penggunaan perintah `login` untuk mengonfigurasi manajer paket `npm` Anda dengan repositori `my-repo`.

```
aws codeartifact login --tool npm --repository my-repo --domain my-domain --domain-
owner 111122223333
```

Anda akan menerima output yang mengonfirmasi Anda berhasil masuk.

```
Successfully configured npm to use AWS CodeArtifact repository https://my-domain-111122223333.d.codeartifact.us-east-2.amazonaws.com/npm/my-repo/  
Login expires in 12 hours at 2020-10-08 02:45:33-04:00
```

Untuk informasi selengkapnya, lihat [Konfigurasi dan gunakan npm dengan CodeArtifact](#).

- Gunakan CLI npm untuk menginstal paket npm. Misalnya, untuk menginstal paket npm populer `Lodash`, gunakan perintah berikut.

```
npm install Lodash
```

- Gunakan perintah `list-packages` untuk melihat paket yang baru saja Anda instal di repositori `my-repo`.

#### Note

Mungkin ada penundaan antara kapan `npm install` perintah selesai dan ketika paket terlihat di repositori Anda. Untuk detail tentang latensi tipikal saat mengambil paket dari repositori publik, lihat [Latensi koneksi lebih masensi koneksi lebih serensi](#).

```
aws codeartifact list-packages --domain my-domain --repository my-repo
```


Data berformat JSON muncul dalam output dengan format dan nama paket yang Anda instal.

```
{  
  "packages": [  
    {  
      "format": "npm",  
      "package": "Lodash"  
    }  
  ]  
}
```

Anda sekarang memiliki tiga CodeArtifact sumber daya:

- Domain `my-domain`.

- Repositori `my-repo` yang berada di `my-domain`. Repositori ini memiliki paket npm yang tersedia untuk itu.
  - Repositori `npm-store` yang berada di `my-domain`. Repositori ini memiliki koneksi eksternal ke repositori npm publik dan dikaitkan sebagai repositori hulu dengan repositori `my-repo`.
9. Untuk menghindari biaya AWS lebih lanjut, hapus sumber daya yang Anda gunakan selama tutorial ini:

 Note

Anda tidak dapat menghapus domain yang berisi repositori, sehingga Anda harus menghapus `my-repo` dan `npm-store` sebelum Anda menghapus `my-domain`.

- a. Gunakan perintah `delete-repository` untuk menghapus repositori `npm-store`.

```
aws codeartifact delete-repository --domain my-domain --domain-owner 111122223333 --repository my-repo
```

Data berformat JSON muncul dalam output dengan detail mengenai repositori yang dihapus.

```
{
  "repository": {
    "name": "my-repo",
    "administratorAccount": "111122223333",
    "domainName": "my-domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/my-domain/my-repo",
    "upstreams": [
      {
        "repositoryName": "npm-store"
      }
    ],
    "externalConnections": []
  }
}
```

- b. Gunakan perintah `delete-repository` untuk menghapus repositori `npm-store`.

```
aws codeartifact delete-repository --domain my-domain --domain-owner 111122223333 --repository npm-store
```

Data berformat JSON muncul dalam output dengan detail mengenai repositori yang dihapus.

```
{
  "repository": {
    "name": "npm-store",
    "administratorAccount": "111122223333",
    "domainName": "my-domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/my-domain/npm-store",
    "upstreams": [],
    "externalConnections": [
      {
        "externalConnectionName": "public:npmjs",
        "packageFormat": "npm",
        "status": "AVAILABLE"
      }
    ]
  }
}
```

- c. Gunakan perintah delete-domain untuk menghapus repositori *my-domain*.

```
aws codeartifact delete-domain --domain my-domain --domain-owner 111122223333
```

Data berformat JSON muncul dalam output dengan detail tentang domain yang dihapus.

```
{
  "domain": {
    "name": "my-domain",
    "owner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:domain/my-domain",
    "status": "Deleted",
    "createdTime": "2020-10-07T15:36:35.194000-04:00",
    "encryptionKey": "arn:aws:kms:us-west-2:111122223333:key/your-kms-key",
    "repositoryCount": 0,
    "assetSizeBytes": 0
  }
}
```

```
}
```



# Bekerja dengan repositori di CodeArtifact

Topik ini menunjukkan cara menggunakan CodeArtifact konsol, dan CodeArtifact API untuk membuat AWS CLI, membuat daftar, memperbarui, dan menghapus repositori.

Topik

- [Buatlah sebuah repositori](#)
- [Terhubung ke repositori](#)
- [Hapus sebuah repositori](#)
- [Mencantumkan repositori](#)
- [Melihat atau mengubah konfigurasi repositori](#)
- [Kebijakan repositori](#)
- [Menandai repositori di CodeArtifact](#)

## Buatlah sebuah repositori

Karena semua paket CodeArtifact disimpan dalam [repositori](#), untuk digunakan CodeArtifact, Anda harus membuatnya. Anda dapat membuat repositori menggunakan CodeArtifact konsol, AWS Command Line Interface (AWS CLI), atau AWS CloudFormation. Setiap repositori dikaitkan dengan AWS akun yang Anda gunakan saat Anda membuatnya. [Anda dapat memiliki beberapa repositori, dan mereka dibuat dan dikelompokkan dalam domain.](#) Ketika Anda membuat repositori, repositori tidak berisi paket apa pun. Repositori adalah polyglot, yang berarti bahwa satu repositori dapat berisi paket dari jenis apa pun yang didukung.

Untuk informasi tentang batas CodeArtifact layanan, seperti jumlah maksimum repositori yang diizinkan dalam satu domain, lihat [Kuota diAWSCodeArtifact](#). Jika Anda menekan jumlah maksimum repositori yang diizinkan, Anda dapat [menghapus repositori](#) untuk memberi ruang bagi lebih banyak.

Sebuah repositori dapat memiliki satu atau lebih CodeArtifact repositori yang terkait dengannya sebagai repositori upstream. Hal ini memungkinkan klien manajer paket mengakses paket yang terdapat di lebih dari satu repositori menggunakan titik akhir URL tunggal. Untuk informasi selengkapnya, lihat [Bekerja dengan repositori hulu di CodeArtifact](#).

Untuk informasi selengkapnya tentang mengelola CodeArtifact repositori dengan CloudFormation, lihat [Menciptakan CodeArtifact sumber daya dengan AWS CloudFormation](#)

**Note**

Setelah membuat repositori, Anda tidak dapat mengubah namanya, akun AWS, atau domain.

## Topik

- [Membuat repositori \(konsol\)](#)
- [Membuat repositori \(AWS CLI\)](#)
- [Membuat repositori dengan repositori hulu](#)

## Membuat repositori (konsol)

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Di panel navigasi, pilih Repository (Repositori) dan pilih Create repository (Buat repositori).
3. Untuk Repository name (Nama repositori), masukkan nama yang unik untuk repositori Anda.
4. (Opsional) Dalam Repository description (Deskripsi repositori), masukkan deskripsi opsional untuk repositori Anda.
5. (Opsional) Dalam Publish upstream repositories (Publikasikan repositori hulu), tambahkan repositori perantara yang menghubungkan repositori Anda ke otoritas paket seperti Maven Central atau npmjs.com.
6. Pilih Berikutnya.
7. Di AWS Account (Akun AWS), pilih This AWS account (Akun AWS ini) jika Anda masuk ke akun yang memiliki domain. Pilih Different AWS account (Akun AWS yang berbeda) jika akun AWS lain menjadi pemilik domain.
8. Di Domain, pilih domain tempat repositori akan dibuat.

Jika tidak ada domain di akun, Anda harus membuatnya. Masukkan nama untuk domain baru di Domain name (Nama domain).

Perluas Additional configuration (Konfigurasi tambahan).

Anda harus menggunakan AWS KMS key (kunci KMS) untuk mengenkripsi semua aset di domain Anda. Anda dapat menggunakan Kunci yang dikelola AWS atau kunci KMS yang Anda kelola:

**⚠ Important**

CodeArtifact hanya mendukung tombol [KMS simetris](#). Anda tidak dapat menggunakan [kunci KMS asimetris](#) untuk mengenkripsi domain Anda. CodeArtifact Untuk bantuan menentukan apakah kunci KMS simetris atau asimetris, lihat [Mengidentifikasi kunci KMS simetris dan asimetris](#).

- Pilih kunci terkelola AWS jika Anda ingin menggunakan default Kunci yang dikelola AWS.
- Pilih Customer managed key jika Anda ingin menggunakan kunci KMS yang Anda kelola. Untuk menggunakan kunci KMS yang Anda kelola, di ARN kunci terkelola Pelanggan, cari dan pilih kunci KMS.

Untuk informasi selengkapnya, lihat [Kunci yang dikelola AWS](#) dan [kunci yang dikelola pelanggan](#) di Panduan AWS Key Management Service Pengembang.

9. Pilih Berikutnya.

10. Di Tinjau dan buat, tinjau CodeArtifact apa yang dibuat untuk Anda.

- Package flow (Aliran paket) menunjukkan bagaimana domain dan repositori Anda terhubung.
- Langkah 1: Buat repositori menunjukkan detail tentang repositori dan repositori hulu opsional yang akan dibuat.
- Langkah 2: Pilih domain menunjukkan detail tentang `my_domain`.

Saat Anda siap, pilih Create repository (Buat repositori).

## Membuat repositori (AWS CLI)

Gunakan perintah `create-repository` untuk membuat repositori di domain Anda.

```
aws codeartifact create-repository --domain my_domain --domain-owner 111122223333 --  
repository my_repo --description "My new repository"
```

Contoh output:

```
{
```

```

"repository": {
  "name": "my_repo",
  "administratorAccount": "123456789012",
  "domainName": "my_domain",
  "domainOwner": "111122223333",
  "arn": "arn:aws:codeartifact:region-
id:111122223333:repository/my_domain/my_repo",
  "description": "My new repository",
  "upstreams": "[]",
  "externalConnections" "[]"
}
}

```

Repository baru tidak berisi paket apa pun. Setiap repository dikaitkan dengan akun AWS tempat Anda diautentikasi saat repository dibuat.

## Membuat repository dengan tanda

Untuk membuat repository dengan tanda, tambahkan parameter `--tags` ke perintah `create-domain`.

```

aws codeartifact create-repository --domain my_domain --domain-owner 111122223333 --
repository my_repo --tags key=k1,value=v1 key=k2,value=v2

```

## Membuat repository dengan repository hulu

Anda dapat menentukan satu atau beberapa repository hulu saat membuat repository.

```

aws codeartifact create-repository --domain my_domain --domain-owner 111122223333 --
repository my_repo \
  --upstreams repositoryName=my-upstream-repo --repository-description "My new
repository"

```

Contoh output:

```

{
  "repository": {
    "name": "my_repo",
    "administratorAccount": "123456789012",
    "domainName": "my_domain",
    "domainOwner": "111122223333",

```

```
"arn": "arn:aws:codeartifact:region-  
id:111122223333:repository/my_domain/my_repo",  
"description": "My new repository",  
"upstreams": [  
  {  
    "repositoryName": "my-upstream-repo"  
  }  
],  
"externalConnections": []  
}
```

### Note

Untuk membuat repositori dengan hulu, Anda harus memiliki izin untuk tindakan `AssociateWithDownstreamRepository` pada repositori hulu.

Untuk menambahkan hulu ke repositori setelah dibuat, lihat [Menambahkan atau menghapus repositori hulu \(konsol\)](#) dan [Menambahkan atau menghapus repositori hulu \(AWS CLI\)](#).

## Terhubung ke repositori

Setelah Anda mengonfigurasi profil dan kredensi Anda untuk mengautentikasi ke AWS akun Anda, putuskan repositori mana yang akan digunakan. CodeArtifact Anda memiliki opsi berikut:

- Buat repositori. Untuk informasi selengkapnya, lihat [Membuat Repositori](#).
- Gunakan repositori yang sudah ada di akun Anda. Anda dapat menggunakan perintah `list-repositories` untuk menemukan repositori yang dibuat di akun AWS Anda. Untuk informasi selengkapnya, lihat [Mencantumkan repositori](#).
- Gunakan repositori di akun yang berbeda AWS . Untuk informasi selengkapnya, lihat [Kebijakan repositori](#).

## Menggunakan klien manajer paket

Setelah Anda mengetahui repositori mana yang ingin Anda gunakan, lihat salah satu topik berikut.

- [Menggunakan CodeArtifact dengan Maven](#)

- [Menggunakan CodeArtifact dengan npm](#)
- [Menggunakan CodeArtifact dengan NuGet](#)
- [Menggunakan CodeArtifact dengan Python](#)

## Hapus sebuah repositori

Anda dapat menghapus repositori menggunakan CodeArtifact konsol atau file. AWS CLI Setelah repositori dihapus, Anda tidak dapat lagi mendorong paket ke sana atau menarik paket darinya. Semua paket dalam repositori menjadi tidak tersedia secara permanen dan tidak dapat dipulihkan. Anda dapat membuat repositori dengan nama yang sama, namun isinya akan kosong.

Topik

- [Menghapus repositori \(konsol\)](#)
- [Menghapus repositori \(AWS CLI\)](#)

### Menghapus repositori (konsol)

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Pada panel navigasi, pilih Repositories (Repositori), lalu pilih repositori yang ingin Anda hapus.
3. Pilih Delete (Hapus) dan kemudian ikuti langkah-langkah untuk menghapus domain.

### Menghapus repositori (AWS CLI)

Gunakan perintah `delete-repository` untuk menghapus repositori.

```
aws codeartifact delete-repository --domain my_domain --domain-owner 111122223333 --  
repository my_repo
```

Contoh output:

```
{  
  "repository": {  
    "name": "my_repo",  
    "administratorAccount": "123456789012",  
    "domainName": "my_domain",  
    "domainOwner": "123456789012",
```

```
    "arn": "arn:aws:codeartifact:region-  
id:123456789012:repository/my_domain/my_repo",  
    "description": "My new repository",  
    "upstreams": [],  
    "externalConnections": []  
  }  
}
```

## Mencantumkan repositori

Gunakan perintah dalam topik ini untuk membuat daftar repositori di AWS akun atau domain.

### Mencantumkan repositori dalam akun AWS

Gunakan perintah ini untuk membuat daftar semua repositori di akun Anda AWS .

```
aws codeartifact list-repositories
```

Contoh output:

```
{  
  "repositories": [  
    {  
      "name": "repo1",  
      "administratorAccount": "123456789012",  
      "domainName": "my_domain",  
      "domainOwner": "123456789012",  
      "arn": "arn:aws:codeartifact:region-  
id:123456789012:repository/my_domain/repo1",  
      "description": "Description of repo1"  
    },  
    {  
      "name": "repo2",  
      "administratorAccount": "123456789012",  
      "domainName": "my_domain",  
      "domainOwner": "123456789012",  
      "arn": "arn:aws:codeartifact:region-  
id:123456789012:repository/my_domain/repo2",  
      "description": "Description of repo2"  
    },  
  ],  
}
```

```
{
  "name": "repo3",
  "administratorAccount": "123456789012",
  "domainName": "my_domain2",
  "domainOwner": "123456789012",
  "arn": "arn:aws:codeartifact:region-
id:123456789012:repository/my_domain2/repo3",
  "description": "Description of repo3"
}
]
```

Anda dapat memberi nomor halaman pada respons dari `list-repositories` menggunakan `--max-results` dan parameter `--next-token`. Untuk `--max-results`, tentukan bilangan bulat dari 1 sampai 1000 untuk menentukan jumlah hasil yang dikembalikan dalam satu halaman. Defaultnya adalah 50. Untuk mengembalikan halaman berikutnya, jalankan `list-repositories` lagi dan teruskan nilai `nextToken` yang diterima dalam perintah output sebelumnya untuk `--next-token`. Saat opsi `--next-token` tidak digunakan, halaman pertama hasil selalu dikembalikan.

## Mencantumkan repositori di domain

Gunakan `list-repositories-in-domain` untuk mendapatkan daftar semua repositori dalam domain.

```
aws codeartifact list-repositories-in-domain --domain my_domain --domain-
owner 123456789012 --max-results 3
```

Output menunjukkan bahwa beberapa repositori dikelola oleh akun AWS yang berbeda.

```
{
  "repositories": [
    {
      "name": "repo1",
      "administratorAccount": "123456789012",
      "domainName": "my_domain",
      "domainOwner": "111122223333",
      "arn": "arn:aws:codeartifact:region-
id:111122223333:repository/my_domain/repo1",
      "description": "Description of repo1"
    },
    {
```



```

    "name": "repo2",
    "administratorAccount": "444455556666",
    "domainName": "my_domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:region-
id:111122223333:repository/my_domain/repo2",
    "description": "Description of repo2"
  },
  {
    "name": "repo3",
    "administratorAccount": "444455556666",
    "domainName": "my_domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:region-
id:111122223333:repository/my_domain/repo3",
    "description": "Description of repo3"
  }
]
}

```

Anda dapat memberi nomor halaman pada respons dari `list-repositories-in-domain` menggunakan `--max-results` dan parameter `--next-token`. Untuk `--max-results`, tentukan bilangan bulat dari 1 sampai 1000 untuk menentukan jumlah hasil yang dikembalikan dalam satu halaman. Defaultnya adalah 50. Untuk mengembalikan halaman berikutnya, jalankan `list-repositories-in-domain` lagi dan teruskan nilai `nextToken` yang diterima dalam perintah output sebelumnya untuk `--next-token`. Saat opsi `--next-token` tidak digunakan, halaman pertama hasil selalu dikembalikan.

Untuk menampilkan output nama repositori dalam daftar yang lebih ringkas, coba perintah berikut.

```

aws codeartifact list-repositories-in-domain --domain my_domain --domain-
owner 111122223333 \
  --query 'repositories[*].[name]' --output text

```

Contoh output:

```

repo1
repo2
repo3

```

Contoh berikut menampilkan output ID akun selain nama repositori.

```
aws codeartifact list-repositories-in-domain --domain my_domain --domain-  
owner 111122223333 \  
--query 'repositories[*].[name,administratorAccount]' --output text
```

Contoh output:

```
repo1 710221105108  
repo2 710221105108  
repo3 532996949307
```

Untuk informasi selengkapnya tentang `--query` parameter, lihat [ListRepositories](#) di Referensi CodeArtifact API.

## Melihat atau mengubah konfigurasi repositori

Anda dapat melihat dan memperbarui detail tentang repositori Anda menggunakan CodeArtifact konsol atau AWS Command Line Interface (AWS CLI).

### Note

Setelah membuat repositori, Anda tidak dapat mengubah namanya, akun AWS, atau domain.

Topik

- [Melihat atau mengubah konfigurasi repositori \(konsol\)](#)
- [Melihat atau mengubah konfigurasi repositori \(AWS CLI\)](#)

## Melihat atau mengubah konfigurasi repositori (konsol)

Anda dapat melihat detail tentang dan memperbarui repositori Anda menggunakan konsol CodeArtifact

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Di panel navigasi, pilih Repositories (Repositori), kemudian pilih nama repositori yang ingin Anda lihat atau ubah.

### 3. Perluas Details (Detail) untuk melihat hal berikut:

- Domain repositori. Pilih nama domain untuk mempelajari selengkapnya tentang hal itu.
- Kebijakan sumber daya repositori. Pilih Apply a repository policy (Terapkan kebijakan repositori) untuk menambahkan.
- Amazon Resource Name (ARN) repositori.
- Jika repositori Anda memiliki koneksi eksternal, Anda dapat memilih koneksi untuk mempelajari selengkapnya tentang hal itu. Sebuah repositori hanya dapat memiliki satu koneksi eksternal. Untuk informasi selengkapnya, lihat [Connect CodeArtifact repositori ke repositori publik](#).
- Jika repositori Anda memiliki repositori hulu, Anda dapat memilih salah satu untuk melihat detailnya. Sebuah repositori dapat memiliki maksimal 10 repositori hulu langsung. Untuk informasi selengkapnya, lihat [Bekerja dengan repositori hulu di CodeArtifact](#).

#### Note

Repositori dapat memiliki koneksi eksternal atau repositori hulu, tetapi tidak keduanya.

4. Dalam Packages (Paket), Anda dapat melihat paket yang tersedia untuk repositori ini. Pilih paket untuk mempelajari selengkapnya tentang hal itu.
5. Pilih Lihat instruksi koneksi, lalu pilih manajer paket untuk mempelajari cara mengonfigurasinya CodeArtifact.
6. Pilih Apply repository policy (Terapkan kebijakan repositori) untuk memperbarui atau menambahkan kebijakan sumber daya ke repositori Anda. Untuk informasi selengkapnya, lihat [Kebijakan repositori](#).
7. Pilih Edit untuk menambahkan atau memperbarui hal berikut.
  - Deskripsi repositori.
  - Tanda yang terkait dengan repositori.
  - Jika repositori Anda memiliki koneksi eksternal, Anda dapat mengubah repositori publik yang terhubung dengannya. Jika tidak, Anda dapat menambahkan satu atau beberapa repositori yang ada sebagai repositori hulu. Atur mereka dalam urutan yang Anda inginkan mereka diprioritaskan CodeArtifact ketika paket diminta. Untuk informasi selengkapnya, lihat [Urutan prioritas repositori hulu](#).

## Melihat atau mengubah konfigurasi repositori (AWS CLI)

Untuk melihat konfigurasi repositori saat ini CodeArtifact, gunakan perintah. `describe-repository`

```
aws codeartifact describe-repository --domain my_domain --domain-owner 111122223333 --  
repository my_repo
```

Contoh output:

```
{  
  "repository": {  
    "name": "my_repo",  
    "administratorAccount": "123456789012",  
    "domainName": "my_domain",  
    "domainOwner": "111122223333",  
    "arn": "arn:aws:codeartifact:region-  
id:111122223333:repository/my_domain/my_repo"  
    "upstreams": [],  
    "externalConnections": []  
  }  
}
```

## Mengubah konfigurasi hulu repositori

Repositori hulu memungkinkan klien manajer paket mengakses paket yang berada dalam lebih dari satu repositori menggunakan titik akhir URL tunggal. Untuk menambah atau mengubah hubungan hulu repositori, gunakan perintah `update-repository`.

```
aws codeartifact update-repository --domain my_domain --domain-owner 111122223333 --  
repository my_repo \  
  --upstreams repositoryName=my-upstream-repo
```

Contoh output:

```
{  
  "repository": {  
    "name": "my_repo",  
    "administratorAccount": "123456789012",  
    "domainName": "my_domain",
```

```
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:region-
id:111122223333:repository/my_domain/my_repo"
    "upstreams": [
      {
        "repositoryName": "my-upstream-repo"
      }
    ],
    "externalConnections": []
  }
}
```

### Note

Untuk menambahkan repositori hulu, Anda harus memiliki izin untuk tindakan `AssociateWithDownstreamRepository` pada repositori hulu.

Untuk menghapus hubungan hulu repositori, gunakan daftar kosong sebagai argumen untuk opsi `--upstreams`.

```
aws codeartifact update-repository --domain my_domain --domain-owner 111122223333 --
repository my_repo --upstreams []
```

Contoh output:

```
{
  "repository": {
    "name": "my_repo",
    "administratorAccount": "123456789012",
    "domainName": "my_domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:region-
id:111122223333:repository/my_domain/my_repo"
    "upstreams": [],
    "externalConnections": []
  }
}
```

## Kebijakan repositori

CodeArtifact menggunakan izin berbasis sumber daya untuk mengontrol akses. Izin berbasis sumber daya memungkinkan Anda menentukan siapa yang memiliki akses ke repositori dan tindakan apa yang dapat mereka lakukan di situ. Secara default, hanya pemilik repositori yang memiliki akses ke repositori. Anda dapat menerapkan dokumen kebijakan yang memungkinkan prinsipal IAM lainnya mengakses repositori Anda.

Untuk informasi selengkapnya, lihat [Kebijakan Berbasis Sumber Daya](#) dan [Kebijakan Berbasis Identitas dan Kebijakan Berbasis Sumber Daya](#).

### Membuat kebijakan sumber daya untuk memberikan akses baca

Kebijakan sumber daya adalah file teks dalam format JSON. File harus menentukan prinsipal (aktor), satu atau beberapa tindakan, dan efek (Allow atau Deny). Sebagai contoh, sumber daya berikut kebijakan memberikan izin 123456789012 pada akun untuk mengunduh paket dari repositori.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "codeartifact:ReadFromRepository"
      ],
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      },
      "Resource": "*"
    }
  ]
}
```

Karena kebijakan dievaluasi hanya untuk operasi terhadap repositori yang dilampirkan, Anda tidak perlu menentukan sumber daya. Karena sumber daya tersirat, Anda dapat mengatur Resource ke \*. Agar manajer paket dapat mengunduh paket dari repositori ini, kebijakan domain untuk akses lintas akun juga perlu dibuat. Kebijakan domain harus memberikan setidaknya `codeartifact:GetAuthorizationToken` dan `sts:GetServiceBearerToken` izin kepada kepala sekolah. Untuk contoh kebijakan domain lengkap untuk memberikan akses lintas akun, lihat ini. [Contoh kebijakan domain](#)

**Note**

Tindakan `codeartifact:ReadFromRepository` hanya dapat digunakan pada sumber daya repositori. Anda tidak dapat menempatkan Amazon Resource Name (ARN) paket sebagai sumber daya dengan `codeartifact:ReadFromRepository` sebagai tindakan untuk mengizinkan akses baca ke subset paket dalam repositori. Sebuah prinsipal tertentu dapat membaca semua paket dalam repositori atau tidak satu pun.

Karena satu-satunya tindakan yang ditentukan dalam repositori adalah `ReadFromRepository`, pengguna dan peran dari akun `1234567890` dapat mengunduh paket dari repositori. Namun, tindakan lain tidak dapat dilakukan (misalnya, mencantumkan nama dan versi paket). Biasanya, Anda memberikan izin dalam kebijakan berikut selain `ReadFromRepository` karena pengguna yang mengunduh paket dari repositori perlu berinteraksi dengannya dengan cara lain juga.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "codeartifact:DescribePackageVersion",
        "codeartifact:DescribeRepository",
        "codeartifact:GetPackageVersionReadme",
        "codeartifact:GetRepositoryEndpoint",
        "codeartifact:ListPackages",
        "codeartifact:ListPackageVersions",
        "codeartifact:ListPackageVersionAssets",
        "codeartifact:ListPackageVersionDependencies",
        "codeartifact:ReadFromRepository"
      ],
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      },
      "Resource": "*"
    }
  ]
}
```

## Menetapkan kebijakan

Setelah Anda membuat dokumen kebijakan, gunakan perintah `put-repository-permissions-policy` untuk melampirkannya ke repositori:

```
aws codeartifact put-repository-permissions-policy --domain my_domain --domain-owner 111122223333 \
    --repository my_repo --policy-document file:///PATH/TO/policy.json
```

Ketika Anda memanggil `put-repository-permissions-policy`, kebijakan sumber daya pada repositori diabaikan ketika mengevaluasi izin. Hal ini memastikan bahwa pemilik domain tidak dapat mengunci diri dari repositori, yang akan mencegah mereka memperbarui kebijakan sumber daya.

### Note

Anda tidak dapat memberikan izin ke AWS akun lain untuk memperbarui kebijakan sumber daya di repositori menggunakan kebijakan sumber daya, karena kebijakan sumber daya diabaikan saat memanggil `put-repository-permissions-policy`.

Contoh output:

```
{
  "policy": {
    "resourceArn": "arn:aws:codeartifact:region-id:111122223333:repository/my_domain/my_repo",
    "document": "{ ...policy document content...}",
    "revision": "MQLyyTQRASRU3HB58gBtSDHXG7Q3hvxxxxxxxxx="
  }
}
```

Output dari perintah berisi Amazon Resource Name (ARN) dari sumber daya repositori, isi lengkap dokumen kebijakan, dan pengidentifikasi revisi. Anda dapat meneruskan pengidentifikasi revisi ke `put-repository-permissions-policy` menggunakan opsi `--policy-revision`. Hal ini memastikan bahwa revisi dokumen yang sedang ditimpa, dan bukan versi yang lebih baru yang ditetapkan oleh penulis lain.



## Membaca kebijakan

Gunakan perintah `get-repository-permissions-policy` untuk membaca versi dokumen kebijakan yang ada. Untuk memformat output agar dapat dibaca, gunakan `--output` dan `--query policy.document` bersama-sama dengan modul `json.tool` Python.

```
aws codeartifact get-repository-permissions-policy --domain my_domain --domain-owner 111122223333 \  
    --repository my_repo --output text --query policy.document | python -m  
    json.tool
```

Contoh output:

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Principal": {  
        "AWS": "arn:aws:iam::123456789012:root"  
      },  
      "Action": [  
        "codeartifact:DescribePackageVersion",  
        "codeartifact:DescribeRepository",  
        "codeartifact:GetPackageVersionReadme",  
        "codeartifact:GetRepositoryEndpoint",  
        "codeartifact:ListPackages",  
        "codeartifact:ListPackageVersions",  
        "codeartifact:ListPackageVersionAssets",  
        "codeartifact:ListPackageVersionDependencies",  
        "codeartifact:ReadFromRepository"  
      ],  
      "Resource": "*"   
    }  
  ]  
}
```

## Menghapus kebijakan

Gunakan perintah `delete-repository-permissions-policy` untuk menghapus kebijakan dari repositori.

```
aws codeartifact delete-repository-permissions-policy --domain my_domain --domain-owner 111122223333 \  
    --repository my_repo
```

Format output sama dengan perintah `get-repository-permissions-policy`.

## Memberikan akses baca ke prinsipal

Bila Anda menetapkan pengguna root akun sebagai prinsipal dalam dokumen kebijakan, Anda memberikan akses ke semua pengguna dan peran dalam akun tersebut. Untuk membatasi akses ke pengguna atau peran yang dipilih, gunakan ARN-nya di bagian `Principal` dari kebijakan. Misalnya, gunakan berikut ini untuk memberikan akses baca ke pengguna IAM bob dalam akun 123456789012.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Action": [  
        "codeartifact:ReadFromRepository"  
      ],  
      "Effect": "Allow",  
      "Principal": {  
        "AWS": "arn:aws:iam::123456789012:user/bob"  
      },  
      "Resource": "*"   
    }  
  ]  
}
```

## Memberikan akses tulis ke paket

Tindakan `codeartifact:PublishPackageVersion` digunakan untuk mengontrol izin untuk memublikasikan versi baru dari sebuah paket. Sumber daya yang digunakan dengan tindakan ini harus berupa paket. Format CodeArtifact paket ARN adalah sebagai berikut.

```
arn:aws:codeartifact:region-id:111122223333:package/my_domain/my_repo/package-format/package-namespace/package-name
```

Contoh berikut menunjukkan ARN untuk paket npm dengan lingkup @parity dan nama ui dalam repositori example-repo dalam domain my\_domain.

```
arn:aws:codeartifact:region-id:111122223333:package/my_domain/example-repo/npm/parity/ui
```

ARN untuk paket npm tanpa lingkup memiliki string kosong untuk bidang namespace. Misalnya, berikut ini adalah ARN untuk paket tanpa ruang lingkup dan dengan nama react dalam repositori example-repo dalam domain my\_domain.

```
arn:aws:codeartifact:region-id:111122223333:package/my_domain/example-repo/npm//react
```

Kebijakan berikut memberikan akun 123456789012 izin untuk memublikasikan versi @parity/ui dalam repositori example-repo.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "codeartifact:PublishPackageVersion"
      ],
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      },
      "Resource": "arn:aws:codeartifact:region-id:111122223333:package/my_domain/example-repo/npm/parity/ui"
    }
  ]
}
```

### Important

Untuk memberikan izin untuk menerbitkan versi Maven dan NuGet paket, tambahkan izin berikut sebagai tambahan. `codeartifact:PublishPackageVersion`

1. NuGet: `codeartifact:ReadFromRepository` dan tentukan sumber daya repositori
2. Maven: `codeartifact:PutPackageMetadata`

Karena kebijakan ini menentukan domain dan repositori sebagai bagian dari sumber daya, kebijakan ini hanya mengizinkan publikasi bila dilampirkan ke repositori tersebut.

## Memberikan akses tulis ke repositori

Anda dapat menggunakan wildcard untuk memberikan izin tulis untuk semua paket dalam repositori. Sebagai contoh, gunakan kebijakan berikut untuk memberikan izin akun untuk menulis ke semua paket di repositori `example-repo`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "codeartifact:PublishPackageVersion"
      ],
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      },
      "Resource": "arn:aws:codeartifact:region-id:111122223333:package/my_domain/my_repo/*"
    }
  ]
}
```

## Menandai repositori di CodeArtifact

Tanda adalah pasangan kunci-nilai yang terkait dengan sumber daya AWS. Anda dapat menerapkan tag ke repositori Anda di CodeArtifact Untuk informasi tentang penandaan CodeArtifact sumber daya, kasus penggunaan, kunci tag dan batasan nilai, serta jenis sumber daya yang didukung, lihat.

[Penandaan pada sumber daya](#)

Anda dapat menggunakan CLI untuk menentukan tanda saat membuat repositori. Anda dapat menggunakan konsol atau CLI untuk menambah atau menghapus tanda, dan memperbarui nilai tanda dalam repositori. Anda dapat menambahkan hingga 50 tanda ke setiap repositori.

Topik

- [Menandai repositori \(CLI\)](#)

- [Menandai repositori \(konsol\)](#)

## Menandai repositori (CLI)

Anda dapat menggunakan CLI untuk mengelola tanda repositori.

Topik

- [Menambahkan tanda ke repositori \(CLI\)](#)
- [Melihat tanda untuk repositori \(CLI\)](#)
- [Mengedit tanda untuk repositori \(CLI\)](#)
- [Menghapus tanda dari repositori \(CLI\)](#)

## Menambahkan tanda ke repositori (CLI)

Anda dapat menggunakan konsol atau AWS CLI untuk menandai repositori.

Untuk menambahkan tag ke repositori saat Anda membuatnya, lihat [Buatlah sebuah repositori](#).

Dalam langkah-langkah ini, kami menganggap bahwa Anda telah menginstal versi terbaru dari AWS CLI atau diperbarui ke versi terkini. Untuk informasi lebih lanjut, lihat [Menginstal AWS Command Line Interface](#).

Pada terminal atau baris perintah, jalankan perintah `tag-resource`, yang menentukan Amazon Resource Name (ARN) repositori tempat Anda ingin menambahkan tanda dan kunci dan nilai tanda yang ingin Anda tambahkan.

### Note

Untuk mendapatkan ARN repositori, jalankan perintah `describe-repository`:

```
aws codeartifact describe-repository --domain my_domain --repository my_repo --  
query repository.arn
```

Anda dapat menambahkan lebih dari satu tanda ke repositori. Misalnya, untuk menandai repositori bernama *my\_repo* di domain bernama *my\_domain* dengan dua tanda, sebuah kunci tanda bernama *key1* dengan nilai tanda *value1*, dan kunci tanda bernama *key2* dengan nilai tanda *value2*:

```
aws codeartifact tag-resource --resource-arn arn:aws:codeartifact:us-west-2:111122223333:repository/my_domain/my_repo --tags key=key1,value=value1 key=key2,value=value2
```

Jika berhasil, perintah ini tidak memiliki output.

## Melihat tanda untuk repositori (CLI)

Ikuti langkah-langkah ini untuk menggunakan AWS CLI untuk melihat AWS tag untuk repositori. Jika tidak ada tanda yang telah ditambahkan, daftar yang ditampilkan kosong.

Pada terminal atau baris perintah, jalankan perintah `list-tags-for-resource`.

### Note

Untuk mendapatkan ARN repositori, jalankan perintah `describe-repository`:

```
aws codeartifact describe-repository --domain my_domain --repository my_repo --query repository.arn
```

Misalnya, untuk melihat daftar kunci tanda dan nilai tanda untuk repositori bernama `my_repo` di domain bernama `my_domain` dengan `arn:aws:codeartifact:us-west-2:111122223333:repository/my_domain/my_repo` nilai ARN:

```
aws codeartifact list-tags-for-resource --resource-arn arn:aws:codeartifact:us-west-2:111122223333:repository/my_domain/my_repo
```

Jika berhasil, perintah ini mengembalikan informasi yang serupa dengan berikut ini:

```
{
  "tags": {
    "key1": "value1",
    "key2": "value2"
  }
}
```

## Mengedit tanda untuk repositori (CLI)

Ikuti langkah-langkah ini untuk menggunakan AWS CLI untuk mengedit tag untuk repositori. Anda dapat mengubah nilai untuk kunci yang ada atau menambahkan kunci lain.

Pada terminal atau baris perintah, jalankan perintah `tag-resource`, yang menentukan ARN repositori tempat Anda ingin memperbarui tanda dan menentukan kunci tanda dan nilai tanda.

### Note

Untuk mendapatkan ARN repositori, jalankan perintah `describe-repository`:

```
aws codeartifact describe-repository --domain my_domain --repository my_repo --query repository.arn
```

```
aws codeartifact tag-resource --resource-arn arn:aws:codeartifact:us-west-2:111122223333:repository/my_domain/my_repo --tags key=key1,value=newvalue1
```

Jika berhasil, perintah ini tidak memiliki output.

## Menghapus tanda dari repositori (CLI)

Ikuti langkah-langkah ini untuk menggunakan AWS CLI untuk menghapus tag dari repositori.

### Note

Jika Anda menghapus repositori, semua asosiasi tanda dihapus dari repositori yang dihapus. Anda tidak perlu menghapus tanda sebelum menghapus repositori.

Pada terminal atau baris perintah, jalankan perintah `untag-resource`, yang menentukan ARN repositori tempat Anda ingin menghapus tanda dan kunci tanda dari tanda yang ingin Anda hapus.

### Note

Untuk mendapatkan ARN repositori, jalankan perintah `describe-repository`:

```
aws codeartifact describe-repository --domain my_domain --repository my_repo --query repository.arn
```

Misalnya, untuk menghapus beberapa tanda pada repositori bernama *my\_repo* di domain bernama *my\_domain* dengan kunci tanda *key1* dan *key2*:

```
aws codeartifact untag-resource --resource-arn arn:aws:codeartifact:us-west-2:111122223333:repository/my_domain/my_repo --tag-keys key1 key2
```

Jika berhasil, perintah ini tidak memiliki output. Setelah menghapus tanda, Anda dapat melihat tanda yang tersisa pada repositori menggunakan perintah `list-tags-for-resource`.

## Menandai repositori (konsol)

Anda dapat menggunakan konsol atau CLI untuk menandai sumber daya.

### Topik

- [Menambahkan tanda ke repositori \(konsol\)](#)
- [Melihat tanda untuk repositori \(konsol\)](#)
- [Mengedit tanda untuk repositori \(konsol\)](#)
- [Menghapus tanda dari repositori \(konsol\)](#)

## Menambahkan tanda ke repositori (konsol)

Anda dapat menggunakan konsol untuk menambahkan tanda ke repositori yang ada.

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Pada halaman Repositories (Repositori), pilih repositori yang ingin Anda tambahkan tanda.
3. Perluas bagian Details (Detail).
4. Di bawah Repository tags (Tanda repositori), jika tidak ada tanda pada repositori, pilih Add repository tags (Tambahkan tanda repositori). Jika ada tanda pada repositori, pilih View and edit repository tags (Lihat dan edit tanda repositori).
5. Pilih Add new tag (Tambahkan tanda baru).



6. Di bidang Key (Kunci) dan Value (Nilai), masukkan teks untuk setiap tanda yang ingin Anda tambahkan. (Bidang Value (Nilai) bersifat opsional.) Contohnya, dalam Key (Kunci), masukkan **Name**. Dalam Value (Nilai), masukkan **Test**.

Developer Tools > CodeArtifact > Repositories > reponame > Edit repository

## Edit reponame [Info](#)

### Repository

Repository description - *optional*

1000 character limit

### Tags

Tags - *optional*

Key Value - *optional*

<input type="text" value="Name"/>	<input type="text" value="Test"/>	<input type="button" value="Remove"/>
-----------------------------------	-----------------------------------	---------------------------------------

You can add 49 more tags.

▶ **AWS reserved tags**  
Resource tags added by other AWS services. These tags cannot be modified.

### Upstream repositories - *optional*

Repository name

1. <input type="checkbox"/>	reponame
-----------------------------	----------

[How to use this input ?](#)

7. (Opsional) Pilih Add tag (Tambahkan tanda) untuk menambahkan lebih banyak baris dan memasukkan lebih banyak tanda.
8. Pilih Update repository (Perbarui repositori).

## Melihat tanda untuk repositori (konsol)

Anda dapat menggunakan konsol untuk mencantumkan tanda untuk repositori yang ada.

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Pada halaman Repositories (Repositori), pilih repositori tempat Anda ingin melihat tanda.
3. Perluas bagian Details (Detail).
4. Di bagian Repository tags (Tag repositori), pilih View and edit repository tags (Lihat dan edit tanda repositori).

### Note

Jika tidak ada tanda yang ditambahkan ke repositori ini, konsol akan membaca Add repository tags (Tambahkan tanda repositori).

## Mengedit tanda untuk repositori (konsol)

Anda dapat menggunakan konsol untuk mengedit tanda yang telah ditambahkan ke repositori.

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Pada halaman Repositories (Repositori), pilih repositori tempat Anda ingin memperbarui tanda.
3. Perluas bagian Details (Detail).
4. Di bagian Repository tags (Tag repositori), pilih View and edit repository tags (Lihat dan edit tanda repositori).

### Note

Jika tidak ada tanda yang ditambahkan ke repositori ini, konsol akan membaca Add repository tags (Tambahkan tanda repositori).

5. Di bidang Key (Kunci) dan Value (Nilai), perbarui nilai di setiap bidang yang diperlukan. Misalnya, untuk kunci **Name**, di Value (Nilai), ubah **Test** menjadi **Prod**.

6. Pilih Update repository (Perbarui repositori).

## Menghapus tanda dari repositori (konsol)

Anda dapat menggunakan konsol untuk menghapus tanda dari repositori.

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Pada halaman Repositories (Repositori), pilih repositori tempat Anda ingin menghapus tanda.
3. Perluas bagian Details (Detail).
4. Di bagian Repository tags (Tag repositori), pilih View and edit repository tags (Lihat dan edit tanda repositori).

### Note

Jika tidak ada tanda yang ditambahkan ke repositori ini, konsol akan membaca Add repository tags (Tambahkan tanda repositori).

5. Di samping kunci dan nilai untuk setiap tanda yang ingin Anda hapus, pilih Remove (Hapus).
6. Pilih Update repository (Perbarui repositori).

# Bekerja dengan repositori hulu di CodeArtifact

Sebuah repositori dapat menggunakan AWS CodeArtifact repositori lain sebagai repositori hulu. Hal ini memungkinkan klien manajer paket mengakses paket yang terdapat dalam lebih dari satu repositori menggunakan titik akhir repositori tunggal.

Anda dapat menambahkan satu atau beberapa repositori hulu ke AWS CodeArtifact repositori menggunakan AWS Management Console, AWS CLI, atau SDK. Untuk mengaitkan repositori dengan repositori hulu, Anda harus memiliki izin untuk tindakan `AssociateWithDownstreamRepository` pada repositori hulu. Untuk informasi selengkapnya, lihat [Membuat repositori dengan repositori hulu](#) dan [Menambahkan atau menghapus repositori hulu](#).

Jika repositori hulu memiliki koneksi eksternal ke repositori publik, repositori yang berada di hilirnya dapat menarik paket dari repositori publik tersebut. Misalnya, repositori `my_repo` memiliki repositori hulu bernama `upstream`, dan `upstream` memiliki koneksi eksternal ke repositori npm publik. Dalam hal ini, manajer paket yang terhubung ke `my_repo` dapat menarik paket dari repositori publik npm. Untuk informasi selengkapnya tentang meminta paket dari repositori hulu atau koneksi eksternal, lihat [Meminta versi paket dengan repositori hulu](#) atau [Meminta paket dari koneksi eksternal](#).

## Topik

- [Apa perbedaan antara repositori hulu dan koneksi eksternal?](#)
- [Menambahkan atau menghapus repositori hulu](#)
- [Connect CodeArtifact repositori ke repositori publik](#)
- [Meminta versi paket dengan repositori hulu](#)
- [Meminta paket dari koneksi eksternal](#)
- [Urutan prioritas repositori hulu](#)
- [Perilaku API dengan repositori hulu](#)

## Apa perbedaan antara repositori hulu dan koneksi eksternal?

Di CodeArtifact, repositori hulu dan koneksi eksternal berperilaku sebagian besar sama, tetapi ada beberapa perbedaan penting.

1. Anda dapat menambahkan hingga 10 repositori hulu ke CodeArtifact repositori. Anda hanya dapat menambahkan satu koneksi eksternal.

2. Ada panggilan API terpisah untuk menambahkan repositori hulu atau koneksi eksternal.
3. Perilaku retensi paket sedikit berbeda, karena paket yang diminta dari repositori upstream disimpan di repositori tersebut. Untuk informasi selengkapnya, lihat [Retensi paket dalam repositori menengah](#).

## Menambahkan atau menghapus repositori hulu

Ikuti langkah-langkah di bagian berikut untuk menambah atau menghapus repositori hulu ke atau dari CodeArtifact repositori. Untuk informasi lebih lanjut tentang repositori hulu, lihat [Bekerja dengan repositori hulu di CodeArtifact](#).

Panduan ini berisi informasi tentang mengonfigurasi CodeArtifact repositori lainnya sebagai repositori hulu. Untuk informasi tentang cara mengonfigurasi koneksi eksternal ke repositori publik seperti npmjs.com, Galeri Nuget, Maven Central, atau PyPI, lihat [Menambahkan koneksi eksternal](#).

## Menambahkan atau menghapus repositori hulu (konsol)

Lakukan langkah-langkah dalam prosedur berikut untuk menambahkan repositori sebagai repositori hulu menggunakan CodeArtifact konsol. Untuk informasi tentang menambahkan repositori hulu dengan AWS CLI, lihat [Menambahkan atau menghapus repositori hulu \(AWS CLI\)](#).

Untuk menambahkan repositori hulu menggunakan CodeArtifact konsol

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Di panel navigasi, pilih Domain, lalu pilih nama domain yang berisi repositori Anda.
3. Pilih nama repositori Anda.
4. Pilih Edit.
5. Di Repositori hulu, pilih Kaitkan repositori hulu dan tambahkan repositori yang ingin Anda tambahkan sebagai repositori hulu. Anda hanya dapat menambahkan repositori di domain yang sama dengan repositori hulu.
6. Pilih Update repository (Perbarui repositori).

Untuk menghapus repositori hulu menggunakan CodeArtifact konsol

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Di panel navigasi, pilih Domain, lalu pilih nama domain yang berisi repositori Anda.

3. Pilih nama repositori Anda.
4. Pilih Edit.
5. Di repositori Upstream, temukan entri daftar repositori hulu yang ingin Anda hapus dan pilih Disassociate.

 Important

Setelah Anda menghapus repositori hulu dari CodeArtifact repositori, manajer paket tidak akan memiliki akses ke paket di repositori hulu atau repositori upstream.

6. Pilih Update repository (Perbarui repositori).

## Menambahkan atau menghapus repositori hulu (AWS CLI)

Anda dapat menambahkan atau menghapus CodeArtifact repositori hulu menggunakan AWS Command Line Interface (AWS CLI). Untuk melakukannya, gunakan perintah `update-repository`, dan tentukan repositori hulu menggunakan parameter `--upstreams`.

Anda hanya dapat menambahkan repositori di domain yang sama dengan repositori hulu.

Untuk menambahkan repositori hulu (AWS CLI)

1. Jika belum, ikuti langkah-langkah [Mengatur dengan AWS CodeArtifact](#) untuk mengatur dan mengkonfigurasi AWS CLI dengan CodeArtifact.
2. Gunakan `aws codeartifact update-repository` perintah dengan `--upstreams` bendera untuk menambahkan repositori upstream.

 Note

Memanggil `update-repository` perintah menggantikan repositori hulu dikonfigurasi yang ada dengan daftar repositori yang disediakan dengan `--upstreams` bendera. Jika Anda ingin menambahkan repositori upstream dan menyimpan repositori yang sudah ada, Anda harus menyertakan repositori upstream yang ada dalam panggilan.

Contoh perintah berikut menambahkan dua repositori hulu ke repositori bernama `my_repo` yang ada di domain bernama `my_domain`. Urutan repositori hulu dalam `--upstreams` parameter

menentukan prioritas pencarian ketika CodeArtifact meminta paket dari `my_repo` repositori. Untuk informasi selengkapnya, lihat [Urutan prioritas repositori hulu](#).

```
aws codeartifact update-repository --repository my_repo --domain my_domain --  
domain-owner 111122223333 \  
--upstreams repositoryName=upstream-1 repositoryName=upstream-2
```

Output berisi repositori hulu, sebagai berikut.

```
{  
  "repository": {  
    "name": "my_repo",  
    "administratorAccount": "123456789012",  
    "domainName": "my_domain",  
    "domainOwner": "111122223333",  
    "arn": "arn:aws:codeartifact:us-east-2:111122223333:repository/my_domain/my_repo",  
    "upstreams": [  
      {  
        "repositoryName": "upstream-1"  
      },  
      {  
        "repositoryName": "upstream-2"  
      }  
    ],  
    "externalConnections": []  
  }  
}
```

Untuk menghapus repositori upstream (AWS CLI)

1. Jika belum, ikuti langkah-langkah [Mengatur dengan AWS CodeArtifact](#) untuk mengatur dan mengkonfigurasi AWS CLI dengan CodeArtifact.
2. Untuk menghapus repositori hulu dari CodeArtifact repositori, gunakan `update-repository` perintah dengan `--upstreams` bendera. Daftar repositori yang disediakan untuk perintah akan menjadi set baru repositori hulu untuk CodeArtifact repositori. Sertakan repositori upstream yang ada yang ingin Anda simpan, dan hilangkan repositori upstream yang ingin Anda hapus.

Untuk menghapus semua repositori hulu dari repositori, gunakan perintah `update-repository` dan sertakan `--upstreams` tanpa argumen. Tindakan berikut menghapus

repositori hulu dari repositori bernama `my_repo` yang berada dalam domain bernama `my_domain`.

```
aws codeartifact update-repository --repository my_repo --domain my_domain --  
domain-owner 111122223333 --upstreams
```

Output menunjukkan bahwa daftar `upstreams` kosong.

```
{  
  "repository": {  
    "name": "my_repo",  
    "administratorAccount": "123456789012",  
    "domainName": "my_domain",  
    "domainOwner": "111122223333",  
    "arn": "arn:aws:codeartifact:us-  
east-2:111122223333:repository/my_domain/my_repo",  
    "upstreams": [],  
    "externalConnections": []  
  }  
}
```

## Connect CodeArtifact repositori ke repositori publik

Anda dapat menambahkan koneksi eksternal antara CodeArtifact repositori dan repositori eksternal, publik seperti <https://npmjs.com> atau [Repositori Maven Central](#). Kemudian, ketika Anda meminta paket dari CodeArtifact repositori yang belum ada dalam repositori, paket dapat diambil dari koneksi eksternal. Hal ini memungkinkan untuk menggunakan dependensi sumber terbuka yang digunakan oleh aplikasi Anda.

Di CodeArtifact, cara yang dimaksudkan untuk menggunakan koneksi eksternal adalah memiliki satu repositori per domain dengan koneksi eksternal ke repositori publik tertentu. Misalnya, jika Anda ingin terhubung ke [npmjs.com](https://npmjs.com), konfigurasi satu repositori di domain Anda dengan koneksi eksternal ke [npmjs.com](https://npmjs.com) dan konfigurasi semua repositori lain dengan upstream ke sana. Dengan cara ini, semua repositori dapat menggunakan paket yang telah diambil dari [npmjs.com](https://npmjs.com), daripada mengambil dan menyimpannya lagi.

Topik

- [Connect ke repositori](#)



- [Connect ke repositori](#)
- [Repositori koneksi eksternal yang didukung](#)
- [Menghapus koneksi eksternal \(CLI\)](#)

## Connect ke repositori

Ketika Anda menggunakan konsol untuk menambahkan koneksi ke repositori eksternal, berikut ini akan terjadi:

1. Sebuah `-store` repositori untuk repositori eksternal akan dibuat di CodeArtifact domain Anda jika salah satu tidak ada sudah. `-store`Repository ini berperilaku sebagai repositori perantara antara repositori Anda dan repositori eksternal dan memungkinkan Anda untuk terhubung ke lebih dari satu repositori eksternal.
2. `-store`Repository yang sesuai ditambahkan sebagai hulu ke repositori Anda.

Daftar berikut berisi setiap `-store` repositori CodeArtifact dan masing-masing repositori eksternal mereka terhubung ke.

1. `commonsware-store`terhubung ke CommonsWare Android Repository.
2. `google-android-store`terhubung ke Google Android.
3. `gradle-plugins-store`terhubung ke plugin Gradle.
4. `maven-central-store`terhubung ke Maven Central Repository.
5. `clojars-store`terhubung ke Clojars Repository.
6. `npm-store`terhubung ke npmjs.com.
7. `nuget-store`terhubung ke nuget.org.
8. `pypi-store`terhubung ke Python Packaging Authority.

Untuk menyambung ke repositori (konsol)

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Di panel navigasi, pilih Domain, lalu pilih nama domain yang berisi repositori Anda.
3. Pilih nama repositori Anda.
4. Pilih Edit.

5. Di repositori Upstream, pilih Associate upstream repository dan tambahkan-store repositori yang sesuai yang terhubung sebagai upstream.
6. Pilih Update repository (Perbarui repositori).

Setelah-store repositori ditambahkan sebagai repositori hulu, manajer paket yang terhubung ke CodeArtifact repositori Anda dapat mengambil paket dari repositori eksternal masing-masing.

## Connect ke repositori

Anda dapat menggunakan AWS CLI untuk menghubungkan CodeArtifact repositori Anda ke repositori eksternal dengan menambahkan koneksi eksternal langsung ke repositori. Ini akan memungkinkan pengguna terhubung ke CodeArtifact repositori, atau repositori hilir, untuk mengambil paket dari repositori eksternal yang dikonfigurasi. Setiap CodeArtifact repositori hanya dapat memiliki satu koneksi eksternal.

Disarankan untuk memiliki satu repositori per domain dengan koneksi eksternal ke repositori publik yang diberikan. Untuk menghubungkan repositori lain ke repositori publik, tambahkan repositori dengan koneksi eksternal sebagai upstream untuk mereka. Jika Anda atau orang lain di domain Anda telah mengonfigurasi koneksi eksternal di konsol, domain Anda kemungkinan sudah memiliki-store repositori dengan koneksi eksternal ke repositori publik yang ingin Anda sambungkan. Untuk informasi lebih lanjut tentang-store repositori dan menghubungkan dengan konsol, lihat [Connect ke repositori](#).

Untuk menambahkan koneksi eksternal ke CodeArtifact repositori (CLI)

- Gunakan `associate-external-connection` untuk menambahkan koneksi eksternal. Contoh berikut menghubungkan repositori ke registri publik npm, npmjs.com. Untuk daftar repositori eksternal yang didukung, lihat [Repositori koneksi eksternal yang didukung](#).

```
aws codeartifact associate-external-connection --external-connection public:npmjs \  
--domain my_domain --domain-owner 111122223333 --repository my_repo
```

Contoh keluaran:

```
{  
  "repository": {  
    "name": my_repo  
    "administratorAccount": "123456789012",  
    "domainName": "my_domain",
```

```

    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:us-
west-2:111122223333:repository/my_domain/my_repo",
    "description": "A description of my_repo",
    "upstreams": [],
    "externalConnections": [
      {
        "externalConnectionName": "public:npmjs",
        "packageFormat": "npm",
        "status": "AVAILABLE"
      }
    ]
  }
}

```

Setelah menambahkan koneksi eksternal, lihat [Meminta paket dari koneksi eksternal](#) informasi tentang meminta paket dari repositori eksternal dengan koneksi eksternal.

## Repositori koneksi eksternal yang didukung

CodeArtifact mendukung koneksi eksternal ke repositori publik berikut. Untuk menggunakan CodeArtifact CLI guna menentukan sambungan eksternal, gunakan nilai di kolom Nama untuk `--external-connection` parameter ketika Anda menjalankan `associate-external-connection` perintah.

Jenis repositori	Deskripsi	Nama
npm	registri publik npm	public:npmjs
Python	Indeks Paket Python	public:pypi
Maven	Maven Central	public:maven-central
Maven	Repositori Google Android	public:maven-googleandroid
Maven	Repositori plugin Gradle	public:maven-gradleplugins

Jenis repositori	Deskripsi	Nama
Maven	CommonsWare Repositori	public:maven-commonsware
Maven	Repositori	public:maven-clojars
NuGet	NuGet Galeri	public:nuget-org

## Menghapus koneksi eksternal (CLI)

Untuk menghapus koneksi eksternal yang ditambahkan dengan menggunakan `associate-external-connection` perintah di AWS CLI, gunakan `disassociate-external-connection`.

```
aws codeartifact disassociate-external-connection --external-connection public:npmjs \
  --domain my_domain --domain-owner 111122223333 --repository my_repo
```

Contoh keluaran:

```
{
  "repository": {
    "name": my_repo,
    "administratorAccount": "123456789012",
    "domainName": "my_domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:repository/my_domain/my_repo",
    "description": "A description of my_repo",
    "upstreams": [],
    "externalConnections": []
  }
}
```

## Meminta versi paket dengan repositori hulu

Ketika klien (misalnya, npm) meminta versi paket dari CodeArtifact repositori bernama `my_repo` yang memiliki beberapa repositori hulu, hal berikut ini dapat terjadi:

- Jika `my_repo` berisi versi paket yang diminta, akan dikembalikan ke klien.
- Jika `my_repo` tidak berisi versi paket yang diminta CodeArtifact, cari `my_repo` repositori hulu. Jika versi paket ditemukan, referensi untuk itu akan disalin ke `my_repo`, dan versi paket dikembalikan ke klien.
- Jika `my_repo` atau repositori hulu tidak berisi versi paket, respons HTTP 404 Not Found dikembalikan ke klien.

Ketika Anda menambahkan repositori hulu menggunakan perintah `create-repository` atau `update-repository`, urutan penerusan ke parameter `--upstreams` menentukan prioritas ketika versi paket diminta. Tentukan repositori hulu `--upstreams` dalam urutan yang CodeArtifact ingin Anda gunakan ketika versi paket diminta. Untuk informasi selengkapnya, lihat [Urutan prioritas repositori hulu](#).

Jumlah maksimum repositori hulu langsung yang diizinkan untuk satu repositori adalah 10. Karena repositori hulu langsung juga dapat memiliki repositori hulu langsung mereka sendiri, CodeArtifact dapat mencari lebih dari 10 repositori untuk versi paket. Jumlah maksimum repositori yang CodeArtifact dilihat ketika versi paket diminta adalah 25.

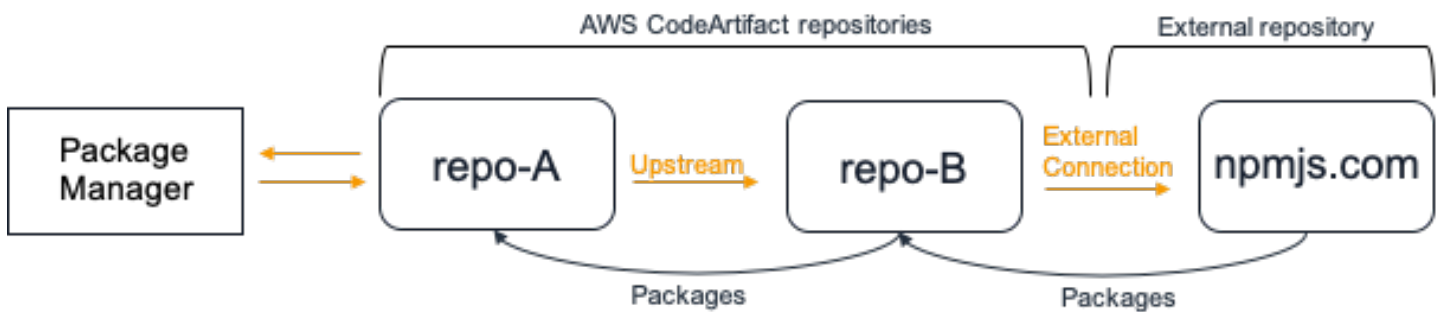
## Retensi paket dari repositori hulu

Jika versi paket yang diminta ditemukan di repositori hulu, referensi untuk itu dipertahankan dan selalu tersedia dari repositori hilir. Versi paket yang dipertahankan tidak terpengaruh oleh salah satu dari berikut ini:

- Menghapus repositori hulu.
- Memutuskan koneksi repositori hulu dari repositori hilir.
- Menghapus versi paket dari repositori hulu.
- Mengedit versi paket di repositori hulu (misalnya, dengan menambahkan aset baru ke dalamnya).

## Mengambil paket melalui hubungan hulu

Jika CodeArtifact repositori memiliki hubungan hulu dengan repositori yang memiliki koneksi eksternal, permintaan untuk paket yang tidak berada di repositori hulu disalin dari repositori eksternal. Misalnya, pertimbangkan konfigurasi berikut ini: repositori bernama `repo-A` memiliki repositori hulu bernama `repo-B`. `repo-B` memiliki koneksi eksternal ke <https://npmjs.com>.



Jika npm dikonfigurasi untuk menggunakan repositori `repo-A`, menjalankan `npm install` akan memicu penyalinan paket dari <https://npmjs.com> ke `repo-B`. Versi yang dipasang juga ditarik ke dalam `repo-A`. Contoh berikut menginstal `lodash`.

```

$ npm config get registry
https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/npm/my-downstream-repo/
$ npm install lodash
+ lodash@4.17.20
added 1 package from 2 contributors in 6.933s
  
```

Setelah menjalankan `npm install`, `repo-A` hanya berisi versi terbaru (`lodash 4.17.20`) karena itu adalah versi yang diambil oleh npm dari `repo-A`.

```

aws codeartifact list-package-versions --repository repo-A --domain my_domain \
  --domain-owner 111122223333 --format npm --package lodash
  
```

Contoh keluaran:

```

{
  "package": "lodash",
  "format": "npm",
  "versions": [
    {
      "version": "4.17.15",
      "revision": "REVISION-1-SAMPLE-6C81EFF7DA55CC",
      "status": "Published"
    }
  ]
}
  
```

Karena repo-B memiliki koneksi eksternal ke <https://npmjs.com>, semua versi paket yang diimpor dari <https://npmjs.com> disimpan di repo-B. Versi paket ini bisa saja diambil oleh repositori hilir mana pun dengan hubungan hulu ke repo-B.

Isi repo-B menyediakan cara untuk melihat semua paket dan versi paket yang diimpor dari <https://npmjs.com> dari waktu ke waktu. Misalnya, untuk melihat semua versi paket `lodash` yang diimpor dari waktu ke waktu, Anda dapat menggunakan `list-package-versions`, sebagai berikut.

```
aws codeartifact list-package-versions --repository repo-B --domain my_domain \
  --domain-owner 111122223333 --format npm --package lodash --max-results 5
```

Contoh keluaran:

```
{
  "package": "lodash",
  "format": "npm",
  "versions": [
    {
      "version": "0.10.0",
      "revision": "REVISION-1-SAMPLE-6C81EFF7DA55CC",
      "status": "Published"
    },
    {
      "version": "0.2.2",
      "revision": "REVISION-2-SAMPLE-6C81EFF7DA55CC",
      "status": "Published"
    },
    {
      "version": "0.2.0",
      "revision": "REVISION-3-SAMPLE-6C81EFF7DA55CC",
      "status": "Published"
    },
    {
      "version": "0.2.1",
      "revision": "REVISION-4-SAMPLE-6C81EFF7DA55CC",
      "status": "Published"
    },
    {
      "version": "0.1.0",
      "revision": "REVISION-5-SAMPLE-6C81EFF7DA55CC",
      "status": "Published"
    }
  ]
}
```

```

  ],
  "nextToken": "eyJsaXN0UGFja2FnZVZlcnNpb25zVG9rZW4iOiIwLjIuMiJ9"
}

```

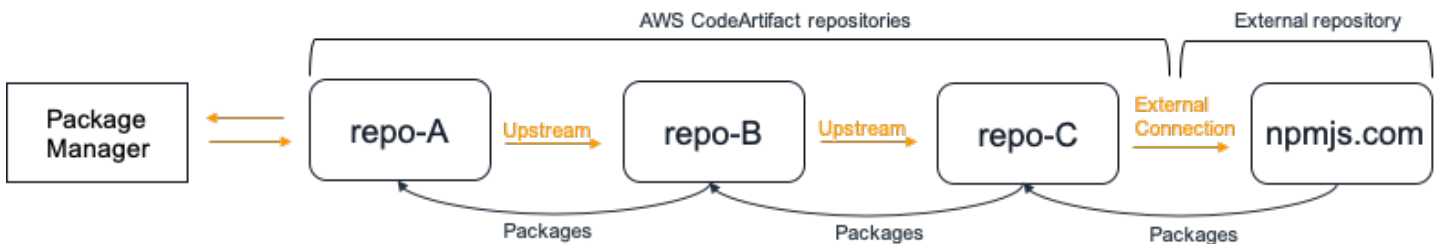
## Retensi paket dalam repositori menengah

CodeArtifact memungkinkan rantai repositori hulu. Misalnya, `repo-A` dapat memiliki `repo-B` sebagai hulu dan `repo-B` dapat memiliki `repo-C` sebagai hulu. Konfigurasi ini membuat versi paket di `repo-B` dan `repo-C` tersedia dari `repo-A`.



Ketika manajer paket terhubung ke repositori `repo-A` dan mengambil versi paket dari repositori `repo-C`, versi paket tidak akan disimpan dalam repositori `repo-B`. Versi paket hanya akan disimpan di repositori paling hilir, dalam contoh ini, `repo-A`. Versi paket tidak akan disimpan dalam repositori menengah. Hal ini juga berlaku untuk rantai yang lebih panjang; misalnya jika ada empat repositori `repo-A`, `repo-B`, `repo-C`, dan `repo-D` dan manajer paket yang terhubung ke `repo-A` mengambil versi paket dari `repo-D`, versi paket akan disimpan di `repo-A` tapi tidak di `repo-B` atau `repo-C`.

Perilaku penyimpanan paket serupa dengan saat menarik versi paket dari repositori eksternal, kecuali bahwa versi paket selalu disimpan di repositori yang memiliki koneksi eksternal terpasang. Misalnya, `repo-A` memiliki `repo-B` sebagai hulu. `repo-B` memiliki `repo-C` sebagai hulu, dan `repo-C` juga memiliki `npmjs.com` yang dikonfigurasi sebagai koneksi eksternal; lihat diagram berikut.



Jika manajer paket yang terhubung ke `repo-A` meminta versi paket, `lodash 4.17.20` misalnya, dan versi paket tidak ada di salah satu dari tiga repositori, akan diambil dari `npmjs.com`. Saat `lodash 4.17.20` diambil, akan disimpan di `repo-A` karena itu adalah repositori paling hilir dan `repo-C` karena memiliki koneksi eksternal ke `npmjs.com` terlampir. `lodash 4.17.20` tidak akan disimpan di `repo-B` karena merupakan repositori menengah.



# Meminta paket dari koneksi eksternal

Bagian berikut menjelaskan cara meminta paket dari koneksi eksternal dan CodeArtifact perilaku yang diharapkan saat meminta paket.

## Topik

- [Mengambil paket dari koneksi eksternal](#)
- [Latensi koneksi lebih masensi koneksi lebih serensi](#)
- [CodeArtifact perilaku ketika repositori eksternal tidak tersedia](#)
- [Ketersediaan versi paket baru](#)
- [Mengimpor versi paket dengan lebih dari satu aset](#)

## Mengambil paket dari koneksi eksternal

Untuk mengambil paket dari koneksi eksternal setelah Anda menambahkannya ke CodeArtifact repositori Anda seperti yang dijelaskan dalam [Connect CodeArtifact repositori ke repositori publik](#), konfigurasi manajer paket Anda untuk menggunakan repositori Anda dan menginstal paket.

### Note

Petunjuk berikut digunakannpm, untuk melihat konfigurasi dan petunjuk penggunaan untuk jenis paket lainnya, lihat [Menggunakan CodeArtifact dengan Maven](#), [Menggunakan CodeArtifact dengan NuGet](#), atau [Menggunakan CodeArtifact dengan Python](#).

## Mengambil paket dari koneksi eksternal

1. Konfigurasi dan autentikasi manajer paket Anda dengan CodeArtifact repositori Anda. Untuknpm, gunakanaws codeartifact login perintah berikut.

```
aws codeartifact login --tool npm --domain my_domain --domain-owner 111122223333 --  
repository my_repo
```

2. Meminta paket dari repositori publik. Untuknpm, gunakannpm install perintah berikut, ganti *Lodash* dengan paket yang ingin Anda instal.

```
npm install lodash
```

3. Setelah paket disalin ke CodeArtifact repositori Anda, Anda dapat menggunakan `list-package-versions` perintah `list-packages` dan untuk melihatnya.

```
aws codeartifact list-packages --domain my_domain --domain-owner 111122223333 --repository my_repo
```

Contoh keluaran:

```
{
  "packages": [
    {
      "format": "npm",
      "package": "lodash"
    }
  ]
}
```

`list-package-versions` Perintah mencantumkan semua versi paket yang disalin ke CodeArtifact repositori Anda.

```
aws codeartifact list-package-versions --domain my_domain --domain-owner 111122223333 --repository my_repo --format npm --package lodash
```

Contoh keluaran:

```
{
  "defaultDisplayVersion": "1.2.5"
  "format": "npm",
  "package": "lodash",
  "namespace": null,
  "versions": [
    {
      "version": "1.2.5",
      "revision": "REVISION-1-SAMPLE-6C81EFF7DA55CC",
      "status": "Published"
    }
  ]
}
```

## Latensi koneksi lebih masensi koneksi lebih serensi

Ketika mengambil paket dari repositori publik menggunakan koneksi eksternal, ada penundaan dari kapan paket diambil dari repositori publik dan ketika disimpan dalam CodeArtifact repositori Anda. Misalnya, Anda telah menginstal versi 1.2.5 dari paket npm “lodash” seperti yang dijelaskan dalam [Mengambil paket dari koneksi eksternal](#). Meskipun perintah `npm install lodash` berhasil diselesaikan, versi paket mungkin belum muncul di CodeArtifact repositori Anda. Biasanya dibutuhkan sekitar 3 menit agar versi paket muncul di repositori Anda, meskipun kadang-kadang bisa memakan waktu lebih lama.

Karena latensi ini, Anda mungkin telah berhasil mengambil versi paket, tetapi mungkin belum dapat melihat versi di repositori Anda di CodeArtifact konsol atau saat memanggil operasi `ListPackages` dan `ListPackageVersions` API. CodeArtifact Setelah asynchronously bertahan versi paket, itu akan terlihat di konsol dan melalui permintaan API.

## CodeArtifact perilaku ketika repositori eksternal tidak tersedia

Terkadang, repositori eksternal akan mengalami pemadaman yang berarti CodeArtifact tidak dapat mengambil paket darinya, atau pengambilan paket jauh lebih lambat dari biasanya. Ketika ini terjadi, versi paket yang telah ditarik dari repositori eksternal (misalnya `npmjs.com`) dan disimpan dalam CodeArtifact repositori akan terus tersedia untuk diunduh dari repositori CodeArtifact. Namun, paket yang belum disimpan CodeArtifact mungkin tidak tersedia, bahkan ketika koneksi eksternal ke repositori tersebut telah dikonfigurasi. Misalnya, CodeArtifact repositori Anda mungkin berisi versi paket `npm-lodash 4.17.19` karena itulah yang telah Anda gunakan dalam aplikasi Anda sejauh ini. Ketika Anda ingin meningkatkan ke `4.17.20`, biasanya CodeArtifact akan mengambil versi baru dari `npmjs.com` dan menyimpannya di CodeArtifact repositori Anda. Namun, jika `npmjs.com` mengalami pemadaman, versi baru ini tidak akan tersedia. Satu-satunya solusi adalah mencoba lagi nanti setelah `npmjs.com` pulih.

Pemadaman repositori eksternal juga dapat memengaruhi publikasi versi paket baru untuk CodeArtifact. Dalam repositori dengan koneksi eksternal dikonfigurasi, tidak CodeArtifact akan mengizinkan publikasi versi paket yang sudah ada di repositori eksternal. Untuk informasi selengkapnya, lihat [Gambaran umum paket](#). Namun, dalam kasus yang jarang terjadi, pemadaman repositori eksternal mungkin berarti bahwa CodeArtifact tidak memiliki up-to-date informasi tentang paket dan versi paket yang ada di repositori eksternal. Dalam hal ini, CodeArtifact mungkin mengizinkan publikasi versi paket yang biasanya ditolak.

## Ketersediaan versi paket baru

Untuk versi paket di repositori publik seperti npmjs.com tersedia melalui CodeArtifact repositori, itu harus terlebih dahulu ditambahkan ke cache metadata paket Regional. Cache ini dikelola oleh CodeArtifact di setiap AWS Wilayah dan berisi metadata yang menjelaskan isi repositori publik yang didukung. Karena cache ini, ada penundaan antara kapan versi paket baru dipublikasikan ke repositori publik dan kapan tersedia dari CodeArtifact. Penundaan ini bervariasi menurut jenis paket.

Untuk paket npm, Python, dan Nuget, mungkin ada penundaan hingga 30 menit sejak versi paket baru diterbitkan ke npmjs.com, pypi.org, atau nuget.org dan kapan tersedia untuk instalasi dari CodeArtifact repositori. CodeArtifact secara otomatis menyinkronkan metadata dari dua repositori ini untuk memastikan bahwa cache diperbarui.

Untuk paket Maven, mungkin ada penundaan hingga 3 jam dari saat versi paket baru diterbitkan ke repositori publik dan ketika tersedia untuk instalasi dari CodeArtifact repositori. CodeArtifact akan memeriksa versi baru dari paket paling banyak sekali setiap 3 jam. Permintaan pertama untuk nama paket tertentu setelah masa hidup cache 3 jam telah berakhir akan menyebabkan semua versi baru dari paket yang akan diimpor ke cache Regional.

Untuk paket Maven yang umum digunakan, versi baru biasanya akan diimpor setiap 3 jam karena tingginya tingkat permintaan berarti bahwa cache akan sering diperbarui segera setelah masa cache telah berakhir. Untuk paket yang jarang digunakan, cache tidak akan memiliki versi terbaru sampai versi paket diminta dari CodeArtifact repositori. Pada permintaan pertama, hanya versi yang diimpor sebelumnya yang akan tersedia CodeArtifact, tetapi permintaan ini akan menyebabkan cache diperbarui. Pada permintaan selanjutnya, versi baru paket akan ditambahkan ke cache dan akan tersedia untuk diunduh.

## Mengimpor versi paket dengan lebih dari satu aset

Kedua paket Maven dan Python dapat memiliki beberapa aset per versi paket. Hal ini membuat mengimpor paket format ini lebih kompleks daripada npm dan NuGet paket, yang hanya memiliki satu aset per versi paket. Untuk deskripsi aset mana yang diimpor untuk jenis paket ini dan bagaimana aset yang baru ditambahkan tersedia, lihat [Meminta paket Python dari upstream dan koneksi eksternal](#) dan [Meminta paket Maven dari upstream dan koneksi eksternal](#).

## Urutan prioritas repositori hulu

Ketika Anda meminta versi paket dari repositori dengan satu atau beberapa repositori hulu, prioritasnya sesuai dengan urutan yang terdaftar ketika memanggil perintah `create-repository`

atau `update-repository`. Ketika versi paket yang diminta ditemukan, pencarian berhenti, meskipun tidak mencari semua repositori hulu. Untuk informasi selengkapnya, lihat [Menambahkan atau menghapus repositori hulu \(AWS CLI\)](#).

Gunakan perintah `describe-repository` untuk melihat urutan prioritas.

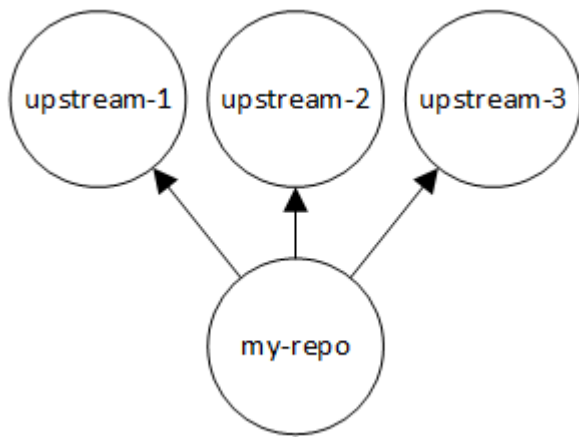
```
aws codeartifact describe-repository --repository my_repo --domain my_domain --domain-owner 111122223333
```

Hasilnya mungkin sebagai berikut. Hal ini menunjukkan bahwa prioritas repositori hulu adalah pertama `upstream-1`, kedua `upstream-2`, dan ketiga `upstream-3`.

```
{
  "repository": {
    "name": "my_repo",
    "administratorAccount": "123456789012",
    "domainName": "my_domain",
    "domainOwner": "111122223333",
    "arn": "arn:aws:codeartifact:us-east-1:111122223333:repository/my_domain/my_repo",
    "description": "My new repository",
    "upstreams": [
      {
        "repositoryName": "upstream-1"
      },
      {
        "repositoryName": "upstream-2"
      },
      {
        "repositoryName": "upstream-3"
      }
    ],
    "externalConnections": []
  }
}
```

## Contoh urutan prioritas sederhana

Pada diagram berikut, repositori `my_repo` memiliki tiga repositori hulu. Urutan prioritas dari repositori hulu adalah `upstream-1`, `upstream-2`, `upstream-3`.



Permintaan untuk versi paket di `my_repo` mencari repositori dalam urutan berikut sampai ditemukan, atau sampai respons HTTP 404 Not Found dikembalikan ke klien:

1. `my_repo`
2. `upstream-1`
3. `upstream-2`
4. `upstream-3`

Jika versi paket ditemukan, pencarian berhenti, bahkan jika tidak terlihat di semua repositori hulu. Misalnya, jika versi paket ditemukan di `upstream-1`, pencarian berhenti dan CodeArtifact tidak mencari di `upstream-2` atau `upstream-3`.

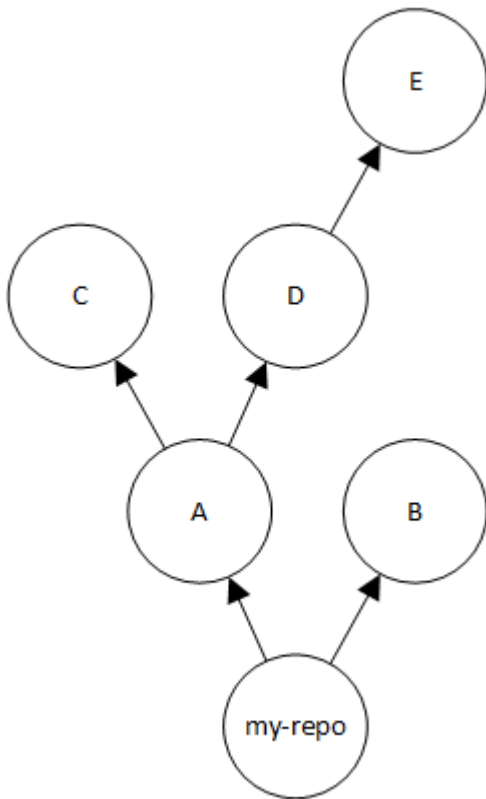
Jika Anda menggunakan perintah AWS CLI `list-package-versions` untuk mencantumkan versi paket di `my_repo`, perintah tersebut hanya melihat di `my_repo`. Versi paket tidak dicantumkan di repositori hulu.

## Contoh urutan prioritas kompleks

Jika repositori hulu memiliki repositori hulu sendiri, logika yang sama digunakan untuk menemukan versi paket sebelum pindah ke repositori hulu berikutnya. Misalnya, anggaplah repositori `my_repo` memiliki dua repositori hulu, A dan B. Jika repositori A memiliki repositori hulu, permintaan untuk versi paket di `my_repo` pertama melihat di `my_repo`, kedua di A, lalu di repositori hulu A, dan sebagainya.

Pada diagram berikut, repositori `my_repo` berisi repositori hulu. Repositori hulu A memiliki dua repositori hulu, dan D memiliki satu repositori hulu. Repositori hulu pada tingkat yang sama dalam diagram muncul dalam urutan prioritas mereka, kiri ke kanan (repositori A memiliki urutan prioritas

yang lebih tinggi dari repositori B, dan repositori C memiliki urutan prioritas yang lebih tinggi dari repositori D).



Dalam contoh ini, permintaan untuk versi paket di `my_repo` melihat di repositori dalam urutan berikut sampai ditemukan, atau sampai manajer paket mengembalikan respons HTTP 404 Not Found kepada klien:

1. `my_repo`
2. A
3. C
4. D
5. E
6. B

## Perilaku API dengan repositori hulu

Ketika Anda memanggil CodeArtifact API tertentu pada repositori yang terhubung ke repositori hulu, perilaku mungkin berbeda tergantung apakah paket atau versi paket atau versi paket disimpan dalam repositori target atau repositori hulu. Perilaku API ini didokumentasikan di sini.

Untuk informasi selengkapnya tentang CodeArtifact API, lihat [ReferensiCodeArtifact API](#).

Kebanyakan API yang mereferensi paket atau versi paket akan mengembalikan kesalahan `ResourceNotFound` jika versi paket yang ditentukan tidak ada dalam repositori target. Hal ini berlaku bahkan jika paket atau versi paket ada dalam repositori hulu. Secara efektif, repositori hulu diabaikan ketika memanggil API ini. API ini adalah:

- `DeletePackageVersions`
- `DescribePackageVersion`
- `GetPackageVersionAsset`
- `GetPackageVersionReadme`
- `ListPackages`
- `ListPackageVersionAssets`
- `ListPackageVersionDependencies`
- `ListPackageVersions`
- `UpdatePackageVersionsStatus`

Untuk menunjukkan perilaku ini, kita memiliki dua repositori: `target-repo` dan `upstream-repo`. `target-repo` kosong dan mengonfigurasi `upstream-repo` sebagai repositori hulu. `upstream-repo` berisi paket npm `lodash`.

Ketika memanggil API `DescribePackageVersion` pada `upstream-repo`, yang berisi paket `lodash`, kita mendapatkan output berikut:

```
{
  "packageVersion": {
    "format": "npm",
    "packageName": "lodash",
    "displayName": "lodash",
    "version": "4.17.20",
    "summary": "Lodash modular utilities.",
    "homePage": "https://lodash.com/",
    "sourceCodeRepository": "https://github.com/lodash/lodash.git",
    "publishedTime": "2020-10-14T11:06:10.370000-04:00",
    "licenses": [
      {
        "name": "MIT"
      }
    ]
  }
}
```



```
  ],  
  "revision": "Ciqe5/9yicvkJT13b5/LdLpCyE6fqA7poa9qp+FilPs=",  
  "status": "Published"  
}
```

Ketika memanggil API yang sama pada `target-repo`, yang kosong tetapi mengonfigurasi `upstream-repo` sebagai hulu, kita mendapatkan output sebagai berikut:

```
An error occurred (ResourceNotFoundException) when calling the DescribePackageVersion  
operation:  
Package not found in repository. RepoId: repo-id, Package =  
PackageCoordinate{packageType=npm, packageName=lodash},
```

Perilaku API `CopyPackageVersions` berbeda. Secara default, API `CopyPackageVersions` hanya menyalin versi paket yang disimpan dalam repositori target. Jika disimpan dalam repositori hulu tetapi tidak dalam repositori target, versi paket tidak akan disalin. Untuk menyertakan paket versi dari paket yang hanya disimpan di repositori hulu, tetapkan nilai `includeFromUpstream` ke `true` dalam permintaan API Anda.

Untuk informasi API `CopyPackageVersions` lebih lanjut, lihat [Menyalin paket antar-repositori](#).

# Bekerja dengan paket di CodeArtifact

Topik ini menunjukkan kepada Anda cara membuat daftar, menyalin, menghapus, dan mencari paket menggunakan CodeArtifact CLI dan API.

## Topik

- [Gambaran umum paket](#)
- [Mencantumkan nama paket](#)
- [Mencantumkan versi paket](#)
- [Mencantumkan aset versi paket](#)
- [Mengunduh aset versi paket](#)
- [Menyalin paket antar-repositori](#)
- [Menghapus paket](#)
- [Melihat dan memperbarui detail versi paket dan dependensi](#)
- [Memperbarui status versi paket](#)
- [Mengedit kontrol asal paket](#)

## Gambaran umum paket

Paket adalah bundel perangkat lunak dan metadata yang diperlukan untuk menyelesaikan dependensi dan menginstal perangkat lunak. Dalam CodeArtifact, paket terdiri dari nama paket, [namespace](#) opsional seperti `@types in@types/node`, satu set versi paket, dan metadata tingkat paket seperti tag npm.

## Daftar Isi

- [Format paket yang didukung](#)
- [Publikasi paket](#)
  - [Izin publikasi](#)
  - [Penimpanan aset paket](#)
  - [Paket privat dan repositori publik](#)
  - [Memublikasikan versi paket dengan patch](#)
  - [Batas ukuran aset untuk penerbitan](#)
  - [Latensi penerbitan](#)

- [Status versi paket](#)
- [Nama paket, versi paket, dan normalisasi nama aset](#)

## Format paket yang didukung

AWS CodeArtifact [mendukung npm, PyPI, Maven NuGet, dan format paket generik.](#)

## Publikasi paket

Anda dapat mempublikasikan versi baru dari [format paket apa pun yang didukung](#) ke CodeArtifact repositori menggunakan alat seperti npm, twine, Maven Gradle, dan dotnet.

## Izin publikasi

Pengguna atau peran AWS Identity and Access Management (IAM) Anda harus memiliki izin untuk mempublikasikan ke repositori tujuan. Izin berikut diperlukan untuk mempublikasikan paket:

- Maven: `codeartifact:PublishPackageVersion` dan `codeartifact:PutPackageMetadata`
- npm: `codeartifact:PublishPackageVersion`
- NuGet: `codeartifact:PublishPackageVersion` dan `codeartifact:ReadFromRepository`
- Python: `codeartifact:PublishPackageVersion`
- generik: `codeartifact:PublishPackageVersion`

Dalam daftar izin sebelumnya, kebijakan IAM Anda harus menentukan package sumber daya untuk izin `codeartifact:PublishPackageVersion` dan `codeartifact:PutPackageMetadata`. Itu juga harus menentukan repository sumber daya untuk `codeartifact:ReadFromRepository` izin.

Untuk informasi selengkapnya tentang izin di CodeArtifact, lihat [AWS CodeArtifact referensi izin](#).

## Penyimpanan aset paket

Anda tidak dapat mempublikasikan kembali aset paket yang sudah ada dengan konten yang berbeda. Misalnya, anggaplah Anda sudah mempublikasikan paket Maven dengan aset `JAR mypackage-1.0.jar`. Anda hanya dapat mempublikasikan aset itu lagi jika checksum aset lama dan baru identik. Untuk mempublikasikan aset yang sama dengan konten baru, hapus versi paket

menggunakan perintah `delete-package-versions` terlebih dahulu. Mencoba untuk memublikasikan ulang nama aset yang sama dengan konten yang berbeda akan mengakibatkan kesalahan konflik HTTP 409.

Untuk format paket yang mendukung beberapa aset (generik, PyPI, dan Maven), Anda dapat menambahkan aset baru dengan nama berbeda ke versi paket yang ada, dengan asumsi Anda memiliki izin yang diperlukan. Untuk paket generik, Anda dapat menambahkan aset baru selama versi paket dalam `Unfinished` status. Karena npm hanya mendukung aset tunggal per versi paket, untuk mengubah versi paket yang sudah dipublikasikan dengan cara apa pun, Anda harus terlebih dahulu menghapusnya menggunakan `delete-package-versions`.

Jika Anda mencoba untuk memublikasikan kembali aset yang sudah ada (misalnya, `mypackage-1.0.jar`), dan isi dari aset yang dipublikasikan dan aset baru identik, operasi akan berhasil karena operasi idempoten.

## Paket privat dan repositori publik

CodeArtifact tidak memublikasikan paket yang disimpan dalam CodeArtifact repositori ke repositori publik seperti `npmjs.com` atau `Maven Central`. CodeArtifact mengimpor paket dari repositori publik ke CodeArtifact repositori, tetapi tidak pernah memindahkan paket ke arah lain. Paket yang Anda publikasikan ke CodeArtifact repositori tetap pribadi dan hanya tersedia untuk AWS akun, peran, dan pengguna yang telah Anda berikan akses.

## Memublikasikan versi paket dengan patch

Terkadang Anda mungkin ingin memublikasikan versi paket yang dimodifikasi, kemungkinan yang tersedia di repositori publik. Misalnya, Anda mungkin menemukan bug dalam dependensi aplikasi penting yang disebut `mydep 1.1`, dan Anda harus memperbaikinya lebih cepat dari yang dapat ditinjau vendor paket dan menerima perubahan. Seperti yang dijelaskan sebelumnya, CodeArtifact mencegah Anda menerbitkan `mydep 1.1` di repositori Anda jika CodeArtifact repositori publik dapat dijangkau dari repositori Anda melalui CodeArtifact repositori hulu dan koneksi eksternal.

Untuk mengatasinya, publikasikan versi paket ke repositori lain di mana CodeArtifact repositori publik tidak dapat dijangkau. Kemudian gunakan `copy-package-versions` API untuk menyalin versi tambalan `mydep 1.1` ke CodeArtifact repositori tempat Anda akan menggunakannya.

## Batas ukuran aset untuk penerbitan

Ukuran maksimum aset paket yang dapat dipublikasikan dibatasi oleh kuota maksimum ukuran file Aset yang ditunjukkan pada [Kuota diAWSCodeArtifact](#). Misalnya, Anda tidak dapat memublikasikan

roda Maven JAR atau Python yang lebih besar dari kuota maksimum ukuran file aset Anda saat ini. Jika Anda perlu menyimpan aset yang lebih besar CodeArtifact, mintalah peningkatan kuota.

Selain kuota maksimum ukuran file aset, ukuran maksimum permintaan penerbitan untuk paket npm adalah 2 GB. Batas ini tidak tergantung pada kuota maksimum ukuran file aset dan tidak dapat dinaikkan dengan peningkatan kuota. Dalam permintaan penerbitan npm (HTTP PUT), metadata paket dan konten arsip tar paket npm dibundel bersama. Karena itu, ukuran maksimum sebenarnya dari paket npm yang dapat dipublikasikan bervariasi dan tergantung pada ukuran metadata yang disertakan.

#### Note

Paket npm yang diterbitkan dibatasi hingga ukuran maksimum kurang dari 2 GB.

## Latensi penerbitan

Versi Package yang diterbitkan ke CodeArtifact repositori sering tersedia untuk diunduh dalam waktu kurang dari satu detik. Misalnya, jika Anda mempublikasikan versi paket npm ke CodeArtifact with `npm publish`, versi tersebut harus tersedia untuk `npm install` perintah dalam waktu kurang dari satu detik. Namun, penerbitan bisa tidak konsisten dan terkadang bisa memakan waktu lebih lama. Jika Anda harus menggunakan versi paket segera setelah penerbitan, gunakan percobaan ulang untuk memastikan bahwa unduhan dapat diandalkan. Misalnya, setelah menerbitkan versi paket, ulangi unduhan hingga tiga kali jika versi paket yang baru saja diterbitkan pada awalnya tidak tersedia pada upaya unduhan pertama.

#### Note

Mengimpor versi paket dari repositori publik biasanya membutuhkan waktu lebih lama daripada penerbitan. Untuk informasi selengkapnya, lihat [Latensi koneksi lebih masensi koneksi lebih serensi](#).

## Status versi paket

Setiap versi paket CodeArtifact memiliki status yang menjelaskan status saat ini dan ketersediaan versi paket. Anda dapat mengubah status versi paket di AWS CLI dan SDK. Untuk informasi selengkapnya, lihat [Memperbarui status versi paket](#).

Berikut ini adalah nilai yang mungkin untuk status versi paket:

- **Diterbitkan** - Versi paket berhasil diterbitkan dan dapat diminta menggunakan manajer paket. Versi paket akan disertakan dalam daftar versi paket yang dikembalikan ke manajer paket, misalnya, dalam `outputnpm view <package-name> versions`. Semua aset versi paket tersedia dari repositori.
- **Belum Selesai** — Klien telah mengunggah satu atau lebih aset untuk versi paket, tetapi belum menyelesaikannya dengan memindahkannya ke negara bagian. `Published` Saat ini hanya versi paket generik dan Maven yang dapat memiliki status. `Unfinished` Untuk paket Maven, ini dapat terjadi ketika klien mengunggah satu atau lebih aset untuk versi paket tetapi tidak mempublikasikan `maven-metadata.xml` file untuk paket yang menyertakan versi itu. Ketika versi paket Maven `Belum Selesai`, itu tidak akan disertakan dalam daftar versi yang dikembalikan ke klien seperti itu `mvn` atau `gradle`, sehingga tidak dapat digunakan sebagai bagian dari build. Paket generik dapat dengan sengaja disimpan dalam `Unfinished` status dengan memberikan `unfinished` tanda saat memanggil API. [PublishPackageVersion](#) Paket generik dapat diubah ke `Published` status dengan menghilangkan `unfinished` bendera, atau dengan memanggil API. [UpdatePackageVersionsStatus](#)
- **Tidak terdaftar** - Aset versi paket tersedia untuk diunduh dari repositori, tetapi versi paket tidak termasuk dalam daftar versi yang dikembalikan ke manajer paket. Misalnya, untuk paket `npm`, `output npm view <package-name> versions` tidak akan menyertakan versi paket. Ini berarti bahwa logika resolusi dependensi `npm` tidak akan memilih versi paket karena versi tidak muncul dalam daftar versi yang tersedia. Namun, jika versi paket `Tidak Terdaftar` telah direferensikan dalam file `npm package-lock.json`, versi paket masih boleh diunduh dan diinstal, misalnya, ketika menjalankan `npm ci`.
- **Diarsipkan** — Aset versi paket tidak dapat lagi diunduh. Versi paket tidak akan dimasukkan dalam daftar versi yang dikembalikan ke manajer paket. Karena aset tidak tersedia, konsumsi versi paket oleh klien diblokir. Jika build aplikasi Anda tergantung pada versi yang diperbarui ke `Diarsipkan`, build akan rusak, dengan asumsi versi paket belum di-cache secara lokal. [Anda tidak dapat menggunakan pengelola paket atau alat pembuatan untuk menerbitkan ulang versi paket yang Diarsipkan karena masih ada di repositori, tetapi Anda dapat mengubah status versi paket kembali ke Tidak Terdaftar atau Diterbitkan dengan API. UpdatePackageVersionsStatus](#)
- **Disposed** — Versi paket tidak muncul dalam daftar dan aset tidak dapat diunduh dari repositori. Perbedaan utama antara `Disposed` dan `Archived` adalah bahwa dengan status `Disposed`, aset versi paket akan dihapus secara permanen oleh CodeArtifact Untuk alasan ini, Anda tidak dapat memindahkan versi paket dari `Dibuang` ke `Diarsipkan`, `Tidak Terdaftar`, atau `Dipublikasikan`. Versi

paket tidak dapat lagi digunakan karena aset telah dihapus. Setelah versi paket ditandai sebagai `Disposed`, Anda tidak akan lagi ditagih untuk penyimpanan aset paket.

Versi Package dari semua status akan dikembalikan secara default saat memanggil `list-package-versions` tanpa `--status` parameter.

Terlepas dari status yang tercantum sebelumnya, versi paket juga dapat dihapus dengan [DeletePackageVersionsAPI](#). Setelah dihapus, versi paket tidak lagi ada di repositori dan Anda dapat dengan bebas menerbitkan ulang versi paket tersebut menggunakan pengelola paket atau alat pembuatan. Setelah versi paket dihapus, Anda tidak akan lagi ditagih untuk penyimpanan aset versi paket tersebut.

## Nama paket, versi paket, dan normalisasi nama aset

CodeArtifact menormalkan nama paket, versi paket, dan nama aset sebelum menyimpannya, yang berarti nama atau versi CodeArtifact mungkin berbeda dari nama atau versi yang disediakan saat paket diterbitkan. Untuk informasi selengkapnya tentang cara nama dan versi dinormalisasi CodeArtifact untuk setiap jenis paket, lihat dokumentasi berikut:

- [Normalisasi nama paket Python](#)
- [NuGetnama paket, versi, dan normalisasi nama aset](#)

CodeArtifact tidak melakukan normalisasi pada format paket lainnya.

## Mencantumkan nama paket

Gunakan `list-packages` perintah CodeArtifact untuk mendapatkan daftar semua nama paket dalam repositori. Perintah ini hanya mengembalikan nama paket, bukan versi.

```
aws codeartifact list-packages --domain my_domain --domain-owner 111122223333 --  
repository my_repo
```

Contoh output:

```
{  
  "nextToken": "eyJidWNrZXRJZCI6I...",  
  "packages": [  
    {  
      "package": "acorn",
```

```
    "format": "npm",
    "originConfiguration": {
      "restrictions": {
        "publish": "BLOCK",
        "upstream": "ALLOW"
      }
    },
    {
      "package": "acorn-dynamic-import",
      "format": "npm",
      "originConfiguration": {
        "restrictions": {
          "publish": "BLOCK",
          "upstream": "ALLOW"
        }
      },
      {
        "package": "ajv",
        "format": "npm",
        "originConfiguration": {
          "restrictions": {
            "publish": "BLOCK",
            "upstream": "ALLOW"
          }
        },
        {
          "package": "ajv-keywords",
          "format": "npm",
          "originConfiguration": {
            "restrictions": {
              "publish": "BLOCK",
              "upstream": "ALLOW"
            }
          },
          {
            "package": "anymatch",
            "format": "npm",
            "originConfiguration": {
              "restrictions": {
                "publish": "BLOCK",
                "upstream": "ALLOW"
              }
            }
          },
          {
```



```
    "package": "ast",
    "namespace": "webassemblyjs",
    "format": "npm",
    "originConfiguration": {
      "restrictions": {
        "publish": "BLOCK",
        "upstream": "ALLOW"
      }
    }
  ]
}
```

## Mencantumkan nama paket npm

Untuk mencantumkan hanya nama paket npm, tetapkan nilai opsi `--format` ke `npm`.

```
aws codeartifact list-packages --domain my_domain --domain-owner 111122223333 --
repository my_repo \
  --format npm
```

Untuk daftar paket npm di namespace (lingkup npm), gunakan opsi `--namespace` dan `--format`.

### Important

Nilai untuk opsi `--namespace` tidak boleh mencakup `@` di awal. Untuk mencari namespace `@types`, tetapkan nilai ke *jenis*.

### Note

`--namespace`Opsi filter dengan awalan namespace. Paket npm apa pun dengan cakupan yang dimulai dengan nilai yang diteruskan ke `--namespace` opsi akan dikembalikan dalam `list-packages` respons.

```
aws codeartifact list-packages --domain my_domain --domain-owner 111122223333 --
repository my_repo \
  --format npm --namespace types
```

Contoh output:

```
{
  "nextToken": "eyJidWNrZXRJZ...",
  "packages": [
    {
      "package": "3d-bin-packing",
      "namespace": "types",
      "format": "npm"
    },
    {
      "package": "a-big-triangle",
      "namespace": "types",
      "format": "npm"
    },
    {
      "package": "a11y-dialog",
      "namespace": "types",
      "format": "npm"
    }
  ]
}
```

## Mencantumkan nama paket Maven

Untuk mencantumkan hanya nama paket Maven, tetapkan nilai opsi `--format` ke `maven`. Anda juga harus menentukan ID grup Maven di opsi. `--namespace`

### Note

`--namespace`Opsi filter dengan awalan namespace. Paket npm apa pun dengan cakupan yang dimulai dengan nilai yang diteruskan ke `--namespace` opsi akan dikembalikan dalam `list-packages` respons.

```
aws codeartifact list-packages --domain my_domain --domain-owner 111122223333 --
repository my_repo \
  --format maven --namespace org.apache.commons
```

Contoh output:

```
{
  "nextToken": "eyJidWNrZXRJZ...",
  "packages": [
    {
      "package": "commons-lang3",
      "namespace": "org.apache.commons",
      "format": "maven"
    },
    {
      "package": "commons-collections4",
      "namespace": "org.apache.commons",
      "format": "maven"
    },
    {
      "package": "commons-compress",
      "namespace": "org.apache.commons",
      "format": "maven"
    }
  ]
}
```

## Mencantumkan nama paket Python

Untuk mencantumkan hanya nama Python, tetapkan nilai opsi `--format` ke `pypi`.

```
aws codeartifact list-packages --domain my_domain --domain-owner 111122223333 --
repository my_repo \
  --format pypi
```

## Filter berdasarkan prefiks nama paket

Untuk mengembalikan paket yang dimulai dengan string tertentu, Anda dapat menggunakan opsi `--package-prefix`.

```
aws codeartifact list-packages --domain my_domain --domain-owner 111122223333 --
repository my_repo \
  --format npm --package-prefix pat
```

## Contoh output:

```
{
  "nextToken": "eyJidWNrZXRJZ...",
  "packages": [
    {
      "package": "path",
      "format": "npm"
    },
    {
      "package": "pat-test",
      "format": "npm"
    },
    {
      "package": "patch-math3",
      "format": "npm"
    }
  ]
}
```

## Kombinasi opsi penelusuran yang didukung

Anda dapat menggunakan opsi `--format`, `--namespace`, dan `--package-prefix` dalam kombinasi apa pun, kecuali bahwa `--namespace` tidak dapat digunakan sendiri. Mencari semua paket npm dengan cakupan yang dimulai dengan `@types` memerlukan `--format` opsi yang akan ditentukan. Menggunakan `--namespace` sendiri menghasilkan kesalahan.

Tidak menggunakan satu pun dari tiga pilihan ini juga didukung oleh `list-packages` dan akan mengembalikan semua paket dari semua format yang ada dalam repositori.

## Output format

Anda dapat menggunakan parameter yang tersedia untuk semua AWS CLI perintah untuk membuat `list-packages` respons ringkas dan lebih mudah dibaca. Gunakan parameter `--query` untuk menentukan format setiap versi paket yang dikembalikan. Gunakan parameter `--output` untuk memformat respons sebagai plaintext.

```
aws codeartifact list-packages --domain my_domain --domain-owner 111122223333 --  
repository my_repo \  
  --output text --query 'packages[*].[package]'
```

Contoh output:

```
accepts  
array-flatten  
body-parser  
bytes  
content-disposition  
content-type  
cookie  
cookie-signature
```

Untuk informasi lebih lanjut, lihat [Mengontrol output perintah dari AWS CLI](#) dalam Panduan Pengguna AWS Command Line Interface .

## Default dan opsi lainnya

Secara default, jumlah maksimum hasil yang dikembalikan oleh `list-packages` adalah 100. Anda dapat mengubah batas hasil ini menggunakan opsi `--max-results`.

```
aws codeartifact list-packages --domain my_domain --domain-owner 111122223333 --  
repository my_repo --max-results 20
```

Nilai maksimum `--max-results` yang diizinkan adalah 1.000. Untuk mengizinkan mencantumkan paket dalam repositori dengan lebih dari 1.000 paket, `list-packages` mendukung pemberian nomor halaman menggunakan bidang `nextToken` dalam respons. Jika jumlah paket dalam repositori lebih dari nilai `--max-results`, Anda dapat meneruskan nilai `nextToken` ke invokasi lain dari `list-packages` untuk mendapatkan halaman hasil berikutnya.

```
aws codeartifact list-packages --domain my_domain --domain-owner 111122223333 --  
repository my_repo \  
  --next-token r00ABXNyAEjbj...
```

## Mencantumkan versi paket

Gunakan `list-package-versions` perintah AWS CodeArtifact untuk mendapatkan daftar semua versi nama paket dalam repositori.

```
aws codeartifact list-package-versions --package kind-of \  
--domain my_domain --domain-owner 111122223333 \  
--repository my_repository --format npm
```

Contoh output:

```
{  
  "defaultDisplayVersion": "1.0.1",  
  "format": "npm",  
  "package": "kind-of",  
  "versions": [  
    {  
      "version": "1.0.1",  
      "revision": "REVISION-SAMPLE-1-C7F4S5E9B772FC",  
      "status": "Published",  
      "origin": {  
        "domainEntryPoint": {  
          "externalConnectionName": "public:npmjs"  
        },  
        "originType": "EXTERNAL"  
      }  
    },  
    {  
      "version": "1.0.0",  
      "revision": "REVISION-SAMPLE-2-C752BEEF6D2CFC",  
      "status": "Published",  
      "origin": {  
        "domainEntryPoint": {  
          "externalConnectionName": "public:npmjs"  
        },  
        "originType": "EXTERNAL"  
      }  
    },  
    {  
      "version": "0.1.2",  
      "revision": "REVISION-SAMPLE-3-654S65A5C5E1FC",  
      "status": "Published",  
    }  
  ]  
}
```

```
    "origin": {
      "domainEntryPoint": {
        "externalConnectionName": "public:npmjs"
      },
      "originType": "EXTERNAL"
    }
  },
  {
    "version": "0.1.1",
    "revision": "REVISION-SAMPLE-1-C7F4S5E9B772FC",
    "status": "Published",
    "origin": {
      "domainEntryPoint": {
        "externalConnectionName": "public:npmjs"
      },
      "originType": "EXTERNAL"
    }
  },
  {
    "version": "0.1.0",
    "revision": "REVISION-SAMPLE-4-AF669139B772FC",
    "status": "Published",
    "origin": {
      "domainEntryPoint": {
        "externalConnectionName": "public:npmjs"
      },
      "originType": "EXTERNAL"
    }
  }
]
}
```

Anda dapat menambahkan parameter `--status` ke panggilan `list-package-versions` untuk memfilter hasil berdasarkan status versi paket. Untuk informasi selengkapnya tentang status versi paket, lihat [Status versi paket](#).

Anda dapat memberi nomor halaman pada respons dari `list-package-versions` menggunakan `--max-results` dan parameter `--next-token`. Untuk `--max-results`, tentukan bilangan bulat dari 1 sampai 1000 untuk menentukan jumlah hasil yang dikembalikan dalam satu halaman. Defaultnya adalah 50. Untuk mengembalikan halaman berikutnya, jalankan `list-package-versions` lagi dan teruskan nilai `nextToken` yang diterima dalam perintah output sebelumnya

untuk `--next-token`. Saat opsi `--next-token` tidak digunakan, halaman pertama hasil selalu dikembalikan.

Perintah `list-package-versions` tidak mencantumkan versi paket di repositori hulu. Namun, referensi ke versi paket dalam repositori hulu yang disalin ke repositori Anda selama permintaan versi paket tercantum. Untuk informasi selengkapnya, lihat [Bekerja dengan repositori hulu di CodeArtifact](#).

## Daftar versi paket npm

Untuk mencantumkan semua versi paket untuk paket npm, tetapkan nilai `--format` opsi `kenpm`.

```
aws codeartifact list-package-versions --package my_package --domain my_domain \  
--domain-owner 111122223333 --repository my_repo --format npm
```

Untuk mencantumkan versi paket npm di namespace tertentu (lingkup npm), gunakan opsi `--namespace`. Nilai untuk opsi `--namespace` tidak boleh mencakup `@` di awal. Untuk mencari namespace `@types`, tetapkan nilai ke *jenis*.

```
aws codeartifact list-package-versions --package my_package --domain my_domain \  
--domain-owner 111122223333 --repository my_repo --format npm \  
--namespace types
```

## Daftar versi paket Maven

Untuk mencantumkan semua versi paket untuk paket Maven, tetapkan nilai `--format` opsi ke `maven`. Anda juga harus menentukan ID grup Maven di opsi `--namespace`.

```
aws codeartifact list-package-versions --package my_package --domain my_domain \  
--domain-owner 111122223333 --repository my_repo --format maven \  
--namespace org.apache.commons
```

## Mengurutkan versi

`list-package-versions` dapat memberikan output versi yang diurutkan dalam urutan menurun berdasarkan waktu publikasi (versi yang paling baru dipublikasikan dicantumkan pertama). Gunakan parameter `--sort-by` dengan nilai `PUBLISHED_TIME`, sebagai berikut.

```
aws codeartifact list-package-versions --domain my_domain --domain-owner 111122223333 \  
--repository my_repository \  
--sort-by PUBLISHED_TIME
```



```
--format npm --package webpack --max-results 5 --sort-by PUBLISHED_TIME
```

Contoh output:

```
{  
  
  "defaultDisplayVersion": "4.41.2",  
  "format": "npm",  
  "package": "webpack",  
  "versions": [  
    {  
      "version": "5.0.0-beta.7",  
      "revision": "REVISION-SAMPLE-1-C7F4S5E9B772FC",  
      "status": "Published"  
    },  
    {  
      "version": "5.0.0-beta.6",  
      "revision": "REVISION-SAMPLE-2-C752BEEF6D2CFC",  
      "status": "Published"  
    },  
    {  
      "version": "5.0.0-beta.5",  
      "revision": "REVISION-SAMPLE-3-654S65A5C5E1FC",  
      "status": "Published"  
    },  
    {  
      "version": "5.0.0-beta.4",  
      "revision": "REVISION-SAMPLE-4-AF669139B772FC",  
      "status": "Published"  
    },  
    {  
      "version": "5.0.0-beta.3",  
      "revision": "REVISION-SAMPLE-5-C752BEE9B772FC",  
      "status": "Published"  
    }  
  ],  
  "nextToken": "eyJsaXN0UGF...."  
}
```

## Versi tampilan default

Nilai yang dikembalikan untuk `defaultDisplayVersion` bergantung pada format paket:

- Untuk paket generik, Maven, dan PyPI, ini adalah versi paket yang paling baru diterbitkan.
- Untuk paket npm, itu adalah versi yang direferensikan oleh tanda `latest`. Jika tanda `latest` tidak diatur, nilai adalah versi paket yang paling baru dipublikasikan.

## Output format

Anda dapat menggunakan parameter yang tersedia untuk semua AWS CLI perintah untuk membuat `list-package-versions` respons ringkas dan lebih mudah dibaca. Gunakan parameter `--query` untuk menentukan format setiap versi paket yang dikembalikan. Gunakan `--output` parameter untuk memformat respons sebagai teks biasa.

```
aws codeartifact list-package-versions --package my-package-name --domain my_domain --  
domain-owner 111122223333 \  
--repository my_repo --format npm --output text --query 'versions[*].[version]'
```

Contoh output:

```
0.1.1  
0.1.2  
0.1.0  
3.0.0
```

Untuk informasi lebih lanjut, lihat [Mengontrol output perintah dari AWS CLI](#) dalam Panduan Pengguna AWS Command Line Interface .

## Mencantumkan aset versi paket

Aset adalah file individual (misalnya, file npm atau `.tgz` file Maven POM atau JAR) CodeArtifact yang disimpan di dalamnya terkait dengan versi paket. Anda dapat menggunakan perintah `list-package-version-assets` untuk mencantumkan aset di setiap versi paket.

Jalankan `list-package-version-assets` perintah untuk mengembalikan informasi berikut tentang setiap aset di AWS akun Anda dan AWS Wilayah Anda saat ini:

- Namanya.
- Ukurannya, dalam byte.
- Satu set nilai hash yang digunakan untuk validasi checksum.

Misalnya, gunakan perintah berikut untuk mencantumkan aset paket Python `flatten-json`, versi `0.1.7`.

```
aws codeartifact list-package-version-assets --domain my_domain --domain-
owner 111122223333 \
--repository my_repo --format pypi --package flatten-json \
--package-version 0.1.7
```

Berikut ini menunjukkan output.

```
{
  "format": "pypi",
  "package": "flatten-json",
  "version": "0.1.7",
  "versionRevision": "REVISION-SAMPLE-1-C7F4S5E9B772FC",
  "assets": [
    {
      "name": "flatten_json-0.1.7-py3-none-any.whl",
      "size": 31520,
      "hashes": {
        "MD5": "41bba98d5b9219c43089eEXAMPLE-MD5",
        "SHA-1": "69b215c25dd4cda1d997a786ec6EXAMPLE-SHA-1",
        "SHA-256": "43f24850b7b7b7d79c5fa652418518fbdf427e602b1edabe6EXAMPLE-
SHA-256",
        "SHA-512":
"3947382ac2c180ee3f2aba4f8788241527c8db9dfe9f4b039abe9fc560aaf5a1fced7bd1e80a0dca9ce320d95f086
SHA-512"
      }
    },
    {
      "name": "flatten_json-0.1.7.tar.gz",
      "size": 2865,
      "hashes": {
        "MD5": "41bba98d5b9219c43089eEXAMPLE-MD5",
        "SHA-1": "69b215c25dd4cda1d997a786ec6EXAMPLE-SHA-1",
        "SHA-256": "43f24850b7b7b7d79c5fa652418518fbdf427e602b1edabe6EXAMPLE-
SHA-256",
        "SHA-512":
"3947382ac2c180ee3f2aba4f8788241527c8db9dfe9f4b039abe9fc560aaf5a1fced7bd1e80a0dca9ce320d95f086
SHA-512"
      }
    }
  ]
}
```



```
--asset guava-27.1-jre.jar \  
guava-27.1-jre.jar
```

*Dalam contoh ini, nama file ditentukan sebagai `guava-27.1-jre.jar` oleh argumen terakhir dalam perintah sebelumnya, sehingga aset yang diunduh akan diberi nama `guava-27.1-jre.jar`.*

Output dari perintah yaitu:

```
{  
  "assetName": "guava-27.1-jre.jar",  
  "packageVersion": "27.1-jre",  
  "packageVersionRevision": "YGp9ck2tmy03PGSxioclYzQ0BfTLR9zzhQJtERv62I="  
}
```

#### Note

Untuk mengunduh aset dari paket npm tercakup, sertakan cakupan dalam opsi. `--namespace` Simbol `@` harus dihilangkan ketika menggunakan `--namespace`. Misalnya, jika ruang lingkupnya `@types`, gunakan `--namespace types`.

Mengunduh aset menggunakan `get-package-version-asset` membutuhkan izin `codeartifact:GetPackageVersionAsset` pada sumber daya paket. Untuk informasi selengkapnya tentang kebijakan izin berbasis sumber daya, lihat [Resource-based policies](#) di Panduan Pengguna AWS Identity and Access Management .

## Menyalin paket antar-repositori

Anda dapat menyalin versi paket dari satu repositori ke repositori lainnya di CodeArtifact. Hal ini dapat membantu untuk skenario seperti alur kerja promosi paket atau berbagi versi paket antara tim atau proyek. Repositori sumber dan tujuan harus berada dalam domain yang sama untuk menyalin versi paket.

## Izin IAM yang diperlukan untuk menyalin paket

Untuk menyalin versi paket CodeArtifact, pengguna panggilan harus memiliki izin IAM yang diperlukan dan kebijakan berbasis sumber daya yang dilampirkan ke repositori sumber dan tujuan

harus memiliki izin yang diperlukan. Untuk informasi selengkapnya tentang kebijakan dan repositori izin berbasis sumber daya, lihat. CodeArtifact [Kebijakan repositori](#)

Panggilan pengguna `copy-package-versions` harus memiliki `ReadFromRepository` izin pada repositori sumber dan `CopyPackageVersions` izin pada repositori tujuan.

Repositori sumber harus memiliki `ReadFromRepository` izin dan repositori tujuan harus memiliki izin yang ditetapkan ke akun IAM atau paket `CopyPackageVersions` penyalinan pengguna. Kebijakan berikut adalah contoh kebijakan repositori yang akan ditambahkan ke repositori sumber atau repositori tujuan dengan perintah. `put-repository-permissions-policy` Ganti **111122223333** dengan ID panggilan akun. `copy-package-versions`

#### Note

Panggilan `put-repository-permissions-policy` akan menggantikan kebijakan repositori saat ini jika ada. Anda dapat menggunakan `get-repository-permissions-policy` perintah untuk melihat apakah ada kebijakan, untuk informasi selengkapnya lihat [Membaca kebijakan](#). Jika ada kebijakan, Anda mungkin ingin menambahkan izin ini ke sana alih-alih menggantinya.

#### Contoh kebijakan izin repositori sumber

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "codeartifact:ReadFromRepository"
      ],
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Resource": "*"
    }
  ]
}
```

#### Contoh kebijakan izin repositori tujuan

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "codeartifact:CopyPackageVersions"
      ],
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Resource": "*"
    }
  ]
}
```

## Menyalin versi paket

Gunakan `copy-package-versions` perintah CodeArtifact untuk menyalin satu atau beberapa versi paket dari repositori sumber ke repositori tujuan di domain yang sama. Contoh berikut akan menyalin versi 6.0.2 dan 4.0.0 dari paket npm bernama `my-package` dari repositori `my_repo` ke repositori `repo-2`.

```
aws codeartifact copy-package-versions --domain my_domain --domain-owner 111122223333
--source-repository my_repo \
--destination-repository repo-2 --package my-package --format npm \
--versions 6.0.2 4.0.0
```

Anda dapat menyalin beberapa versi dari nama paket yang sama dalam satu operasi. Untuk menyalin versi dari nama paket yang berbeda, Anda harus memanggil `copy-package-versions` untuk setiap nama paket.

Perintah sebelumnya akan menghasilkan output berikut, dengan asumsi kedua versi berhasil disalin.

```
{
  "successfulVersions": {
    "6.0.2": {
      "revision": "REVISION-1-SAMPLE-6C81EFF7DA55CC",
      "status": "Published"
    },
  },
}
```

```
    "4.0.0": {
      "revision": "REVISION-1-SAMPLE-6C81EFF7DA55CC",
      "status": "Published"
    }
  },
  "failedVersions": {}
}
```

## Menyalin paket dari repositori hulu

Biasanya, `copy-package-versions` hanya melihat di repositori yang ditentukan oleh opsi `--source-repository` untuk versi yang akan disalin. Namun, Anda dapat menyalin versi dari kedua repositori sumber dan repositori hulu dengan menggunakan opsi `--include-from-upstream`. Jika Anda menggunakan CodeArtifact SDK, panggil `CopyPackageVersions` API dengan `includeFromUpstream` parameter yang disetel ke `true`. Untuk informasi selengkapnya, lihat [Bekerja dengan repositori hulu di CodeArtifact](#).

## Menyalin paket npm dalam lingkup

Untuk menyalin versi paket npm dalam lingkup, gunakan opsi `--namespace` untuk menentukan ruang lingkup. Misalnya, untuk menyalin paket `@types/react`, gunakan `--namespace types`. Simbol `@` harus dihilangkan ketika menggunakan `--namespace`.

```
aws codeartifact copy-package-versions --domain my_domain --domain-owner 111122223333
--source-repository repo-1 \
--destination-repository repo-2 --format npm --namespace types \
--package react --versions 0.12.2
```

## Menyalin versi paket Maven

Untuk menyalin versi paket Maven antar repositori, tentukan paket yang akan disalin dengan meneruskan ID grup Maven dengan `--namespace` opsi dan Maven ArtifactID dengan opsi `--name`. Sebagai contoh, untuk menyalin satu versi `com.google.guava:guava`:

```
aws codeartifact copy-package-versions --domain my_domain --domain-owner 111122223333
\
--source-repository my_repo --destination-repository repo-2 --format maven --
namespace com.google.guava \
--package guava --versions 27.1-jre
```



Jika versi paket berhasil disalin, output akan serupa dengan berikut ini.

```
{
  "successfulVersions": {
    "27.1-jre": {
      "revision": "REVISION-1-SAMPLE-6C81EFF7DA55CC",
      "status": "Published"
    }
  },
  "failedVersions": {}
}
```

## Versi yang tidak ada dalam repositori sumber

Jika Anda menentukan versi yang tidak ada di repositori sumber, penyalinan akan gagal. Jika beberapa versi ada di repositori sumber dan beberapa tidak ada, semua versi akan gagal disalin. Dalam contoh berikut, versi 0.2.0 paket npm `array-unique` ada dalam repositori sumber, tetapi versi 5.6.7 tidak:

```
aws codeartifact copy-package-versions --domain my_domain --domain-owner 111122223333 \
  --source-repository my_repo --destination-repository repo-2 --format npm \
  --package array-unique --versions 0.2.0 5.6.7
```

Output dalam skenario ini akan serupa dengan berikut ini.

```
{
  "successfulVersions": {},
  "failedVersions": {
    "0.2.0": {
      "errorCode": "SKIPPED",
      "errorMessage": "Version 0.2.0 was skipped"
    },
    "5.6.7": {
      "errorCode": "NOT_FOUND",
      "errorMessage": "Could not find version 5.6.7"
    }
  }
}
```

Kode kesalahan `SKIPPED` digunakan untuk menunjukkan bahwa versi tidak disalin ke repositori tujuan karena versi lain tidak dapat disalin.

## Versi yang sudah ada di repositori tujuan

Ketika versi paket disalin ke repositori yang sudah ada, CodeArtifact bandingkan aset paket dan metadata tingkat versi paketnya di dua repositori.

Jika aset dan metadata versi paket identik dalam repositori sumber dan tujuan, penyalinan tidak dilakukan tetapi operasi dianggap berhasil. Ini berarti bahwa `copy-package-versions` idempoten. Ketika ini terjadi, versi yang sudah ada di repositori sumber dan tujuan tidak akan tercantum dalam output `copy-package-versions`.

Dalam contoh berikut, dua versi paket npm `array-unique` ada dalam repositori sumber `repo-1`. Versi 0.2.1 juga ada dalam repositori tujuan `dest-repo` dan versi 0.2.0 tidak.

```
aws codeartifact copy-package-versions --domain my_domain --domain-owner 111122223333 \  
  --source-repository my_repo --destination-repository repo-2 --format npm --  
package array-unique \  
  --versions 0.2.1 0.2.0
```

Output dalam skenario ini akan serupa dengan berikut ini.

```
{  
  "successfulVersions": {  
    "0.2.0": {  
      "revision": "Yad+B1QcBq2kdEVrx1E1vSfHJVh8Pr61hBUkoWPGWX0=",  
      "status": "Published"  
    }  
  },  
  "failedVersions": {}  
}
```

Versi 0.2.0 tercantum dalam `successfulVersions` karena berhasil disalin dari sumber ke repositori tujuan. Versi 0.2.1 tidak ditampilkan dalam output karena sudah ada di repositori tujuan.

Jika aset atau metadata versi paket berbeda di repositori sumber dan tujuan, operasi penyalinan akan gagal. Anda dapat menggunakan parameter `--allow-overwrite` untuk menimpa dengan paksa.

Jika beberapa versi ada di repositori tujuan dan beberapa tidak, semua versi akan gagal disalin. Dalam contoh berikut, versi 0.3.2 paket npm `array-unique` ada di repositori sumber dan tujuan, tetapi isi versi paket berbeda. Versi 0.2.1 ada dalam repositori sumber tetapi tidak di repositori tujuan.

```
aws codeartifact copy-package-versions --domain my_domain --domain-owner 111122223333 \  
  --source-repository my_repo --destination-repository repo-2 --format npm --  
package array-unique \  
  --versions 0.3.2 0.2.1
```

Output dalam skenario ini akan serupa dengan berikut ini.

```
{  
  "successfulVersions": {},  
  "failedVersions": {  
    "0.2.1": {  
      "errorCode": "SKIPPED",  
      "errorMessage": "Version 0.2.1 was skipped"  
    },  
    "0.3.2": {  
      "errorCode": "ALREADY_EXISTS",  
      "errorMessage": "Version 0.3.2 already exists"  
    }  
  }  
}
```

Versi 0.2.1 ditandai sebagai SKIPPED karena tidak disalin ke repositori tujuan. Versi tidak disalin karena salinan versi 0.3.2 gagal karena sudah ada di repositori tujuan, tetapi tidak identik dalam repositori sumber dan tujuan.

## Menentukan revisi versi paket

Revisi versi paket adalah string yang menentukan serangkaian aset dan metadata tertentu untuk versi paket. Anda dapat menentukan revisi versi paket untuk menyalin versi paket yang berada dalam status tertentu. Untuk menentukan revisi versi paket, gunakan parameter `--version-revisions` untuk meneruskan satu atau lebih versi paket yang dipisahkan koma dan pasangan revisi versi paket ke perintah `copy-package-versions`.

### Note

Anda harus menentukan parameter `--versions` atau `--version-revisions` dengan `copy-package-versions`. Anda tidak dapat menentukan keduanya.

Contoh berikut hanya akan menyalin versi 0.3.2 dari paket `my-package` jika ada di repositori sumber dengan revisi versi paket `REVISION-1-SAMPLE-6C81EFF7DA55CC`.

```
aws codeartifact copy-package-versions --domain my_domain --domain-owner 111122223333
--source-repository repo-1 \
--destination-repository repo-2 --format npm --namespace my-namespace \
--package my-package --version-revisions 0.3.2=REVISION-1-SAMPLE-6C81EFF7DA55CC
```

Contoh berikut menyalin dua versi paket `my-package`, 0.3.2 dan 0.3.13. Salinan hanya akan berhasil jika di repositori sumber, versi 0.3.2 `my-package` mempunyai revisi `REVISION-1-SAMPLE-6C81EFF7DA55CC` dan versi 0.3.13 mempunyai revisi `REVISION-2-SAMPLE-55C752BEE772FC`.

```
aws codeartifact copy-package-versions --domain my_domain --domain-owner 111122223333
--source-repository repo-1 \
--destination-repository repo-2 --format npm --namespace my-namespace \
--package my-package --version-revisions 0.3.2=REVISION-1-SAMPLE-6C81EFF7DA55CC,0.3.13=REVISION-2-SAMPLE-55C752BEE772FC
```

Untuk menemukan revisi versi paket, gunakan perintah `describe-package-version` atau `list-package-versions`.

Untuk informasi selengkapnya, lihat [Revisi versi paket](#) dan [CopyPackageVersion](#) di Referensi CodeArtifact API.

## Menyalin paket npm

Untuk informasi selengkapnya tentang `copy-package-versions` perilaku dengan paket npm, lihat [tag npm](#) dan [API. CopyPackageVersions](#)

## Menghapus paket

Anda dapat menghapus satu atau beberapa versi paket sekaligus menggunakan perintah `delete-package-versions`. Untuk menghapus paket dari repositori sepenuhnya, termasuk semua versi dan konfigurasi terkait, gunakan perintah `delete-package`. Sebuah paket dapat ada di repositori tanpa versi paket apa pun. Ini dapat terjadi ketika semua versi dihapus menggunakan `delete-package-versions` perintah, atau jika paket dibuat tanpa versi apa pun yang menggunakan operasi `put-package-origin-configuration` API (lihat [Mengedit kontrol asal paket](#)).

## Topik

- [Menghapus paket \(\)AWS CLI](#)
- [Menghapus versi paket \(\)AWS CLI](#)
- [Menghapus paket \(konsol\)](#)
- [Menghapus versi paket \(konsol\)](#)
- [Menghapus paket npm](#)
- [Menghapus paket Maven](#)

## Menghapus paket ()AWS CLI

Anda dapat menghapus paket, termasuk semua versi paket dan konfigurasi, menggunakan `delete-package` perintah. Contoh berikut menghapus paket PyPI yang `my-package` dinamai dalam `my_repo` repo di domain: `my_domain`

```
aws codeartifact delete-package --domain my_domain --domain-owner 111122223333 \  
--repository my_repo --format pypi \  
--package my-package
```

Contoh output:

```
{  
  "deletedPackage": {  
    "format": "pypi",  
    "originConfiguration": {  
      "restrictions": {  
        "publish": "ALLOW",  
        "upstream": "BLOCK"  
      }  
    },  
    "package": "my-package"  
  }  
}
```

Anda dapat mengonfirmasi bahwa paket telah dihapus dengan menjalankan `describe-package` nama paket yang sama:

```
aws codeartifact describe-package --domain my_domain --domain-owner 111122223333 \  

```

```
--repository my_repo --format pypi --package my-package
```

## Menghapus versi paket (AWS CLI)

Anda dapat menghapus satu atau beberapa versi paket sekaligus menggunakan perintah `delete-package-versions`. Contoh berikut menghapus versi `4.0.0`, `4.0.1`, dan `5.0.0` paket PyPI `my-package` bernama dalam `my_repo` `my_domain` domain:

```
aws codeartifact delete-package-versions --domain my_domain --domain-owner 111122223333 \  
 \  
--repository my_repo --format pypi \  
--package my-package --versions 4.0.0 4.0.1 5.0.0
```

Contoh output:

```
{  
  "successfulVersions": {  
    "4.0.0": {  
      "revision": "oxwwYC9dDeuBoCt6+PDSwL60MZ7rXeIXy44BM32Iawo=",  
      "status": "Deleted"  
    },  
    "4.0.1": {  
      "revision": "byaaQR748wrsdBaT+PDSwL60MZ7rXeIBKM0551aqWmo=",  
      "status": "Deleted"  
    },  
    "5.0.0": {  
      "revision": "yubm34QWeST345ts+ASeioPI354rXeISWr734PotwRw=",  
      "status": "Deleted"  
    }  
  },  
  "failedVersions": {}  
}
```

Anda dapat mengonfirmasi bahwa versi telah dihapus dengan menjalankan `list-package-versions` untuk nama paket yang sama:

```
aws codeartifact list-package-versions --domain my_domain --domain-owner 111122223333 \  
--repository my_repo --format pypi --package my-package
```

## Menghapus paket (konsol)

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Di panel navigasi, pilih Repositori.
3. Pilih Repositori dari mana Anda ingin menghapus paket.
4. Pilih Package yang ingin Anda hapus.
5. Pilih Hapus Package.

## Menghapus versi paket (konsol)

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Di panel navigasi, pilih Repositori.
3. Pilih Repositori dari mana Anda ingin menghapus versi paket.
4. Pilih Package dari mana Anda ingin menghapus versi.
5. Pilih Package Version yang ingin Anda hapus.
6. Pilih Hapus.

### Note

Di konsol, Anda hanya dapat menghapus satu versi paket sekaligus. Untuk menghapus lebih dari satu per satu, gunakan CLI.

## Menghapus paket npm

Untuk menghapus paket npm atau versi paket individual, atur `--format` opsi `npm`. Untuk menghapus versi paket dalam paket npm cakupan, gunakan `--namespace` opsi untuk menentukan ruang lingkup. Misalnya, untuk menghapus paket@types/react, gunakan `--namespace types`. Hilangkan `@` simbol saat menggunakan `--namespace`.

```
aws codeartifact delete-package-versions --domain my_domain --domain-owner 111122223333 \
--repository my_repo --format npm --namespace types \
--package react --versions 0.12.2
```

Untuk menghapus paket@types/react, termasuk semua versinya:

```
aws codeartifact delete-package --domain my_domain --domain-owner 111122223333 \  
--repository my_repo --format npm --namespace types \  
--package react
```

## Menghapus paket Maven

Untuk menghapus paket Maven atau versi paket individual, atur `--format` opsi ke maven dan tentukan paket yang akan dihapus dengan meneruskan ID grup Maven dengan `--namespace` opsi dan Maven ArtifactID dengan opsi. `--name` Misalnya, berikut ini menunjukkan cara menghapus satu `versicom.google.guava:guava`:

```
aws codeartifact delete-package-versions --domain my_domain --domain-  
owner 111122223333 \  
--repository my_repo --format maven --namespace com.google.guava \  
--package guava --versions 27.1-jre
```

Contoh berikut menunjukkan cara menghapus paket `com.google.guava:guava`, termasuk semua versinya:

```
aws codeartifact delete-package --domain my_domain --domain-owner 111122223333 \  
--repository my_repo --format maven --namespace com.google.guava \  
--package guava
```

## Melihat dan memperbarui detail versi paket dan dependensi

Anda dapat melihat informasi tentang versi paket, termasuk dependensi, di CodeArtifact Anda juga dapat memperbarui status versi paket. Untuk informasi selengkapnya tentang status versi paket, lihat [Status versi paket](#).

### Melihat detail versi paket

Gunakan perintah `describe-package-version` untuk melihat detail tentang versi paket. Rincian versi Package diekstraksi dari paket saat dipublikasikan ke CodeArtifact. Detail dalam paket yang berbeda bervariasi dan bergantung pada format dan berapa banyak informasi yang ditambahkan penulis ke dalamnya.

Sebagian besar informasi dalam output perintah `describe-package-version` bergantung pada format paket. Misalnya, `describe-package-version` mengekstrak informasi paket npm dari file



package.json. Revisi dibuat oleh CodeArtifact. Untuk informasi selengkapnya, lihat [Menentukan revisi versi paket](#).

Dua versi paket dengan nama yang sama bisa berada di repositori yang sama jika masing-masing berada di namespace yang berbeda. Gunakan parameter `--namespace` opsional untuk menentukan namespace. Untuk informasi selengkapnya, lihat [Lihat detail versi paket npm](#) atau [Lihat detail versi paket Maven](#).

Contoh berikut mengembalikan detail tentang versi `1.9.0` paket Python bernama `pyhamcrest` yang ada di repositori `my_repo`.

```
aws codeartifact describe-package-version --domain my_domain --domain-owner 111122223333 --repository my_repo \
--format pypi --package pyhamcrest --package-version 1.9.0
```

Output mungkin terlihat seperti berikut ini.

```
{
  "format": "pypi",
  "package": "PyHamcrest",
  "displayName": "PyHamcrest",
  "version": "1.9.0",
  "summary": "Hamcrest framework for matcher objects",
  "homePage": "https://github.com/hamcrest/PyHamcrest",
  "publishedTime": 1566002944.273,
  "licenses": [
    {
      "id": "license-id",
      "name": "license-name"
    }
  ],
  "revision": "REVISION-SAMPLE-55C752BEE9B772FC"
}
```

## Lihat detail versi paket npm

Untuk melihat detail tentang versi paket npm, tetapkan nilai `--format` opsi `kenpm`. Secara opsional, sertakan namespace versi paket (lingkup npm) dalam opsi `--namespace`. Nilai untuk opsi `--namespace` tidak boleh mencakup `@` di awal. Untuk mencari namespace `@types`, tetapkan nilai ke *jenis*.

Berikut ini mengembalikan rincian tentang 4.41.5 versi paket npm bernama webpack dalam @types lingkup.

```
aws codeartifact describe-package-version --domain my_domain --domain-owner 111122223333 --repository my_repo \
--format npm --package webpack --namespace types --package-version 4.41.5
```

Output mungkin terlihat seperti berikut ini.

```
{
  "format": "npm",
  "namespace": "types",
  "package": "webpack",
  "displayName": "webpack",
  "version": "4.41.5",
  "summary": "Packs CommonJs/AMD modules for the browser. Allows ... further output omitted for brevity",
  "homePage": "https://github.com/webpack/webpack",
  "sourceCodeRepository": "https://github.com/webpack/webpack.git",
  "publishedTime": 1577481261.09,
  "licenses": [
    {
      "id": "license-id",
      "name": "license-name"
    }
  ],
  "revision": "REVISION-SAMPLE-55C752BEE9B772FC",
  "status": "Published",
  "origin": {
    "domainEntryPoint": {
      "externalConnectionName": "public:npmjs"
    },
    "originType": "EXTERNAL"
  }
}
```

## Lihat detail versi paket Maven

Untuk melihat detail tentang versi paket Maven, tetapkan nilai `--format` opsi ke maven dan sertakan namespace versi paket dalam opsi. `--namespace`

Contoh berikut mengembalikan detail tentang versi 1.2 paket Maven bernama `commons-rng-client-api` yang ada di namespace `org.apache.commons` dan repositori `my_repo`.

```
aws codeartifact describe-package-version --domain my_domain --domain-owner 111122223333 --repository my_repo \
--format maven --namespace org.apache.commons --package commons-rng-client-api --package-version 1.2
```

Output mungkin terlihat seperti berikut ini.

```
{
  "format": "maven",
  "namespace": "org.apache.commons",
  "package": "commons-rng-client-api",
  "displayName": "Apache Commons RNG Client API",
  "version": "1.2",
  "summary": "API for client code that uses random numbers generators.",
  "publishedTime": 1567920624.849,
  "licenses": [],
  "revision": "REVISION-SAMPLE-55C752BEE9B772FC"
}
```

#### Note

CodeArtifact tidak mengekstrak informasi detail versi paket dari file POM induk. Metadata untuk versi paket tertentu hanya akan menyertakan informasi dalam POM untuk versi paket yang tepat, bukan untuk POM induk atau POM lain yang direferensikan secara transitif menggunakan tag POM. `parent` Ini berarti bahwa output `describe-package-version` akan menghilangkan metadata (seperti informasi lisensi) untuk versi paket Maven yang mengandalkan `parent` referensi untuk berisi metadata ini.

## Melihat dependensi versi paket

Gunakan perintah `list-package-version-dependencies` untuk mendapatkan daftar dependensi versi paket. Perintah berikut mencantumkan dependensi paket npm bernama `my-package`, versi `4.41.5`, di repositori `my_repo`, dalam domain `my_domain`.

```
aws codeartifact list-package-version-dependencies --domain my_domain --domain-owner 111122223333 --repository my_repo \
```

```
--format npm --package my-package --package-version 4.41.5
```

Output mungkin terlihat seperti berikut ini.

```
{
  "dependencies": [
    {
      "namespace": "webassemblyjs",
      "package": "ast",
      "dependencyType": "regular",
      "versionRequirement": "1.8.5"
    },
    {
      "namespace": "webassemblyjs",
      "package": "helper-module-context",
      "dependencyType": "regular",
      "versionRequirement": "1.8.5"
    },
    {
      "namespace": "webassemblyjs",
      "package": "wasm-edit",
      "dependencyType": "regular",
      "versionRequirement": "1.8.5"
    }
  ],
  "versionRevision": "REVISION-SAMPLE-55C752BEE9B772FC"
}
```

Untuk rentang nilai yang didukung untuk bidang `DependencyType`, lihat tipe [PackageDependency](#) data di API. CodeArtifact

## Melihat file readme versi paket

Beberapa format paket, seperti npm, mencakup file README. Gunakan `get-package-version-readme` untuk mendapatkan file README versi paket. Perintah berikut mengembalikan file README paket npm bernama `my-package`, versi `4.41.5`, di repositori `my_repo`, dalam domain `my_domain`.

### Note

CodeArtifact tidak mendukung menampilkan file readme dari paket generik atau Maven.

```
aws codeartifact get-package-version-readme --domain my_domain --domain-owner 111122223333 --repository my_repo \
--format npm --package my-package --package-version 4.41.5
```

Output mungkin terlihat seperti berikut ini.

```
{
  "format": "npm",
  "package": "my-package",
  "version": "4.41.5"
  "readme": "<div align=\"center\">\n  <a href=\"https://github.com/webpack/webpack\n\"> ... more content ... \n",
  "versionRevision": "REVISION-SAMPLE-55C752BEE9B772FC"
}
```

## Memperbarui status versi paket

Setiap versi paket CodeArtifact memiliki status yang menjelaskan status saat ini dan ketersediaan versi paket. Anda dapat mengubah status versi paket menggunakan konsol AWS CLI dan konsol.

### Note

Untuk informasi selengkapnya tentang status versi paket, termasuk daftar status yang tersedia, lihat [Status versi paket](#).

## Memperbarui status versi paket

Mengatur status versi paket memungkinkan mengontrol bagaimana versi paket dapat digunakan tanpa menghapusnya sepenuhnya dari repositori. Misalnya, ketika versi paket memiliki status `Unlisted`, itu masih dapat diunduh seperti biasa, tetapi tidak akan muncul dalam daftar versi paket yang dikembalikan ke perintah seperti `npm view`. [UpdatePackageVersionsStatus API](#) memungkinkan pengaturan status versi paket dari beberapa versi paket yang sama dalam satu panggilan API. Untuk deskripsi status yang berbeda, lihat [Gambaran umum paket](#).

Gunakan `update-package-versions-status` perintah untuk mengubah status versi paket menjadi `Published`, `Unlisted`, atau `Archived`. Untuk melihat izin IAM yang diperlukan untuk menggunakan perintah, lihat [Izin IAM yang diperlukan untuk memperbarui status versi paket](#) Contoh berikut menetapkan status versi 4.1.0 dari paket `chalk` npm ke `Archived`

```
aws codeartifact update-package-versions-status --domain my_domain
--domain-owner 111122223333 --repository my_repo --format npm --package chalk
--versions 4.1.0 --target-status Archived
```

Contoh output:

```
{
  "successfulVersions": {
    "4.1.0": {
      "revision": "+0z8skWbwY3k8M6SrNIqNj6bVH/ax+CxvkJx+No5j8I=",
      "status": "Archived"
    }
  },
  "failedVersions": {}
}
```

Contoh ini menggunakan paket npm, tetapi perintah bekerja identik untuk format lain. Beberapa versi dapat dipindahkan ke status target yang sama menggunakan satu perintah, lihat contoh berikut.

```
aws codeartifact update-package-versions-status --domain my_domain
--domain-owner 111122223333 --repository my_repo --format npm --package chalk
--versions 4.1.0 4.1.1 --target-status Archived
```

Contoh output:

```
{
  "successfulVersions": {
    "4.1.0": {
      "revision": "25/UjBleHs1DZewk+zozoeqH/R80Rc9gL1P8vbzVMJ4=",
      "status": "Archived"
    },
    "4.1.1": {
      "revision": "+0z8skWbwY3k8M6SrNIqNj6bVH/ax+CxvkJx+No5j8I=",
      "status": "Archived"
    }
  },
  "failedVersions": {}
}
```

Perhatikan bahwa setelah diterbitkan, versi paket tidak dapat dipindahkan kembali ke Unfinished status, sehingga status ini tidak diizinkan sebagai nilai untuk `--target-status` parameter. Untuk

memindahkan versi paket ke Disposed status, gunakan `dispose-package-versions` perintah sebagai gantinya seperti yang dijelaskan di bawah ini.

## Izin IAM yang diperlukan untuk memperbarui status versi paket

`update-package-versions-status` Untuk memanggil paket, Anda harus memiliki `codeartifact:UpdatePackageVersionsStatus` izin pada sumber daya paket. Ini berarti Anda dapat memberikan izin untuk menelepon `update-package-versions-status` berdasarkan per paket. Misalnya, kebijakan IAM yang memberikan izin untuk memanggil `update-package-versions-status` *kapur* paket npm akan menyertakan pernyataan seperti berikut ini.

```
{
  "Action": [
    "codeartifact:UpdatePackageVersionsStatus"
  ],
  "Effect": "Allow",
  "Resource": "arn:aws:codeartifact:us-east-1:111122223333:package/my_domain/my_repo/
npm//chalk"
}
```

## Memperbarui status untuk paket npm cakupan

Untuk memperbarui status versi paket dari versi paket npm dengan cakupan, gunakan `--namespace` parameter. Misalnya, untuk membatalkan daftar versi 8.0.0 dari `@nestjs/core`, gunakan perintah berikut.

```
aws codeartifact update-package-versions-status --domain my_domain
--domain-owner 111122223333 --repository my_repo --format npm --namespace nestjs
--package core --versions 8.0.0 --target-status Unlisted
```

## Memperbarui status untuk paket Maven

Paket Maven selalu memiliki ID grup, yang disebut sebagai namespace di CodeArtifact. Gunakan `--namespace` parameter untuk menentukan ID grup Maven saat memanggil `update-package-versions-status`. Misalnya, untuk mengarsipkan versi 2.13.1 dari paket `Mavenorg.apache.logging.log4j:log4j`, gunakan perintah berikut.

```
aws codeartifact update-package-versions-status --domain my_domain
--domain-owner 111122223333 --repository my_repo --format maven
```

```
--namespace org.apache.logging.log4j --package log4j
--versions 2.13.1 --target-status Archived
```

## Menentukan revisi versi paket

Revisi versi paket adalah string yang menentukan serangkaian aset dan metadata tertentu untuk versi paket. Anda dapat menentukan revisi versi paket untuk memperbarui status versi paket yang berada dalam keadaan tertentu. Untuk menentukan revisi versi paket, gunakan `--version-revisions` parameter untuk meneruskan satu atau beberapa versi paket yang dipisahkan koma dan pasangan revisi versi paket. Status versi paket hanya akan diperbarui jika revisi versi paket saat ini cocok dengan nilai yang ditentukan.

### Note

`--versions` parameter juga harus ditentukan saat menggunakan `--version-revisions` parameter.

```
aws codeartifact update-package-versions-status --domain my_domain
--domain-owner 111122223333 --repository my_repo --format npm --package chalk
--version-revisions "4.1.0=25/UjBleHs1DZewk+zozoeqH/R80Rc9gL1P8bzVMJ4="
--versions 4.1.0 --target-status Archived
```

Untuk memperbarui beberapa versi dengan satu perintah, teruskan daftar pasangan revisi versi dan versi yang dipisahkan koma ke opsi `--version-revisions`. Contoh perintah berikut mendefinisikan dua versi paket yang berbeda dan pasangan revisi versi paket.

```
aws codeartifact update-package-versions-status --domain my_domain
--domain-owner 111122223333 --repository my_repo --format npm
--package chalk
--version-revisions "4.1.0=25/UjBleHs1DZewk+zozoeqH/R80Rc9gL1P8vbzVMJ4=,4.0.0=E3lhBp0R0bRTut4pkjV5c1AQGkgSA70xtil6hMMzelc="
--versions 4.1.0 4.0.0 --target-status Published
```

Contoh output:

```
{
  "successfulVersions": {
    "4.0.0": {
```



```

    "revision": "E31hBp0R0bRTut4pkjV5c1AQGkgSA70xti16hMMzelc=",
    "status": "Published"
  },
  "4.1.0": {
    "revision": "25/UjBleHs1DZewk+zozoeqH/R80Rc9gL1P8vbzVMJ4=",
    "status": "Published"
  }
},
"failedVersions": {}
}

```

Saat memperbarui beberapa versi paket, versi yang diteruskan `--version-revisions` harus sama dengan versi yang diteruskan `--versions`. Jika revisi ditentukan secara tidak benar, versi itu tidak akan diperbarui statusnya.

## Menggunakan parameter status yang diharapkan

`update-package-versions-status` Perintah menyediakan `--expected-status` parameter yang mendukung menentukan status saat ini yang diharapkan dari versi paket. Jika status saat ini tidak sesuai dengan nilai yang diteruskan `--expected-status`, status versi paket tersebut tidak akan diperbarui.

Misalnya, di `my_repo`, versi 4.0.0 dan 4.1.0 dari paket npm saat ini memiliki status `chalk Published`. Panggilan `update-package-versions-status` yang menentukan status yang diharapkan `Unlisted` akan gagal memperbarui kedua versi paket karena ketidakcocokan status.

```

aws codeartifact update-package-versions-status --domain my_domain
--domain-owner 111122223333 --repository my_repo --format npm --package chalk
--versions 4.1.0 4.0.0 --target-status Archived --expected-status Unlisted

```

Contoh output:

```

{
  "successfulVersions": {},
  "failedVersions": {
    "4.0.0": {
      "errorCode": "MISMATCHED_STATUS",
      "errorMessage": "current status: Published, expected status: Unlisted"
    },
    "4.1.0": {
      "errorCode": "MISMATCHED_STATUS",

```

```

      "errorMessage": "current status: Published, expected status: Unlisted"
    }
  }
}

```

## Kesalahan dengan versi paket individual

Ada beberapa alasan mengapa status versi paket tidak akan diperbarui saat menelepon `update-package-versions-status`. Misalnya, revisi versi paket mungkin telah ditentukan secara tidak benar, atau status yang diharapkan tidak cocok dengan status saat ini. Dalam kasus ini, versi akan disertakan dalam `failedVersions` peta dalam respons API. Jika satu versi gagal, versi lain yang ditentukan dalam panggilan yang sama `update-package-versions-status` mungkin dilewati dan statusnya tidak diperbarui. Versi tersebut juga akan dimasukkan dalam `failedVersions` peta dengan `errorCode` `aSKIPPED`.

Dalam implementasi saat ini `update-package-versions-status`, jika satu atau lebih versi tidak dapat diubah statusnya, semua versi lain akan dilewati. Artinya, semua versi berhasil diperbarui atau tidak ada versi yang diperbarui. Perilaku ini tidak dijamin dalam kontrak API; di masa depan, beberapa versi mungkin berhasil sementara versi lain gagal dalam satu panggilan ke `update-package-versions-status`.

Perintah contoh berikut mencakup kegagalan pembaruan status versi yang disebabkan oleh ketidakcocokan revisi versi paket. Kegagalan pembaruan itu menyebabkan panggilan pembaruan status versi lain dilewati.

```

aws codeartifact update-package-versions-status --domain my_domain
--domain-owner 111122223333 --repository my_repo
--format npm --package chalk
--version-revisions "4.1.0=25/UjBleHs1DZewk+zozoeqH/
R80Rc9gL1P8vbzVMJ=,4.0.0=E3lhBp0R0bRTut4pkjV5c1AQGkgSA70xtil6hMMzelc="
--versions 4.1.0 4.0.0 --target-status Archived

```

Contoh output:

```

{
  "successfulVersions": {},
  "failedVersions": {
    "4.0.0": {
      "errorCode": "SKIPPED",
      "errorMessage": "version 4.0.0 is skipped"
    }
  }
}

```

```

    },
    "4.1.0": {
      "errorCode": "MISMATCHED_REVISION",
      "errorMessage": "current revision: 25/UjBleHs1DZewk+zozoeqH/
R80Rc9gL1P8vbzVMJ4=, expected revision: 25/UjBleHs1DZewk+zozoeqH/R80Rc9gL1P8vbzVMJ="
    }
  }
}

```

## Membuang versi paket

Status `Disposed` paket memiliki perilaku yang mirip dengan `Archived`, kecuali bahwa aset paket akan dihapus secara permanen CodeArtifact sehingga akun pemilik domain tidak lagi ditagih untuk penyimpanan aset. Untuk informasi selengkapnya tentang setiap status versi paket, lihat [Status versi paket](#). Untuk mengubah status versi paket ke `Disposed`, gunakan `dispose-package-versions` perintah. Kemampuan ini terpisah dari `update-package-versions-status` karena membuang versi paket tidak dapat dibalik. Karena aset paket akan dihapus, status versi tidak dapat diubah kembali ke `Archived`, `Unlisted`, atau `Published`. Satu-satunya tindakan yang dapat diambil pada versi paket yang telah dibuang adalah untuk menghapusnya menggunakan `delete-package-versions` perintah.

`dispose-package-versions` Agar berhasil menelepon, prinsipal IAM pemanggil harus memiliki `codeartifact:DisposePackageVersions` izin pada sumber daya paket.

Perilaku `dispose-package-versions` perintah mirip dengan `update-package-versions-status`, termasuk perilaku `--version-revisions` dan `--expected-status` opsi yang dijelaskan dalam [revisi versi](#) dan bagian [status yang diharapkan](#). Misalnya, perintah berikut mencoba untuk membuang versi paket tetapi gagal karena status yang diharapkan tidak cocok.

```

aws codeartifact dispose-package-versions --domain my_domain --domain-
owner 111122223333
--repository my_repo --format npm --package chalk --versions 4.0.0
--expected-status Unlisted

```

Contoh output:

```

{
  "successfulVersions": {},
  "failedVersions": {
    "4.0.0": {

```

```
        "errorCode": "MISMATCHED_STATUS",
        "errorMessage": "current status: Published, expected status: Unlisted"
    }
}
```

Jika perintah yang sama dijalankan lagi dengan a `--expected-status ofPublished`, pembuangan akan berhasil.

```
aws codeartifact dispose-package-versions --domain my_domain --domain-
owner 111122223333
--repository my_repo --format npm --package chalk --versions 4.0.0
--expected-status Published
```

Contoh output:

```
{
  "successfulVersions": {
    "4.0.0": {
      "revision": "E31hBp0R0bRTut4pkjV5c1AQGkgSA70xtil6hMMzelc=",
      "status": "Disposed"
    }
  },
  "failedVersions": {}
}
```

## Mengedit kontrol asal paket

Dalam AWS CodeArtifact, versi paket dapat ditambahkan ke repositori dengan menerbitkannya secara langsung, menariknya ke bawah dari repositori hulu, atau melannya dari repositori publik eksternal. Mengizinkan versi paket dari sebuah paket ditambahkan baik dengan penerbitan langsung maupun menelan dari repositori publik membuat Anda rentan terhadap serangan substitusi ketergantungan. Untuk informasi selengkapnya, lihat [Serangan substitusi ketergantungan](#). Untuk melindungi diri Anda dari serangan substitusi dependensi, Anda dapat mengonfigurasi kontrol asal paket pada paket dalam repositori untuk membatasi bagaimana versi paket tersebut dapat ditambahkan ke repositori.

Mengonfigurasi kontrol asal paket harus dipertimbangkan oleh tim mana pun yang ingin mengizinkan versi baru dari paket yang berbeda berasal dari kedua sumber internal, seperti penerbitan langsung, dan sumber eksternal, seperti repositori publik. Secara default, kontrol asal paket akan dikonfigurasi berdasarkan bagaimana versi pertama paket ditambahkan ke repositori. Untuk informasi tentang pengaturan kontrol asal paket dan nilai defaultnya, lihat [Pengaturan kontrol asal paket](#).

Untuk menghapus catatan paket setelah menggunakan operasi `put-package-origin-configuration` API, gunakan `delete-package` (lihat [Menghapus paket](#)).

## Skenario kontrol akses paket umum

Bagian ini mencakup beberapa skenario umum ketika versi paket ditambahkan ke CodeArtifact repositori. Pengaturan kontrol asal paket akan diatur untuk paket baru tergantung pada bagaimana versi paket pertama ditambahkan.

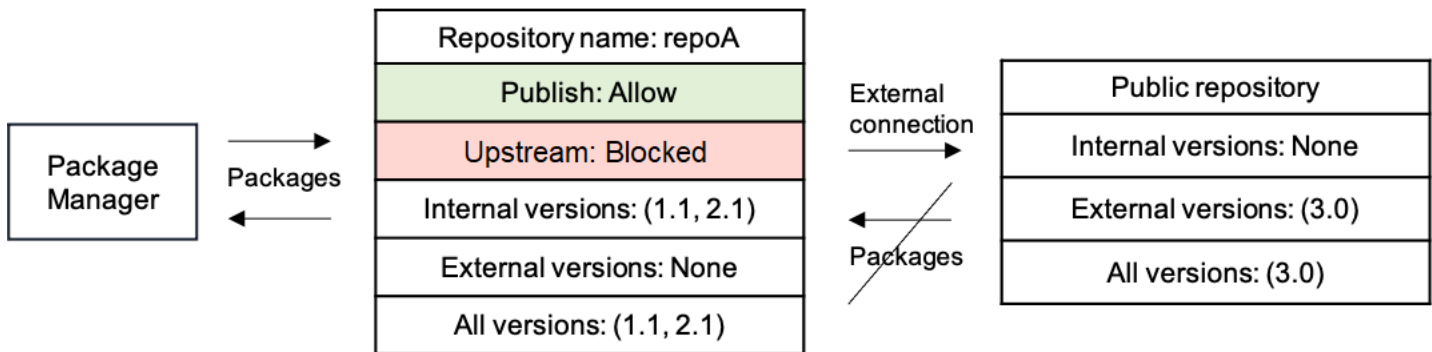
Dalam skenario berikut, paket internal adalah paket yang diterbitkan langsung dari manajer paket ke repositori Anda, seperti paket yang Anda atau penulis tim Anda dan pertahankan. Paket eksternal adalah paket yang ada di repositori publik yang dapat dicerna ke dalam repositori Anda dengan koneksi eksternal.

Versi paket eksternal diterbitkan untuk paket internal yang ada

Dalam skenario ini, pertimbangkan paket internal, PackageA. Tim Anda menerbitkan versi paket pertama untuk PackageA ke repositori. CodeArtifact Karena ini adalah versi paket pertama untuk paket itu, pengaturan kontrol asal paket secara otomatis diatur ke `Publish: Allow and Upstream: Block`. Setelah paket ada di repositori Anda, paket dengan nama yang sama dipublikasikan ke

repositori publik yang terhubung ke repositori Anda. CodeArtifact Ini bisa berupa percobaan serangan substitusi ketergantungan terhadap paket internal, atau bisa juga kebetulan. Terlepas dari itu, kontrol asal paket dikonfigurasi untuk memblokir konsumsi versi eksternal baru untuk melindungi diri dari serangan potensial.

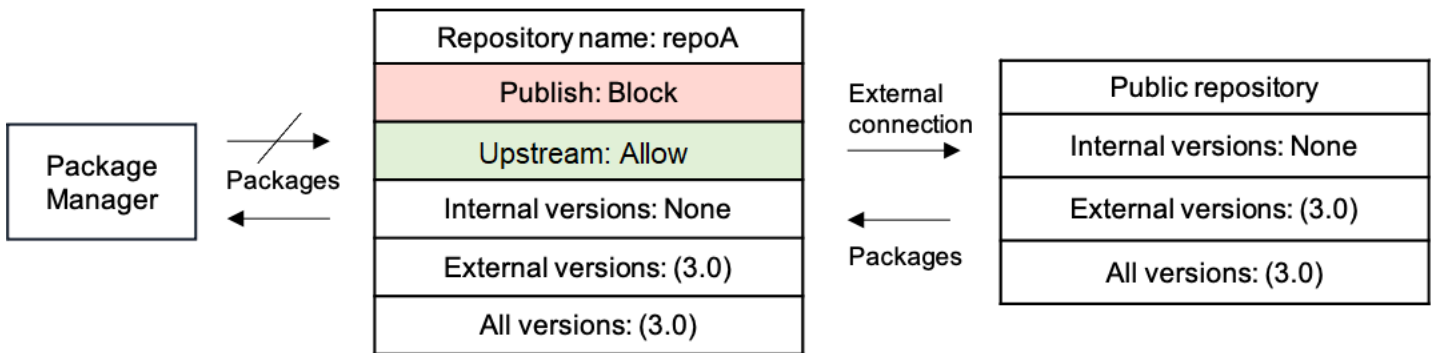
Pada gambar berikut, RePoA adalah CodeArtifact repositori Anda dengan koneksi eksternal ke repositori publik. Repositori Anda berisi versi 1.1 dan 2.1 dari PackageA, tetapi versi 3.0 dipublikasikan ke repositori publik. Biasanya, RepOA akan menelan versi 3.0 setelah paket diminta oleh manajer paket. Karena konsumsi paket diatur ke Blokir, versi 3.0 tidak tertelan ke dalam CodeArtifact repositori Anda dan tidak tersedia untuk manajer paket yang terhubung dengannya.



Versi paket internal diterbitkan untuk paket eksternal yang ada

Dalam skenario ini, sebuah paket, PackageB ada secara eksternal di repositori publik yang telah Anda sambungkan ke repositori Anda. Ketika manajer paket yang terhubung ke repositori Anda meminta PackageB, versi paket diserap ke dalam repositori Anda dari repositori publik. Karena ini adalah versi paket pertama dari PackageB yang ditambahkan ke repositori Anda, pengaturan asal paket dikonfigurasi untuk Publish: BLOCK dan Upstream: ALLOW. Kemudian, Anda mencoba mempublikasikan versi dengan nama paket yang sama ke repositori. Entah Anda tidak mengetahui paket publik dan mencoba menerbitkan paket yang tidak terkait dengan nama yang sama, atau Anda mencoba menerbitkan versi yang ditambal, atau Anda mencoba untuk secara langsung menerbitkan versi paket persis yang sudah ada secara eksternal. CodeArtifact akan menolak versi yang Anda coba terbitkan, tetapi memungkinkan Anda untuk secara eksplisit mengganti penolakan dan mempublikasikan versi jika perlu.

Pada gambar berikut, RePoA adalah CodeArtifact repositori Anda dengan koneksi eksternal ke repositori publik. Repositori Anda berisi versi 3.0 yang dicerna dari repositori publik. Anda ingin mempublikasikan versi 1.1 ke repositori Anda. Biasanya, Anda dapat mempublikasikan versi 1.2 ke RePoA, tetapi karena penerbitan diatur ke Blokir, versi 1.2 tidak dapat dipublikasikan.



Menerbitkan versi paket yang ditambah dari paket eksternal yang ada

Dalam skenario ini, sebuah paket, PackageB ada secara eksternal di repositori publik yang telah Anda sambungkan ke repositori Anda. Ketika manajer paket yang terhubung ke repositori Anda meminta PackageB, versi paket diserap ke dalam repositori Anda dari repositori publik. Karena ini adalah versi paket pertama dari PackageB yang ditambahkan ke repositori Anda, pengaturan asal paket dikonfigurasi untuk **Publish: BLOCK** dan **Upstream: ALLOW**. Tim Anda memutuskan bahwa perlu mempublikasikan versi paket yang ditambah dari paket ini ke repositori. Untuk dapat mempublikasikan versi paket secara langsung, tim Anda mengubah pengaturan kontrol asal paket menjadi **Publish: ALLOW** dan **Upstream: BLOCK**. Versi paket ini sekarang dapat dipublikasikan langsung ke repositori Anda dan dicerna dari repositori publik. Setelah tim Anda menerbitkan versi paket yang ditambah, tim Anda mengembalikan setelan asal paket ke **Publish: BLOCK** dan **Upstream: ALLOW**.

## Pengaturan kontrol asal paket

Dengan kontrol asal paket, Anda dapat mengonfigurasi bagaimana versi paket dapat ditambahkan ke repositori. Daftar berikut mencakup pengaturan dan nilai kontrol asal paket yang tersedia.

### Publikasikan

Pengaturan ini mengonfigurasi apakah versi paket dapat dipublikasikan langsung ke repositori menggunakan manajer paket atau alat serupa.

- **ALLOW:** Versi Package dapat dipublikasikan secara langsung.
- **BLOCK:** Versi Package tidak dapat dipublikasikan secara langsung.

### Hulu

Pengaturan ini mengonfigurasi apakah versi paket dapat dicerna dari eksternal, repositori publik, atau disimpan dari repositori hulu saat diminta oleh manajer paket.

- **ALLOW:** Setiap versi paket dapat dipertahankan dari repositori lain yang dikonfigurasi sebagai CodeArtifact repositori upstream atau dicerna dari sumber publik dengan koneksi eksternal.
- **BLOCK:** Versi Package tidak dapat dipertahankan dari CodeArtifact repositori lain yang dikonfigurasi sebagai repositori upstream atau dicerna dari sumber publik dengan koneksi eksternal.

## Pengaturan kontrol asal paket default

Kontrol asal paket default untuk sebuah paket akan didasarkan pada bagaimana versi pertama paket itu ditambahkan ke repositori.

### Note

Paket yang ada di CodeArtifact repositori sebelum sekitar Mei 2022 akan memiliki kontrol asal paket default Publish: ALLOW dan Upstream: ALLOW. Package origin control harus diatur secara manual untuk paket-paket tersebut. Nilai default saat ini telah ditetapkan pada paket baru sejak saat itu, dan mulai diberlakukan ketika fitur diluncurkan pada 14 Juli 2022. Untuk informasi selengkapnya tentang pengaturan kontrol asal paket, lihat [Mengedit kontrol asal paket](#).

- Jika versi paket pertama diterbitkan secara langsung oleh manajer paket, pengaturannya adalah Publish: ALLOW dan Upstream: BLOCK.
- Jika versi paket pertama dicerna dari sumber publik, pengaturannya adalah Publish: BLOCK dan Upstream: ALLOW.

## Mengedit kontrol asal paket

Kontrol asal paket dikonfigurasi secara otomatis berdasarkan bagaimana versi paket pertama dari sebuah paket ditambahkan ke repositori, untuk informasi selengkapnya lihat. [Pengaturan kontrol asal paket default](#) Untuk menambah atau mengedit kontrol asal paket untuk paket dalam CodeArtifact repositori, lakukan langkah-langkah dalam prosedur berikut.



Untuk menambah atau mengedit kontrol asal paket (konsol)

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Di panel navigasi, pilih Repositori, dan pilih repositori yang berisi paket yang ingin Anda edit.
3. Dalam tabel Paket, cari dan pilih paket yang ingin Anda edit.
4. Dari halaman ringkasan paket, di kontrol Origin, pilih Edit.
5. Di kontrol Edit asal, pilih kontrol asal paket yang ingin Anda atur untuk paket ini. Kedua pengaturan kontrol asal paket, Publish dan Upstream, harus diatur pada saat yang sama.
  - Untuk mengizinkan penerbitan versi paket secara langsung, di Publikasikan, pilih Izinkan. Untuk memblokir penerbitan versi paket, pilih Blokir.
  - Untuk memungkinkan konsumsi paket dari repositori eksternal dan menarik paket dari repositori upstream, di sumber Upstream, pilih Izinkan. Untuk memblokir semua konsumsi dan penarikan versi paket dari repositori eksternal dan upstream, pilih Blokir.

Untuk menambah atau mengedit kontrol asal paket (AWS CLI)

1. Jika belum, konfigurasi AWS CLI dengan mengikuti langkah-langkah di [Mengatur dengan AWS CodeArtifact](#).
2. Gunakan `put-package-origin-configuration` perintah untuk menambah atau mengedit kontrol asal paket. Ganti bidang berikut:
  - Ganti *my\_domain* dengan CodeArtifact domain yang berisi paket yang ingin Anda perbarui.
  - Ganti *my\_repo* dengan CodeArtifact repositori yang berisi paket yang ingin Anda perbarui.
  - Ganti *npm* dengan format paket paket yang ingin Anda perbarui.
  - Ganti *my\_package* dengan nama paket yang ingin Anda perbarui.
  - Ganti *ALLOW* dan *BLOCK* dengan pengaturan kontrol asal paket yang Anda inginkan.

```
aws codeartifact put-package-origin-configuration --domain my_domain \  
--repository my_repo --format npm --package my_package \  
--restrictions publish=ALLOW,upstream=BLOCK
```

## Publikasi dan repositori hulu

CodeArtifact tidak mengizinkan penerbitan versi paket yang ada di repositori upstream yang dapat dijangkau atau repositori publik. Misalnya, anggaplah Anda ingin memublikasikan paket `com.mycompany.mypackage:1.0` ke repositori `myrepo`, dan `myrepo` memiliki repositori hulu dengan koneksi eksternal ke Maven Central. Pertimbangkan skenario berikut.

1. Pengaturan kontrol asal paket aktif `com.mycompany.mypackage` adalah `Publish: ALLOW` dan `Upstream: ALLOW`. Jika `com.mycompany.mypackage:1.0` ada di repositori hulu atau di Maven Central, CodeArtifact menolak setiap upaya untuk memublikasikannya dengan kesalahan konflik 409. `myrepo` Anda masih dapat memublikasikan versi yang berbeda, seperti `com.mycompany.mypackage:1.1`.
2. Pengaturan kontrol asal paket aktif `com.mycompany.mypackage` adalah `Publish: ALLOW` dan `Upstream: BLOCK`. Anda dapat memublikasikan versi apa pun `com.mycompany.mypackage` ke repositori Anda yang belum ada karena versi paket tidak dapat dijangkau.
3. Pengaturan kontrol asal paket aktif `com.mycompany.mypackage` adalah `Publish: BLOCK` dan `Upstream: ALLOW`. Anda tidak dapat memublikasikan versi paket apa pun langsung ke repositori Anda.

# Bekerja dengan domain di CodeArtifact

CodeArtifact domain membuatnya lebih mudah untuk mengelola beberapa repositori di seluruh organisasi. Anda dapat menggunakan domain untuk menerapkan izin di banyak repositori yang dimiliki oleh akun AWS yang berbeda. Aset disimpan hanya sekali dalam domain, bahkan jika tersedia dari beberapa repositori.

Meskipun Anda dapat memiliki beberapa domain, kami merekomendasikan satu domain produksi yang berisi semua artefak yang dipublikasikan sehingga tim pengembangan Anda dapat menemukan dan berbagi paket. Anda dapat menggunakan domain praproduksi kedua untuk menguji perubahan konfigurasi domain produksi.

Topik ini menjelaskan cara menggunakan CodeArtifact konsol, konsol AWS CLI, dan AWS CloudFormation untuk membuat atau mengonfigurasi CodeArtifact domain.

Topik

- [Gambaran umum domain](#)
- [Membuat domain](#)
- [Menghapus domain](#)
- [Kebijakan domain](#)
- [Menandai domain di CodeArtifact](#)

## Gambaran umum domain

Saat Anda bekerja dengan CodeArtifact, domain berguna untuk hal-hal berikut:

- Penyimpanan deduplikat: Aset hanya perlu disimpan sekali dalam domain, meskipun tersedia dalam 1 atau 1.000 repositori. Itu berarti Anda hanya membayar satu kali untuk penyimpanan.
- Penyalinan cepat: Saat Anda menarik paket dari CodeArtifact repositori upstream ke hilir atau menggunakan [CopyPackageVersions API](#), hanya catatan metadata yang harus diperbarui. Tidak ada aset yang disalin. Hal ini membuat penyiapan repositori baru untuk staging atau pengujian menjadi cepat. Untuk informasi selengkapnya, lihat [Bekerja dengan repositori hulu di CodeArtifact](#).
- Berbagi dengan mudah di seluruh repositori dan tim: Semua aset dan metadata dalam domain dienkripsi dengan satu (kunci KMS). AWS KMS key Anda tidak perlu mengelola kunci untuk setiap repositori atau memberikan akses ke satu kunci untuk beberapa akun.

- Terapkan kebijakan di beberapa repositori: Administrator domain dapat menerapkan kebijakan di seluruh domain. Ini termasuk membatasi akun mana yang memiliki akses ke repositori di domain, dan siapa yang dapat mengonfigurasi koneksi ke repositori publik untuk digunakan sebagai sumber paket. Untuk informasi selengkapnya tentang kebijakan, lihat [Kebijakan domain](#).
- Nama repositori unik: Domain menyediakan namespace untuk repositori. Nama repositori hanya perlu unik dalam domain. Anda harus menggunakan nama bermakna yang mudah dipahami.

Nama domain harus unik dalam akun.

Anda tidak dapat membuat repositori tanpa domain. Saat Anda menggunakan [CreateRepository](#) API untuk membuat repositori, Anda harus menentukan nama domain. Anda tidak dapat memindahkan repositori dari satu domain ke domain lainnya.

Repositori dapat dimiliki oleh AWS akun yang sama yang memiliki domain, atau akun yang berbeda. Jika akun pemilik berbeda, akun pemilik repositori harus diberikan izin `CreateRepository` di sumber daya domain. Anda dapat melakukan ini dengan menambahkan kebijakan sumber daya ke domain menggunakan [PutDomainPermissionsPolicy](#) perintah.

Meskipun sebuah organisasi dapat memiliki beberapa domain, direkomendasikan untuk memiliki domain produksi tunggal yang berisi semua artefak yang dipublikasikan sehingga tim pengembangan dapat menemukan dan berbagi paket di seluruh organisasi mereka. Domain pra-produksi kedua dapat bermanfaat untuk menguji perubahan konfigurasi domain produksi.

## Domain lintas akun

Nama domain hanya perlu unik dalam akun, yang berarti mungkin ada beberapa domain dalam wilayah yang memiliki nama yang sama. Karena hal ini, jika ingin mengakses domain yang dimiliki oleh akun yang tidak Anda miliki autentikasinya, Anda harus memberikan ID pemilik domain bersama dengan nama domain di CLI dan konsol. Lihat contoh CLI berikut.

Akses domain yang dimiliki oleh akun yang Anda autentikasi:

Saat mengakses domain dalam akun yang Anda miliki autentikasinya, Anda hanya perlu menentukan nama domain. Contoh berikut mencantumkan paket di repositori *my\_repo* di domain *my\_domain* yang dimiliki oleh akun Anda.

```
aws codeartifact list-packages --domain my_domain --repository my_repo
```

Akses domain yang dimiliki oleh akun yang tidak Anda autentikasi:

Saat mengakses domain yang dimiliki oleh akun yang tidak Anda miliki autentikasinya, Anda harus menentukan pemilik domain serta nama domain. Contoh berikut mencantumkan paket di repositori *other-repo* di domain *other-domain* yang dimiliki oleh akun yang tidak Anda miliki autentikasinya. Perhatikan penambahan parameter `--domain-owner`.

```
aws codeartifact list-packages --domain other-domain --domain-owner 111122223333 --  
repository other-repo
```

## Jenis AWS KMS kunci yang didukung di CodeArtifact

CodeArtifact hanya mendukung tombol [KMS simetris](#). Anda tidak dapat menggunakan [kunci KMS asimetris](#) untuk mengenkripsi domain Anda. CodeArtifact Untuk informasi selengkapnya, lihat [Mengidentifikasi kunci KMS simetris dan asimetris](#). Untuk mempelajari cara membuat kunci terkelola pelanggan baru, lihat [Membuat kunci KMS enkripsi simetris di Panduan AWS Key Management Service](#) Pengembang.

CodeArtifact mendukung Toko Kunci AWS KMS Eksternal (XKS). Anda bertanggung jawab atas ketersediaan, daya tahan, dan latensi operasi kunci dengan kunci XKS, yang dapat memengaruhi ketersediaan, daya tahan, dan latensi dengan. CodeArtifact Beberapa contoh efek menggunakan kunci XKS dengan CodeArtifact:

- Karena setiap aset dari paket yang diminta dan semua dependensinya tunduk pada latensi dekripsi, latensi build dapat ditingkatkan secara substansional dengan peningkatan latensi operasi XKS.
- Karena semua aset dienkripsi CodeArtifact, hilangnya materi kunci XKS akan mengakibatkan hilangnya semua aset yang terkait dengan domain menggunakan kunci XKS.

Untuk informasi selengkapnya tentang kunci XKS, lihat [Penyimpanan kunci eksternal](#) di Panduan AWS Key Management Service Pengembang.

## Membuat domain

Anda dapat membuat domain menggunakan CodeArtifact konsol, AWS Command Line Interface (AWS CLI), atau AWS CloudFormation. Ketika Anda membuat domain, domain tidak berisi repositori apa pun. Untuk informasi selengkapnya, lihat [Buatlah sebuah repositori](#). Untuk informasi selengkapnya tentang mengelola CodeArtifact domain dengan CloudFormation, lihat [Menciptakan CodeArtifact sumber daya dengan AWS CloudFormation](#).

## Topik

- [Membuat domain \(konsol\)](#)
- [Membuat domain \(AWS CLI\)](#)

## Membuat domain (konsol)

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Pada panel navigasi, pilih Domains (Domain), lalu pilih Create domain (Buat domain).
3. Dalam Name (Nama), masukkan nama untuk domain Anda.
4. Perluas Additional configuration (Konfigurasi tambahan).
5. Gunakan AWS KMS key (kunci KMS) untuk mengenkripsi semua aset di domain Anda. Anda dapat menggunakan kunci KMS AWS terkelola atau kunci KMS yang Anda kelola. Untuk informasi selengkapnya tentang jenis kunci KMS yang didukung CodeArtifact, lihat [Jenis AWS KMS kunci yang didukung di CodeArtifact](#).
  - Pilih kunci terkelola AWS jika Anda ingin menggunakan default Kunci yang dikelola AWS.
  - Pilih Customer managed key jika Anda ingin menggunakan kunci KMS yang Anda kelola. Untuk menggunakan kunci KMS yang Anda kelola, di ARN kunci terkelola Pelanggan, cari dan pilih kunci KMS.

Untuk informasi selengkapnya, lihat [Kunci yang dikelola AWS](#) dan [Kunci terkelola pelanggan](#) di Panduan AWS Key Management Service Pengembang.

6. Pilih Create domain (Buat domain).

## Membuat domain (AWS CLI)

Untuk membuat domain dengan AWS CLI, gunakan `create-domain` perintah. Anda harus menggunakan AWS KMS key (kunci KMS) untuk mengenkripsi semua aset di domain Anda. Anda dapat menggunakan kunci KMS AWS terkelola atau kunci KMS yang Anda kelola. Jika Anda menggunakan kunci KMS AWS terkelola, jangan gunakan `--encryption-key` parameter.

Untuk informasi selengkapnya tentang jenis kunci KMS yang didukung CodeArtifact, lihat [Jenis AWS KMS kunci yang didukung di CodeArtifact](#). Untuk informasi selengkapnya tentang kunci KMS, lihat [Kunci yang dikelola AWS](#) dan [Kunci terkelola pelanggan](#) di Panduan AWS Key Management Service Pengembang.

```
aws codeartifact create-domain --domain my_domain
```

Data berformat JSON muncul dalam output dengan detail tentang domain baru Anda.

```
{
  "domain": {
    "name": "my_domain",
    "owner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:domain/my_domain",
    "status": "Active",
    "encryptionKey": "arn:aws:kms:us-west-2:111122223333:key/your-kms-key",
    "repositoryCount": 0,
    "assetSizeBytes": 0,
    "createdTime": "2020-10-12T16:51:18.039000-04:00"
  }
}
```

Jika Anda menggunakan kunci KMS yang Anda kelola, sertakan Nama Sumber Daya Amazon (ARN) dengan `--encryption-key` parameternya.

```
aws codeartifact create-domain --domain my_domain --encryption-key arn:aws:kms:us-west-2:111122223333:key/your-kms-key
```

Data berformat JSON muncul dalam output dengan detail tentang domain baru Anda.

```
{
  "domain": {
    "name": "my_domain",
    "owner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:domain/my_domain",
    "status": "Active",
    "encryptionKey": "arn:aws:kms:us-west-2:111122223333:key/your-kms-key",
    "repositoryCount": 0,
    "assetSizeBytes": 0,
    "createdTime": "2020-10-12T16:51:18.039000-04:00"
  }
}
```

## Membuat domain dengan tanda

Untuk membuat domain dengan tanda, tambahkan parameter `--tags` ke perintah `create-domain` Anda.

```
aws codeartifact create-domain --domain my_domain --tags key=k1,value=v1  
key=k2,value=v2
```

## Menghapus domain

Anda dapat menghapus domain menggunakan CodeArtifact konsol atau AWS Command Line Interface (AWS CLI).

Topik

- [Pembatasan penghapusan domain](#)
- [Menghapus domain \(konsol\)](#)
- [Menghapus domain \(AWS CLI\)](#)

## Pembatasan penghapusan domain

Biasanya, Anda tidak dapat menghapus domain yang berisi repositori. Sebelum menghapus domain, Anda harus terlebih dahulu menghapus repositori. Untuk informasi selengkapnya, lihat [Hapus sebuah repositori](#).

Namun, jika CodeArtifact tidak lagi memiliki akses ke kunci KMS domain, Anda dapat menghapus domain meskipun masih berisi repositori. Situasi ini akan terjadi jika Anda menghapus kunci KMS domain atau mencabut [hibah KMS](#) yang CodeArtifact digunakan untuk mengakses kunci. Dalam keadaan ini, Anda tidak dapat mengakses repositori di domain atau paket yang disimpan di dalamnya. Pencatatan dan penghapusan repositori juga tidak dimungkinkan ketika tidak CodeArtifact dapat mengakses kunci KMS domain. Untuk alasan ini, penghapusan domain tidak memeriksa apakah domain berisi repositori ketika kunci KMS domain tidak dapat diakses.

### Note

Ketika domain yang masih berisi repositori dihapus, CodeArtifact akan menghapus repositori secara asinkron dalam waktu 15 menit. Setelah domain dihapus, repositori akan tetap



terlihat di CodeArtifact konsol dan dalam output `list-repositories` perintah sampai pembersihan repositori otomatis terjadi.

## Menghapus domain (konsol)

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Di panel navigasi, pilih Domains (Domain), lalu pilih domain yang ingin Anda hapus.
3. Pilih Hapus.

## Menghapus domain (AWS CLI)

Gunakan perintah `delete-domain` untuk menghapus domain.

```
aws codeartifact delete-domain --domain my_domain --domain-owner 111122223333
```

Data berformat JSON muncul dalam output dengan detail tentang domain yang dihapus.

```
{
  "domain": {
    "name": "my_domain",
    "owner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:domain/my_domain",
    "status": "Active",
    "encryptionKey": "arn:aws:kms:us-west-2:111122223333:key/your-kms-key",
    "repositoryCount": 0,
    "assetSizeBytes": 0,
    "createdTime": "2020-10-12T16:51:18.039000-04:00"
  }
}
```

## Kebijakan domain

CodeArtifact mendukung penggunaan izin berbasis sumber daya untuk mengontrol akses. Dengan kebijakan berbasis sumber daya, Anda dapat menentukan siapa yang memiliki akses ke sumber daya tersebut dan tindakan apa yang dapat dilakukan pada sumber daya tersebut. Secara default, hanya akun AWS yang memiliki domain dapat membuat dan mengakses repositori dalam domain.

Anda dapat menerapkan dokumen kebijakan ke domain untuk mengizinkan prinsipal IAM lain mengaksesnya.

Untuk informasi selengkapnya, lihat [Kebijakan dan Izin](#) dan [Kebijakan Berbasis Identitas dan Kebijakan Berbasis Sumber Daya](#).

## Topik

- [Mengaktifkan akses lintas akun ke domain](#)
- [Contoh kebijakan domain](#)
- [Contoh kebijakan domain dengan AWS Organizations](#)
- [Menetapkan kebijakan domain](#)
- [Membaca kebijakan domain](#)
- [Menghapus kebijakan domain](#)

## Mengaktifkan akses lintas akun ke domain

Kebijakan sumber daya adalah file teks dalam format JSON. File harus menentukan prinsipal (aktor), satu atau beberapa tindakan, dan efek (Allow atau Deny). Untuk membuat repositori di domain yang dimiliki oleh akun lain, prinsipal harus diberikan izin `CreateRepository` pada sumber daya domain.

Sebagai contoh, kebijakan sumber daya berikut memberikan izin `123456789012` pada akun untuk membuat repositori di domain.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "codeartifact:CreateRepository"
      ],
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      },
      "Resource": "*"
    }
  ]
}
```

Untuk memungkinkan pembuatan repositori dengan tanda, Anda harus menyertakan izin `codeartifact:TagResource`. Ini juga akan memberikan akses kepada akun untuk menambahkan tanda ke domain dan semua repositori di dalamnya.

Karena kebijakan dievaluasi hanya untuk operasi terhadap domain tempatnya dilampirkan, Anda tidak perlu menentukan sumber daya. Karena sumber daya tersirat, `Resource` dapat diatur ke `*`.

Untuk mengakses paket di domain yang dimiliki oleh akun lain, prinsipal harus diberikan izin `GetAuthorizationToken` pada sumber daya domain. Hal ini memungkinkan pemilik domain untuk melakukan kontrol atas akun yang dapat membaca isi repositori di domain.

Sebagai contoh, sumber daya berikut kebijakan memberikan izin `123456789012` kepada akun untuk mengambil token auth untuk repositori apa pun di domain.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "codeartifact:GetAuthorizationToken"
      ],
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      },
      "Resource": "*"
    }
  ]
}
```

#### Note

Prinsipal yang ingin mengambil paket dari titik akhir repositori harus diberikan izin `ReadFromRepository` pada sumber daya repositori selain izin `GetAuthorizationToken` pada domain. Demikian pula, prinsipal yang ingin memublikasikan paket ke titik akhir repositori harus diberikan izin `PublishPackageVersion` selain `GetAuthorizationToken`.

Untuk informasi selengkapnya tentang izin `ReadFromRepository` dan `PublishPackageVersion`, lihat [Kebijakan Repositori](#).

## Contoh kebijakan domain

Ketika beberapa akun menggunakan domain, akun harus diberikan serangkaian izin dasar untuk memungkinkan penggunaan domain sepenuhnya. Kebijakan sumber daya berikut mencantumkan serangkaian izin yang memungkinkan penggunaan domain sepenuhnya.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "BasicDomainPolicy",
      "Action": [
        "codeartifact:GetDomainPermissionsPolicy",
        "codeartifact:ListRepositoriesInDomain",
        "codeartifact:GetAuthorizationToken",
        "codeartifact:DescribeDomain",
        "codeartifact:CreateRepository"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      }
    }
  ]
}
```

### Note

Anda tidak perlu membuat kebijakan domain jika domain dan semua repositori dimiliki oleh satu akun dan hanya perlu digunakan dari akun tersebut.

## Contoh kebijakan domain dengan AWS Organizations

Anda dapat menggunakan kunci `aws:PrincipalOrgID` kondisi untuk memberikan akses ke CodeArtifact domain dari semua akun di organisasi Anda, sebagai berikut.

```
{
  "Version": "2012-10-17",
  "Statement": {
```

```
"Sid": "DomainPolicyForOrganization",
"Effect": "Allow",
"Principal": "*",
"Action": [
    "codeartifact:GetDomainPermissionsPolicy",
    "codeartifact:ListRepositoriesInDomain",
    "codeartifact:GetAuthorizationToken",
    "codeartifact:DescribeDomain",
    "codeartifact:CreateRepository"
],
"Resource": "*",
"Condition": {
    "StringEquals": { "aws:PrincipalOrgID":["o-xxxxxxxxxxxxx"]}
}
}
```

Untuk informasi selengkapnya tentang kunci syarat `aws:PrincipalOrgID`, lihat [Kunci Konteks Syarat Global AWS](#) di Panduan Pengguna IAM.

## Menetapkan kebijakan domain

Anda dapat menggunakan perintah `put-domain-permissions-policy` untuk melampirkan kebijakan ke domain.

```
aws codeartifact put-domain-permissions-policy --domain my_domain --domain-
owner 111122223333 \
--policy-document file://</PATH/TO/policy.json>
```

Ketika Anda memanggil `put-domain-permissions-policy`, kebijakan sumber daya pada domain diabaikan ketika mengevaluasi izin. Hal ini memastikan bahwa pemilik domain tidak dapat mengunci diri dari domain, yang akan mencegah mereka memperbarui kebijakan sumber daya.

### Note

Anda tidak dapat memberikan izin ke AWS akun lain untuk memperbarui kebijakan sumber daya pada domain menggunakan kebijakan sumber daya, karena kebijakan sumber daya diabaikan saat memanggil `put-domain-permissions-policy`.

Contoh output:

```
{
  "policy": {
    "resourceArn": "arn:aws:codeartifact:region-id:111122223333:domain/my_domain",
    "document": "{ ...policy document content...}",
    "revision": "MQLyyTQRASRU3HB58gBtSDHXG7Q3hvxxxxxxxxx="
  }
}
```

Output dari perintah berisi Amazon Resource Name (ARN) dari sumber daya domain, isi lengkap dokumen kebijakan, dan pengidentifikasi revisi. Pengidentifikasi revisi dapat diteruskan ke `put-domain-permissions-policy` menggunakan opsi `--policy-revision`. Hal ini memastikan bahwa revisi dokumen yang sedang ditimpa, dan bukan versi yang lebih baru yang ditetapkan oleh penulis lain.

## Membaca kebijakan domain

Untuk membaca versi dokumen kebijakan yang ada, gunakan perintah `get-domain-permissions-policy`. Untuk memformat output agar dapat dibaca, gunakan `--output` dan `--query policy.document` bersama-sama dengan modul `json.tool` Python, sebagai berikut.

```
aws codeartifact get-domain-permissions-policy --domain my_domain --domain-owner 111122223333 \
  --output text --query policy.document | python -m json.tool
```

Contoh output:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "BasicDomainPolicy",
      "Action": [
        "codeartifact:GetDomainPermissionsPolicy",
        "codeartifact:ListRepositoriesInDomain",
        "codeartifact:GetAuthorizationToken",
        "codeartifact:CreateRepository"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      }
    }
  ]
}
```

```
    }  
  }  
]  
}
```

## Menghapus kebijakan domain

Gunakan perintah `delete-domain-permissions-policy` untuk menghapus kebijakan dari domain.

```
aws codeartifact delete-domain-permissions-policy --domain my_domain --domain-owner 111122223333
```

Format output sama dengan perintah `get-domain-permissions-policy` dan `delete-domain-permissions-policy`.

## Menandai domain di CodeArtifact

Tanda adalah pasangan kunci-nilai yang terkait dengan sumber daya AWS. Anda dapat menerapkan tag ke domain Anda di CodeArtifact. Untuk informasi tentang penandaan CodeArtifact sumber daya, kasus penggunaan, kunci tag dan batasan nilai, serta jenis sumber daya yang didukung, lihat.

[Penandaan pada sumber daya](#)

Anda dapat menggunakan CLI untuk menentukan tanda saat membuat domain. Anda dapat menggunakan konsol atau CLI untuk menambah atau menghapus tanda, dan memperbarui nilai tanda dalam domain. Anda dapat menambahkan hingga 50 tanda ke setiap domain.

Topik

- [Menandai domain \(CLI\)](#)
- [Menandai domain \(konsol\)](#)

## Menandai domain (CLI)

Anda dapat menggunakan CLI untuk mengelola tanda domain.

Topik

- [Menambahkan tanda ke domain \(CLI\)](#)

- [Melihat tanda untuk domain \(CLI\)](#)
- [Mengedit tanda untuk domain \(CLI\)](#)
- [Menghapus tanda dari domain \(CLI\)](#)

## Menambahkan tanda ke domain (CLI)

Anda dapat menggunakan konsol atau AWS CLI untuk menandai domain.

Untuk menambahkan tanda ke domain saat Anda membuatnya, lihat [Buatlah sebuah repositori](#).

Dalam langkah-langkah ini, kami menganggap bahwa Anda telah menginstal versi terbaru AWS CLI atau memperbaruinya ke versi terkini. Untuk informasi lebih lanjut, lihat [Menginstal AWS Command Line Interface](#).

Pada terminal atau baris perintah, jalankan perintah `tag-resource`, yang menentukan Amazon Resource Name (ARN) domain tempat Anda ingin menambahkan tanda dan kunci dan nilai tanda yang ingin Anda tambahkan.

### Note

Untuk mendapatkan ARN domain, jalankan perintah `describe-domain`:

```
aws codeartifact describe-domain --domain my_domain --query domain.arn
```

Anda dapat menambahkan lebih dari satu tanda ke domain. Misalnya, untuk menandai domain bernama *my\_domain* dengan dua tanda, sebuah kunci tanda bernama *key1* dengan nilai tanda *value1*, dan kunci tanda bernama *key2* dengan nilai tanda *value2*:

```
aws codeartifact tag-resource --resource-arn arn:aws:codeartifact:us-west-2:123456789012:domain/my_domain --tags key=key1,value=value1 key=key2,value=value2
```

Jika berhasil, perintah ini tidak memiliki output.

## Melihat tanda untuk domain (CLI)

Ikuti langkah-langkah ini untuk menggunakan AWS CLI untuk melihat AWS tag untuk domain. Jika tidak ada tanda yang telah ditambahkan, daftar yang ditampilkan kosong.



Pada terminal atau baris perintah, jalankan perintah `list-tags-for-resource` dengan Amazon Resource Name (ARN) domain.

#### Note

Untuk mendapatkan ARN domain, jalankan perintah `describe-domain`:

```
aws codeartifact describe-domain --domain my_domain --query domain.arn
```

Misalnya, untuk melihat daftar kunci tanda dan nilai tanda untuk domain bernama *my\_domain* dengan `arn:aws:codeartifact:us-west-2:123456789012:domain/my_domain` nilai ARN:

```
aws codeartifact list-tags-for-resource --resource-arn arn:aws:codeartifact:us-west-2:123456789012:domain/my_domain
```

Jika berhasil, perintah ini mengembalikan informasi yang serupa dengan berikut ini:

```
{
  "tags": {
    "key1": "value1",
    "key2": "value2"
  }
}
```

## Mengedit tanda untuk domain (CLI)

Ikuti langkah-langkah ini untuk menggunakan AWS CLI untuk mengedit tag untuk domain. Anda dapat mengubah nilai untuk kunci yang ada atau menambahkan kunci lain. Anda juga dapat menghapus tanda dari domain, seperti yang ditunjukkan di bagian berikutnya.

Pada terminal atau baris perintah, jalankan perintah `tag-resource`, yang menentukan ARN domain tempat Anda ingin memperbarui tanda dan menentukan kunci tanda dan nilai tanda:

#### Note

Untuk mendapatkan ARN domain, jalankan perintah `describe-domain`:

```
aws codeartifact describe-domain --domain my_domain --query domain.arn
```

```
aws codeartifact tag-resource --resource-arn arn:aws:codeartifact:us-west-2:123456789012:domain/my_domain --tags key=key1,value=newvalue1
```

Jika berhasil, perintah ini tidak memiliki output.

## Menghapus tanda dari domain (CLI)

Ikuti langkah-langkah ini untuk menggunakan AWS CLI untuk menghapus tag dari domain.

### Note

Jika Anda menghapus domain, semua asosiasi tanda dihapus dari domain yang dihapus. Anda tidak perlu menghapus tanda sebelum menghapus domain.

Pada terminal atau baris perintah, jalankan perintah `untag-resource`, yang menentukan ARN domain tempat Anda ingin menghapus tanda dan kunci tanda dari tanda yang ingin Anda hapus.

### Note

Untuk mendapatkan ARN domain, jalankan perintah `describe-domain`:

```
aws codeartifact describe-domain --domain my_domain --query domain.arn
```

Misalnya, untuk menghapus beberapa tanda pada domain bernama *mydomain* dengan kunci tanda *key1* dan *key2*:

```
aws codeartifact untag-resource --resource-arn arn:aws:codeartifact:us-west-2:123456789012:domain/my_domain --tag-keys key1 key2
```

Jika berhasil, perintah ini tidak memiliki output. Setelah menghapus tanda, Anda dapat melihat tanda yang tersisa pada repositori menggunakan perintah `list-tags-for-resource`.

## Menandai domain (konsol)

Anda dapat menggunakan konsol atau CLI untuk menandai sumber daya.

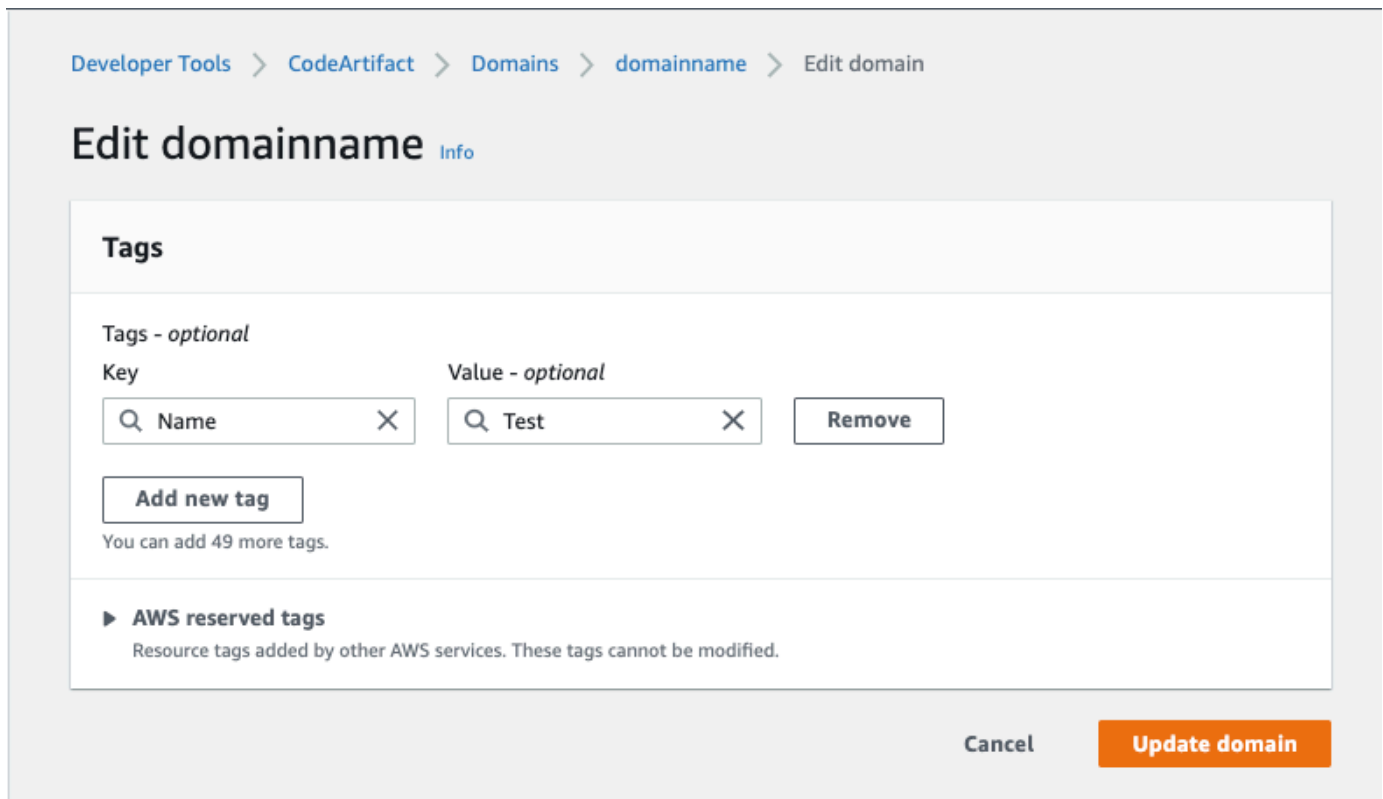
## Topik

- [Menambahkan tanda ke domain \(konsol\)](#)
- [Melihat tanda untuk domain \(konsol\)](#)
- [Mengedit tanda untuk domain \(konsol\)](#)
- [Menghapus tanda dari domain \(konsol\)](#)

## Menambahkan tanda ke domain (konsol)

Anda dapat menggunakan konsol untuk menambahkan tanda ke domain yang ada.

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Pada halaman Domains (Domain), pilih domain yang ingin Anda tambahkan tanda.
3. Perluas bagian Details (Detail).
4. Di bagian Domain tags (Tanda domain), pilih Add domain tags (Tambahkan tanda domain) jika tidak ada tanda pada domain, atau pilih View and edit domain tags (Lihat dan edit tanda domain) jika ada.
5. Pilih Add new tag (Tambahkan tanda baru).
6. Di bidang Key (Kunci) dan Value (Nilai), masukkan teks untuk setiap tanda yang ingin Anda tambahkan. (Bidang Value (Nilai) bersifat opsional.) Contohnya, dalam Key (Kunci), masukkan **Name**. Dalam Value (Nilai), masukkan **Test**.



7. (Opsional) Pilih Add tag (Tambahkan tanda) untuk menambahkan lebih banyak baris dan memasukkan lebih banyak tanda.
8. Pilih Update domain (Perbarui domain).

## Melihat tanda untuk domain (konsol)

Anda dapat menggunakan konsol untuk mencantumkan tanda untuk domain yang ada.

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Pada halaman Domains (Domain), pilih domain tempat Anda ingin melihat tanda.
3. Perluas bagian Details (Detail).
4. Di bagian Domain tags (Tanda domain), pilih View and edit domain tags (Lihat dan edit tanda repositori).

### Note

Jika tidak ada tanda yang ditambahkan ke domain ini, konsol akan membaca Add domain tags (Tambahkan tanda domain).

## Mengedit tanda untuk domain (konsol)

Anda dapat menggunakan konsol untuk mengedit tanda yang telah ditambahkan ke domain.

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Pada halaman Domains (Domain), pilih domain tempat Anda ingin memperbarui tanda.
3. Perluas bagian Details (Detail).
4. Di bagian Domain tags (Tanda domain), pilih View and edit domain tags (Lihat dan edit tanda repositori).

### Note

Jika tidak ada tanda yang ditambahkan ke domain ini, konsol akan membaca Add domain tags (Tambahkan tanda domain).

5. Di bidang Key (Kunci) dan Value (Nilai), perbarui nilai di setiap bidang yang diperlukan. Misalnya, untuk kunci **Name**, di Value (Nilai), ubah **Test** menjadi **Prod**.
6. Pilih Update domain (Perbarui domain).

## Menghapus tanda dari domain (konsol)

Anda dapat menggunakan konsol untuk menghapus tanda dari domain.

1. Buka AWS CodeArtifact konsol di <https://console.aws.amazon.com/codesuite/codeartifact/home>.
2. Pada halaman Domains (Domain), pilih domain tempat Anda ingin menghapus tanda.
3. Perluas bagian Details (Detail).
4. Di bagian Domain tags (Tanda domain), pilih View and edit domain tags (Lihat dan edit tanda repositori).

### Note

Jika tidak ada tanda yang ditambahkan ke domain ini, konsol akan membaca Add domain tags (Tambahkan tanda domain).

5. Di samping kunci dan nilai untuk setiap tanda yang ingin Anda hapus, pilih Remove (Hapus).
6. Pilih Update domain (Perbarui domain).

# Menggunakan CodeArtifact dengan npm

Topik ini menjelaskan cara menggunakan npm, manajer paket Node.js, dengan CodeArtifact.

## Note

CodeArtifact mendukung node v4.9.1 dan yang lebih baru serta npm v5.0.0 dan yang lebih baru.

## Topik

- [Konfigurasi dan gunakan npm dengan CodeArtifact](#)
- [Konfigurasi dan gunakan Yarn dengan CodeArtifact](#)
- [dukungan perintah npm](#)
- [penanganan tanda npm](#)
- [Dukungan untuk manajer paket yang kompatibel dengan npm](#)

## Konfigurasi dan gunakan npm dengan CodeArtifact

Setelah Anda membuat repositori di CodeArtifact, Anda dapat menggunakan klien npm untuk menginstal dan menerbitkan paket. Metode yang disarankan untuk mengonfigurasi npm dengan titik akhir repositori dan token otorisasi Anda adalah dengan menggunakan perintah `aws codeartifact login`. Anda juga dapat mengonfigurasi npm secara manual.

## Daftar Isi

- [Mengkonfigurasi npm dengan perintah login](#)
- [Mengonfigurasi npm tanpa menggunakan perintah login](#)
- [Menjalankan perintah npm](#)
- [Memverifikasi otentikasi dan otorisasi npm](#)
- [Mengubah kembali ke registri npm default](#)
- [Memecahkan masalah pemasangan lambat dengan npm 8.x atau lebih tinggi](#)

## Mengkonfigurasi npm dengan perintah login

Gunakan `aws codeartifact login` perintah untuk mengambil kredensial untuk digunakan dengan npm.

### Note

Jika Anda mengakses repositori di domain milik Anda, Anda tidak perlu menyertakan `--domain-owner`. Untuk informasi selengkapnya, lihat [Domain lintas akun](#).

### Important

Jika Anda menggunakan npm 10.x atau yang lebih baru, Anda harus menggunakan AWS CLI versi 2.9.5 atau yang lebih baru untuk berhasil menjalankan perintah. `aws codeartifact login`

```
aws codeartifact login --tool npm --domain my_domain --domain-owner 111122223333 --  
repository my_repo
```

Perintah ini membuat perubahan berikut pada file `~/.npmrc` Anda:

- Menambahkan token otorisasi setelah mengambilnya dari CodeArtifact menggunakan kredensial AndaAWS.
- Menetapkan registri npm ke repositori yang ditentukan oleh opsi `--repository`.
- Untuk npm 6 dan yang lebih rendah: Menambahkan `"always-auth=true"` sehingga token otorisasi dikirim untuk setiap perintah npm.

Periode otorisasi default setelah memanggil `login` adalah 12 jam, dan `login` harus dipanggil untuk menyegarkan token secara berkala. Untuk informasi selengkapnya tentang token otorisasi yang dibuat dengan perintah `login`, lihat [Token dibuat dengan perintah login](#).

## Mengonfigurasi npm tanpa menggunakan perintah login

Anda dapat mengonfigurasi npm dengan CodeArtifact repositori Anda tanpa `aws codeartifact login` perintah dengan memperbarui konfigurasi npm secara manual.

## Untuk mengkonfigurasi npm tanpa menggunakan perintah login

1. Di baris perintah, ambil token CodeArtifact otorisasi dan simpan dalam variabel lingkungan. npm akan menggunakan token ini untuk mengautentikasi dengan repositori Anda. CodeArtifact

### Note

Perintah berikut adalah untuk mesin macOS atau Linux. Untuk informasi tentang mengonfigurasi variabel lingkungan pada mesin Windows, lihat [Teruskan token auth menggunakan variabel lingkungan](#).

```
CODEARTIFACT_AUTH_TOKEN=`aws codeartifact get-authorization-token --  
domain my_domain --domain-owner 111122223333 --query authorizationToken --output  
text`
```

2. Dapatkan endpoint CodeArtifact repositori Anda dengan menjalankan perintah berikut. Endpoint repositori Anda digunakan untuk mengarahkan npm ke repositori Anda untuk menginstal atau menerbitkan paket.

- Ganti *my\_domain* dengan nama CodeArtifact domain Anda.
- Ganti *111122223333* dengan ID AWS akun pemilik domain. Jika Anda mengakses repositori di domain milik Anda, Anda tidak perlu menyertakan `--domain-owner`. Untuk informasi selengkapnya, lihat [Domain lintas akun](#).
- Ganti *my\_repo* dengan nama repositori Anda CodeArtifact .

```
aws codeartifact get-repository-endpoint --domain my_domain --domain-  
owner 111122223333 --repository my_repo --format npm
```

URL berikut adalah contoh titik akhir repositori.

```
https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/npm/my_repo/
```



**⚠ Important**

URL registri harus diakhiri dengan garis miring (/). Jika tidak, Anda tidak dapat terhubung ke repositori.

- Gunakan `npm config set` perintah untuk mengatur registri ke CodeArtifact repositori Anda. Ganti URL dengan URL endpoint repositori dari langkah sebelumnya.

```
npm config set
registry=https://my_domain-111122223333.d.codeartifact.region.amazonaws.com/
npm/my_repo/
```

- Gunakan `npm config set` perintah untuk menambahkan token otorisasi Anda ke konfigurasi npm Anda.

```
npm config set //my_domain-111122223333.d.codeartifact.region.amazonaws.com/
npm/my_repo/:_authToken=$CODEARTIFACT_AUTH_TOKEN
```

Untuk npm 6 atau lebih rendah: Untuk membuat npm selalu meneruskan token auth ke CodeArtifact, bahkan untuk GET permintaan, setel variabel `always-auth` konfigurasi dengan `npm config set`

```
npm config set //my_domain-111122223333.d.codeartifact.region.amazonaws.com/
npm/my_repo/:always-auth=true
```

**Contoh file konfigurasi npm () `.npmrc`**

Berikut ini adalah `.npmrc` file contoh setelah mengikuti instruksi sebelumnya untuk mengatur titik akhir CodeArtifact registri, menambahkan token otentikasi, dan mengkonfigurasi `always-auth`

```
registry=https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/npm/my-
cli-repo/
//my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/npm/
my_repo/:_authToken=eyJ2ZX...
//my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/npm/my_repo/:always-
auth=true
```

## Menjalankan perintah npm

Setelah Anda mengkonfigurasi klien npm, Anda dapat menjalankan perintah npm. Dengan asumsi bahwa sebuah paket ada dalam repositori Anda atau salah satu repositori hulu, Anda dapat menginstalnya dengan npm `install`. Misalnya, gunakan hal berikut untuk menginstal paket `lodash`.

```
npm install lodash
```

Gunakan perintah berikut untuk menerbitkan paket npm baru ke CodeArtifact repositori.

```
npm publish
```

Untuk informasi tentang cara membuat paket npm, lihat [Membuat Modul Node.js](#) di situs web dokumentasi npm. Untuk daftar perintah npm yang didukung oleh CodeArtifact, lihat [npm Command Support](#).

## Memverifikasi otentikasi dan otorisasi npm

Memanggil npm `ping` perintah adalah cara untuk memverifikasi hal-hal berikut:

- Anda telah mengonfigurasi kredensialnya dengan benar sehingga Anda dapat mengautentikasi ke repositori. CodeArtifact
- Konfigurasi otorisasi memberikan Anda izin `ReadFromRepository`.

Output dari panggilan npm `ping` yang berhasil terlihat seperti berikut.

```
$ npm -d ping
npm info it worked if it ends with ok
npm info using npm@6.4.1
npm info using node@v9.5.0
npm info attempt registry request try #1 at 4:30:59 PM
npm http request GET https://<domain>.d.codeartifact.us-west-2.amazonaws.com/npm/
shared/-/ping?write=true
npm http 200 https:///npm/shared/-/ping?write=true
Ping success: {}
npm timing npm Completed in 716ms
npm info ok
```

-dOpsi ini menyebabkan npm mencetak informasi debug tambahan, termasuk URL repositori. Informasi ini memudahkan untuk mengonfirmasi bahwa npm dikonfigurasi untuk menggunakan repositori yang Anda harapkan.

## Mengubah kembali ke registri npm default

Mengkonfigurasi npm dengan CodeArtifact menetapkan registri npm ke repositori yang ditentukan. CodeArtifact Anda dapat menjalankan perintah berikut untuk mengatur registri npm kembali ke registri default ketika Anda selesai menghubungkan ke CodeArtifact.

```
npm config set registry https://registry.npmjs.com/
```

## Memecahkan masalah pemasangan lambat dengan npm 8.x atau lebih tinggi

Ada masalah yang diketahui di npm versi 8.x dan lebih besar di mana jika permintaan dibuat ke repositori paket, dan repositori mengarahkan klien ke Amazon S3 alih-alih mengalirkan aset secara langsung, klien npm dapat hang selama beberapa menit per ketergantungan.

Karena CodeArtifact repositori dirancang untuk selalu mengarahkan permintaan ke Amazon S3, terkadang masalah ini terjadi, yang menyebabkan waktu pembuatan yang lama karena waktu pemasangan npm yang lama. Contoh perilaku ini akan muncul sebagai bilah kemajuan yang ditampilkan selama beberapa menit.

Untuk menghindari masalah ini, gunakan `progress=false` tanda `--no-progress` atau dengan perintah npm cli, seperti yang ditunjukkan pada contoh berikut.

```
npm install lodash --no-progress
```

## Konfigurasi dan gunakan Yarn dengan CodeArtifact

Setelah Anda membuat repositori, Anda dapat menggunakan klien Yarn untuk mengelola paket npm.

### Note

Yarn 1.X membaca dan menggunakan informasi dari file konfigurasi npm Anda (`.npmrc`), sementara Yarn 2.X tidak. Konfigurasi untuk Yarn 2.X harus didefinisikan dalam file `yarnrc.yml`.

## Daftar Isi

- [Konfigurasi Yarn 1.X dengan `aws codeartifact login` perintah](#)
- [Konfigurasi Yarn 2.X dengan `yarn config set` perintah](#)

## Konfigurasi Yarn 1.X dengan `aws codeartifact login` perintah

Untuk Yarn 1.X, Anda dapat mengkonfigurasi Yarn dengan CodeArtifact menggunakan `aws codeartifact login` perintah. Yang `login` perintah akan mengkonfigurasi file `~/.npmrc` Anda dengan CodeArtifact informasi endpoint repositori dan kredensi. Dengan Yarn 1.X, `yarn` perintah menggunakan informasi konfigurasi dari file `~/.npmrc`.

Untuk mengkonfigurasi Yarn 1.X dengan perintah `login`

1. Jika Anda belum melakukannya, konfigurasi AWS kredensi untuk digunakan dengan AWS CLI, seperti yang dijelaskan dalam [Memulai dengan CodeArtifact](#).
2. Untuk menjalankan `aws codeartifact login` perintah berhasil, npm harus diinstal. Lihat [Mengunduh dan menginstal Node.js dan npm](#) di dalam dokumentasi npm untuk petunjuk instalasi.
3. Gunakan `aws codeartifact login` perintah untuk mengambil CodeArtifact kredensi dan konfigurasi file `~/.npmrc` Anda.
  - Ganti `my_domain` dengan Anda CodeArtifact nama domain.
  - Ganti `111122223333` dengan AWS ID akun pemilik domain. Jika Anda mengakses repositori di domain milik Anda, Anda tidak perlu menyertakan `--domain-owner`. Untuk informasi selengkapnya, lihat [Domain lintas akun](#).
  - Ganti `repo` dengan Anda CodeArtifact nama repositori.

```
aws codeartifact login --tool npm --domain my_domain --domain-owner 111122223333 --
repository my_repo
```

Yang `login` perintah membuat perubahan berikut ke file `~/.npmrc` Anda:

- Menambahkan token otorisasi setelah mengambilnya dari CodeArtifact menggunakan AWS mandat.
- Menetapkan registri npm ke repositori yang ditentukan oleh opsi `--repository`.

- Untuk npm 6 dan lebih rendah: Menambahkan `"always-auth=true"` sehingga token otorisasi dikirim untuk setiap perintah npm.

Periode otorisasi default setelah menelepon `login` adalah 12 jam, dan `login` harus dipanggil untuk menyegarkan token secara berkala. Untuk informasi selengkapnya tentang token otorisasi yang dibuat dengan perintah `login`, lihat [Token dibuat dengan perintah login](#).

4. Untuk npm 7.X dan 8.X, Anda harus menambahkan `always-auth=true` ke file `~/.npmrc` Anda untuk menggunakan Yarn.
  - Buka file `~/.npmrc` Anda di editor teks dan tambahkan `always-auth=true` pada baris baru.

Anda dapat menggunakan `yarn config list` perintah untuk memeriksa bahwa Yarn menggunakan konfigurasi yang benar. Setelah menjalankan perintah, periksa nilai di `info npm config` bagian. Isi harus terlihat mirip dengan cuplikan berikut.

```
info npm config
{
  registry: 'https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/npm/my_repo/',
  '//my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/npm/my_repo/:_authToken': 'eyJ2ZXI...',
  'always-auth': true
}
```

## Konfigurasi Yarn 2.X dengan `yarn config set` komando

Prosedur berikut merinci cara mengkonfigurasi Yarn 2.X dengan memperbarui `yarnrc.yml` konfigurasi dari baris perintah dengan `yarn config set` perintah.

Untuk memperbarui `yarnrc.yml` konfigurasi dari baris perintah

1. Jika Anda belum melakukannya, konfigurasi AWS kredensi untuk digunakan dengan AWS CLI, seperti yang dijelaskan dalam [Memulai dengan CodeArtifact](#).
2. Gunakan `aws codeartifact get-repository-endpoint` perintah untuk mendapatkan CodeArtifact endpoint repositori.
  - Ganti `my_domain` dengan Anda CodeArtifact nama domain.

- Ganti `111122223333` dengan AWS ID akun pemilik domain. Jika Anda mengakses repositori di domain milik Anda, Anda tidak perlu menyertakan `--domain-owner`. Untuk informasi selengkapnya, lihat [Domain lintas akun](#).
- Ganti `repo` dengan AndaCodeArtifact nama repositori.

```
aws codeartifact get-repository-endpoint --domain my_domain --domain-owner 111122223333 --repository my_repo --format npm
```

3. Perbarui `npmRegistryServer` nilai dalam `file.yarnrc.yml` dengan endpoint repositori.

```
yarn config set npmRegistryServer "https://my_domain-111122223333.d.codeartifact.region.amazonaws.com/npm/my_repo/"
```

4. Ambil CodeArtifact token otorisasi dan menyimpannya dalam variabel lingkungan.

#### Note

Perintah berikut adalah untuk macOS atau Linux mesin. Untuk informasi tentang mengonfigurasi variabel lingkungan pada mesin Windows, lihat [Teruskan token auth menggunakan variabel lingkungan](#).

- Ganti `my_domain` dengan AndaCodeArtifact nama domain.
- Ganti `111122223333` dengan AWS ID akun pemilik domain. Jika Anda mengakses repositori di domain milik Anda, Anda tidak perlu menyertakan `--domain-owner`. Untuk informasi selengkapnya, lihat [Domain lintas akun](#).
- Ganti `repo` dengan AndaCodeArtifact nama repositori.

```
export CODEARTIFACT_AUTH_TOKEN=`aws codeartifact get-authorization-token --domain my_domain --domain-owner 111122223333 --query authorizationToken --output text`
```

5. Gunakan `yarn config set` perintah untuk menambahkan CodeArtifact token otentikasi ke `file.yarnrc.yml`. Ganti URL dalam perintah berikut dengan URL endpoint repositori Anda dari Langkah 2.

```
yarn config set
  'npmRegistries["https://my_domain-
111122223333.d.codeartifact.region.amazonaws.com/npm/my_repo/"] .npmAuthToken '
"${CODEARTIFACT_AUTH_TOKEN}"
```

- Gunakan `yarn config set` perintah untuk mengatur nilai `npmAlwaysAuth` ke `true`. Ganti URL dalam perintah berikut dengan URL endpoint repositori Anda dari Langkah 2.

```
yarn config set
  'npmRegistries["https://my_domain-
111122223333.d.codeartifact.region.amazonaws.com/npm/my_repo/"] .npmAlwaysAuth '
"true"
```

Setelah mengkonfigurasi, file konfigurasi `yarnrc.yml`mu harus memiliki konten yang mirip dengan cuplikan berikut.

```
npmRegistries:
  "https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/npm/my_repo/":
    npmAlwaysAuth: true
    npmAuthToken: eyJ2ZXI...

npmRegistryServer: "https://my_domain-111122223333.d.codeartifact.us-
west-2.amazonaws.com/npm/my_repo/"
```

Anda juga dapat menggunakan `yarn config` perintah untuk memeriksa nilai `npmRegistries` dan `npmRegistryServer`.

## dukungan perintah npm

Bagian berikut meringkas perintah npm yang didukung oleh CodeArtifact repositori, selain perintah khusus yang tidak didukung.

### Daftar Isi

- [Perintah yang didukung yang berinteraksi dengan repositori](#)
- [Perintah sisi klien yang didukung](#)
- [Perintah tidak didukung](#)

## Perintah yang didukung yang berinteraksi dengan repositori

Bagian ini berisi daftar npm tempat klien npm membuat satu atau beberapa permintaan ke registri yang telah dikonfigurasi (misalnya, dengan `npm config set registry`). Perintah ini telah diverifikasi agar berfungsi dengan benar ketika dipanggil terhadap CodeArtifact repositori.

Perintah	Deskripsi
<a href="#">bug</a>	Mencoba menebak lokasi URL pelacak bug paket, dan kemudian mencoba membukanya.
<a href="#">ci</a>	Menginstal proyek dari awal.
<a href="#">mencela</a>	Menghentikan penggunaan versi paket.
<a href="#">dist-tag</a>	Memodifikasi tanda distribusi paket.
<a href="#">dokumen</a>	Mencoba menebak lokasi URL dokumentasi paket, dan kemudian mencoba untuk membukanya menggunakan parameter konfigurasi <code>--browser</code> .
<a href="#">dokter</a>	Menjalankan serangkaian pemeriksaan untuk memastikan bahwa penginstalan npm Anda memiliki apa yang dibutuhkan untuk mengelola JavaScript paket.
<a href="#">memasang</a>	Menginstal paket.
<a href="#">install-ci-test</a>	Menginstal proyek dari awal dan menjalankan pengujian. Alias: <code>npm ci</code> . Perintah ini menjalankan <code>npm ci</code> yang langsung diikuti oleh <code>npm test</code> .
<a href="#">menginstal-tes</a>	Menginstal paket dan menjalankan tes. Menjalankan <code>npm install</code> yang langsung diikuti oleh <code>npm test</code> .



Perintah	Deskripsi
<a href="#">ketinggalan jaman</a>	Memeriksa registri yang dikonfigurasi untuk melihat apakah paket yang diinstal saat ini sudah usang.
<a href="#">ping</a>	Ping registri npm yang dikonfigurasi atau diberikan dan memverifikasi autentikasi.
<a href="#">mempublikasikan</a>	Memublikasikan versi paket ke registri.
<a href="#">memperbarui</a>	Menebak lokasi URL repositori paket, dan kemudian mencoba membukanya menggunakan parameter <code>config --browser</code> .
<a href="#">pandangan</a>	Menampilkan metadata paket. Dapat digunakan untuk mencetak properti metadata.

## Perintah sisi klien yang didukung

Perintah ini tidak memerlukan interaksi langsung dengan repositori, sehingga CodeArtifact tidak perlu melakukan apa pun untuk mendukungnya.

Perintah	Deskripsi
<a href="#">membangun</a>	Membangun paket.
<a href="#">tembolok</a>	Memanipulasi cache paket.
<a href="#">penyelesaian</a>	Memungkinkan penyelesaian tab di semua perintah npm.
<a href="#">konfigurasi</a>	Memperbarui isi pengguna dan file <code>npmrc</code> global.
<a href="#">menyimpulkan</a>	Mencari pohon paket lokal dan mencoba menyederhanakan struktur dengan memindahkan dependensi semakin ke atas pohon, yang

Perintah	Deskripsi
	dapat lebih efektif dibagi oleh beberapa paket dependen.
<a href="#">mengedit</a>	Mengedit paket yang diinstal. Memilih dependensi dalam direktori kerja saat ini dan membuka folder paket di editor default.
<a href="#">mengeksplorasi</a>	Menelusuri paket yang diinstal. Memunculkan subshell di direktori paket terinstal yang ditentukan. Jika ditentukan, perintah dijalankan di subshell, yang kemudian segera berakhir.
<a href="#">membantu</a>	Mendapat bantuan mengenai npm.
<a href="#">bantuan-pencarian</a>	Menelusuri dokumentasi bantuan npm.
<a href="#">init</a>	Membuat file <code>package.json</code> .
<a href="#">tautan</a>	Symlink folder paket.
<a href="#">ls</a>	Daftar paket yang diinstal.
<a href="#">pak</a>	Membuat tarball dari sebuah paket.
<a href="#">awalan</a>	Menampilkan prefiks. Ini adalah direktori induk terdekat untuk memuat file <code>package.json</code> kecuali <code>-g</code> juga ditentukan.
<a href="#">memangkas</a>	Menghapus paket yang tidak tercantum pada daftar dependensi paket induk.
<a href="#">membangun kembali</a>	Menjalankan perintah <code>npm build</code> pada folder yang cocok.
<a href="#">memulai kembali</a>	Menjalankan skrip berhenti, mulai ulang, dan mulai untuk paket dan skrip pra- dan post-skrip terkait.

Perintah	Deskripsi
<a href="#">akar</a>	Mencetak folder <code>node_modules</code> untuk standar keluar.
<a href="#">run-script</a>	Menjalankan skrip paket arbitrer.
<a href="#">shrinkwrap</a>	Mengunci versi dependensi untuk publikasi.
<a href="#">mencopot pemasangan</a>	Meng-uninstall paket.

## Perintah tidak didukung

Perintah npm ini tidak didukung oleh CodeArtifact repositori.

Perintah	Deskripsi	Catatan
<a href="#">akses</a>	Menetapkan tingkat akses pada paket yang dipublikasikan.	CodeArtifact menggunakan model izin yang berbeda dari repositori npmjs publik.
<a href="#">adduser</a>	Menambahkan akun pengguna registri	CodeArtifact menggunakan model pengguna yang berbeda dari repositori npmjs publik.
<a href="#">audit</a>	Menjalankan audit keamanan.	CodeArtifact Saat ini tidak menawarkan data kelemahan keamanan.
<a href="#">kait</a>	Mengelola kait npm, termasuk menambahkan, menghapus, mendaftar, dan memperbarui.	CodeArtifact Saat ini tidak mendukung segala jenis mekanisme notifikasi perubahan.
<a href="#">masuk</a>	Mengautentikasi pengguna. Ini adalah nama lain untuk npm <code>adduser</code> .	CodeArtifact menggunakan model autentikasi yang berbeda dari repositori npmjs

Perintah	Deskripsi	Catatan
		publik. Untuk informasi, lihat <a href="#">Autentikasi dengan npm</a> .
<a href="#">logout</a>	Keluar dari registri.	CodeArtifact menggunakan model autentikasi yang berbeda dari repositori npmjs publik. Tidak ada cara untuk keluar dari CodeArtifact repositori, tetapi token autentikasi berakhir setelah waktu kedaluwarsa yang dapat dikonfigurasi. Durasi token default adalah 12 jam.
<a href="#">pemilik</a>	Mengelola pemilik paket.	CodeArtifact menggunakan model perizinan yang berbeda dari repositori npmjs publik.
<a href="#">profil</a>	Mengubah pengaturan pada profil registri Anda.	CodeArtifact menggunakan model pengguna yang berbeda dari repositori npmjs publik.
<a href="#">pencarian</a>	Mencari registri untuk paket yang cocok dengan istilah pencarian.	CodeArtifact mendukung fungsionalitas pencarian terbatas dengan perintah <a href="#">list-packages</a> .
<a href="#">bintang</a>	Menandai paket favorit Anda.	CodeArtifact saat ini tidak mendukung segala jenis mekanisme favorit.
<a href="#">bintang</a>	Melihat paket yang ditandai sebagai favorit.	CodeArtifact saat ini tidak mendukung segala jenis mekanisme favorit.

Perintah	Deskripsi	Catatan
<a href="#">tim</a>	Mengelola tim organisasi dan keanggotaan tim.	CodeArtifact menggunakan model keanggotaan pengguna dan grup yang berbeda dari repositori npmjs publik. Untuk informasi, lihat <a href="#">Identitas (Pengguna, Grup, dan Peran)</a> di Panduan Pengguna IAM.
<a href="#">token</a>	Mengelola token autentikasi Anda.	CodeArtifact menggunakan model yang berbeda untuk mendapatkan token autentikasi. Untuk informasi, lihat <a href="#">Autentikasi dengan npm</a> .
<a href="#">membatalkan publikasi</a>	Menghapus paket dari registri.	CodeArtifact tidak mendukung penghapusan versi paket dari repositori menggunakan klien npm. Anda dapat menggunakan perintah <a href="#">delete-package-version</a> .
<a href="#">whoami</a>	Menampilkan nama pengguna npm.	CodeArtifact menggunakan model pengguna yang berbeda dari repositori npmjs publik.

## penanganan tanda npm

registri npm mendukung tanda, yang merupakan alias string untuk versi paket. Anda dapat menggunakan tanda untuk memberikan alias, bukan nomor versi. Misalnya, Anda mungkin memiliki proyek dengan beberapa aliran pengembangan dan menggunakan tanda yang berbeda (misalnya, `stable`, `beta`, `dev`, `canary`) untuk setiap aliran. Untuk informasi selengkapnya, lihat [dist-tag](#) di situs web npm.

Secara default, npm menggunakan tanda `latest` untuk mengidentifikasi versi paket saat ini. npm `install pkg` (tanpa penentu `@version` atau `@tag`) menginstal tanda terbaru. Biasanya, proyek menggunakan tanda terbaru hanya untuk versi rilis stabil. Tanda lain digunakan untuk versi yang tidak stabil atau pra-rilis.

## Edit tanda dengan klien npm

Ketiga perintah npm `dist-tag` (`add`, `rm`, dan `ls`) memiliki fungsi yang serupa dalam repositori CodeArtifact seperti yang mereka lakukan di [registri npm default](#).

## Tanda npm dan API CopyPackageVersions

Ketika Anda menggunakan API `CopyPackageVersions` untuk menyalin versi paket npm, semua tanda alias versi tersebut disalin ke repositori tujuan. Ketika versi yang sedang disalin memiliki tanda yang juga ada di tujuan, operasi penyalinan menetapkan nilai tanda di repositori tujuan untuk mencocokkan nilai di repositori sumber.

Sebagai contoh, anggap repositori S dan repositori D berisi satu versi paket `web-helper` dengan set tanda terbaru seperti yang ditunjukkan dalam tabel ini.

Repositori	Nama paket	Tanda paket
D	<code>web-helper</code>	terbaru (alias untuk versi 1.0.1)
D	<code>web-helper</code>	terbaru (alias untuk versi 1.0.0)

`CopyPackageVersions` dipanggil untuk menyalin `web-helper 1.0.1` dari S ke D. Setelah operasi selesai, tanda `latest` pada `web-helper` dalam repositori alias D adalah 1.0.1, bukan 1.0.0.

Jika Anda perlu mengubah tanda setelah menyalin, gunakan perintah npm `dist-tag` untuk mengubah tanda secara langsung di repositori tujuan. Untuk informasi selengkapnya tentang API `CopyPackageVersions`, lihat [Copying Packages Between Repositories](#).

## tanda npm dan repositori hulu

Saat npm meminta tanda untuk paket dan versi paket yang juga ada di repositori hulu, CodeArtifact menggabungkan tanda sebelum mengembalikannya ke klien. Misalnya, repositori bernama R

memiliki repositori hulu bernama U. Tabel berikut menunjukkan tanda untuk paket bernama `web-helper` yang ada di kedua repositori.

Repositori	Nama paket	Tanda paket
R	<code>web-helper</code>	terbaru (alias untuk versi 1.0.0)
U	<code>web-helper</code>	alfa (alias untuk versi 1.0.1)

Dalam kasus ini, ketika mengambil tanda untuk paket `web-helper` dari repositori R, klien npm menerima tanda terbaru dan alfa. Versi yang ditunjukkan oleh tanda tidak akan berubah.

Ketika tanda yang sama ada pada paket yang sama di repositori hulu dan hilir, CodeArtifact menggunakan tanda yang ada dalam repositori hulu. Misalnya, anggap tanda pada `webhelper` telah dimodifikasi agar terlihat seperti berikut ini.

Repositori	Nama paket	Tanda paket
R	<code>web-helper</code>	terbaru (alias untuk versi 1.0.0)
U	<code>web-helper</code>	terbaru (alias untuk versi 1.0.1)

Dalam hal ini, ketika klien npm mengambil tanda untuk paket `web-helper` dari repositori R, tanda terbaru akan membuat alias versi 1.0.1 karena itulah yang ada di repositori hulu. Hal ini memudahkan penggunaan versi paket baru di repositori hulu yang belum ada di repositori hilir dengan menjalankan `npm update`.

Menggunakan tanda di repositori hulu dapat menjadi masalah ketika memublikasikan versi baru dari sebuah paket di repositori hilir. Sebagai contoh, anggap bahwa tanda terbaru pada paket `web-helper` sama di R dan U.

Repositori	Nama paket	Tanda paket
R	web-helper	terbaru (alias untuk versi 1.0.1)
U	web-helper	terbaru (alias untuk versi 1.0.1)

Saat versi 1.0.2 dipublikasikan ke R, npm memperbaruiterbarutanda untuk 1.0.2.

Repositori	Nama paket	Tanda paket
R	web-helper	terbaru (alias untuk versi 1.0.2)
U	web-helper	terbaru (alias untuk versi 1.0.1)

Namun, klien npm tidak pernah melihat nilai tanda ini karena nilaiterbarudi U adalah 1.0.1.

Menjalankan `npm install` terhadap repositori R segera setelah publikasi 1.0.2 akan menginstal 1.0.1, bukan versi yang baru saja dipublikasikan. Untuk menginstal versi yang paling baru dipublikasikan, Anda harus menentukan versi paket yang tepat, sebagai berikut.

```
npm install web-helper@1.0.2
```

## Dukungan untuk manajer paket yang kompatibel dengan npm

Manajer paket lain ini kompatibel dengan CodeArtifact dan berfungsi dengan format paket npm dan protokol kabel npm:

- [manajer paket pnpm](#). Versi terbaru yang dikonfirmasi berfungsi dengan CodeArtifact adalah 3.3.4, yang dirilis pada 18 Mei 2019.
- [Manajer paket Yarn](#). Versi terbaru yang dikonfirmasi berfungsi dengan CodeArtifact adalah 1.21.1, yang dirilis pada 11 Desember 2019.



**Note**

Sebaiknya gunakan Yarn 2.x dengan CodeArtifact. Benang 1.x tidak memiliki HTTP retries, yang berarti lebih rentan terhadap kesalahan layanan intermiten yang mengakibatkan kode status 500 tingkat atau kesalahan. Tidak ada cara untuk mengkonfigurasi strategi coba ulang yang berbeda untuk Yarn 1.x, tetapi ini telah ditambahkan dalam Yarn 2.x. Anda dapat menggunakan Yarn 1.x, tetapi Anda mungkin perlu menambahkan percobaan ulang tingkat yang lebih tinggi dalam skrip build. Misalnya, menjalankan perintah benang Anda dalam satu lingkaran sehingga akan mencoba lagi jika paket download gagal.

# Menggunakan CodeArtifact dengan Python

Topik-topik ini menjelaskan cara menggunakan `pip`, manajer paket Python, dan `twine`, utilitas penerbitan paket Python, dengan CodeArtifact.

Topik

- [Konfigurasi dan gunakan `pip` dengan CodeArtifact](#)
- [Konfigurasi dan gunakan `benang` dengan CodeArtifact](#)
- [Normalisasi nama paket Python](#)
- [Kompatibilitas Python](#)
- [Meminta paket Python dari upstream dan koneksi eksternal](#)

## Konfigurasi dan gunakan `pip` dengan CodeArtifact

`pip` adalah penginstal paket untuk paket Python. Untuk menggunakan `pip` untuk menginstal paket Python dari Anda CodeArtifact repositori, Anda harus terlebih dahulu mengkonfigurasi klien `pip` dengan CodeArtifact informasi repositori dan kredensial.

`pip` hanya dapat digunakan untuk menginstal paket Python. Untuk mempublikasikan paket Python, Anda dapat menggunakan `benang`. Untuk informasi selengkapnya, lihat [Konfigurasi dan gunakan `benang` dengan CodeArtifact](#).

## Konfigurasi `pip` dengan `login` komando

Pertama, konfigurasi kredensial AWS untuk digunakan dengan AWS CLI, seperti yang dijelaskan dalam [Memulai dengan CodeArtifact](#). Kemudian, gunakan CodeArtifact `login` perintah untuk mengambil kredensial dan mengkonfigurasi `pip` dengan mereka.

### Note

Jika Anda mengakses repositori di domain milik Anda, Anda tidak perlu menyertakan `--domain-owner`. Untuk informasi selengkapnya, lihat [Domain lintas akun](#).

Untuk mengkonfigurasi `pip`, jalankan perintah berikut.

```
aws codeartifact login --tool pip --domain my_domain --domain-owner 111122223333 --  
repository my_repo
```

login mengambil token otorisasi dari CodeArtifact menggunakan Anda AWS kredensialnya. The login perintah akan mengkonfigurasi pip untuk digunakan dengan CodeArtifact dengan mengedit `~/.config/pip/pip.conf` untuk mengatur `index-url` ke repositori yang ditentukan oleh `--repository` pilihan.

Periode otorisasi default setelah memanggil login adalah 12 jam, dan login harus dipanggil untuk menyegarkan token secara berkala. Untuk informasi selengkapnya tentang token otorisasi yang dibuat dengan perintah login, lihat [Token dibuat dengan perintah login](#).

## Mengonfigurasi pip tanpa perintah login

Jika Anda tidak dapat menggunakan perintah login untuk mengonfigurasi pip, Anda dapat menggunakan `pip config`.

1. Gunakan AWS CLI untuk mengambil token otorisasi baru.

### Note

Jika Anda mengakses repositori di domain yang Anda miliki, Anda tidak perlu menyertakan `--domain-owner`. Untuk informasi selengkapnya, lihat [Domain lintas akun](#).

```
CODEARTIFACT_AUTH_TOKEN=`aws codeartifact get-authorization-token --  
domain my_domain --domain-owner 111122223333 --query authorizationToken --output  
text`
```

2. Gunakan `pip config` untuk mengatur CodeArtifact URL registri dan kredensialnya. Perintah berikut akan memperbarui file konfigurasi lingkungan saat ini saja. Untuk memperbarui file konfigurasi seluruh sistem, ganti `site` dengan `global`.

```
pip config set site.index-url https://aws:  
$CODEARTIFACT_AUTH_TOKEN@my_domain-  
111122223333.d.codeartifact.region.amazonaws.com/pypi/my_repo/simple/
```

**⚠ Important**

URL registri harus diakhiri dengan garis miring (/). Jika tidak, Anda tidak dapat terhubung ke repositori.

## Contoh file konfigurasi pip

Berikut ini adalah contoh dari `pip.conf` file setelah mengatur `CodeArtifactURL` registri dan kredensialnya.

```
[global]
index-url = https://aws:eyJ2ZX...@my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/pypi/my_repo/simple/
```

## Menjalankan pip

Untuk menjalankan `pip` perintah, Anda harus mengkonfigurasi `pip` bersama `CodeArtifact`. Untuk informasi selengkapnya, lihat dokumentasi berikut.

1. Ikuti langkah-langkah di bagian [Mengatur dengan AWS CodeArtifact](#) untuk mengonfigurasi akun AWS, alat, dan izin.
2. Konfigurasi `twine` dengan mengikuti langkah-langkah di [Konfigurasi dan gunakan `benang` dengan `CodeArtifact`](#).

Dengan asumsi bahwa sebuah paket ada dalam repositori Anda atau salah satu repositori hulu, Anda dapat menginstalnya dengan `pip install`. Misalnya, gunakan perintah berikut untuk menginstal paket `requests`.

```
pip install requests
```

Gunakan `-i` opsi untuk sementara kembali ke menginstal paket dari <https://pypi.org> bukannya `CodeArtifact` repositori.

```
pip install -i https://pypi.org/simple requests
```

# Konfigurasi dan gunakan benang dengan CodeArtifact

[benang](#) adalah utilitas penerbitan paket untuk paket Python. Untuk menggunakan benang untuk mempublikasikan paket Python ke Anda CodeArtifact repositori, Anda harus terlebih dahulu mengkonfigurasi benang dengan CodeArtifact informasi repositori dan kredensial.

benang hanya dapat digunakan untuk mempublikasikan paket Python. Untuk menginstal paket Python, Anda dapat menggunakan [pip](#). Untuk informasi selengkapnya, lihat [Konfigurasi dan gunakan pip dengan CodeArtifact](#).

## Konfigurasi benang dengan `login` komando

Pertama, konfigurasi kredensial AWS untuk digunakan dengan AWS CLI, seperti yang dijelaskan dalam [Memulai dengan CodeArtifact](#). Kemudian, gunakan CodeArtifact `login` perintah untuk mengambil kredensial dan mengkonfigurasi benang dengan mereka.

### Note

Jika Anda mengakses repositori di domain milik Anda, Anda tidak perlu menyertakan `--domain-owner`. Untuk informasi selengkapnya, lihat [Domain lintas akun](#).

Untuk mengkonfigurasi benang, jalankan perintah berikut.

```
aws codeartifact login --tool twine --domain my_domain --domain-owner 111122223333 --  
repository my_repo
```

`login` mengambil token otorisasi dari CodeArtifact menggunakan Anda AWS kredensialnya. The `login` perintah mengkonfigurasi benang untuk digunakan dengan CodeArtifact dengan mengedit `~/.pypirc` untuk menambahkan repositori yang ditentukan oleh `--repository` opsi dengan kredensial.

Periode otorisasi default setelah memanggil `login` adalah 12 jam, dan `login` harus dipanggil untuk menyegarkan token secara berkala. Untuk informasi selengkapnya tentang token otorisasi yang dibuat dengan perintah `login`, lihat [Token dibuat dengan perintah login](#).

## Konfigurasi benang tanpa `login` komando

Jika Anda tidak dapat menggunakan `login` perintah untuk mengkonfigurasi benang, Anda dapat menggunakan `~/.pypirc` variabel file atau lingkungan. Untuk menggunakan file `~/.pypirc`,

tambahkan entri berikut ke file. Kata sandi harus berupa token auth yang diperoleh oleh API `get-authorization-token`.

```
[distutils]
index-servers =
  codeartifact
[codeartifact]
repository = https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/
pypi/my_repo/
password = auth-token
username = aws
```

Untuk menggunakan variabel lingkungan, lakukan hal berikut.

### Note

Jika Anda mengakses repositori di domain yang Anda miliki, Anda tidak perlu menyertakan `--domain-owner`. Untuk informasi selengkapnya, lihat [Domain lintas akun](#).

```
export TWINE_USERNAME=aws
export TWINE_PASSWORD=`aws codeartifact get-authorization-token --domain my_domain --
domain-owner 111122223333 --query authorizationToken --output text`
export TWINE_REPOSITORY_URL=`aws codeartifact get-repository-endpoint --
domain my_domain --domain-owner 111122223333 --repository my_repo --format pypi --query
repositoryEndpoint --output text`
```

## Menjalankan twine

Sebelum menggunakan benang untuk menerbitkan aset paket Python, Anda harus mengkonfigurasi terlebih dahulu CodeArtifact dan sumber daya.

1. Ikuti langkah-langkah di bagian [Mengatur dengan AWS CodeArtifact](#) untuk mengonfigurasi akun AWS, alat, dan izin.
2. Konfigurasi benang dengan mengikuti langkah-langkah di [Konfigurasi benang dengan logikomando](#) atau [Konfigurasi benang tanpa logikomando](#).

Setelah Anda mengkonfigurasi benang, Anda dapat menjalankan `twine` perintah. Gunakan perintah berikut untuk memublikasikan aset paket Python.

```
twine upload --repository codeartifact mypackage-1.0.tgz
```

Untuk informasi tentang cara membangun dan mengemas aplikasi Python Anda, lihat [Generating Distribution Archives](#) di situs web Python Packaging Authority.

## Normalisasi nama paket Python

CodeArtifact menormalkan nama paket sebelum menyimpannya, yang berarti nama paket di CodeArtifact mungkin berbeda dari nama yang diberikan saat paket diterbitkan.

Untuk paket Python, saat melakukan normalisasi, nama paket diberi huruf kecil dan semua instance karakter `.`, `-`, dan `_` diganti dengan satu karakter. Jadi nama paketnya `pigeon_cli` dan `pigeon.cl` dinormalisasi dan disimpan sebagai `pigeon-cli`. Nama yang tidak dinormalisasi dapat digunakan oleh pip dan benang tetapi nama yang dinormalisasi harus digunakan di CodeArtifact Permintaan CLI atau API (seperti `list-package-versions`) dan di ARN. Untuk informasi selengkapnya tentang normalisasi nama paket Python, lihat [SEMANGAT 503](#) dalam dokumentasi Python.

## Kompatibilitas Python

CodeArtifact tidak mendukung PyPI XML-RPC atau JSON API.

CodeArtifact mendukung PyPI Legacy API, kecuali `simple` API. Sementara CodeArtifact tidak mendukung `/simple/` titik akhir API, itu mendukung `/simple/<project>/` titik akhir.

Untuk informasi lebih lanjut, lihat berikut ini di Python Packaging Authority GitHub repositori.

- [XML-RPC API](#)
- [API JSON](#)
- [API Legacy](#)

## dukungan perintah pip

Bagian berikut merangkum perintah pip yang didukung, oleh CodeArtifact repositori, selain perintah tertentu yang tidak didukung.

Topik

- [Perintah yang didukung yang berinteraksi dengan repositori](#)
- [Perintah sisi klien yang didukung](#)

## Perintah yang didukung yang berinteraksi dengan repositori

Bagian ini berisi daftar `pip` tempat klien `pip` membuat satu atau beberapa permintaan ke registri yang telah dikonfigurasi. Perintah-perintah ini telah diverifikasi untuk berfungsi dengan benar ketika dipanggil terhadap CodeArtifact repositori.

Perintah	Deskripsi
<a href="#">menginstal</a>	Menginstal paket.
<a href="#">unduh</a>	Mengunduh paket.

CodeArtifact tidak menerapkan `pip search`. Jika Anda telah mengonfigurasi `pip` dengan a CodeArtifact repositori, berjalan `pip search` akan mencari dan menampilkan paket dari [PyPI](#).

## Perintah sisi klien yang didukung

Perintah ini tidak memerlukan interaksi langsung dengan repositori, jadi CodeArtifact tidak perlu melakukan apa pun untuk mendukung mereka.

Perintah	Deskripsi
<a href="#">hapus instalasi</a>	Meng-uninstall paket.
<a href="#">membeku</a>	Output paket terinstal dalam format persyaratan.
<a href="#">daftar</a>	Daftar paket terinstal.
<a href="#">memperlihatkan</a>	Menampilkan informasi tentang paket terinstal.
<a href="#">memeriksa</a>	Memverifikasi apakah paket terinstal memiliki dependensi yang kompatibel.
<a href="#">konfigurasi</a>	Mengelola konfigurasi lokal dan global.



Perintah	Deskripsi
<a href="#">roda</a>	Membangun wheel dari kebutuhan Anda.
<a href="#">pagar</a>	Menghitung hash dari arsip paket.
<a href="#">penyelesaian</a>	Membantu penyelesaian perintah.
<a href="#">awakutu</a>	Menampilkan informasi yang berguna untuk debugging.
help	Menampilkan bantuan untuk perintah.

## Meminta paket Python dari upstream dan koneksi eksternal

Saat mengimpor versi paket Python [dari](#) pypi.org CodeArtifact, akan mengimpor semua aset dalam versi paket itu. Sementara sebagian besar paket Python berisi sejumlah kecil aset, beberapa berisi lebih dari 100 aset, biasanya untuk mendukung beberapa arsitektur perangkat keras dan interpreter Python.

Adalah umum untuk aset baru untuk dipublikasikan ke pypi.org untuk versi paket yang ada. Misalnya, beberapa proyek menerbitkan aset baru ketika versi baru Python dirilis. Ketika paket Python diinstal dari CodeArtifact with `pip install`, versi paket yang disimpan dalam CodeArtifact repositori diperbarui untuk mencerminkan kumpulan aset terbaru dari pypi.org.

Demikian pula, jika aset baru tersedia untuk versi paket di repositori upstream yang tidak ada di CodeArtifact repositori saat ini, mereka akan disimpan di CodeArtifact repositori saat ini ketika dijalankan. `pip install`

## Versi paket yang ditarik

Beberapa versi paket di pypi.org ditandai sebagai ditarik, yang berkomunikasi dengan penginstal paket (seperti pip) bahwa versi tersebut tidak boleh diinstal kecuali itu adalah satu-satunya yang cocok dengan penentu versi (menggunakan salah satu atau). `==` Lihat [PEP\\_592](#) untuk informasi lebih lanjut.

Jika versi paket pada CodeArtifact awalnya diambil dari koneksi eksternal ke [pypi.org](#), saat Anda menginstal versi paket dari CodeArtifact repositori, pastikan bahwa metadata menarik yang CodeArtifact diperbarui dari versi paket diambil dari pypi.org.

## Bagaimana cara mengetahui apakah versi paket ditarik

Untuk memeriksa apakah versi paket ditarik CodeArtifact, Anda dapat mencoba menginstalnya. `pip install packageName==packageVersion` Jika versi paket ditarik, Anda akan menerima pesan peringatan yang mirip dengan berikut ini:

```
WARNING: The candidate selected for download or install is a yanked version
```

Untuk memeriksa apakah versi paket ditarik di [pypi.org](https://pypi.org), Anda dapat mengunjungi daftar pypi.org versi paket di <https://pypi.org/project/packageName/packageVersion/>

## Menyetel status yang ditarik pada paket pribadi

CodeArtifact tidak mendukung pengaturan metadata yang ditarik untuk paket yang diterbitkan langsung ke repositori. CodeArtifact

## Mengapa CodeArtifact tidak mengambil metadata atau aset terbaru yang ditarik untuk versi paket?

Biasanya, CodeArtifact memastikan bahwa ketika versi paket Python diambil dari CodeArtifact repositori, metadata yang ditarik adalah dengan nilai terbaru di pypi.org. up-to-date Selain itu, daftar aset dalam versi paket juga terus diperbarui dengan set terbaru di pypi.org dan repositori hulu CodeArtifact apa pun. Ini benar apakah Anda menginstal versi paket untuk pertama kalinya dan CodeArtifact mengimpornya dari pypi.org ke CodeArtifact repositori Anda, atau jika Anda telah menginstal paket sebelumnya. Namun, ada kasus ketika klien manajer paket, seperti pip, tidak akan menarik metadata terbaru yang ditarik dari pypi.org atau repositori upstream. Sebaliknya, CodeArtifact akan mengembalikan data yang sudah disimpan di repositori Anda. Bagian ini menjelaskan tiga cara hal ini dapat terjadi:

**Konfigurasi hulu:** Jika koneksi eksternal ke pypi.org dihapus dari repositori atau upstream menggunakan [disassociate-external-connection](#), metadata yang ditarik tidak akan lagi di-refresh dari pypi.org. Demikian pula, jika Anda menghapus repositori upstream, aset dari repositori yang dihapus dan upstream repositori yang dihapus tidak akan lagi tersedia untuk repositori saat ini. Hal yang sama berlaku jika Anda menggunakan [kontrol asal CodeArtifact paket](#) untuk mencegah versi baru dari paket tertentu ditarik—pengaturan `upstream=BLOCK` akan memblokir metadata yang ditarik agar tidak disegarkan.

**Status versi paket:** Jika Anda menyetel status versi paket ke apa pun kecuali `Published` atau `Unlisted`, metadata dan aset yang ditarik dari versi paket tidak akan di-refresh. Demikian

pula, jika Anda mengambil versi paket tertentu (katakanlah `torch 2.0.1`) dan versi paket yang sama ada di repositori upstream dengan status yang tidak `Published` atau `Unlisted`, ini juga akan memblokir metadata yang ditarik dan propagasi aset dari repositori upstream ke repositori saat ini. Ini karena status versi paket lainnya merupakan indikasi bahwa versi tersebut tidak dimaksudkan untuk dikonsumsi lagi di repositori apa pun.

**Penerbitan langsung:** Jika Anda mempublikasikan versi paket tertentu langsung ke CodeArtifact repositori, ini akan mencegah metadata yang ditarik dan penyegaran aset untuk versi paket dari repositori hulu dan `pypi.org`. Misalnya, Anda mengunduh aset dari versi paket `torch 2.0.1`, seperti `torch-2.0.1-cp311-none-macosx_11_0_arm64.whl`, menggunakan browser web dan kemudian mempublikasikannya ke CodeArtifact repositori Anda menggunakan benang sebagai `torch 2.0.1`. CodeArtifact melacak bahwa versi paket memasuki domain dengan menerbitkan langsung ke repositori Anda, bukan dari koneksi eksternal ke `pypi.org` atau repositori upstream. Dalam hal ini, metadata yang ditarik CodeArtifact tidak disinkronkan dengan repositori upstream atau `pypi.org`. Hal yang sama berlaku jika Anda mempublikasikan `torch 2.0.1` ke repositori upstream—keberadaan versi paket akan memblokir propagasi metadata dan aset yang ditarik ke repositori lebih jauh ke bawah grafik hulu.

# Menggunakan CodeArtifact dengan Maven

Format repositori Maven digunakan oleh banyak bahasa yang berbeda, termasuk Java, Kotlin, Scala, dan Clojure. Ini didukung oleh berbagai alat pembangun, termasuk Maven, Gradle, Scala SBT, Apache Ivy, dan Leiningen.

Kami telah menguji dan mengkonfirmasi kompatibilitas dengan CodeArtifact untuk versi berikut:

- Maven terbaru versi: 3.6.3.
- Gradle terbaru versi: 6.4.1. 5.5.1 juga telah diuji.
- Versi Clojure terbaru: 1.11.1 juga telah diuji.

## Topik

- [GunakanCodeArtifactdengan Gradle](#)
- [Gunakan CodeArtifact dengan mvn](#)
- [Gunakan CodeArtifact dengan deps.edn](#)
- [Publikasi dengan curl](#)
- [Menggunakan checksum Maven](#)
- [Menggunakan snapshot Maven](#)
- [Meminta paket Maven dari upstream dan koneksi eksternal](#)
- [Pemecahan masalah Maven](#)

## GunakanCodeArtifactdengan Gradle

Setelah Anda memilikiCodeArtifacttoken autentikasi dalam variabel lingkungan seperti yang dijelaskan dalam[Lulus token autentikasi menggunakan variabel lingkungan](#), ikuti petunjuk ini untuk menggunakan paket Maven dari, dan menerbitkan paket baru ke, aCodeArtifactrepositori.

## Topik

- [Mengambil dependensi](#)
- [Mengambil plugin](#)
- [Memublikasikan artefak](#)
- [Menjalankan build Gradle di IntelliJ IDEA](#)

## Mengambil dependensi

Untuk mengambil dependensi dari CodeArtifact dalam build Gradle, gunakan prosedur berikut.

Untuk mengambil dependensi dari CodeArtifact dalam build Gradle

1. Jika belum, buat dan simpan CodeArtifact token autentikasi dalam variabel lingkungan dengan mengikuti prosedur di [Teruskan token auth menggunakan variabel lingkungan](#).
2. Menambahkan maven bagian ke repositories bagian dalam proyek build.gradle berkas.

```
maven {
    url 'https://my_domain-111122223333.d.codeartifact.region.amazonaws.com/
maven/my_repo/'
    credentials {
        username "aws"
        password System.env.CODEARTIFACT_AUTH_TOKEN
    }
}
```

The url Dalam contoh sebelumnya adalah CodeArtifact titik akhir repositori. Gradle menggunakan titik akhir untuk terhubung ke repositori Anda. Dalam sampel, `my_domain` adalah nama domain Anda, `111122223333` adalah ID pemilik domain, dan `my_repo` adalah nama repositori Anda. Anda dapat mengambil titik akhir repositori dengan menggunakan perintah `get-repository-endpoint` AWS CLI.

Misalnya, dengan repositori bernama `my_repo` di dalam domain bernama `my_domain`, perintah yang digunakan adalah sebagai berikut:

```
aws codeartifact get-repository-endpoint --domain my_domain --domain-
owner 111122223333 --repository my_repo --format maven
```

Parameter perintah `get-repository-endpoint` akan mengembalikan titik akhir repositori:

```
url 'https://my_domain-111122223333.d.codeartifact.region.amazonaws.com/
maven/my_repo/'
```

The `credentials` objek dalam contoh sebelumnya termasuk CodeArtifact token autentikasi yang Anda buat di Langkah 1 yang digunakan Gradle untuk mengautentikasi CodeArtifact.

3. (Opsional) - Untuk menggunakan CodeArtifact repository sebagai satu-satunya sumber untuk dependensi proyek Anda, hapus bagian lain di `repositories` dari `build.gradle`. Jika Anda memiliki lebih dari satu repository, Gradle menelusuri setiap repository untuk dependensi sesuai urutan pencantumannya.
4. Setelah mengonfigurasi repository, Anda dapat menambahkan dependensi proyek ke bagian `dependencies` dengan sintaks Gradle standar.

```
dependencies {
    implementation 'com.google.guava:guava:27.1-jre'
    implementation 'commons-cli:commons-cli:1.4'
    testImplementation 'org.testng:testng:6.14.3'
}
```

## Mengambil plugin

Secara default Gradle akan menyelesaikan plugin dari [Portal Plugin Gradle](#) publik. Untuk menarik plugin dari CodeArtifact repository, gunakan prosedur berikut.

Untuk menarik plugin dari CodeArtifact repository

1. Jika belum, buat dan simpan CodeArtifact token autentikasi dalam variabel lingkungan dengan mengikuti prosedur di [Teruskan token auth menggunakan variabel lingkungan](#).
2. Menambahkan `pluginManagement` blok ke `Anda/settings.gradle` berkas. The `pluginManagement` blok harus muncul sebelum pernyataan lain di `settings.gradle`, lihat cuplikan berikut:

```
pluginManagement {
    repositories {
        maven {
            name 'my_repo'
            url
            'https://my_domain-111122223333.d.codeartifact.region.amazonaws.com/
maven/my_repo/'
            credentials {
                username 'aws'
                password System.env.CODEARTIFACT_AUTH_TOKEN
            }
        }
    }
}
```

```
}
```

Hal ini akan memastikan bahwa Gradle menyelesaikan plugin dari repositori yang ditentukan. Repositori harus memiliki repositori hulu dengan koneksi eksternal ke Portal Plugin Gradle (mis. `gradle-plugins-store`) sehingga plugin Gradle yang umum diperlukan tersedia untuk build. Untuk informasi selengkapnya, lihat [Dokumentasi Gradle](#).

## Memublikasikan artefak

Bagian ini menjelaskan cara memublikasikan pustaka Java yang dibangun dengan Gradle ke CodeArtifact repository.

Pertama, tambahkan plugin `maven-publish` ke bagian `plugins` dari file `build.gradle` proyek.

```
plugins {  
    id 'java-library'  
    id 'maven-publish'  
}
```

Selanjutnya, tambahkan bagian `publishing` ke file `build.gradle` proyek.

```
publishing {  
    publications {  
        mavenJava(MavenPublication) {  
            groupId = 'group-id'  
            artifactId = 'artifact-id'  
            version = 'version'  
            from components.java  
        }  
    }  
    repositories {  
        maven {  
            url 'https://my_domain-111122223333.d.codeartifact.region.amazonaws.com/  
maven/my_repo/'  
            credentials {  
                username "aws"  
                password System.env.CODEARTIFACT_AUTH_TOKEN  
            }  
        }  
    }  
}
```

Plugin `maven-publish` menghasilkan file POM berdasarkan `groupId`, `artifactId`, dan `version` yang ditentukan dalam bagian `publishing`.

Setelah perubahan ke `build.gradle` selesai, jalankan perintah berikut untuk membangun proyek dan mengunggah ke repositori.

```
./gradlew publish
```

Gunakan `list-package-versions` untuk memeriksa bahwa paket berhasil dipublikasikan.

```
aws codeartifact list-package-versions --domain my_domain --domain-owner 111122223333
--repository my_repo --format maven\
--namespace com.company.framework --package my-package-name
```

Contoh output:

```
{
  "format": "maven",
  "namespace": "com.company.framework",
  "package": "example",
  "versions": [
    {
      "version": "1.0",
      "revision": "REVISION-SAMPLE-1-C7F4S5E9B772FC",
      "status": "Published"
    }
  ]
}
```

Untuk informasi selengkapnya, lihat topik berikut di situs web Gradle:

- [Membangun Perpustakaan Java](#)
- [Menerbitkan proyek sebagai modul](#)

## Menjalankan build Gradle di IntelliJ IDEA

Anda dapat menjalankan build Gradle di IntelliJ IDEA yang menarik dependensi dari CodeArtifact. Untuk mengautentikasi dengan CodeArtifact, Anda harus memberikan Gradle dengan CodeArtifact token otorisasi. Ada tiga metode untuk menyediakan token auth.



- Metode 1: Menyimpan token autentikasi di `gradle.properties`. Gunakan metode ini jika Anda dapat menimpa atau menambah `isgradle.properties` berkas.
- Metode 2: Menyimpan token autentikasi dalam file terpisah. Gunakan metode ini jika Anda tidak ingin memodifikasi file `gradle.properties`.
- Metode 3: Menghasilkan token autentikasi baru untuk setiap proses dengan menjalankan `aws` sebagai skrip inline di `build.gradle`. Gunakan metode ini jika Anda ingin skrip Gradle mengambil token baru pada setiap proses. Token tidak akan disimpan di sistem file.

## Token stored in `gradle.properties`

### Metode 1: Menyimpan token autentikasi di `gradle.properties`

#### Note

Contoh menunjukkan file `gradle.properties` yang berada di `GRADLE_USER_HOME`.

1. Perbarui file `build.gradle` dengan cuplikan berikut:

```
repositories {
    maven {
        url
        'https://my_domain-111122223333.d.codeartifact.region.amazonaws.com/
maven/my_repo/'
        credentials {
            username "aws"
            password "$codeartifactToken"
        }
    }
}
```

2. Untuk mengambil plugin dari CodeArtifact, tambahkan `pluginManagement` blok ke `Andasettings.gradle` berkas. The `pluginManagement` blok harus muncul sebelum pernyataan lain di `settings.gradle`.

```
pluginManagement {
    repositories {
        maven {
            name 'my_repo'
```

```

        url
        'https://my_domain-111122223333.codeartifact.region.amazonaws.com/
maven/my_repo/'
        credentials {
            username 'aws'
            password "$codeartifactToken"
        }
    }
}
}

```

### 3. Ambil CodeArtifact token autentikasi:

```

export CODEARTIFACT_AUTH_TOKEN=`aws codeartifact get-authorization-token --
domain my_domain --domain-owner 111122223333 --query authorizationToken --output
text --profile profile-name`

```

### 4. Tulis token auth ke dalam file gradle.properties:

```

echo "codeartifactToken=$CODEARTIFACT_AUTH_TOKEN" > ~/.gradle/gradle.properties

```

## Token stored in separate file

### Metode 2: Menyimpan token autentikasi dalam file terpisah

#### 1. Perbarui file build.gradle dengan cuplikan berikut:

```

def props = new Properties()
file("file").withInputStream { props.load(it) }

repositories {

    maven {
        url
        'https://my_domain-111122223333.d.codeartifact.region.amazonaws.com/
maven/my_repo/'
        credentials {
            username "aws"
            password props.getProperty("codeartifactToken")
        }
    }
}

```

```
}

```

2. Untuk mengambil plugin dari CodeArtifact, tambahkan pluginManagement blok ke `Anda/settings.gradle` berkas. The `pluginManagement` blok harus muncul sebelum pernyataan lain di `settings.gradle`.

```
pluginManagement {
    def props = new Properties()
    file("file").withInputStream { props.load(it) }
    repositories {
        maven {
            name 'my_repo'
            url
            'https://my_domain-111122223333.codeartifact.region.amazonaws.com/
maven/my_repo/'
            credentials {
                username 'aws'
                password props.getProperty("codeartifactToken")
            }
        }
    }
}

```

3. Ambil CodeArtifact token autentikasi:

```
export CODEARTIFACT_AUTH_TOKEN=`aws codeartifact get-authorization-token --
domain my_domain --domain-owner 111122223333 --query authorizationToken --output
text --profile profile-name`

```

4. Tulis token auth ke dalam file yang ditentukan dalam file `build.gradle` Anda:

```
echo "codeartifactToken=$CODEARTIFACT_AUTH_TOKEN" > file

```

Token generated for each run in `build.gradle`

Metode 3: Menghasilkan token autentikasi baru untuk setiap proses dengan menjalankan `aws` sebagai skrip inline di `build.gradle`

1. Perbarui file `build.gradle` dengan cuplikan berikut:

```
def codeartifactToken = "aws codeartifact get-authorization-token --
domain my_domain --domain-owner 111122223333 --query authorizationToken --output
text --profile profile-name".execute().text
    repositories {
        maven {
            url
            'https://my_domain-111122223333.d.codeartifact.region.amazonaws.com/
maven/my_repo/'
            credentials {
                username "aws"
                password codeartifactToken
            }
        }
    }
}
```

2. Untuk mengambil plugin dari CodeArtifact, tambahkan `pluginManagement` blok ke `AndSettings.gradle` berkas. The `pluginManagement` blok harus muncul sebelum pernyataan lain di `settings.gradle`.

```
pluginManagement {
    def codeartifactToken = "aws codeartifact get-authorization-token --
domain my_domain --domain-owner 111122223333 --query authorizationToken --output
text --profile profile-name".execute().text
    repositories {
        maven {
            name 'my_repo'
            url
            'https://my_domain-111122223333.codeartifact.region.amazonaws.com/
maven/my_repo/'
            credentials {
                username 'aws'
                password codeartifactToken
            }
        }
    }
}
```

# Gunakan CodeArtifact dengan mvn

Anda menggunakan perintah mvn untuk mengeksekusi build Maven. Bagian ini menunjukkan cara mengkonfigurasi mvn untuk menggunakan CodeArtifact repositori.

Topik

- [Mengambil dependensi](#)
- [Memublikasikan artefak](#)
- [Memublikasikan artefak pihak ketiga](#)
- [Batasi unduhan ketergantungan Maven ke repositori CodeArtifact](#)
- [Informasi Proyek Apache Maven](#)

## Mengambil dependensi

Untuk mengonfigurasi mvn untuk mengambil dependensi dari CodeArtifact repositori, Anda harus mengedit file konfigurasi Maven, dan secara opsional `settings.xml`, POM proyek Anda.

1. Jika belum, buat dan simpan token CodeArtifact autentikasi dalam variabel lingkungan seperti yang dijelaskan [Teruskan token auth menggunakan variabel lingkungan](#) untuk menyiapkan autentikasi ke repositori Anda CodeArtifact .
2. Dalam `settings.xml` (biasanya ditemukan di `~/.m2/settings.xml`), tambahkan bagian `<servers>` dengan referensi ke variabel lingkungan `CODEARTIFACT_AUTH_TOKEN` sehingga Maven meneruskan token dalam permintaan HTTP.

```
<settings>
...
  <servers>
    <server>
      <id>codeartifact</id>
      <username>aws</username>
      <password>${env.CODEARTIFACT_AUTH_TOKEN}</password>
    </server>
  </servers>
...
</settings>
```

3. Tambahkan titik akhir URL untuk CodeArtifact repositori Anda dalam sebuah elemen `<repository>` Anda dapat melakukan hal ini di `settings.xml` atau file POM proyek Anda.

Anda dapat mengambil titik akhir repositori Anda menggunakan perintah `get-repository-endpoint` AWS CLI.

Misalnya, dengan repositori bernama `my_repo` di dalam domain bernama `my_domain`, perintah yang digunakan adalah sebagai berikut:

```
aws codeartifact get-repository-endpoint --domain my_domain --repository my_repo --format maven
```

Parameter perintah `get-repository-endpoint` akan mengembalikan titik akhir repositori:

```
url 'https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/maven/my_repo/'
```

Tambahkan titik akhir repositori ke `settings.xml` sebagai berikut.

```
<settings>
...
  <profiles>
    <profile>
      <id>default</id>
      <repositories>
        <repository>
          <id>codeartifact</id>
          <url>https://my_domain-111122223333.d.codeartifact.us-
west-2.amazonaws.com/maven/my_repo/</url>
        </repository>
      </repositories>
    </profile>
  </profiles>
  <activeProfiles>
    <activeProfile>default</activeProfile>
  </activeProfiles>
  ...
</settings>
```

Atau, Anda dapat menambahkan `<repositories>` bagian ke file POM proyek untuk digunakan hanya CodeArtifact untuk proyek itu.

```
<project>
```

```
...
  <repositories>
    <repository>
      <id>codeartifact</id>
      <name>codeartifact</name>
      <url>https://my_domain-111122223333.d.codeartifact.us-
west-2.amazonaws.com/maven/my_repo/</url>
    </repository>
  </repositories>
...
</project>
```

### Important

Anda dapat menggunakan nilai apa pun dalam elemen `<id>`, tetapi harus sama di elemen `<server>` dan `<repository>`. Hal ini memungkinkan kredensi yang ditentukan untuk dimasukkan dalam permintaan untuk CodeArtifact

Setelah membuat perubahan konfigurasi ini, Anda dapat membangun proyek.

```
mvn compile
```

Maven mencatat URL lengkap dari semua dependensi yang diunduhnya ke konsol.

```
[INFO] -----< com.example.example:myapp >-----
[INFO] Building myapp 1.0
[INFO] -----[ jar ]-----
Downloading from codeartifact: https://<domain>.d.codeartifact.us-west-2.amazonaws.com/
maven/myrepo/commons-cli/commons-cli/1.4/commons-cli-1.4.pom
Downloaded from codeartifact: https://<domain>.d.codeartifact.us-west-2.amazonaws.com/
maven/myrepo/commons-cli/commons-cli/1.4/commons-cli-1.4.pom (11 kB at 3.9 kB/s)
Downloading from codeartifact: https://<domain>.d.codeartifact.us-west-2.amazonaws.com/
maven/myrepo/org/apache/commons/commons-parent/42/commons-parent-42.pom
Downloading from codeartifact: https://<domain>.d.codeartifact.us-west-2.amazonaws.com/
maven/myrepo/org/apache/commons/commons-parent/42/commons-parent-42.pom
Downloaded from codeartifact: https://<domain>.d.codeartifact.us-west-2.amazonaws.com/
maven/myrepo/org/apache/commons/commons-parent/42/commons-parent-42.pom (68 kB at 123
kB/s)
Downloading from codeartifact: https://<domain>.d.codeartifact.us-west-2.amazonaws.com/
maven/myrepo/commons-cli/commons-cli/1.4/commons-cli-1.4.jar
```

Downloaded from codeartifact: https://<domain>.d.codeartifact.us-west-2.amazonaws.com/maven/myrepo/commons-cli/commons-cli/1.4/commons-cli-1.4.jar (54 kB at 134 kB/s)

## Memublikasikan artefak

Untuk memublikasikan artefak Maven dengan mvn ke CodeArtifact repositori, Anda juga harus mengedit ~/.m2/settings.xml dan proyek POM.

1. Jika belum, buat dan simpan token CodeArtifact autentikasi dalam variabel lingkungan seperti yang dijelaskan [Teruskan token auth menggunakan variabel lingkungan](#) untuk menyiapkan autentikasi ke repositori Anda CodeArtifact .
2. Tambahkan bagian <servers> ke settings.xml dengan referensi ke variabel lingkungan CODEARTIFACT\_AUTH\_TOKEN sehingga Maven meneruskan token dalam permintaan HTTP.

```
<settings>
...
  <servers>
    <server>
      <id>codeartifact</id>
      <username>aws</username>
      <password>${env.CODEARTIFACT_AUTH_TOKEN}</password>
    </server>
  </servers>
...
</settings>
```

3. Tambahkan bagian <distributionManagement> ke pom.xml.

```
<project>
...
  <distributionManagement>
    <repository>
      <id>codeartifact</id>
      <name>codeartifact</name>
      <url>https://my_domain-111122223333.d.codeartifact.us-
west-2.amazonaws.com/maven/my_repo/</url>
    </repository>
  </distributionManagement>
...
</project>
```



Setelah membuat perubahan konfigurasi ini, Anda dapat membangun proyek dan memublikasikannya ke repositori yang ditentukan.

```
mvn deploy
```

Gunakan `list-package-versions` untuk memeriksa bahwa paket berhasil dipublikasikan.

```
aws codeartifact list-package-versions --domain my_domain --domain-owner 111122223333
--repository my_repo --format maven \
--namespace com.company.framework --package my-package-name
```

Contoh output:

```
{
  "defaultDisplayVersion": null,
  "format": "maven",
  "namespace": "com.company.framework",
  "package": "my-package-name",
  "versions": [
    {
      "version": "1.0",
      "revision": "REVISION-SAMPLE-1-C7F4S5E9B772FC",
      "status": "Published"
    }
  ]
}
```

## Memublikasikan artefak pihak ketiga

Anda dapat memublikasikan artefak Maven pihak ketiga ke repositori dengan CodeArtifact `.mvn deploy:deploy-file`. Hal ini dapat membantu pengguna yang ingin memublikasikan artefak dan hanya memiliki file JAR dan tidak memiliki akses ke kode sumber paket atau file POM.

Perintah `mvn deploy:deploy-file` akan menghasilkan file POM berdasarkan informasi yang dikirimkan pada baris perintah.

### Memublikasikan artefak Maven pihak ketiga

1. Jika belum, buat dan simpan token CodeArtifact autentikasi dalam variabel lingkungan seperti yang dijelaskan [Teruskan token auth menggunakan variabel lingkungan](#) untuk menyiapkan otentikasi ke repositori Anda CodeArtifact .

2. Buat file `~/.m2/settings.xml` dengan konten berikut ini:

```
<settings>
  <servers>
    <server>
      <id>codeartifact</id>
      <username>aws</username>
      <password>${env.CODEARTIFACT_AUTH_TOKEN}</password>
    </server>
  </servers>
</settings>
```

3. Jalankan perintah `mvn deploy:deploy-file`:

```
mvn deploy:deploy-file -DgroupId=commons-cli \
-DartifactId=commons-cli \
-Dversion=1.4 \
-Dfile=./commons-cli-1.4.jar \
-Dpackaging=jar \
-DrepositoryId=codeartifact \
-Durl=https://my_domain-111122223333.d.codeartifact.region.amazonaws.com/
maven/repo-name/
```

#### Note

Contoh di atas memublikasikan `commons-cli 1.4`. Ubah `groupId`, `artifactId`, `versi`, dan `file` argumen untuk memublikasikan JAR yang berbeda.

Instruksi ini didasarkan pada contoh dalam [Panduan men-deploy JAR pihak ketiga ke repositori jarak jauh](#) dari Dokumentasi Apache Maven.

## Batasi unduhan ketergantungan Maven ke repositori CodeArtifact

Jika paket tidak dapat diambil dari repositori yang dikonfigurasi, secara default, perintah `mvn` mengambilnya dari Maven Central. Tambahkan `mirrors` elemen `settings.xml` untuk membuat `mvn` selalu menggunakan CodeArtifact repositori Anda.

```
<settings>
  ...
  <mirrors>
    <mirror>
      <id>central-mirror</id>
      <name>CodeArtifact Maven Central mirror</name>
      <url>https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/
maven/my_repo/</url>
      <mirrorOf>central</mirrorOf>
    </mirror>
  </mirrors>
  ...
</settings>
```

Jika Anda menambahkan elemen `mirrors`, Anda juga harus memiliki elemen `pluginRepository` di `settings.xml` atau `pom.xml`. Contoh berikut mengambil dependensi aplikasi dan plugin Maven dari repositori. CodeArtifact

```
<settings>
  ...
  <profiles>
    <profile>
      <pluginRepositories>
        <pluginRepository>
          <id>codeartifact</id>
          <name>CodeArtifact Plugins</name>
          <url>https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/
maven/my_repo/</url>
          <releases>
            <enabled>true</enabled>
          </releases>
          <snapshots>
            <enabled>true</enabled>
          </snapshots>
        </pluginRepository>
      </pluginRepositories>
    </profile>
  </profiles>
  ...
</settings>
```

Contoh berikut mengambil dependensi aplikasi dari CodeArtifact repositori dan mengambil plugin Maven dari Maven Central.

```
<profiles>
  <profile>
    <id>default</id>
    ...
    <pluginRepositories>
      <pluginRepository>
        <id>central-plugins</id>
        <name>Central Plugins</name>
        <url>https://repo.maven.apache.org/maven2/</url>
        <releases>
          <enabled>true</enabled>
        </releases>
        <snapshots>
          <enabled>true</enabled>
        </snapshots>
      </pluginRepository>
    </pluginRepositories>
    ....
  </profile>
</profiles>
```

## Informasi Proyek Apache Maven

Untuk informasi lebih lanjut tentang Maven, lihat topik ini di situs web Proyek Apache Maven:

- [Menyiapkan Beberapa Repositori](#)
- [Pengaturan Referensi](#)
- [Manajemen Distribusi](#)
- [Profil](#)

## Gunakan CodeArtifact dengan deps.edn

Anda gunakan `deps.edn` dengan `clj` untuk mengelola dependensi untuk proyek Clojure. Bagian ini menunjukkan cara mengonfigurasi `deps.edn` untuk menggunakan CodeArtifact repositori.

Topik

- [Mengambil dependensi](#)

- [Memublikasikan artefak](#)

## Mengambil dependensi

Untuk mengonfigurasi Clojure untuk mengambil dependensi dari CodeArtifact repositori, Anda harus mengedit file konfigurasi `MavenSettings.xml`.

1. Di `settings.xml`, tambahkan `<servers>` bagian dengan referensi ke variabel `CODEARTIFACT_AUTH_TOKEN` lingkungan sehingga Clojure melewati token dalam permintaan HTTP.

### Note

Clojure mengharapkan file `settings.xml` akan berada di `~/.m2/settings.xml`. Jika di tempat lain, buat file di lokasi ini.

```
<settings>
...
  <servers>
    <server>
      <id>codeartifact</id>
      <username>aws</username>
      <password>${env.CODEARTIFACT_AUTH_TOKEN}</password>
    </server>
  </servers>
...
</settings>
```

2. Jika Anda tidak memiliki satu sudah, menghasilkan POM xml untuk proyek Anda menggunakan `clj -Spom`.
3. Dalam file `deps.edn` konfigurasi Anda, tambahkan repositori yang cocok dengan id server dari `MavenSettings.xml`.

```
:mvn/repos {
  "clojars" nil
  "central" nil
  "codeartifact" {:url "https://my_domain-111122223333.d.codeartifact.us-
west-2.amazonaws.com/maven/my_repo/"}
```

```
}
```

### Note

- `tools.deps` menjamin bahwa `central` dan `clojars` repositori akan diperiksa pertama untuk perpustakaan Maven. Setelah itu, repositori lain yang tercantum dalam `deps.edn` akan diperiksa.
- Untuk mencegah download dari Clojars dan Maven Central langsung, `central` dan `clojars` perlu diatur `kenil`.

Pastikan Anda memiliki token CodeArtifact Auth dalam variabel lingkungan (lihat [Teruskan token auth menggunakan variabel lingkungan](#)). Ketika membangun paket setelah perubahan ini, dependensi di `deps.edn` akan diambil dari CodeArtifact.

## Memublikasikan artefak

1. Memperbarui pengaturan Maven Anda dan `deps.edn` untuk menyertakan CodeArtifact sebagai server maven-diakui (lihat [Mengambil dependensi](#)). Anda dapat menggunakan alat seperti [deps-deploy](#) untuk mengunggah artefak ke CodeArtifact.
2. Di `Anda/build.clj`, tambahkan `deploy` tugas untuk mengunggah artefak yang diperlukan ke codeartifact repositori pengaturan sebelumnya.

```
(ns build
  (:require [deps-deploy.deps-deploy :as dd]))

(defn deploy [_]
  (dd/deploy {:installer :remote
              :artifact "PATH_TO_JAR_FILE.jar"
              :pom-file "pom.xml" ;; pom containing artifact coordinates
              :repository "codeartifact"}))
```

3. Publikasikan artefak dengan menjalankan perintah: `clj -T:build deploy`

Untuk informasi lebih lanjut tentang memodifikasi repositori default, lihat [Memodifikasi repositori default](#) di Clojure Deps dan CLI Reference Rationale.

## Publikasi dengan curl

Bagian ini menunjukkan cara menggunakan klien HTTP `curl` untuk menerbitkan artefak Maven ke CodeArtifact repositori. Memublikasikan artefak dengan `curl` dapat berguna jika Anda tidak memiliki atau ingin menginstal klien Maven di lingkungan Anda.

Memublikasikan artefak Maven dengan **curl**

1. Mengambil CodeArtifact token otorisasi dengan mengikuti langkah-langkah di [Teruskan token auth menggunakan variabel lingkungan](#) dan kembali ke langkah-langkah ini.
2. Gunakan berikut `curl` perintah untuk memublikasikan JAR ke CodeArtifact repositori:

Di masing-masing `curl` perintah dalam prosedur ini, mengganti placeholder berikut:

- Ganti `my_domain` dengan CodeArtifact nama domain.
- Ganti `111122223333` dengan ID dari pemilik CodeArtifact domain.
- Ganti `us-west-2` dengan wilayah tempat CodeArtifact domain berada.
- Ganti `my_repo` dengan CodeArtifact nama repositori.

```
curl --request PUT https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/maven/my_repo/com/mycompany/app/my-app/1.0/my-app-1.0.jar \
  --user "aws:$CODEARTIFACT_AUTH_TOKEN" --header "Content-Type: application/octet-stream" \
  --data-binary @my-app-1.0.jar
```

### Important

Anda harus awalan nilai `--data-binary` parameter dengan `@` karakter. Ketika menempatkan nilai dalam tanda petik, `@` harus disertakan di dalam tanda kutip.

3. Gunakan berikut `curl` perintah untuk memublikasikan POM ke CodeArtifact repositori:

```
curl --request PUT https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/maven/my_repo/com/mycompany/app/my-app/1.0/my-app-1.0.pom \
  --user "aws:$CODEARTIFACT_AUTH_TOKEN" --header "Content-Type: application/octet-stream" \
  --data-binary @my-app-1.0.pom
```

4. Pada titik ini, artefak Maven akan berada di artefak Anda CodeArtifact repositori dengan status `Unfinished`. Untuk dapat menggunakan paket, artefak tersebut harus berstatus `Published`. Anda dapat memindahkan paket dari `Unfinished` ke `Published` dengan mengunggah file `maven-metadata.xml` ke paket Anda, atau memanggil [API UpdatePackageVersionsStatus](#) untuk mengubah status.
  - a. Opsi 1: Gunakan berikut `curl` perintah untuk menambahkan `maven-metadata.xml` file ke paket Anda:

```
curl --request PUT
  https://my_domain-111122223333.d.codeartifact.region.amazonaws.com/
  maven/my_repo/com/mycompany/app/my-app/maven-metadata.xml \
    --user "aws:$CODEARTIFACT_AUTH_TOKEN" --header "Content-Type: application/
  octet-stream" \
    --data-binary @maven-metadata.xml
```

Berikut ini adalah contoh isi file `maven-metadata.xml`:

```
<metadata modelVersion="1.1.0">
  <groupId>com.mycompany.app</groupId>
  <artifactId>my-app</artifactId>
  <versioning>
    <latest>1.0</latest>
    <release>1.0</release>
    <versions>
      <version>1.0</version>
    </versions>
    <lastUpdated>20200731090423</lastUpdated>
  </versioning>
</metadata>
```

- b. Opsi 2: Perbarui status paket ke `Published` dengan `UpdatePackageVersionsStatus` API.

```
aws codeartifact update-package-versions-status \
  --domain my_domain \
  --domain-owner 111122223333 \
  --repository my_repo \
  --format maven \
  --namespace com.mycompany.app \
  --package my-app \
```



```
--versions 1.0 \  
--target-status Published
```

Jika Anda hanya memiliki file JAR artefak ini, Anda dapat memublikasikan versi paket yang dapat digunakan ke CodeArtifact repositori menggunakan mvn. Hal ini dapat berguna jika Anda tidak memiliki akses ke kode sumber artefak atau POM. Lihat [Memublikasikan artefak pihak ketiga](#) untuk rincian selengkapnya.

## Menggunakan checksum Maven

Ketika artefak Maven dipublikasikan ke AWS CodeArtifact repositori, checksum yang terkait dengan masing-masing aset atau file dalam paket digunakan untuk memvalidasi unggahan. Contoh aset adalah file jar, pom, dan war. Untuk setiap aset, artefak Maven berisi beberapa file checksum yang menggunakan nama aset dengan ekstensi tambahan, seperti md5 atau sha1. Sebagai contoh, file checksum untuk file bernama my-maven-package.jar mungkin my-maven-package.jar.md5 dan my-maven-package.jar.sha1.

### Note

Maven menggunakan istilah *artifact*. Dalam panduan ini, paket Maven sama dengan artefak Maven. Untuk informasi selengkapnya, lihat [AWS CodeArtifact paket](#).

## Penyimpanan checksum

CodeArtifact tidak menyimpan checksum Maven sebagai aset. Ini berarti bahwa checksum tidak muncul sebagai aset individual dalam output [ListPackageVersionAssets API](#). Sebagai gantinya, checksum yang dihitung oleh CodeArtifact tersedia untuk setiap aset di semua jenis checksum yang didukung. Misalnya, bagian dari respon panggilan ListPackageVersionAssets pada versi paket Mavencommons-lang:commons-lang 2.1 adalah:

```
{  
  "name": "commons-lang-2.1.jar",  
  "size": 207723,  
  "hashes": {  
    "MD5": "51591549f1662a64543f08a1d4a0cf87",  
    "SHA-1": "4763ecc9d78781c915c07eb03e90572c7ff04205",
```

```

    "SHA-256": "2ded7343dc8e57decd5e6302337139be020fdd885a2935925e8d575975e480b9",
    "SHA-512":
"a312a5e33b17835f2e82e74ab52ab81f0dec01a7e72a2ba58bb76b6a197ffcd2bb410e341ef7b3720f3b595ce49fd
  }
},
{
  "name": "commons-lang-2.1.pom",
  "size": 9928,
  "hashes": {
    "MD5": "8e41bacdd69de9373c20326d231c8a5d",
    "SHA-1": "a34d992202615804c534953aba402de55d8ee47c",
    "SHA-256": "f1a709cd489f23498a0b6b3dfbfc0d21d4f15904791446dec7f8a58a7da5bd6a",
    "SHA-512":
"1631ce8fe4101b6cde857f5b1db9b29b937f98ba445a60e76cc2b8f2a732ff24d19b91821a052c1b56b73325104e9
  }
},
{
  "name": "maven-metadata.xml",
  "size": 121,
  "hashes": {
    "MD5": "11bb3d48d984f2f49cea1e150b6fa371",
    "SHA-1": "7ef872be17357751ce65cb907834b6c5769998db",
    "SHA-256": "d04d140362ea8989a824a518439246e7194e719557e8d701831b7f5a8228411c",
    "SHA-512":
"001813a0333ce4b2a47cf44900470bc2265ae65123a8c6b5ac5f2859184608596baa4d8ee0696d0a497755dade0f6
  }
}

```

Meskipun checksum tidak disimpan sebagai aset, klien Maven masih dapat mempublikasikan dan mengunduh checksum di lokasi yang diharapkan. Misalnya, jika `commons-lang:commons-lang-2.1` berada di repositori yang disebut `maven-repo`, jalur URL untuk checksum SHA-256 dari file JAR adalah:

```
/maven/maven-repo/commons-lang/commons-lang/2.1/commons-lang-2.1.jar.sha256
```

Jika Anda mengunggah paket Maven yang ada (misalnya, paket yang sebelumnya disimpan di Amazon S3) untuk CodeArtifact menggunakan klien HTTP generik seperti `curl`, Anda tidak perlu mengunggah checksum. CodeArtifact akan menghasilkannya secara otomatis. Jika Anda ingin memverifikasi bahwa aset telah diunggah dengan benar, Anda dapat menggunakan operasi `ListPackageVersionAssets` API untuk membandingkan checksum dalam respons terhadap nilai checksum asli untuk setiap aset.

## Checksum tidak cocok selama penerbitan

Terlepas dari aset dan checksum, artefak Maven juga berisimaven-metadata.xml file. Urutan penerbitan normal untuk paket Maven adalah untuk semua aset dan checksum yang akan di-upload pertama, diikuti olehmaven-metadata.xml. Misalnya, urutan penerbitan untuk versi paket Maven yangcommons-lang 2.1 dijelaskan sebelumnya, dengan asumsi klien dikonfigurasi untuk menerbitkan file checksum SHA-256, akan menjadi:

```
PUT commons-lang-2.1.jar
PUT commons-lang-2.1.jar.sha256
PUT commons-lang-2.1.pom
PUT commons-lang-2.1.pom.sha256
PUT maven-metadata.xml
PUT maven-metadata.xml.sha256
```

Saat mengunggah file checksum untuk aset, seperti file JAR, permintaan unggahan checksum akan gagal dengan respons 400 (Permintaan Buruk) jika ada ketidakcocokan antara nilai checksum yang diunggah dan nilai checksum yang dihitung oleh CodeArtifact. Jika aset terkait tidak ada, permintaan akan gagal dengan respons 404 (Not Found). Untuk menghindari kesalahan ini, Anda harus terlebih dahulu mengunggah aset, dan kemudian mengunggah checksum.

Ketikamaven-metadata.xml di-upload, CodeArtifact biasanya mengubah status versi paket Maven dariUnfinished kePublished. Jika ketidakcocokan checksum terdeteksi untuk aset apa pun, CodeArtifact akan mengembalikan 400 (Permintaan Buruk) dalam menanggapi permintaanmaven-metadata.xml penerbitan. Kesalahan ini dapat menyebabkan klien berhenti mengunggah file untuk versi paket tersebut. Jika ini terjadi, danmaven-metadata.xml file tidak diunggah, aset apa pun dari versi paket yang sudah diunggah tidak dapat diunduh. Hal ini karena status versi paket tidak diatur kePublished dan tetapUnfinished.

CodeArtifact memungkinkan menambahkan aset lebih lanjut ke versi paket Maven bahkan setelahmaven-metadata.xml telah di-upload dan status versi paket telah diatur kePublished. Dalam status ini, permintaan untuk mengunggah file checksum yang tidak cocok juga akan gagal dengan respons 400 (Permintaan Buruk). Namun, karena status versi paket telah diaturPublished, Anda dapat mengunduh aset apa pun dari paket, termasuk aset yang gagal diunggah file checksum. Saat mengunduh checksum untuk aset di mana unggahan file checksum gagal, nilai checksum yang diterima klien akan menjadi nilai checksum yang dihitung CodeArtifact berdasarkan data aset yang diunggah.

CodeArtifact perbandingan checksum bersifat case sensitive, dan checksum yang dihitung oleh CodeArtifact diformat dalam huruf kecil. Oleh karena itu, jika checksum909FA780F76DA393E992A3D2D495F468 diunggah, itu akan gagal dengan ketidakcocokan checksum karena CodeArtifact tidak memperlakukannya sama dengan909fa780f76da393e992a3d2d495f468.

## Memulihkan dari checksum

Jika unggahan checksum gagal karena ketidakcocokan checksum, coba salah satu dari berikut ini untuk memulihkan:

- Jalankan perintah yang menerbitkan artefak Maven lagi. Langkah ini mungkin bekerja jika masalah jaringan merusak file checksum. Jika masalah jaringan, checksum cocok dan unduhan berhasil.
- Hapus versi paket dan kemudian mempublikasikannya kembali. Untuk informasi selengkapnya, lihat [DeletePackageVersions](#) di AWS CodeArtifact API Reference.

## Menggunakan snapshot Maven

Snapshot Maven adalah versi khusus dari paket Maven yang mengacu pada kode cabang produksi terbaru. Ini adalah versi pengembangan sebelum versi rilis akhir. Anda dapat mengidentifikasi versi snapshot paket Maven dengan akhiranSNAPSHOT yang ditambahkan ke versi paket. Sebagai contoh, snapshot versi 1.1 adalah 1.1-SNAPSHOT. Untuk informasi selengkapnya, lihat [What is a SNAPSHOT version?](#) di situs Apache Maven Project.

AWSCodeArtifactmendukung lilibikasi dan penggunaan snapshot Maven. Snapshot unik yang menggunakan nomor versi berbasis waktu adalah satu-satunya snapshot yang didukung. CodeArtifacttidak mendukung snapshot non-unik yang dihasilkan oleh Maven 2 klien. Anda dapat mempublikasikan snapshot Maven didukung untuk setiapCodeArtifact repositori.

### Topik

- [Penerbitan snapshot diCodeArtifact](#)
- [Mengonsumsi versi snapshot](#)
- [Menghapus hapus versi snapshot](#)
- [Publilililikasi snapshot dengan curl](#)
- [Snapshot dan koneksi eksternal](#)
- [Snapshot dan repositori hulu hulu](#)

## Penerbitan snapshot diCodeArtifact

AWSCodeArtifactmendukung pola permintaan yang klien, sepertimvn, gunakan saat menerbitkan snapshot. Karena itu, Anda dapat mengikuti dokumentasi untuk alat build atau manajer paket Anda tanpa memiliki pemahaman rinci tentang bagaimana snapshot Maven diterbitkan. Jika Anda melakukan sesuatu yang lebih kompleks, bagian ini menjelaskan secara rinci bagaimanaCodeArtifact menangani snapshot.

Ketika snapshot Maven dipublikasikan keCodeArtifact repositori Maven dipertahankan di versi baru yang disebut build. Setiap kali snapshot Maven dipublikasikan, versi build baru dibuat. Semua versi snapshot sebelumnya dipertahankan dalam versi build. Ketika snapshot Maven dipublikasikan, status versi pakatnya diatur kePublished dan status build yang berisi versi sebelumnya diatur keUnlisted. Perilaku ini hanya berlaku untuk versi paket Maven di mana versi paket memiliki-SNAPSHOT sebagai akhiran.

Misalnya, versi snapshot dari paket maven yang disebutcom.mycompany.myapp:pkg-1 diunggah keCodeArtifact repositori yang disebutmy-maven-repo. Versi snapshot adalah1.0-SNAPSHOT. Sejauh ini,com.mycompany.myapp:pkg-1 belum ada versi yang diterbitkan. Pertama, aset build awal diterbitkan di jalur ini:

```
PUT maven/my-maven-repo/com/mycompany/myapp/pkg-1/1.0-SNAPSHOT/
pkg-1-1.0-20210728.194552-1.jar
PUT maven/my-maven-repo/com/mycompany/myapp/pkg-1/1.0-SNAPSHOT/
pkg-1-1.0-20210728.194552-1.pom
```

Perhatikan bahwa stempel waktu20210728.194552-1 dihasilkan oleh klien yang menerbitkan build snapshot.

Setelah file.pom dan .jar diunggah, satu-satunya versicom.mycompany.myapp:pkg-1 yang ada di repositori adalah1.0-20210728.194552-1. Hal ini terjadi meskipun versi yang ditentukan di jalur sebelumnya adalah1.0-SNAPSHOT. Status versi paket pada saat ini adalahUnfinished.

```
aws codeartifact list-package-versions --domain my-domain --repository \
my-maven-repo --package pkg-1 --namespace com.mycompany.myapp --format maven
{
  "versions": [
    {
      "version": "1.0-20210728.194552-1",
      "revision": "GipMW+599JmwTcTLaXo9YvDsVQ2bcrrk/02rWJhoKUU=",
      "status": "Unfinished"
    }
  ]
}
```

```

    }
  ],
  "defaultDisplayVersion": null,
  "format": "maven",
  "package": "pkg-1",
  "namespace": "com.mycompany.myapp"
}

```

Selanjutnya, klien mengunggah `maven-metadata.xml` file untuk versi paket:

```
PUT my-maven-repo/com/mycompany/myapp/pkg-1/1.0-SNAPSHOT/maven-metadata.xml
```

Ketika file `maven-metadata.xml` berhasil di-upload, CodeArtifact menciptakan versi `1.0-SNAPSHOT` paket dan menetapkan `1.0-20210728.194552-1` versi untuk `Unlisted`.

```

aws codeartifact list-package-versions --domain my-domain --repository \
my-maven-repo --package pkg-1 --namespace com.mycompany.myapp --format maven
{
  "versions": [
    {
      "version": "1.0-20210728.194552-1",
      "revision": "GipMW+599JmwTcTLaXo9YvDsVQ2bcrrk/02rWJhoKUU=",
      "status": "Unlisted"
    },
    {
      "version": "1.0-SNAPSHOT",
      "revision": "tWu8n3IX5HR82vzVZQAx1wcvvA4U/+S80edWNAki124=",
      "status": "Published"
    }
  ],
  "defaultDisplayVersion": "1.0-SNAPSHOT",
  "format": "maven",
  "package": "pkg-1",
  "namespace": "com.mycompany.myapp"
}

```

Pada titik ini, versi snapshot `1.0-SNAPSHOT` dapat dikonsumsi dalam build. Meskipun ada dua versi `com.mycompany.myapp:pkg-1` di repositori `my-maven-repo`, keduanya mengandung aset yang sama.

```
aws codeartifact list-package-version-assets --domain my-domain --repository \
```

```
my-maven-repo --format maven --namespace com.mycompany.myapp \  
--package pkg-1 --package-version 1.0-SNAPSHOT--query 'assets[*].name'  
[  
  "pkg-1-1.0-20210728.194552-1.jar",  
  "pkg-1-1.0-20210728.194552-1.pom"  
]
```

Menjalankan `list-package-version-assets` perintah yang sama seperti yang ditunjukkan sebelumnya dengan `--package-version` parameter berubah menjadi `1.0-20210728.194552-1` hasil dalam output yang identik.

Karena build tambahan `1.0-SNAPSHOT` ditambahkan ke repositori, versi `Unlisted` paket baru dibuat untuk setiap build baru. Aset versi `1.0-SNAPSHOT` diperbarui setiap kali sehingga versi selalu mengacu pada build terbaru untuk versi tersebut. Memperbarui `1.0-SNAPSHOT` dengan aset terbaru dimulai dengan mengunggah `maven-metadata.xml` file untuk build baru.

## Mengonsumsi versi snapshot

Jika Anda meminta snapshot, versi dengan status `Published` dikembalikan. Ini selalu merupakan versi terbaru snapshot Maven. Anda juga dapat meminta build snapshot tertentu menggunakan nomor versi build (misalnya, `1.0-20210728.194552-1`), bukan versi snapshot (misalnya, `1.0-SNAPSHOT`) di jalur URL. Untuk melihat versi build snapshot Maven, gunakan [ListPackageVersions](#) API di Panduan CodeArtifact API dan tetapkan parameter status ke `Unlisted`.

## Menghapus versi snapshot

Untuk menghapus semua versi build dari snapshot Maven, gunakan [DeletePackageVersions](#) API, tentukan versi yang ingin Anda hapus.

## Publikasikan snapshot dengan curl

Jika Anda memiliki versi snapshot yang ada yang disimpan di Amazon Simple Storage Service (Amazon S3) atau produk repositori artefak lainnya, Anda mungkin ingin mempublikasikan ulang AWS CodeArtifact. Karena bagaimana CodeArtifact mendukung snapshot Maven (lihat [Penerbitan snapshot di CodeArtifact](#)), publikasi snapshot dengan klien HTTP generik seperti `curl` lebih kompleks daripada penerbitan versi rilis Maven seperti yang dijelaskan dalam [Publikasi dengan curl](#). Perhatikan bahwa bagian ini tidak relevan jika Anda membangun dan menerapkan versi snapshot dengan klien Maven seperti `mvn` atau `gradle`. Anda harus mengikuti dokumentasi untuk klien itu.

Menerbitkan versi snapshot melibatkan penerbitan satu atau lebih build dari versi snapshot.

Di CodeArtifact, jika ada  $n$  build dari versi snapshot, akan ada versi  $n + 1$  CodeArtifact:  $n$  versi build semua dengan status `Unlisted`, dan satu versi snapshot (build terbaru yang dipublikasikan) dengan status `Published`. Versi snapshot (yaitu, versi dengan string versi yang berisi “-SNAPSHOT”) berisi kumpulan aset yang identik dengan build terbaru yang diterbitkan. Cara termudah untuk membuat struktur ini dengan menggunakan `curl` adalah sebagai berikut:

1. Publikasikan semua aset dari semua build menggunakan `curl`.
2. Publikasikan `maven-metadata.xml` file build terakhir (yaitu, build dengan stempel tanggal-waktu terbaru) dengan `curl`. Ini akan membuat versi dengan “-SNAPSHOT” dalam string versi dan dengan set aset yang benar.
3. Gunakan [UpdatePackageVersionsStatus](#) API untuk menyetel status semua versi build yang tidak terbaru `Unlisted`.

Gunakan `curl` perintah berikut untuk mempublikasikan aset snapshot (seperti `file.jar` dan `.pom`) untuk versi `1.0-SNAPSHOT` snapshot paket `com.mycompany.app:pkg-1`:

```
curl --user "aws:$CODEARTIFACT_AUTH_TOKEN" -H "Content-Type: application/octet-stream" \
  -X PUT https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/maven/my_maven_repo/com/mycompany/app/pkg-1/1.0-SNAPSHOT/pkg-1-1.0-20210729.171330-2.jar \
  --data-binary @pkg-1-1.0-20210728.194552-1.jar
```

```
curl --user "aws:$CODEARTIFACT_AUTH_TOKEN" -H "Content-Type: application/octet-stream" \
  -X PUT https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/maven/my_maven_repo/com/mycompany/app/pkg-1/1.0-SNAPSHOT/pkg-1-1.0-20210729.171330-2.pom \
  --data-binary @pkg-1-1.0-20210728.194552-1.pom
```

Saat menggunakan contoh ini:

- Ganti `my_domain` dengan nama CodeArtifact domain Anda.
- Ganti `111122223333` dengan Akun AWS ID pemilik CodeArtifact domain Anda.
- Ganti `kami-barat-2` dengan tempat CodeArtifact domain Anda berada. Wilayah AWS
- Ganti `my_maven_repo` dengan nama CodeArtifact repositori Anda.





```

<snapshotVersion>
  <extension>jar</extension>
  <value>1.0-20210729.171330-2</value>
  <updated>20210729171330</updated>
</snapshotVersion>
<snapshotVersion>
  <extension>pom</extension>
  <value>1.0-20210729.171330-2</value>
  <updated>20210729171330</updated>
</snapshotVersion>
</snapshotVersions>
</versioning>
</metadata>

```

Setelah `maven-metadata.xml` dipublikasikan, langkah terakhir adalah mengatur semua versi build lainnya (yaitu, semua versi build terpisah dari build terbaru) agar memiliki status versi paket `Unlisted`. Misalnya, jika `1.0-SNAPSHOT` versi memiliki dua build, dengan build pertama `20210728.194552-1`, perintah untuk menyetel build tersebut `Unlisted` adalah:

```

aws codeartifact update-package-versions-status --domain my-domain --domain-owner
111122223333 \
  --repository my-maven-repo --format maven --namespace com.mycompany.app --package
pkg-1 \
  --versions 1.0-20210728.194552-1 --target-status Unlisted

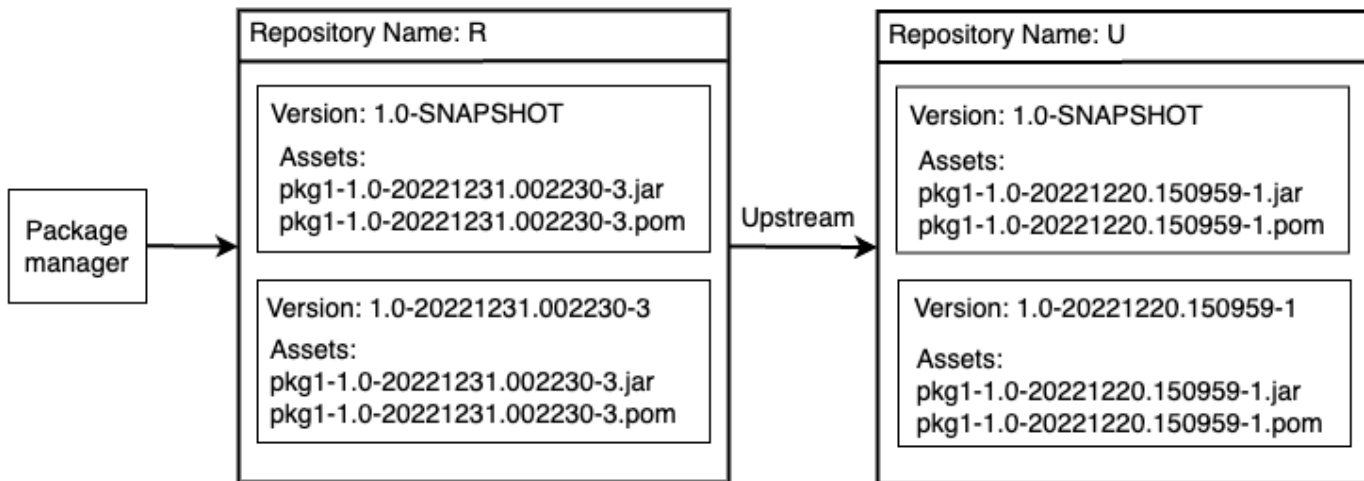
```

## Snapshot dan koneksi eksternal

snapshot Maven tidak dapat diambil dari repositori publik Maven melalui koneksi eksternal. AWS CodeArtifact hanya mendukung mengimpor versi rilis Maven.

## Snapshot dan repositori hulu hulu

Secara umum, snapshot Maven bekerja dengan cara yang sama seperti versi rilis Maven bila digunakan dengan repositori hulu. Misalnya, mengatakan bahwa ada dua repositori dalam `AWS CodeArtifact domain`, `R` dan `U`, di mana `U` adalah hulu dari `R`. Dalam situasi ini, Anda dapat dengan bebas mempublikasikan build snapshot dari paket tertentu (seperti `1.0-SNAPSHOT` dari `com.mycompany.app:pkg-1`) ke keduanya `R` dan `U`. Namun, ada beberapa perilaku penting yang harus dipahami saat mengonsumsi snapshot build dari `R` (repositori hilir).



1. Jika `1.0-SNAPSHOT` hadir di R, maka hanya aset `1.0-SNAPSHOT` di R dapat diambil dengan manajer paket yang dikonfigurasi untuk mengambil paket dari R. Anda tidak dapat mengambil aset `1.0-SNAPSHOT` dalam U melalui R. Ini karena versi snapshot di U dibayangi oleh versi di R. Perilaku ini identik dengan versi rilis Maven dan perilaku format paket lainnya. Dalam diagram, `GET of/maven/R/com/mycompany/myapp/pkg-1/1.0-SNAPSHOT/pkg-1-1.0-20221231.002230-3.jar` akan mengembalikan kode respon HTTP 200 (OK), tetapi `GET of/maven/R/com/mycompany/myapp/pkg-1/1.0-SNAPSHOT/pkg-1-1.0-20221220.150959-1.jar` akan mengembalikan kode respons HTTP 404 (Not Found).
2. Jika `1.0-SNAPSHOT` hadir dalam U tetapi tidak di R, Anda dapat menarik aset `1.0-SNAPSHOT` dari R. Hal ini akan menyebabkan `1.0-SNAPSHOT` untuk dipertahankan dalam R, sama seperti dengan versi rilis.
3. Setelah `1.0-SNAPSHOT` dipertahankan di R, Anda dapat mempublikasikan build tambahan `1.0-SNAPSHOT` di U. Namun, ini tidak akan dapat diakses dari R karena perilaku yang dijelaskan dalam poin (1). Ini berarti alasan standar untuk menggunakan versi snapshot, yaitu, menggunakan build terbaru dari dependensi melalui versi snapshot tertentu, tidak berfungsi seperti yang diharapkan di seluruh hubungan upstream. Meskipun build baru `1.0-SNAPSHOT` dipublikasikan di U, konsumen tidak dapat mengakses build terbaru `1.0-SNAPSHOT` dari R. Untuk bekerja di sekitar ini, baik secara berkala menghapus versi `1.0-SNAPSHOT` dalam R atau mengkonfigurasi klien untuk menarik versi dari `1.0-SNAPSHOT` dari U.
4. Versi build `Unlisted` snapshot dapat diakses dari repositori hulu. Dalam diagram, `GET of/maven/R/com/mycompany/myapp/pkg-1/1.0-20221220.150959-1/pkg-1-1.0-20221220.150959-1.jar` akan mengembalikan kode respon 200 (OK). Meskipun ini meminta aset yang ada di repositori upstream, karena versinya ditangani menggunakan string

versi build (1.0-20221220.150959-1), aset dapat diambil melalui repositori hilir. Ini juga akan menyebabkan versi dipertahankan, dengan status versi paket `Unlisted`.

## Meminta paket Maven dari upstream dan koneksi eksternal

### Mengimpor nama aset standar

Saat mengimpor versi paket Maven dari repositori publik, seperti Maven Central, AWS CodeArtifact mencoba mengimpor semua aset dalam versi paket tersebut. Seperti dijelaskan dalam [Meminta versi paket dengan repositori hulu](#), pengimporan terjadi ketika:

- Klien meminta aset Maven dari repositori. CodeArtifact
- Versi paket belum ada di repositori atau upstreams nya.
- Ada koneksi eksternal yang dapat dijangkau ke repositori Maven publik.

Meskipun klien mungkin hanya meminta satu aset, CodeArtifact upaya untuk mengimpor semua aset yang dapat ditemukan untuk versi paket tersebut. Bagaimana CodeArtifact menemukan aset mana yang tersedia untuk versi paket Maven tergantung pada repositori publik tertentu. Beberapa repositori Maven publik mendukung permintaan daftar aset, tetapi yang lain tidak. Untuk repositori yang tidak menyediakan cara untuk membuat daftar aset, CodeArtifact menghasilkan satu set nama aset yang mungkin ada. Misalnya, ketika aset apa pun dari versi `junit 4.13.2` paket Maven diminta, CodeArtifact akan mencoba mengimpor aset berikut:

- `junit-4.13.2.pom`
- `junit-4.13.2.jar`
- `junit-4.13.2-javadoc.jar`
- `junit-4.13.2-sources.jar`

### Mengimpor nama aset non-standar

Ketika klien Maven meminta aset yang tidak cocok dengan salah satu pola yang dijelaskan di atas, CodeArtifact memeriksa untuk melihat apakah aset tersebut ada di repositori publik. Jika aset ada, itu akan diimpor dan ditambahkan ke catatan versi paket yang ada, jika ada. Misalnya, versi paket `Maven com.android.tools.build:aapt2 7.3.1-8691043` berisi aset berikut:

- `aapt2-7.3.1-8691043.pom`
- `aapt2-7.3.1-8691043-windows.jar`
- `aapt2-7.3.1-8691043-osx.jar`
- `aapt2-7.3.1-8691043-linux.jar`

Ketika klien meminta file POM, jika CodeArtifact tidak dapat mencantumkan aset versi paket, POM akan menjadi satu-satunya aset yang diimpor. Ini karena tidak ada aset lain yang cocok dengan pola nama aset standar. Namun, ketika klien meminta salah satu aset JAR, aset tersebut akan diimpor dan ditambahkan ke versi paket yang ada yang disimpan di CodeArtifact. Versi paket di repositori paling hilir (repositori yang diminta klien) dan repositori dengan koneksi eksternal yang dilampirkan akan diperbarui untuk memuat aset baru, seperti yang dijelaskan dalam [Retensi paket dari repositori hulu](#)

Biasanya, setelah versi paket dipertahankan dalam CodeArtifact repositori, itu tidak terpengaruh oleh perubahan dalam repositori upstream. Untuk informasi selengkapnya, lihat [Retensi paket dari repositori hulu](#). Namun, perilaku aset Maven dengan nama non-standar yang dijelaskan sebelumnya merupakan pengecualian untuk aturan ini. Meskipun versi paket hilir tidak akan berubah tanpa aset tambahan yang diminta oleh klien, dalam situasi ini, versi paket yang dipertahankan dimodifikasi setelah awalnya dipertahankan dan karenanya tidak dapat diubah. Perilaku ini diperlukan karena aset Maven dengan nama non-standar tidak akan dapat diakses melalui CodeArtifact. Perilaku ini juga memungkinkan jika ditambahkan ke versi paket Maven di repositori publik setelah versi paket dipertahankan dalam repositori. CodeArtifact

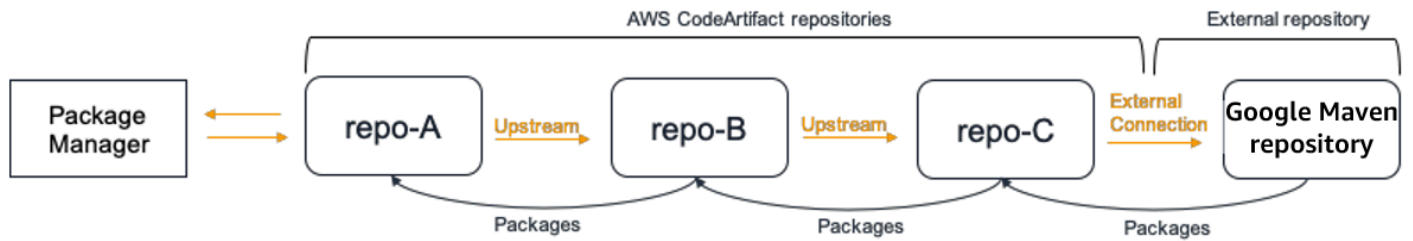
## Memeriksa asal-usul aset

Saat menambahkan aset baru ke versi paket Maven yang dipertahankan sebelumnya, CodeArtifact mengkonfirmasi asal versi paket yang ditahan sama dengan asal aset baru. Ini mencegah pembuatan versi paket “campuran” di mana aset yang berbeda berasal dari repositori publik yang berbeda. Tanpa pemeriksaan ini, pencampuran aset dapat terjadi jika versi paket Maven diterbitkan ke lebih dari satu repositori publik dan repositori tersebut adalah bagian dari grafik hulu repositori. CodeArtifact

## Mengimpor aset baru dan status versi paket di repositori hulu

[Status versi paket versi](#) paket di repositori hulu dapat CodeArtifact mencegah mempertahankan versi tersebut di repositori hilir.

Sebagai contoh, katakanlah sebuah domain memiliki tiga repositori: repo-A, dan repo-B repo-C, di mana repo-B adalah upstream dari repo-A dan repo-C upstream dari repo-B



Package 7.3.1 versi paket Maven `com.android.tools.build:aapt2` hadir di repo-B dan memiliki status `Published`. Itu tidak hadir di repo-A. Jika klien meminta aset versi paket ini dari repo-A, responsnya akan menjadi 200 (OK) dan versi paket Maven 7.3.1 akan dipertahankan di repo-A. Namun, jika status versi paket 7.3.1 di repo-B adalah `Archived` atau `Disposed`, responsnya akan menjadi 404 (Tidak Ditemukan) karena aset versi paket di kedua status tersebut tidak dapat diunduh.

Perhatikan bahwa menyetel [kontrol asal paket](#) ke `upstream=BLOCK` untuk `com.android.tools.build:aapt2` di repo-A, repo-B, dan repo-C akan mencegah aset baru diambil untuk semua versi paket tersebut di repo-A, terlepas dari status versi paket.

## Pemecahan masalah Maven

Informasi berikut dapat membantu Anda memecahkan masalah umum dengan Maven dan CodeArtifact.

### Nonaktifkan parallel puts untuk memperbaiki kesalahan 429: Terlalu Banyak Permintaan

Dimulai dengan versi 3.9.0, Maven mengunggah artefak paket secara paralel (hingga 5 file sekaligus). Hal ini dapat menyebabkan sesekali merespons dengan kode respons kesalahan 429 (Terlalu Banyak Permintaan). Jika Anda mengalami kesalahan ini, Anda dapat menonaktifkan parallel puts untuk memperbaikinya.

Untuk menonaktifkan parallel puts, atur `aeether.connector.basic.parallelPut` properti ke `false` dalam profil Anda di `settings.xml` file Anda seperti yang ditunjukkan oleh contoh berikut:

```
<settings>
  <profiles>
    <profile>
```

```
        <id>default</id>
        <properties>
            <aether.connector.basic.parallelPut>false</
aether.connector.basic.parallelPut>
        </properties>
    </profile>
</profiles>
<settings>
```

Untuk informasi selengkapnya, lihat [Opsi Konfigurasi Resolver Artifact](#) di dokumentasi Maven.

# Menggunakan CodeArtifact dengan NuGet

Topik ini menjelaskan cara menggunakan dan memublikasikan NuGet paket menggunakan CodeArtifact.

## Note

AWS CodeArtifact hanya mendukung [NuGet.exe 4.8](#) dan lebih tinggi.

Topik

- [Menggunakan CodeArtifact dengan Visual Studio](#)
- [Gunakan CodeArtifact dengan nuget atau dotnet CLI](#)
- [NuGet nama paket, versi, dan normalisasi nama aset](#)
- [Kompatibilitas NuGet](#)

# Menggunakan CodeArtifact dengan Visual Studio

Anda dapat menggunakan paket dari CodeArtifact secara langsung di Visual Studio dengan CodeArtifact Credential Provider. Penyedia kredensial menyederhanakan pengaturan dan autentikasi repositori CodeArtifact Anda dalam Visual Studio dan tersedia di [AWS Toolkit for Visual Studio](#).

## Note

Parameter AWS Toolkit for Visual Studio tidak tersedia untuk Visual Studio untuk Mac.

Untuk mengonfigurasi dan menggunakan NuGet dengan alat CLI, lihat [Gunakan CodeArtifact dengan nuget atau dotnet CLI](#).

Topik

- [Mengonfigurasi Visual Studio dengan CodeArtifact Credential Provider](#)
- [Menggunakan konsol Visual Studio Package Manager](#)



## Mengonfigurasi Visual Studio dengan CodeArtifact Credential Provider

CodeArtifact Credential Provider menyederhanakan pengaturan dan autentikasi berkelanjutan antara CodeArtifact dan Visual Studio. Token autentikasi CodeArtifact berlaku selama maksimal 12 jam. Agar tidak perlu menyegarkan token secara manual saat bekerja di Visual Studio, penyedia kredensial secara berkala mengambil token baru sebelum token saat ini berakhir.


### Important

Untuk menggunakan penyedia kredensial, pastikan bahwa semua kredensial AWS CodeArtifact yang ada dibersihkan dari file `nuget.config` yang mungkin telah ditambahkan secara manual atau dengan menjalankan `aws codeartifact login` untuk mengonfigurasi NuGet sebelumnya.

### Menggunakan CodeArtifact di Visual Studio dengan AWS Toolkit for Visual Studio

1. Instal AWS Toolkit for Visual Studio menggunakan langkah-langkah berikut. Kit alat kompatibel dengan Visual Studio 2017 dan 2019 menggunakan langkah-langkah berikut. AWS CodeArtifact tidak mendukung Visual Studio 2015 dan versi sebelumnya.
  1. Toolkit for Visual Studio untuk Visual Studio 2017 dan Visual Studio 2019 didistribusikan di [Marketplace Visual Studio](#). Anda juga dapat menginstal dan memperbarui kit alat dalam Visual Studio dengan menggunakan Tools (Alat) >> Extensions and Updates (Ekstensi dan Pembaruan) (Visual Studio 2017) atau Extensions (Ekstensi) >> Manage Extensions (Kelola Ekstensi) (Visual Studio 2019).
  2. Setelah kit alat diinstal, buka dengan memilih AWSExplorer dari menu View (Lihat).
2. Konfigurasi Toolkit for Visual Studio menggunakan kredensial AWS dengan mengikuti langkah-langkah di [Menyediakan Kredensial AWS](#) dalam Panduan Pengguna AWS Toolkit for Visual Studio.
3. (Opsional) Atur profil AWS yang ingin Anda gunakan dengan CodeArtifact. Jika tidak diatur, CodeArtifact akan menggunakan profil default. Untuk mengatur profil, buka Tools (Alat) > NuGet Package Manager (Manajer Paket NuGet) > Select CodeArtifact Profile (Pilih Profil AWS CodeArtifact).
4. Tambahkan repositori CodeArtifact Anda sebagai sumber paket dalam Visual Studio.

1. Navigasi ke repositori Anda di AWS Explorer, klik kanan dan pilih Copy NuGet Source Endpoint.
2. Gunakan perintah Tools (Alat) > Options (Opsi) dan gulir ke NuGet Package Manager (Manajer Paket NuGet).
3. Pilih simpul Package Sources (Sumber Paket).
4. Pilih +, edit nama, dan tempel titik akhir URL repositori yang disalin di Langkah 3a di kotak (Source) Sumber, dan pilih Update (Perbarui).
5. Pilih kotak centang untuk sumber paket yang baru ditambahkan untuk mengaktifkannya.

 Note

Sebaiknya tambahkan koneksi eksternal ke NuGet.org ke repositori CodeArtifact Anda dan nonaktifkan paket sumber nuget.org dalam Visual Studio. Ketika menggunakan koneksi eksternal, semua paket yang diambil dari NuGet.org akan disimpan dalam repositori CodeArtifact Anda. Jika NuGet.org menjadi tidak tersedia, dependensi aplikasi Anda masih akan tersedia untuk membangun CI dan pengembangan lokal. Untuk informasi selengkapnya tentang koneksi eksternal, lihat [Connect CodeArtifact repositori ke repositori publik](#).

5. Restart Visual Studio agar perubahan diterapkan.

Setelah konfigurasi, Visual Studio dapat menggunakan paket dari repositori CodeArtifact Anda, salah satu repositori hulu, atau dari [NuGet.org](#) jika Anda telah menambahkan koneksi eksternal. Untuk informasi selengkapnya tentang menjelajahi dan menginstal paket NuGet dalam Visual Studio, lihat [Install and manage packages in Visual Studio using the NuGet Package Manager](#) dalam Dokumentasi NuGet.

## Menggunakan konsol Visual Studio Package Manager

Konsol Visual Studio Package Manager tidak akan menggunakan versi Visual Studio CodeArtifact Credential Provider. Untuk menggunakannya, Anda harus mengonfigurasi penyedia kredensial baris perintah. Lihat [Gunakan CodeArtifact dengan nuget atau dotnet CLI](#) untuk informasi selengkapnya.

# Gunakan CodeArtifact dengan nuget atau dotnet CLI

Anda dapat menggunakan alat CLI seperti `nuget` dan `dotnet` untuk memublikasikan dan menggunakan paket dari CodeArtifact. Dokumen ini memberikan informasi tentang mengonfigurasi alat CLI dan menggunakannya untuk memublikasikan atau menggunakan paket.

## Topik

- [Mengonfigurasi nuget atau dotnet CLI](#)
- [Mengonsumsi NuGet paket dari CodeArtifact](#)
- [Publikasikan NuGet paket ke CodeArtifact](#)
- [CodeArtifact NuGet Referensi Penyedia Kredensial](#)
- [CodeArtifact NuGet Versi Credential Provider](#)

## Mengonfigurasi nuget atau dotnet CLI

Anda dapat mengonfigurasi nuget atau dotnet CLI dengan `CodeArtifactNuGet Credential Provider`, dengan `AWS CLI`, atau secara manual. Mengonfigurasi NuGet dengan penyedia kredensial sangat dianjurkan untuk pengaturan sederhana dan autentikasi berkelanjutan.

### Metode 1: Mengonfigurasi dengan Penyedia CodeArtifactNuGet Credential

`CodeArtifactNuGet Credential Provider` menyederhanakan otentikasi dan konfigurasi CodeArtifact dengan alat NuGet CLI. `CodeArtifact` token autentikasi berlaku selama maksimal 12 jam. Agar tidak perlu menyegarkan token secara manual saat menggunakan nuget atau dotnet CLI, penyedia kredensial secara berkala mengambil token baru sebelum token saat ini berakhir.

#### Important

Untuk menggunakan penyedia kredensial, pastikan bahwa semua `AWS CodeArtifact` kredensial yang ada dibersihkan dari `nuget.config` file yang mungkin telah ditambahkan secara manual atau dengan `aws codeartifact login` menjalankan konfigurasi NuGet sebelumnya.

## Instal dan konfigurasi Penyedia CodeArtifactNuGet Kredensial

## dotnet

1. Unduh versi terbaru dari [AWS. CodeArtifact. NuGet. CredentialProvider](#) dari [NuGet.org](#) dengan `dotnet` perintah berikut.

```
dotnet tool install -g AWS.CodeArtifact.NuGet.CredentialProvider
```

2. Gunakan `codeartifact-creds install` perintah untuk menyalin penyedia kredentials ke folder `NuGet` plugin.

```
dotnet codeartifact-creds install
```

3. (Opsional): Atur profil AWS yang ingin Anda gunakan dengan penyedia kredensial. Jika tidak diatur, penyedia kredensial akan menggunakan profil default. Untuk informasi selengkapnya tentang profil AWS CLI, lihat [Named profiles](#).

```
dotnet codeartifact-creds configure set profile profile_name
```

## nuget

Lakukan langkah-langkah berikut untuk menggunakan `NuGet CLI` untuk menginstal `CodeArtifactNuGet Credential Provider` dari bucket Amazon S3 dan mengonfigurasinya. Penyedia kredensial akan menggunakan profil AWS CLI default, untuk informasi selengkapnya tentang profil, lihat [Named profiles](#).

1. Unduh versi terbaru [PenyediaCodeArtifactNuGet Credential \(codeartifact-nuget-credentialprovider.zip\)](#) dari bucket Amazon S3.

Untuk melihat dan mengunduh versi sebelumnya, lihat [CodeArtifactNuGetVersi Credential Provider](#).

2. Buka filenya.
3. Salin `AWS. CodeArtifact. NuGetCredentialProvider` folder dari folder `netfx` ke `%user_profile%/.nuget/plugins/netfx/Windows` atau `~/ .nuget/plugins/netfx` di Linux atau macOS.
4. Salin `AWS. CodeArtifact. NuGetCredentialProvider` folder dari folder `netcore` ke `%user_profile%/.nuget/plugins/netcore/Windows` atau `~/ .nuget/plugins/netcore` di Linux atau macOS.

Setelah membuat repositori dan mengonfigurasi penyedia kredensial, Anda dapat menggunakan alat CLI nuget atau dotnet untuk menginstal dan memublikasikan paket. Untuk informasi selengkapnya, lihat [MengkonsumsiNuGet paket dariCodeArtifact](#) dan [PublikasikanNuGet paket keCodeArtifact](#).

## Metode 2: Mengonfigurasi nuget atau dotnet dengan perintah masuk

`codeartifact login`Perintah diAWS CLI menambahkan titik akhir repositori dan token otorisasi ke fileNuGet konfigurasi Anda. Hal ini memungkinkan nuget atau dotnet terhubung keCodeArtifact repositori Anda. Ini akan memodifikasiNuGet konfigurasi tingkat pengguna yang terletak di%appdata%\NuGet\NuGet.Config untuk Windows dan~/ .config/NuGet/NuGet.Config atau~/ .nuget/NuGet/NuGet.Config untuk Mac/Linux. Untuk informasi selengkapnya tentangNuGet konfigurasi, lihat [NuGetKonfigurasi umum](#).

### Mengonfigurasi nuget atau dotnet dengan perintah **login**

1. Konfigurasi kredensial AWS Anda untuk digunakan dengan AWS CLI, seperti yang dijelaskan dalam [Memulai dengan CodeArtifact](#).
2. Pastikan bahwa alatNuGet CLI (`nuget` atau `dotnet`) telah diinstal dan dikonfigurasi dengan benar. Untuk instruksi, lihat dokumentasi [nuget](#) atau [dotnet](#).
3. Gunakan`CodeArtifactlogin` perintah untuk mengambil kredensial untuk digunakan denganNuGet.

#### Note

Jika Anda mengakses repositori di domain milik Anda, Anda tidak perlu menyertakan `--domain-owner`. Untuk informasi selengkapnya, lihat [Domain lintas akun](#).

#### dotnet

#### Important

Pengguna Linux dan MacOS: Karena enkripsi tidak didukung pada platform non-Windows, kredensial yang diambil akan disimpan sebagai teks biasa dalam file konfigurasi Anda.

```
aws codeartifact login --tool dotnet --domain my_domain --domain-owner 111122223333 --repository my_repo
```

## nuget

```
aws codeartifact login --tool nuget --domain my_domain --domain-owner 111122223333 --repository my_repo
```

Perintah login akan:

- Mengambil token otorisasi dari CodeArtifact menggunakan AWS kredensial Anda.
- Memperbarui NuGet konfigurasi tingkat pengguna Anda dengan entri baru untuk sumber NuGet paket Anda. Sumber yang menunjuk ke titik akhir CodeArtifact repositori Anda akan dipanggil *domain\_name/repo\_name*.

Periode otorisasi default setelah memanggil login adalah 12 jam, dan login harus dipanggil untuk menyegarkan token secara berkala. Untuk informasi selengkapnya tentang token otorisasi yang dibuat dengan perintah login, lihat [Token dibuat dengan perintah login](#).

Setelah membuat repositori dan mengonfigurasi autentikasi, Anda dapat menggunakan nuget, dotnet, atau msbuild CLI klien untuk menginstal dan memublikasikan paket. Untuk informasi selengkapnya, lihat [Mengonsumsi NuGet paket dari CodeArtifact](#) dan [Publikasikan NuGet paket ke CodeArtifact](#).

## Metode 3: Mengonfigurasi nuget atau dotnet tanpa perintah masuk

Untuk konfigurasi manual, Anda harus menambahkan titik akhir repositori dan token otorisasi ke file NuGet konfigurasi untuk memungkinkan nuget atau dotnet terhubung ke CodeArtifact repositori Anda.

Konfigurasi nuget atau dotnet secara manual untuk terhubung ke CodeArtifact repositori Anda.

1. Tentukan endpoint CodeArtifact repositori Anda dengan menggunakan `get-repository-endpoint` AWS CLI perintah.

```
aws codeartifact get-repository-endpoint --domain my_domain --domain-owner 111122223333 --repository my_repo --format nuget
```

Contoh keluaran:

```
{  
  "repositoryEndpoint": "https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/nuget/my_repo/"  
}
```

2. Dapatkan token otorisasi untuk terhubung ke repositori Anda dari manajer paket menggunakan perintah `get-authorization-token` AWS CLI.

```
aws codeartifact get-authorization-token --domain my_domain
```

Contoh keluaran:

```
{  
  "authorizationToken": "eyJ2I...vi0w",  
  "expiration": 1601616533.0  
}
```

3. Buat URL endpoint repositori penuh dengan menambahkan `/v3/index.json` ke URL yang dikembalikan `get-repository-endpoint` pada langkah 3.
4. Konfigurasi nuget atau dotnet untuk menggunakan titik akhir repositori dari Langkah 1 dan token otorisasi dari Langkah 2.

#### Note

URL sumber harus diakhiri `/v3/index.json` agar nuget atau dotnet berhasil terhubung ke CodeArtifact repositori.

## dotnet

Pengguna Linux dan macOS: Karena enkripsi tidak didukung pada platform non-Windows, Anda harus menambahkan tanda `--store-password-in-clear-text` ke perintah berikut. Perhatikan bahwa ini akan menyimpan kata sandi Anda sebagai teks biasa dalam file konfigurasi Anda.

```
dotnet nuget add source https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/nuget/my_repo/v3/index.json --name packageName --password eyJ2I...vi0w --username aws
```

### Note

Untuk memperbarui sumber yang ada, gunakan `dotnet nuget update source` perintah.

## nuget

```
nuget sources add -name domain_name/repo_name -Source https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/nuget/my_repo/v3/index.json -password eyJ2I...vi0w -username aws
```

Contoh keluaran:

```
Package source with Name: domain_name/repo_name added successfully.
```

## MengonsumsiNuGet paket dariCodeArtifact

Setelah [mengonfigurasiNuGetCodeArtifact](#), Anda dapat menggunakanNuGet paket yang disimpan dalamCodeArtifact repositori atau salah satu repositori hulunya.

Untuk menggunakan versi paket dariCodeArtifact repositori atau salah satu repositori hulu dengannuget ataudotnet, jalankan perintah berikut untuk mengganti *packageName* dengan nama paket yang ingin Anda gunakan dan *packageName* dengan nama sumber untukCodeArtifact



repositori Anda dalam fileNuGet konfigurasi Anda. Jika Anda menggunakan `login` perintah untuk mengonfigurasiNuGet konfigurasi, nama sumber adalah *domain\_name/repo\_name*.

#### Note

Ketika sebuah paket diminta,NuGet klien cache versi paket yang ada. Karena perilaku ini, instalasi mungkin gagal untuk paket yang sebelumnya diminta sebelum versi yang diinginkan menjadi tersedia. Untuk menghindari kegagalan ini dan berhasil menginstal paket yang ada, Anda dapat menghapusNuGet cache di depan instalasi dengan `nuget locals all --clear` atau `dotnet nuget locals all --clear`, atau menghindari menggunakan cache selama `install` dan `restore` perintah dengan menyediakan `NoCache` opsi untuk `nuget` atau `--no-cache` opsi untuk `dotnet`.

#### dotnet

```
dotnet add package packageName --source packageSourceName
```

#### nuget

```
nuget install packageName -Source packageSourceName
```

Untuk memasang paket versi tertentu

#### dotnet

```
dotnet add package packageName --version 1.0.0 --source packageSourceName
```

#### nuget

```
nuget install packageName -Version 1.0.0 -Source packageSourceName
```

Lihat [Manage packages using the nuget.exe CLI](#) atau [Install and manage packages using the dotnet CLI](#) dalam Dokumentasi Microsoft untuk informasi selengkapnya.

## MengonsumsiNuGet paket dariNuGet .org

Anda dapat menggunakanNuGet paket dari [NuGet.org](https://www.nuget.org) melaluiCodeArtifact repositori dengan mengkonfigurasi repositori dengan koneksi eksternal ke NuGet.org. Paket yang digunakan dari NuGet.org diserap dan disimpan dalamCodeArtifact repositori Anda. Untuk informasi selengkapnya tentang penambahan sambungan eksternal, lihat [Connect CodeArtifact repositori ke repositori publik](#).

## PublikasikanNuGet paket keCodeArtifact

Setelah Anda telah [dikonfigurasiNuGet denganCodeArtifact](#), Anda dapat menggunakannuget ataudotnet untuk mempublikasikan versi paket keCodeArtifact repositori.

Untuk mendorong versi paket keCodeArtifact repositori, jalankan perintah berikut dengan jalur lengkap ke .nupkg file dan nama sumber untukCodeArtifact repositori Anda dalam fileNuGet konfigurasi Anda. Jika Anda menggunakanlogin perintah untuk mengkonfigurasiNuGet konfigurasi Anda, nama sumbernya adalahdomain\_name/repo\_name.

### Note

Anda dapat membuatNuGet paket jika Anda tidak memilikinya untuk dipublikasikan. Untuk informasi selengkapnya, lihat [alur kerja pembuatan Package](#) di dokumentasi Microsoft.

### dotnet

```
dotnet nuget push path/to/nupkg/SamplePackage.1.0.0.nupkg --source packageSourceName
```

### nuget

```
nuget push path/to/nupkg/SamplePackage.1.0.0.nupkg -Source packageSourceName
```

## CodeArtifactNuGetReferensi Penyedia Kredensial

CodeArtifactNuGetCredential Provider memudahkan konfigurasi dan autentikasiNuGet denganCodeArtifact repositori Anda.

## CodeArtifactNuGetPerintah Penyedia Kredensial

Bagian ini mencakup daftar perintah untuk CodeArtifactNuGet Credential Provider. Perintah ini harus diawali dengan `dotnet codeartifact-creds` seperti contoh berikut.

```
dotnet codeartifact-creds command
```

- `configure set profile profile`: Mengonfigurasi penyedia kredensial untuk menggunakan profil AWS yang disediakan.
- `configure unset profile`: Menghapus profil yang dikonfigurasi jika diatur.
- `install`: Menyalin penyedia kredensial ke folder `plugins`.
- `install --profile profile`: Menyalin penyedia kredensial ke folder `plugins` dan mengonfigurasinya untuk menggunakan profil AWS.
- `uninstall`: Meng-uninstall penyedia kredensial. Ini tidak menghapus perubahan ke file konfigurasi.
- `uninstall --delete-configuration`: Meng-uninstall penyedia kredensial dan menghapus semua perubahan ke file konfigurasi.

## CodeArtifactNuGetLog Penyedia Kredensi

Guna mengaktifkan pencatatan untuk Penyedia CodeArtifactNuGet Credential, Anda harus mengatur file log di lingkungan Anda. Log penyedia kredensial berisi informasi debugging yang bermanfaat seperti:

- Profil AWS yang digunakan untuk membuat koneksi
- Kesalahan autentikasi apa pun
- Jika titik akhir yang diberikan bukan CodeArtifact URL

### Mengatur file log CodeArtifactNuGet Credential Provider

```
export AWS_CODEARTIFACT_NUGET_LOGFILE=/path/to/file
```

Setelah file log diatur, setiap perintah `codeartifact-creds` akan menambahkan output log untuk isi file tersebut.

## CodeArtifactNuGetVersi Credential Provider

Tabel berikut berisi informasi riwayat versi dan tautan unduhan untuk PenyediaCodeArtifactNuGet Kredensi.

Versi	Perubahan	Tanggal diterbitkan	Tautan unduhan (S3)
1.0.1 (terbaru)	Menambahkan dukungan untuk profil net5, net6, dan SSO	03/05/2022	<a href="#">Unduh v1.0.1</a>
1.0.0	Rilis PenyediaCodeArtifactNuGet Kredensi Awal	11/20/2020	<a href="#">Unduh v1.0.0</a>

## NuGetnama paket, versi, dan normalisasi nama aset

CodeArtifactmenormalkan nama paket dan aset dan versi paket sebelum menyimpannya, yang berarti nama atau versi diCodeArtifactmungkin berbeda dari yang disediakan saat paket atau aset diterbitkan.

Normalisasi nama paket: CodeArtifactmenormalisasiNuGetnama paket dengan mengubah semua huruf menjadi huruf kecil.

Normalisasi versi paket: CodeArtifactmenormalisasiNuGetversi paket menggunakan pola yang sama sepertiNuGet. Informasi berikut ini dari[Nomor versi normal](#)dariNuGetdokumentasi.

- Memimpin nol dihapus dari nomor versi:
  - 1.00diperlakukan sebagai1.0
  - 1.01.1diperlakukan sebagai1.1.1
  - 1.00.0.1diperlakukan sebagai1.0.0.1
- Nol di bagian keempat dari nomor versi akan dihilangkan:
  - 1.0.0.0diperlakukan sebagai1.0.0
  - 1.0.01.0diperlakukan sebagai1.0.1
- SemVer2.0.0 build metadata dihapus:
  - 1.0.7+r3456diperlakukan sebagai1.0.7

Paket nama aset normalisasi: CodeArtifactmembangunNuGetnama aset paket dari nama paket dinormalisasi dan versi paket.

Nama paket dan nama versi yang tidak dinormalisasi dapat digunakan dengan permintaan API dan CLI karenaCodeArtifactmelakukan normalisasi pada nama paket dan input versi untuk permintaan tersebut. Misalnya, input`--package Newtonsoft.JSONdan--version 12.0.03.0`akan dinormalisasi dan mengembalikan paket yang memiliki nama paket dinormalisasinewtonsoft.jsondan versi12.0.3.

Anda harus menggunakan nama aset paket yang dinormalisasi dalam permintaan API dan CLI sebagaiCodeArtifacttidak melakukan normalisasi pada`--asset`masukan.

Anda harus menggunakan nama dan versi yang dinormalisasi di ARN.

Untuk menemukan nama paket yang dinormalisasi, gunakanaws codeartifact list-packagesperintah. Untuk informasi selengkapnya, lihat [Mencantumkan nama paket](#).

Untuk menemukan nama paket yang tidak dinormalisasi, gunakanaws codeartifact describe-package-versionperintah. Nama paket yang tidak dinormalisasi dikembalikan dalamdisplayNamebidang. Untuk informasi selengkapnya, lihat [Melihat dan memperbarui detail versi paket dan dependensi](#).

## Kompatibilitas NuGet

Panduan ini berisi informasi tentang kompatibilitas CodeArtifact dengan alat dan versi NuGet yang berbeda.

Topik

- [Kompatibilitas NuGet umum](#)
- [Dukungan baris perintah NuGet](#)

### Kompatibilitas NuGet umum

AWS CodeArtifact mendukung NuGet 4.8 dan lebih tinggi.

AWS CodeArtifact hanya mendukung V3 dari protokol NuGet HTTP. Ini berarti bahwa beberapa perintah CLI yang mengandalkan V2 protokol tidak didukung. Lihat bagian [dukungan perintah nuget.exe](#) untuk informasi selengkapnya.

AWS CodeArtifact tidak mendukung PowerShellGet 2.x.

## Dukungan baris perintah NuGet

AWS CodeArtifact mendukung alat CLI NuGet (`nuget.exe`) dan NET Core (`dotnet`).

### dukungan perintah `nuget.exe`

Karena CodeArtifact hanya mendukung V3 protokol HTTP NuGet, perintah berikut tidak akan bekerja jika digunakan terhadap sumber daya CodeArtifact:

- `list`: Perintah `nuget list` menampilkan daftar paket dari sumber tertentu. Untuk mendapatkan daftar paket dalam repositori CodeArtifact, Anda dapat menggunakan perintah [Mencantumkan nama paket](#) dari AWS CLI.

# Menggunakan CodeArtifact dengan Swift

Topik-topik ini menjelaskan cara menggunakan Swift Package Manager CodeArtifact untuk menginstal dan menerbitkan paket Swift.

## Note

CodeArtifact mendukung Swift 5.8 dan yang lebih baru dan Xcode 14.3 dan yang lebih baru. CodeArtifact merekomendasikan Swift 5.9 dan yang lebih baru dan Xcode 15 dan yang lebih baru.

## Topik

- [Konfigurasi Swift Package Manager dengan CodeArtifact](#)
- [Mengonsumsi dan menerbitkan paket Swift](#)
- [Nama paket Swift dan normalisasi namespace](#)
- [Pemecahan masalah cepat](#)

## Konfigurasi Swift Package Manager dengan CodeArtifact

Untuk menggunakan Swift Package Manager untuk memublikasikan paket ke atau menggunakan paket dari AWS CodeArtifact, Anda harus terlebih dahulu menyiapkan kredensial untuk mengakses repositori Anda. CodeArtifact Metode yang disarankan untuk mengonfigurasi CLI Swift Package Manager dengan CodeArtifact kredensi dan titik akhir repositori Anda adalah dengan menggunakan perintah `aws codeartifact login` Anda juga dapat mengonfigurasi Swift Package Manager secara manual.

## Konfigurasi Swift dengan perintah login

Gunakan `aws codeartifact login` perintah untuk mengkonfigurasi Swift Package Manager dengan CodeArtifact.

## Note

Untuk menggunakan perintah login, Swift 5.8 atau yang lebih baru diperlukan dan Swift 5.9 atau yang lebih baru direkomendasikan.

`aws codeartifact login`Perintah akan melakukan hal berikut:

1. Ambil token otentikasi dari CodeArtifact dan simpan di lingkungan Anda. Bagaimana kredensial disimpan tergantung pada sistem operasi lingkungan:
    - a. macOS: Entri dibuat di aplikasi macOS Keychain.
    - b. Linux dan Windows: Entri dibuat dalam `~/.netrc` file.
- Di semua sistem operasi, jika ada entri kredensial, perintah ini menggantikan entri itu dengan token baru.
2. Ambil URL endpoint CodeArtifact repositori Anda dan tambahkan ke file konfigurasi Swift Anda. Perintah menambahkan URL titik akhir repositori ke file konfigurasi tingkat proyek yang terletak di `/path/to/project/.swiftpm/configuration/registries.json`

#### Note

`aws codeartifact login`Perintah memanggil `swift package-registry` perintah yang harus dijalankan dari direktori yang berisi `Package.swift` file. Karena itu, `aws codeartifact login` perintah harus dijalankan dari dalam proyek Swift.

Untuk mengkonfigurasi Swift dengan perintah `login`

1. Arahkan ke direktori proyek Swift yang berisi `Package.swift` file proyek Anda.
2. Jalankan perintah `aws codeartifact login` berikut.

Jika Anda mengakses repositori di domain milik Anda, Anda tidak perlu menyertakan `--domain-owner`. Untuk informasi selengkapnya, lihat [Domain lintas akun](#).

```
aws codeartifact login --tool swift --domain my_domain \  
--domain-owner 111122223333 --repository my_repo \  
[--namespace my_namespace]
```

`--namespace`Opsi mengonfigurasi aplikasi untuk hanya menggunakan paket dari CodeArtifact repositori Anda jika mereka berada di namespace yang ditentukan. [CodeArtifact namespace](#) identik dengan cakupan, dan digunakan untuk mengatur kode ke dalam grup logis dan untuk mencegah tabrakan nama yang dapat terjadi ketika basis kode Anda menyertakan beberapa pustaka.



Periode otorisasi default setelah memanggil login adalah 12 jam, dan login harus dipanggil untuk menyegarkan token secara berkala. Untuk informasi selengkapnya tentang token otorisasi yang dibuat dengan perintah login, lihat [Token dibuat dengan perintah login](#).

## Konfigurasi Swift tanpa perintah login

Meskipun disarankan agar Anda [mengkonfigurasi Swift dengan aws codeartifact login perintah](#), Anda juga dapat mengkonfigurasi Swift Package Manager tanpa perintah login dengan memperbarui konfigurasi Swift Package Manager secara manual.

Dalam prosedur berikut, Anda akan menggunakan AWS CLI untuk melakukan hal berikut:

1. Ambil token otentikasi dari CodeArtifact dan simpan di lingkungan Anda. Bagaimana kredensial disimpan tergantung pada sistem operasi lingkungan:
  - a. macOS: Entri dibuat di aplikasi macOS Keychain.
  - b. Linux dan Windows: Entri dibuat dalam `~/.netrc` file.
2. Ambil URL titik akhir CodeArtifact repositori Anda.
3. Dalam file `~/.swiftpm/configuration/registries.json` konfigurasi, tambahkan entri dengan URL titik akhir repositori dan jenis otentikasi Anda.

Untuk mengkonfigurasi Swift tanpa perintah login

1. Dalam baris perintah, gunakan perintah berikut untuk mengambil token CodeArtifact otorisasi dan menyimpannya dalam variabel lingkungan.
  - Ganti *my\_domain dengan nama* CodeArtifact domain Anda.
  - Ganti *111122223333* dengan ID AWS akun pemilik domain. Jika Anda mengakses repositori di domain milik Anda, Anda tidak perlu menyertakan `--domain-owner`. Untuk informasi selengkapnya, lihat [Domain lintas akun](#).

macOS and Linux

```
export CODEARTIFACT_AUTH_TOKEN=`aws codeartifact get-authorization-token --
domain my_domain --domain-owner 111122223333 --query authorizationToken --output
text`
```

## Windows

- Windows (menggunakan shell perintah default):

```
for /f %i in ('aws codeartifact get-authorization-token --domain my_domain --domain-owner 111122223333 --query authorizationToken --output text') do set CODEARTIFACT_AUTH_TOKEN=%i
```

- Jendela PowerShell:

```
$env:CODEARTIFACT_AUTH_TOKEN = aws codeartifact get-authorization-token --domain my_domain --domain-owner 111122223333 --query authorizationToken --output text
```

2. Dapatkan endpoint CodeArtifact repositori Anda dengan menjalankan perintah berikut. Endpoint repositori Anda digunakan untuk mengarahkan Swift Package Manager ke repositori Anda untuk mengkonsumsi atau menerbitkan paket.

- Ganti *my\_domain* dengan *nama* CodeArtifact domain Anda.
- Ganti *111122223333* dengan ID AWS akun pemilik domain. Jika Anda mengakses repositori di domain milik Anda, Anda tidak perlu menyertakan `--domain-owner`. Untuk informasi selengkapnya, lihat [Domain lintas akun](#).
- Ganti *my\_repo* dengan *nama repositori* Anda CodeArtifact .

## macOS and Linux

```
export CODEARTIFACT_REPO=`aws codeartifact get-repository-endpoint --domain my_domain --domain-owner 111122223333 --repository my_repo --format swift --query repositoryEndpoint --output text`
```

## Windows

- Windows (menggunakan shell perintah default):

```
for /f %i in ('aws codeartifact get-repository-endpoint --domain my_domain --domain-owner 111122223333 --repository my_repo --format swift --query repositoryEndpoint --output text') do set CODEARTIFACT_REPO=%i
```

- Jendela PowerShell:

```
$env:CODEARTIFACT_REPO = aws codeartifact get-repository-endpoint --  
domain my_domain --domain-owner 111122223333 --repository my_repo --format  
swift --query repositoryEndpoint --output text
```

URL berikut adalah contoh titik akhir repositori.

```
https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com/  
swift/my_repo/
```

### Important

Anda harus menambahkan login ke akhir endpoint URL repositori saat digunakan untuk mengkonfigurasi Swift Package Manager. Ini dilakukan untuk Anda dalam perintah prosedur ini.

3. Dengan dua nilai ini disimpan dalam variabel lingkungan, teruskan ke Swift menggunakan `swift package-registry login` perintah sebagai berikut:

macOS and Linux

```
swift package-registry login ${CODEARTIFACT_REPO}login --token  
${CODEARTIFACT_AUTH_TOKEN}
```

Windows

- Windows (menggunakan shell perintah default):

```
swift package-registry login %CODEARTIFACT_REPO%login --token  
%CODEARTIFACT_AUTH_TOKEN%
```

- Jendela PowerShell:

```
swift package-registry login $Env:CODEARTIFACT_REPO+"login" --token  
$Env:CODEARTIFACT_AUTH_TOKEN
```

4. Selanjutnya, perbarui registri paket yang digunakan oleh aplikasi Anda sehingga ketergantungan apa pun akan ditarik dari CodeArtifact repositori Anda. Perintah ini harus dijalankan di direktori proyek tempat Anda mencoba menyelesaikan ketergantungan paket:

## macOS and Linux

```
$ swift package-registry set ${CODEARTIFACT_REPO} [--scope my_scope]
```

## Windows

- Windows (menggunakan shell perintah default):

```
$ swift package-registry set %CODEARTIFACT_REPO% [--scope my_scope]
```

- Jendela PowerShell:

```
$ swift package-registry set $Env:CODEARTIFACT_REPO [--scope my_scope]
```

--scope Opsi mengonfigurasi aplikasi untuk hanya menggunakan paket dari CodeArtifact repositori Anda jika mereka berada dalam lingkup yang ditentukan. Cakupan identik dengan [CodeArtifact ruang nama](#), dan digunakan untuk mengatur kode ke dalam grup logis dan untuk mencegah tabrakan nama yang dapat terjadi ketika basis kode Anda menyertakan beberapa pustaka.

5. Anda dapat mengonfirmasi bahwa konfigurasi telah diatur dengan benar dengan melihat isi `.swiftpm/configuration/registries.json` file tingkat proyek dengan menjalankan perintah berikut di direktori proyek Anda:

```
$ cat .swiftpm/configuration/registries.json
{
  "authentication" : {

  },
  "registries" : {
    "[default]" : {
      "url" : "https://my-domain-111122223333.d.codeartifact.us-
west-2.amazonaws.com/swift/my-repo/"
    }
  },
  "version" : 1
}
```

Sekarang setelah Anda mengkonfigurasi Swift Package Manager dengan CodeArtifact repositori Anda, Anda dapat menggunakannya untuk mempublikasikan dan menggunakan paket Swift ke dan dari itu. Untuk informasi selengkapnya, lihat [Mengkonsumsi dan menerbitkan paket Swift](#).

## Mengkonsumsi dan menerbitkan paket Swift

### Mengkonsumsi paket Swift dari CodeArtifact

Gunakan prosedur berikut untuk mengonsumsi paket Swift dari AWS CodeArtifact repositori.

Untuk mengonsumsi paket Swift dari repositori CodeArtifact

1. Jika belum, ikuti langkah-langkah [Konfigurasi Swift Package Manager dengan CodeArtifact](#) untuk mengonfigurasi Swift Package Manager untuk menggunakan CodeArtifact repositori Anda dengan kredensi yang tepat.

#### Note

Token otorisasi yang dihasilkan berlaku selama 12 jam. Anda harus membuat yang baru jika 12 jam telah berlalu sejak token dibuat.

2. Edit `Package.swift` file di folder proyek aplikasi Anda untuk memperbarui dependensi paket yang akan digunakan oleh proyek Anda.
  - a. Jika `Package.swift` file tidak berisi `dependencies` bagian, tambahkan satu.
  - b. Di `dependencies` bagian `Package.swift` file, tambahkan paket yang ingin Anda gunakan dengan menambahkan pengenalan paketnya. Pengidentifikasi paket terdiri dari ruang lingkup dan nama paket yang dipisahkan oleh titik. Lihat cuplikan kode mengikuti langkah selanjutnya untuk contoh.

#### Tip

Untuk menemukan pengenalan paket, Anda dapat menggunakan CodeArtifact konsol. Temukan versi paket tertentu yang ingin Anda gunakan dan rujuk petunjuk pintasan Instal pada halaman versi paket.

- c. Jika `Package.swift` file tidak berisi `targets` bagian, tambahkan satu.
- d. Di `targets` bagian ini, tambahkan target yang perlu menggunakan ketergantungan.

Cuplikan berikut adalah contoh cuplikan yang menunjukkan konfigurasi dependencies dan targets bagian dalam file: Package.swift

```
...
],
dependencies: [
    .package(id: "my_scope.package_name", from: "1.0.0")
],
targets: [
    .target(
        name: "MyApp",
        dependencies: ["package_name"]
    ),...
],
...
```

3. Sekarang semuanya sudah dikonfigurasi, gunakan perintah berikut untuk mengunduh dependensi paket dari CodeArtifact

```
swift package resolve
```

## Mengonsumsi paket Swift dari CodeArtifact dalam Xcode

Gunakan prosedur berikut untuk menggunakan paket Swift dari CodeArtifact repositori di Xcode.

Untuk menggunakan paket Swift dari CodeArtifact repositori di Xcode

1. Jika belum, ikuti langkah-langkah [Konfigurasi Swift Package Manager dengan CodeArtifact](#) untuk mengonfigurasi Swift Package Manager untuk menggunakan CodeArtifact repositori Anda dengan kredensi yang tepat.

### Note

Token otorisasi yang dihasilkan berlaku selama 12 jam. Anda harus membuat yang baru jika 12 jam telah berlalu sejak token dibuat.

2. Tambahkan paket sebagai dependensi dalam proyek Anda di Xcode.
  - a. Pilih File > Tambah Paket.

- b. Cari paket Anda menggunakan bilah pencarian. Pencarian Anda harus dalam formulir `package_scope.package_name`.
- c. Setelah ditemukan, pilih paket dan pilih Add Package.
- d. Setelah paket diverifikasi, pilih produk paket yang ingin Anda tambahkan sebagai dependensi, dan pilih Add Package.

Jika Anda mengalami masalah menggunakan CodeArtifact repositori Anda dengan Xcode, lihat [Pemecahan masalah cepat](#) masalah umum dan kemungkinan perbaikan.

## Menerbitkan paket Swift ke CodeArtifact

CodeArtifact merekomendasikan Swift 5.9 atau yang lebih baru dan menggunakan `swift package-registry publish` perintah untuk menerbitkan paket Swift. Jika Anda menggunakan versi sebelumnya, Anda harus menggunakan perintah Curl untuk mempublikasikan paket Swift ke CodeArtifact

## Menerbitkan CodeArtifact paket dengan **swift package-registry publish** perintah

Gunakan prosedur berikut dengan Swift 5.9 atau yang lebih baru untuk mempublikasikan paket Swift ke CodeArtifact repositori dengan Swift Package Manager.

1. Jika belum, ikuti langkah-langkah [Konfigurasi Swift Package Manager dengan CodeArtifact](#) untuk mengonfigurasi Swift Package Manager untuk menggunakan CodeArtifact repositori Anda dengan kredensi yang tepat.

### Note

Token otorisasi yang dihasilkan berlaku selama 12 jam. Anda harus membuat yang baru jika 12 jam telah berlalu sejak dibuat.

2. Arahkan ke direktori proyek Swift yang berisi `Package.swift` file untuk paket Anda.
3. Jalankan `swift package-registry publish` perintah berikut untuk menerbitkan paket. Perintah membuat arsip sumber paket dan menerbitkannya ke CodeArtifact repositori Anda.

```
swift package-registry publish packageScope.packageName packageVersion
```

Sebagai contoh:

```
swift package-registry publish myScope.myPackage 1.0.0
```

4. Anda dapat mengonfirmasi bahwa paket telah diterbitkan dan ada di repositori dengan memeriksa di konsol atau menggunakan `aws codeartifact list-packages` perintah sebagai berikut:

```
aws codeartifact list-packages --domain my_domain --repository my_repo
```

Anda dapat membuat daftar versi tunggal paket menggunakan `aws codeartifact list-package-versions` perintah sebagai berikut:

```
aws codeartifact list-package-versions --domain my_domain --repository my_repo \  
--format swift --namespace my_scope --package package_name
```

## Menerbitkan CodeArtifact paket dengan Curl

Meskipun disarankan untuk menggunakan `swift package-registry publish` perintah yang disertakan dengan Swift 5.9 atau yang lebih baru, Anda juga dapat menggunakan Curl untuk mempublikasikan paket Swift ke CodeArtifact

Gunakan prosedur berikut untuk mempublikasikan paket Swift ke AWS CodeArtifact repositori dengan Curl.

1. Jika belum, buat dan perbarui variabel `CODEARTIFACT_AUTH_TOKEN` dan `CODEARTIFACT_REPO` lingkungan dengan mengikuti langkah-langkahnya [Konfigurasi Swift Package Manager dengan CodeArtifact](#).

### Note

Token otorisasi berlaku selama 12 jam. Anda perlu menyegarkan variabel `CODEARTIFACT_AUTH_TOKEN` lingkungan Anda dengan kredensi baru jika 12 jam telah berlalu sejak dibuat.

2. Pertama, jika Anda tidak memiliki paket Swift yang dibuat, Anda dapat melakukannya dengan menjalankan perintah berikut:



```
mkdir testDir && cd testDir
swift package init
git init .
swift package archive-source
```

- Gunakan perintah Curl berikut untuk mempublikasikan paket Swift Anda ke: CodeArtifact

macOS and Linux

```
curl -X PUT --user "aws:$CODEARTIFACT_AUTH_TOKEN" \
-H "Accept: application/vnd.swift.registry.v1+json" \
-F source-archive="@test_dir_package_name.zip" \
"${CODEARTIFACT_REPO}my_scope/package_name/packageVersion"
```

Windows

```
curl -X PUT --user "aws:%CODEARTIFACT_AUTH_TOKEN%" \
-H "Accept: application/vnd.swift.registry.v1+json" \
-F source-archive="@test_dir_package_name.zip" \
"%CODEARTIFACT_REPO%my_scope/package_name/packageVersion"
```

- Anda dapat mengonfirmasi bahwa paket telah diterbitkan dan ada di repositori dengan memeriksa di konsol atau menggunakan `aws codeartifact list-packages` perintah sebagai berikut:

```
aws codeartifact list-packages --domain my_domain --repository my_repo
```

Anda dapat membuat daftar versi tunggal paket menggunakan `aws codeartifact list-package-versions` perintah sebagai berikut:


```
aws codeartifact list-package-versions --domain my_domain --repository my_repo \
--format swift --namespace my_scope --package package_name
```

## Mengambil paket Swift dari GitHub dan menerbitkan ulang ke CodeArtifact

Gunakan prosedur berikut untuk mengambil paket Swift dari GitHub dan menerbitkannya kembali ke repositori. CodeArtifact

Untuk mengambil paket Swift dari GitHub dan menerbitkannya kembali ke CodeArtifact

1. Jika belum, ikuti langkah-langkah [Konfigurasi Swift Package Manager dengan CodeArtifact](#) untuk mengonfigurasi Swift Package Manager untuk menggunakan CodeArtifact repositori Anda dengan kredensi yang tepat.

 Note

Token otorisasi yang dihasilkan berlaku selama 12 jam. Anda harus membuat yang baru jika 12 jam telah berlalu sejak token dibuat.

2. Kloning repositori git dari paket Swift yang ingin Anda ambil dan terbitkan ulang dengan menggunakan perintah berikut. `git clone` Untuk informasi tentang kloning GitHub repositori, lihat [Mengkloning repositori di](#) Dokumen. GitHub

```
git clone repoURL
```

3. Arahkan ke repositori yang baru saja Anda kloning:

```
cd repoName
```

4. Buat paket dan publikasikan ke CodeArtifact.
  - a. Direkomendasikan: Jika Anda menggunakan Swift 5.9 atau yang lebih baru, Anda dapat menggunakan `swift package-registry publish` perintah berikut untuk membuat paket dan mempublikasikannya ke repositori yang dikonfigurasi CodeArtifact .

```
swift package-registry publish packageScope.packageName versionNumber
```

Sebagai contoh:

```
swift package-registry publish myScope.myPackage 1.0.0
```

- b. Jika Anda menggunakan versi Swift yang lebih tua dari 5.9, Anda harus menggunakan `swift archive-source` perintah untuk membuat paket dan kemudian menggunakan perintah Curl untuk mempublikasikannya.

- i. Jika Anda belum mengonfigurasi variabel `CODEARTIFACT_AUTH_TOKEN` dan `CODEARTIFACT_REPO` lingkungan, atau sudah lebih dari 12 jam sejak Anda memilikinya, ikuti langkah-langkahnya [Konfigurasi Swift tanpa perintah login](#).
- ii. Buat paket Swift dengan menggunakan `swift package archive-source` perintah:

```
swift package archive-source
```

Jika berhasil, Anda akan melihat Created `package_name.zip` di baris perintah.

- iii. Gunakan perintah Curl berikut untuk mempublikasikan paket Swift ke: CodeArtifact macOS and Linux

```
curl -X PUT --user "aws:$CODEARTIFACT_AUTH_TOKEN" \  
-H "Accept: application/vnd.swift.registry.v1+json" \  
-F source-archive="@package_name.zip" \  
"${CODEARTIFACT_REPO}my_scope/package_name/packageVersion"
```

## Windows

```
curl -X PUT --user "aws:%CODEARTIFACT_AUTH_TOKEN%" \  
-H "Accept: application/vnd.swift.registry.v1+json" \  
-F source-archive="@package_name.zip" \  
"%CODEARTIFACT_REPO%my_scope/package_name/packageVersion"
```

5. Anda dapat mengonfirmasi bahwa paket telah diterbitkan dan ada di repositori dengan memeriksa di konsol atau menggunakan `aws codeartifact list-packages` perintah sebagai berikut:

```
aws codeartifact list-packages --domain my_domain --repository my_repo
```

Anda dapat membuat daftar versi tunggal paket menggunakan `aws codeartifact list-package-versions` perintah sebagai berikut:

```
aws codeartifact list-package-versions --domain my_domain --repository my_repo \  
--format swift --namespace my_scope --package package_name
```

## Nama paket Swift dan normalisasi namespace

CodeArtifact menormalkan nama paket dan ruang nama sebelum menyimpannya, yang berarti nama di CodeArtifact mungkin berbeda dari yang disediakan saat paket diterbitkan.

Nama paket dan normalisasi namespace: CodeArtifact menormalkan nama paket Swift dan ruang nama dengan mengonversi semua huruf menjadi huruf kecil.

Normalisasi versi paket: CodeArtifact tidak menormalkan versi paket Swift. [Perhatikan bahwa CodeArtifact hanya mendukung pola versi Semantic Versioning 2.0, untuk informasi selengkapnya tentang Pembuatan Versi Semantik, lihat Versi Semantik 2.0.0.](#)

Nama paket dan namespace yang tidak dinormalisasi dapat digunakan dengan permintaan API dan CLI karena CodeArtifact melakukan normalisasi pada input untuk permintaan tersebut. Misalnya, input `--package myPackage` dan `--namespace myScope` akan dinormalisasi dan mengembalikan paket yang memiliki nama paket dinormalisasi `mypackage` dan namespace dari `myscope`

Anda harus menggunakan nama yang dinormalisasi di ARN, seperti dalam kebijakan IAM.

Untuk menemukan nama paket yang dinormalisasi, gunakan `aws codeartifact list-packages` perintah. Untuk informasi selengkapnya, lihat [Mencantumkan nama paket](#).

## Pemecahan masalah cepat

Informasi berikut dapat membantu Anda memecahkan masalah umum dengan Swift dan CodeArtifact

### Saya mendapatkan kesalahan 401 di Xcode bahkan setelah mengonfigurasi Swift Package Manager

Masalah: [Ketika Anda mencoba menambahkan paket dari CodeArtifact repositori Anda sebagai ketergantungan ke proyek Swift Anda di Xcode, Anda mendapatkan kesalahan 401 yang tidak sah bahkan setelah Anda mengikuti instruksi untuk menghubungkan Swift ke CodeArtifact](#)

Kemungkinan perbaikan: Ini dapat disebabkan oleh masalah dengan aplikasi macOS Keychain, tempat kredensial CodeArtifact Anda disimpan. Untuk memperbaikinya, kami sarankan untuk membuka aplikasi Keychain dan menghapus semua CodeArtifact entri dan mengonfigurasi Swift Package Manager dengan CodeArtifact repositori Anda lagi dengan mengikuti instruksi di [Konfigurasi Swift Package Manager dengan CodeArtifact](#)

## Xcode hang pada mesin CI karena permintaan gantungan kunci untuk kata sandi

Masalah: Saat Anda mencoba menarik paket Swift dari CodeArtifact sebagai bagian dari build Xcode di server continuous integration (CI), seperti dengan GitHub Actions, otentikasi dengan CodeArtifact dapat hang dan akhirnya gagal dengan pesan kesalahan yang mirip dengan berikut ini:

```
Failed to save credentials for
\'https://my_domain-111122223333.d.codeartifact.us-west-2.amazonaws.com\'
to keychain: status -60008
```

Kemungkinan perbaikan: Ini disebabkan oleh kredensial yang tidak disimpan ke gantungan kunci pada mesin CI, dan Xcode hanya mendukung kredensial yang disimpan di Keychain. Untuk memperbaikinya, kami sarankan membuat entri gantungan kunci secara manual menggunakan langkah-langkah berikut:

### 1. Siapkan gantungan kunci.

```
KEYCHAIN_PASSWORD=$(openssl rand -base64 20)
KEYCHAIN_NAME=login.keychain
SYSTEM_KEYCHAIN=/Library/Keychains/System.keychain

if [ -f $HOME/Library/Keychains/"${KEYCHAIN_NAME}"-db ]; then
    echo "Deleting old ${KEYCHAIN_NAME} keychain"
    security delete-keychain "${KEYCHAIN_NAME}"
fi
echo "Create Keychain"
security create-keychain -p "${KEYCHAIN_PASSWORD}" "${KEYCHAIN_NAME}"

EXISTING_KEYCHAINS=( $( security list-keychains | sed -e 's/ *//' | tr '\n' ' ' |
tr -d '"' ) )
sudo security list-keychains -s "${KEYCHAIN_NAME}" "${EXISTING_KEYCHAINS[@]}"

echo "New keychain search list :"
security list-keychain

echo "Configure keychain : remove lock timeout"
security unlock-keychain -p "${KEYCHAIN_PASSWORD}" "${KEYCHAIN_NAME}"
security set-keychain-settings "${KEYCHAIN_NAME}"
```

### 2. Dapatkan token CodeArtifact otentikasi dan titik akhir repositori Anda.

```

export CODEARTIFACT_AUTH_TOKEN=`aws codeartifact get-authorization-token \
    --region us-west-2 \
    --domain my_domain \
    --domain-owner 111122223333 \
    --query authorizationToken \
    --output text`

export CODEARTIFACT_REPO=`aws codeartifact get-repository-endpoint \
    --region us-west-2 \
    --domain my_domain \
    --domain-owner 111122223333 \
    --format swift \
    --repository my_repo \
    --query repositoryEndpoint \
    --output text`

```

### 3. Buat entri Keychain secara manual.

```

SERVER=$(echo $CODEARTIFACT_REPO | sed 's/https://\///g' | sed 's/.com.*$/\.com/g')
AUTHORIZATION=(-T /usr/bin/security -T /usr/bin/codesign -T /usr/bin/xcodebuild -
T /usr/bin/swift \
    -T /Applications/Xcode-15.2.app/Contents/Developer/usr/bin/
xcodebuild)

security delete-internet-password -a token -s $SERVER -r https "${KEYCHAIN_NAME}"

security add-internet-password -a token \
    -s $SERVER \
    -w $CODEARTIFACT_AUTH_TOKEN \
    -r https \
    -U \
    "${AUTHORIZATION[@]}" \
    "${KEYCHAIN_NAME}"

security set-internet-password-partition-list \
    -a token \
    -s $SERVER \
    -S "com.apple.swift-
package,com.apple.security,com.apple.dt.Xcode,apple-tool:,apple:,codesign" \
    -k "${KEYCHAIN_PASSWORD}" "${KEYCHAIN_NAME}"

```

```
security find-internet-password "${KEYCHAIN_NAME}"
```

Untuk informasi selengkapnya tentang kesalahan ini dan solusinya, lihat [https://github.com/apple/swift-package-manager /issues/7236](https://github.com/apple/swift-package-manager/issues/7236).

# Menggunakan CodeArtifact dengan paket generik

Topik ini menunjukkan cara menggunakan dan memublikasikan paket umum. AWS CodeArtifact

Topik

- [Gambaran umum umum](#)
- [Perintah yang didukung untuk paket generik](#)
- [Menerbitkan dan mengonsumsi paket generik](#)

## Gambaran umum umum

Dengan menggunakan format `generic` paket, Anda dapat mengunggah semua jenis file untuk membuat paket di CodeArtifact repositori. Paket umum tidak terkait dengan bahasa pemrograman, jenis file, atau ekosistem manajemen paket tertentu. Ini dapat berguna untuk menyimpan dan membuat versi artefak build sewenang-wenang, seperti installer aplikasi, model pembelajaran mesin, file konfigurasi, dan lainnya.

Paket generik terdiri dari nama paket, namespace, versi, dan satu atau lebih aset (atau file). Paket generik dapat ada bersama paket format lain dalam satu repositori. CodeArtifact

Anda dapat menggunakan AWS CLI atau SDK untuk bekerja dengan paket generik. Untuk daftar lengkap AWS CLI perintah yang bekerja dengan paket generik, lihat [Perintah yang didukung untuk paket generik](#).

## Kendala

- Mereka tidak pernah diambil dari repositori hulu. Mereka hanya dapat diperoleh dari repositori tempat mereka diterbitkan.
- Mereka tidak dapat mendeklarasikan dependensi untuk dikembalikan dari [ListPackageVersionDependencies](#) atau ditampilkan dalam AWS Management Console
- Mereka dapat menyimpan file README dan LICENSE, tetapi mereka tidak ditafsirkan oleh CodeArtifact Informasi dalam file ini tidak dikembalikan dari [GetPackageVersionReadme](#) atau [DescribePackageVersion](#), dan tidak muncul di file AWS Management Console.
- Seperti semua paket CodeArtifact, ada batasan untuk ukuran aset dan jumlah aset per paket. Untuk informasi lebih lanjut tentang batas dan kuota CodeArtifact, lihat [Kuota di AWS CodeArtifact](#).
- Nama aset yang dikandungnya harus mengikuti aturan berikut:



- Nama aset dapat diawali dengan nama dan angka Unicode. Secara khusus, kategori karakter Unicode ini diperbolehkan: Huruf Kecil (`l`), Huruf Pengubah (`L1`), Huruf Lain (`Lm`), Huruf Titlecase (`Lo`), Huruf Besar (`Lt`), Nomor Huruf (`Lu`), dan Nomor Desimal (`N1`). `Nd`
- Karakter khusus berikut diperbolehkan: `~!@^&()-_+[]{}; , .`
- Aset tidak dapat diberi nama `.` atau `..`
- Spasi adalah satu-satunya karakter spasi diperbolehkan. Nama aset tidak dapat diawali atau diakhiri dengan spasi, atau diawali atau diawali atau diakhiri dengan spasi.

## Perintah yang didukung untuk paket generik

Anda dapat menggunakan AWS CLI atau SDK untuk bekerja dengan paket generik. CodeArtifactPerintah berikut bekerja dengan paket generik:

- [copy-package-versions](#)(lihat [Menyalin paket antar-repositori](#))
- [menghapus-paket](#) (lihat) [Menghapus paket \(\)AWS CLI](#)
- [delete-package-versions](#)(lihat [Menghapus versi paket \(\)AWS CLI](#))
- [menjelaskan-paket](#)
- [describe-package-version](#)(lihat [Melihat dan memperbarui detail versi paket dan dependensi](#))
- [dispose-package-versions](#)(lihat [Membuang versi paket](#))
- [get-package-version-asset](#)(lihat [Mengunduh aset versi paket](#))
- [list-package-version-assets](#)(lihat [Mencantumkan aset versi paket](#))
- [list-package-versions](#)(lihat [Mencantumkan versi paket](#))
- [daftar-paket](#) (lihat) [Mencantumkan nama paket](#)
- [publish-package-version](#)(lihat [Menerbitkan paket generik](#))
- [put-package-origin-configuration](#)(lihat [Mengedit kontrol asal paket](#))

### Note

Anda dapat menggunakan pengaturan kontrol `publish` asal untuk mengizinkan atau memblokir penerbitan nama paket generik di repositori. Namun, `upstream` pengaturan tidak berlaku untuk paket generik karena mereka tidak dapat diambil dari repositori `upstream`.

- [update-package-versions-status](#)(lihat [Memperbarui status versi paket](#))

# Menerbitkan dan mengonsumsi paket generik

Untuk mempublikasikan versi paket generik dan aset terkait, gunakan `publish-package-version` perintah. Anda dapat mencantumkan aset paket generik menggunakan `list-package-version-asset` perintah dan mengunduhnya menggunakan `get-package-version-asset`. Topik berikut berisi step-by-step petunjuk untuk mempublikasikan paket generik atau mengunduh aset paket generik menggunakan perintah ini.

## Menerbitkan paket generik

Paket generik terdiri dari nama paket, namespace, versi, dan satu atau lebih aset (atau file). Topik ini menunjukkan bagaimana untuk mempublikasikan sebuah paket bernama `my-package`, dengan namespace `my-ns`, versi `1.0.0`, dan berisi satu aset bernama `asset.tar.gz`

Prasyarat:

- Mengatur dan mengkonfigurasi AWS Command Line Interface dengan CodeArtifact (lihat [Mengatur dengan AWS CodeArtifact](#))
- Memiliki CodeArtifact domain dan repositori (lihat [Memulai menggunakan AWS CLI](#))

Untuk memublikasikan paket generik

1. Gunakan perintah berikut untuk menghasilkan hash SHA256 untuk setiap file yang ingin Anda upload ke versi paket, dan menempatkan nilai dalam variabel lingkungan. Nilai ini digunakan sebagai pemeriksaan integritas untuk memverifikasi bahwa isi file tidak berubah setelah awalnya dikirim.

Linux

```
export ASSET_SHA256=$(sha256sum asset.tar.gz | awk '{print $1;}')
```

macOS

```
export ASSET_SHA256=$(shasum -a 256 asset.tar.gz | awk '{print $1;}')
```

## Windows

```
for /f "tokens=*" %G IN ('certUtil -hashfile asset.tar.gz SHA256 ^| findstr /v "hash"') DO SET "ASSET_SHA256=%G"
```

2. Panggilan `publish-package-version` untuk mengunggah aset dan membuat versi paket baru.

### Note

Jika paket Anda berisi lebih dari satu aset, Anda dapat menelepon satu `publish-package-version` kali untuk setiap aset untuk diunggah. Sertakan `--unfinished` argumen untuk setiap panggilan ke `publish-package-version`, kecuali saat mengunggah aset akhir. Menghilangkan `--unfinished` akan mengatur status versi paket `Published`, dan mencegah aset tambahan diunggah ke sana. Atau, sertakan `--unfinished` untuk setiap panggilan ke `publish-package-version`, lalu atur status versi paket untuk `Published` menggunakan `update-package-versions-status` perintah.

## Linux/macOS

```
aws codeartifact publish-package-version --domain my_domain --repository my_repo \
  --format generic --namespace my-ns --package my-package --package-
version 1.0.0 \
  --asset-content asset.tar.gz --asset-name asset.tar.gz \
  --asset-sha256 $ASSET_SHA256
```

## Windows

```
aws codeartifact publish-package-version --domain my_domain --repository my_repo ^
  --format generic --namespace my-ns --package my-package --package-
version 1.0.0 ^
  --asset-content asset.tar.gz --asset-name asset.tar.gz ^
  --asset-sha256 %ASSET_SHA256%
```

Berikut ini menunjukkan output.

```
{
  "format": "generic",
  "namespace": "my-ns",
  "package": "my-package",
  "version": "1.0.0",
  "versionRevision": "REVISION-SAMPLE-1-C7F4S5E9B772FC",
  "status": "Published",
  "asset": {
    "name": "asset.tar.gz",
    "size": 11,
    "hashes": {
      "MD5": "41bba98d5b9219c43089eEXAMPLE-MD5",
      "SHA-1": "69b215c25dd4cda1d997a786ec6EXAMPLE-SHA-1",
      "SHA-256": "43f24850b7b7b7d79c5fa652418518fbdf427e602b1edabe6EXAMPLE-SHA-256",
      "SHA-512":
"3947382ac2c180ee3f2aba4f8788241527c8db9dfe9f4b039abe9fc560aaf5a1fced7bd1e80a0dca9ce320d95
SHA-512"
    }
  }
}
```

## Mencantumkan aset Mencantumkan aset

Untuk daftar aset yang terkandung dalam paket generik, gunakan `list-package-version-assets` perintah. Untuk informasi selengkapnya, lihat [Mencantumkan aset versi paket](#).

Contoh berikut mencantumkan aset `1.0.0` versi paket `my-package`.

Mencantumkan aset mencantumkan aset

- Panggilan `list-package-version-assets` untuk daftar aset yang terkandung dalam paket generik.

Linux/macOS

```
aws codeartifact list-package-version-assets --domain my_domain \  
--repository my_repo --format generic --namespace my-ns \  

```

```
--package my-package --package-version 1.0.0
```

## Windows

```
aws codeartifact list-package-version-assets --domain my_domain ^  
--repository my_repo --format generic --namespace my-ns ^  
--package my-package --package-version 1.0.0
```

Berikut ini menunjukkan output.

```
{  
  "assets": [  
    {  
      "name": "asset.tar.gz",  
      "size": 11,  
      "hashes": {  
        "MD5": "41bba98d5b9219c43089eEXAMPLE-MD5",  
        "SHA-1": "69b215c25dd4cda1d997a786ec6EXAMPLE-SHA-1",  
        "SHA-256":  
"43f24850b7b7b7d79c5fa652418518fbdf427e602b1edabe6EXAMPLE-SHA-256",  
        "SHA-512":  
"3947382ac2c180ee3f2aba4f8788241527c8db9dfe9f4b039abe9fc560aaf5a1fced7bd1e80a0dca9ce320d95  
SHA-512"  
      }  
    }  
  ],  
  "package": "my-package",  
  "format": "generic",  
  "namespace": "my-ns",  
  "version": "1.0.0",  
  "versionRevision": "REVISION-SAMPLE-1-C7F4S5E9B772FC"  
}
```

## Mengunduh aset Mengunduh aset Mengunduh aset

Untuk mengunduh aset dari paket generik, gunakan `get-package-version-asset` perintah. Untuk informasi selengkapnya, lihat [Mengunduh aset versi paket](#).

Contoh berikut download aset `asset.tar.gz 1.0.0` dari versi paket `my-package` ke direktori kerja saat ini ke dalam file juga bernama `asset.tar.gz`.

## Mengunduh aset dapat diawali

- Panggilan `get-package-version-asset` untuk mengunduh aset dari paket generik.

### Linux/macOS

```
aws codeartifact get-package-version-asset --domain my_domain \  
  --repository my_repo --format generic --namespace my-ns --package my-package \  
  --package-version 1.0.0 --asset asset.tar.gz \  
  asset.tar.gz
```

### Windows

```
aws codeartifact get-package-version-asset --domain my_domain ^  
  --repository my_repo --format generic --namespace my-ns --package my-package ^  
  --package-version 1.0.0 --asset asset.tar.gz ^  
  asset.tar.gz
```

Berikut ini menunjukkan output.

```
{  
  "assetName": "asset.tar.gz",  
  "packageVersion": "1.0.0",  
  "packageVersionRevision": "REVISION-SAMPLE-1-C7F4S5E9B772FC"  
}
```

# Menggunakan CodeArtifact dengan CodeBuild

Topik-topik ini menjelaskan cara menggunakan paket dalam CodeArtifact repositori dalam AWS CodeBuild membangun proyek.

## Topik

- [Menggunakan paket npm di CodeBuild](#)
- [Menggunakan paket Python di CodeBuild](#)
- [Menggunakan paket Maven di CodeBuild](#)
- [Menggunakan NuGet paket di CodeBuild](#)
- [Caching Dependensi](#)

## Menggunakan paket npm di CodeBuild

Langkah-langkah berikut telah diuji dengan sistem operasi yang tercantum dalam [Gambar Docker disediakan oleh CodeBuild](#).

## Mengatur izin dengan IAM role

Langkah-langkah ini diperlukan saat menggunakan paket npm dari CodeArtifact di CodeBuild.

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Peran. Pada Peran halaman, mengedit peran yang digunakan oleh CodeBuild membangun proyek. Peran ini harus memiliki izin berikut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [ "codeartifact:GetAuthorizationToken",
                  "codeartifact:GetRepositoryEndpoint",
                  "codeartifact:ReadFromRepository"
                ],
      "Resource": "*"
    }
  ],
  {
```

```
    "Effect": "Allow",
    "Action": "sts:GetServiceBearerToken",
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "sts:AWSServiceName": "codeartifact.amazonaws.com"
      }
    }
  }
]
```

### Important

Jika Anda juga ingin menggunakan CodeBuild untuk mempublikasikan paket, tambahkan `codeartifact:PublishPackageVersion` izin.

Untuk informasi selengkapnya, lihat [Modifying a Role](#) dalam Panduan Pengguna IAM.

## Masuk dan menggunakan npm

Untuk menggunakan paket npm dari CodeBuild, jalankan `login` perintah dari `pre-build` bagian dari proyek Anda `buildspec.yaml` untuk mengkonfigurasi npm untuk mengambil paket dari CodeArtifact. Untuk informasi selengkapnya, lihat [Autentikasi dengan npm](#).

Setelah login telah berjalan dengan sukses, Anda dapat menjalankan perintah npm dari bagian `build` untuk menginstal paket npm.

### Linux

#### Note

Hal ini hanya diperlukan untuk meng-upgrade AWS CLI bersama `pip3 install awscli --upgrade --user` jika Anda menggunakan yang lebih tua CodeBuild gambar. Jika Anda menggunakan versi gambar terbaru, Anda dapat menghapus baris tersebut.

```
pre_build:
```



```

commands:
  - pip3 install awscli --upgrade --user
  - aws codeartifact login --tool npm --domain my_domain --domain-owner 111122223333
--repository my_repo
build:
  commands:
    - npm install

```

## Windows

```

version: 0.2
phases:
  install:
    commands:
      - '[Net.ServicePointManager]::SecurityProtocol = "Tls12"; Invoke-WebRequest
https://awscli.amazonaws.com/AWSCLIV2.msi -OutFile $env:TEMP/AWSCLIV2.msi'
      - Start-Process -Wait msiexec "/i $env:TEMP\AWSCLIV2.msi /quiet /norestart"
  pre_build:
    commands:
      - '&"C:\Program Files\Amazon\AWSCLIV2\aws" codeartifact login --tool npm --
domain my_domain --domain-owner 111122223333 --repository my_repo'
  build:
    commands:
      - npm install

```

## Menggunakan paket Python diCodeBuild

Langkah-langkah berikut telah diuji dengan sistem operasi yang tercantum dalam [Gambar Docker disediakan olehCodeBuild](#).

### Mengatur izin dengan IAM role

Langkah-langkah ini diperlukan saat menggunakan paket Python dariCodeArtifactdiCodeBuild.

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Peran. PadaPeranhalaman, mengedit peran yang digunakan olehCodeBuildmembangun proyek. Peran ini harus memiliki izin berikut.

```
{
```

```
"Version": "2012-10-17",
"Statement": [
  {
    "Effect": "Allow",
    "Action": [ "codeartifact:GetAuthorizationToken",
               "codeartifact:GetRepositoryEndpoint",
               "codeartifact:ReadFromRepository"
             ],
    "Resource": "*"
  },
  {
    "Effect": "Allow",
    "Action": "sts:GetServiceBearerToken",
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "sts:AWSServiceName": "codeartifact.amazonaws.com"
      }
    }
  }
]
```

### Important

Jika Anda juga ingin menggunakan CodeBuild untuk mempublikasikan paket, tambahkan **codeartifact:PublishPackageVersion** izin.

Untuk informasi selengkapnya, lihat [Modifying a Role](#) dalam Panduan Pengguna IAM.

## Masuk dan gunakan pip atau twine

Untuk menggunakan paket Python dari CodeBuild, jalankan login perintah dari pre-build bagian dari proyek Anda buildspec.yaml file untuk dikonfigurasi pip untuk mengambil paket dari CodeArtifact. Untuk informasi selengkapnya, lihat [Menggunakan CodeArtifact dengan Python](#).

Setelah login telah berjalan dengan sukses, Anda dapat menjalankan perintah pip dari bagian build untuk menginstal atau memublikasikan paket Python.

## Linux

### Note

Hal ini hanya diperlukan untuk meng-upgrade AWS CLI bersama pip3 `install awscli --upgrade --user` jika Anda menggunakan yang lebih tua CodeBuild gambar. Jika Anda menggunakan versi gambar terbaru, Anda dapat menghapus baris tersebut.

Untuk menginstal paket Python menggunakan pip:

```
pre_build:
  commands:
    - pip3 install awscli --upgrade --user
    - aws codeartifact login --tool pip --domain my_domain --domain-owner 111122223333
      --repository my_repo
build:
  commands:
    - pip install requests
```

Untuk memublikasikan paket Python menggunakan twine:

```
pre_build:
  commands:
    - pip3 install awscli --upgrade --user
    - aws codeartifact login --tool twine --domain my_domain --domain-
      owner 111122223333 --repository my_repo
build:
  commands:
    - twine upload --repository codeartifact mypackage
```

## Windows

Untuk menginstal paket Python menggunakan pip:

```
version: 0.2
phases:
  install:
    commands:
      - '[Net.ServicePointManager]::SecurityProtocol = "Tls12"; Invoke-WebRequest
        https://awscli.amazonaws.com/AWSCLIV2.msi -OutFile $env:TEMP/AWSCLIV2.msi'
```

```

- Start-Process -Wait msiexec "/i $env:TEMP\AWSCLIV2.msi /quiet /norestart"
pre_build:
  commands:
    - '&"C:\Program Files\Amazon\AWSCLIV2\aws" codeartifact login --tool pip --
domain my_domain --domain-owner 111122223333 --repository my_repo'
build:
  commands:
    - pip install requests

```

Untuk memublikasikan paket Python menggunakan twine:

```

version: 0.2
phases:
  install:
    commands:
      - '[Net.ServicePointManager]::SecurityProtocol = "Tls12"; Invoke-WebRequest
https://awscli.amazonaws.com/AWSCLIV2.msi -OutFile $env:TEMP/AWSCLIV2.msi'
      - Start-Process -Wait msiexec "/i $env:TEMP\AWSCLIV2.msi /quiet /norestart"
  pre_build:
    commands:
      - '&"C:\Program Files\Amazon\AWSCLIV2\aws" codeartifact login --tool twine --
domain my_domain --domain-owner 111122223333 --repository my_repo'
  build:
    commands:
      - twine upload --repository codeartifact mypackage

```

## Menggunakan paket Maven diCodeBuild

### Mengatur izin dengan IAM role

Langkah-langkah ini diperlukan saat menggunakan paket Maven dariCodeArtifactdiCodeBuild.

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Peran. PadaPeranhalaman, mengedit peran yang digunakan olehCodeBuildmembangun proyek. Peran ini harus memiliki izin berikut.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {

```

```

    "Effect": "Allow",
    "Action": [ "codeartifact:GetAuthorizationToken",
               "codeartifact:GetRepositoryEndpoint",
               "codeartifact:ReadFromRepository"
             ],
    "Resource": "*"
  },
  {
    "Effect": "Allow",
    "Action": "sts:GetServiceBearerToken",
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "sts:AWSServiceName": "codeartifact.amazonaws.com"
      }
    }
  }
]
}

```

### Important

Jika Anda juga ingin menggunakan CodeBuild untuk mempublikasikan paket, tambahkan **codeartifact:PublishPackageVersion** dan **codeartifact:PutPackageMetadata**.

Untuk informasi selengkapnya, lihat [Modifying a Role](#) dalam Panduan Pengguna IAM.

## Menggunakan gradle atau mvn

Untuk menggunakan paket Maven dengan `gradle` atau `mvn`, simpan `CodeArtifactTokenAuth` dalam variabel lingkungan, seperti yang dijelaskan dalam [Lulus token autentikasi dalam variabel lingkungan](#). Berikut adalah contohnya.

### Note

Hal ini hanya diperlukan untuk meng-upgrade AWS CLI bersama `pip3 install awscli --upgrade --user` jika Anda menggunakan yang lebih tua CodeBuild gambar. Jika Anda menggunakan versi gambar terbaru, Anda dapat menghapus baris tersebut.

```
pre_build:
  commands:
    - pip3 install awscli --upgrade --user
    - export CODEARTIFACT_AUTH_TOKEN=`aws codeartifact get-authorization-token --
      domain my_domain --domain-owner 111122223333 --query authorizationToken --output text`
```

Untuk menggunakan Gradle:

Jika Anda merujuk variabel `CODEARTIFACT_AUTH_TOKEN` di Gradle Anda `build.gradle` seperti yang dijelaskan dalam [Menggunakan CodeArtifact dengan Gradle](#), Anda dapat memanggil build Gradle Anda dari `buildspec.yaml` bagian.

```
build:
  commands:
    - gradle build
```

Untuk menggunakan mvn:

Anda harus mengkonfigurasi file konfigurasi Maven Anda (`settings.xml` dan `pom.xml`) mengikuti instruksi di [Menggunakan CodeArtifact dengan mvn](#).

## Menggunakan NuGet paket di CodeBuild

Langkah-langkah berikut telah diuji dengan sistem operasi yang tercantum dalam [Gambar Docker disediakan oleh CodeBuild](#).

Topik

- [Mengatur izin dengan IAM role](#)
- [Mengonsumsi NuGet paket](#)
- [Bangun dengan NuGet paket](#)
- [Publikasikan NuGet paket](#)

## Mengatur izin dengan IAM role

Langkah-langkah ini diperlukan saat menggunakan NuGet paket dari CodeArtifact di CodeBuild.

1. Masuk ke AWS Management Console dan buka konsol IAM di <https://console.aws.amazon.com/iam/>.

- Di panel navigasi, pilih Peran. PadaPeranhalaman, mengedit peran yang digunakan olehCodeBuildmembangun proyek. Peran ini harus memiliki izin berikut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [ "codeartifact:GetAuthorizationToken",
                  "codeartifact:GetRepositoryEndpoint",
                  "codeartifact:ReadFromRepository"
                ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sts:GetServiceBearerToken",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "sts:AWSServiceName": "codeartifact.amazonaws.com"
        }
      }
    }
  ]
}
```

### Important

Jika Anda juga ingin menggunakanCodeBuilduntuk mempublikasikan paket, tambahkan**codeartifact:PublishPackageVersion**izin.

Untuk informasi selengkapnya, lihat [Modifying a Role](#) dalam Panduan Pengguna IAM.

## MengonsumsiNuGetpaket

MengonsumsiNuGetpaket dariCodeBuild, sertakan yang berikut ini di proyek Andabuildspec.yamlberkas.

1. Dalam `install` bagian, instal `CodeArtifactCredential Provider` untuk mengkonfigurasi alat baris perintah seperti `msbuild` dan `dotnet` untuk membangun dan mempublikasikan paket ke `CodeArtifact`.
2. Dalam `pre-build` bagian, tambahkan `CodeArtifact` repositori ke `NuGet` konfigurasi.

Lihat contoh `buildspec.yaml` berikut. Untuk informasi selengkapnya, lihat [Menggunakan CodeArtifact dengan NuGet](#).

Setelah penyedia kredensi diinstal dan sumber repositori Anda ditambahkan, Anda dapat menjalankan `NuGet` perintah alat CLI dari `build` bagian untuk mengonsumsi `NuGet` paket.

## Linux

Mengonsumsi `NuGet` paket menggunakan `dotnet`:

```
version: 0.2

phases:
  install:
    runtime-versions:
      dotnet: latest
    commands:
      - export PATH="$PATH:/root/.dotnet/tools"
      - dotnet tool install -g AWS.CodeArtifact.NuGet.CredentialProvider
      - dotnet codeartifact-creds install
  pre_build:
    commands:
      - dotnet nuget add source -n codeartifact $(aws codeartifact get-repository-endpoint --domain my_domain --domain-owner 111122223333 --repository my_repo --format nuget --query repositoryEndpoint --output text)"v3/index.json"
  build:
    commands:
      - dotnet add package <packageName> --source codeartifact
```

## Windows

Mengonsumsi `NuGet` paket menggunakan `dotnet`:

```
version: 0.2
```



```
phases:
  install:
    commands:
      - dotnet tool install -g AWS.CodeArtifact.NuGet.CredentialProvider
      - dotnet codeartifact-creds install
  pre_build:
    commands:
      - dotnet nuget add source -n codeartifact "$(aws codeartifact get-repository-
endpoint --domain my_domain --domain-owner 111122223333 --repository my_repo --format
nuget --query repositoryEndpoint --output text)v3/index.json"
  build:
    commands:
      - dotnet add package <packageName> --source codeartifact
```

## Bangun denganNuGetpaket

Untuk membangun denganNuGetpaket dariCodeBuild, sertakan yang berikut ini di proyek Andabuildspec.yamlberkas.

1. Dalaminstallbagian, instalCodeArtifactCredential Provider untuk mengkonfigurasi alat baris perintah sepertimsbuilddandotnetuntuk membangun dan mempublikasikan paket keCodeArtifact.
2. Dalampre-buildbagian, tambahkanCodeArtifactrepositori keNuGetkonfigurasi.

Lihat contoh buildspec.yaml berikut. Untuk informasi selengkapnya, lihat [Menggunakan CodeArtifact dengan NuGet](#).

Setelah penyedia kredensi diinstal dan sumber repositori Anda ditambahkan, Anda dapat menjalankanNuGetPerintah alat CLI sepertidotnet builddaribuildbagian.

## Linux

Untuk membangunNuGetpaket menggunakandotnet:

```
version: 0.2

phases:
  install:
    runtime-versions:
      dotnet: latest
    commands:
```

```

- export PATH="$PATH:/root/.dotnet/tools"
- dotnet tool install -g AWS.CodeArtifact.NuGet.CredentialProvider
- dotnet codeartifact-creds install
pre_build:
  commands:
    - dotnet nuget add source -n codeartifact $(aws codeartifact get-repository-
endpoint --domain my_domain --domain-owner 111122223333 --repository my_repo --format
nuget --query repositoryEndpoint --output text)"v3/index.json"
build:
  commands:
    - dotnet build

```

Untuk membangun NuGet paket menggunakan msbuild:

```

version: 0.2

phases:
  install:
    runtime-versions:
      dotnet: latest
    commands:
      - export PATH="$PATH:/root/.dotnet/tools"
      - dotnet tool install -g AWS.CodeArtifact.NuGet.CredentialProvider
      - dotnet codeartifact-creds install
  pre_build:
    commands:
      - dotnet nuget add source -n codeartifact $(aws codeartifact get-repository-
endpoint --domain my_domain --domain-owner 111122223333 --repository my_repo --format
nuget --query repositoryEndpoint --output text)"v3/index.json"
  build:
    commands:
      - msbuild -t:Rebuild -p:Configuration=Release

```

## Windows

Untuk membangun NuGet paket menggunakan dotnet:

```

version: 0.2

phases:
  install:
    commands:
      - dotnet tool install -g AWS.CodeArtifact.NuGet.CredentialProvider

```

```

- dotnet codeartifact-creds install
pre_build:
  commands:
    - dotnet nuget add source -n codeartifact "$(aws codeartifact get-repository-
endpoint --domain my_domain --domain-owner 111122223333 --repository my_repo --format
nuget --query repositoryEndpoint --output text)v3/index.json"
  build:
    commands:
      - dotnet build

```

Untuk membangun NuGet paket menggunakan msbuild:

```

version: 0.2

phases:
  install:
    commands:
      - dotnet tool install -g AWS.CodeArtifact.NuGet.CredentialProvider
      - dotnet codeartifact-creds install
  pre_build:
    commands:
      - dotnet nuget add source -n codeartifact "$(aws codeartifact get-repository-
endpoint --domain my_domain --domain-owner 111122223333 --repository my_repo --format
nuget --query repositoryEndpoint --output text)v3/index.json"
  build:
    commands:
      - msbuild -t:Rebuild -p:Configuration=Release

```

## Publikasikan NuGet paket

Untuk mempublikasikan NuGet paket dari CodeBuild, sertakan yang berikut ini di proyek `Anda/buildspec.yaml` berkas.

1. Dalam `install` bagian, instal `CodeArtifactCredentialProvider` untuk mengkonfigurasi alat baris perintah seperti `msbuild` dan `dotnet` untuk membangun dan mempublikasikan paket ke `CodeArtifact`.
2. Dalam `pre-build` bagian, tambahkan `CodeArtifact` repositori ke `NuGet` konfigurasi.

Lihat contoh `buildspec.yaml` berikut. Untuk informasi selengkapnya, lihat [Menggunakan CodeArtifact dengan NuGet](#).

Setelah penyedia kredensi diinstal dan sumber repositori Anda ditambahkan, Anda dapat menjalankanNuGetPerintah alat CLI dari build bagian dan mempublikasikanNuGetpaket.

## Linux

Untuk mempublikasikanNuGetpaket menggunakandotnet:

```
version: 0.2

phases:
  install:
    runtime-versions:
      dotnet: latest
    commands:
      - export PATH="$PATH:/root/.dotnet/tools"
      - dotnet tool install -g AWS.CodeArtifact.NuGet.CredentialProvider
      - dotnet codeartifact-creds install
  pre_build:
    commands:
      - dotnet nuget add source -n codeartifact $(aws codeartifact get-repository-
endpoint --domain my_domain --domain-owner 111122223333 --repository my_repo --format
nuget --query repositoryEndpoint --output text)"v3/index.json"
  build:
    commands:
      - dotnet pack -o .
      - dotnet nuget push *.nupkg -s codeartifact
```

## Windows

Untuk mempublikasikanNuGetpaket menggunakandotnet:

```
version: 0.2

phases:
  install:
    commands:
      - dotnet tool install -g AWS.CodeArtifact.NuGet.CredentialProvider
      - dotnet codeartifact-creds install
  pre_build:
    commands:
      - dotnet nuget add source -n codeartifact "$(aws codeartifact get-repository-
endpoint --domain my_domain --domain-owner 111122223333 --repository my_repo --format
nuget --query repositoryEndpoint --output text)v3/index.json"
```

```
build:
  commands:
    - dotnet pack -o .
    - dotnet nuget push *.nupkg -s codeartifact
```

## Caching Dependensi

Anda dapat mengaktifkan caching lokal di CodeBuild untuk mengurangi jumlah dependensi yang perlu diambil CodeArtifact untuk setiap membangun. Untuk informasi, lihat [Build Caching in AWS CodeBuild](#) di AWS CodeBuild Panduan Pengguna. Setelah mengaktifkan cache lokal kustom, tambahkan direktori cache ke file `buildspec.yaml` proyek Anda.

Misalnya, jika Anda menggunakan `mvn`, gunakan berikut.

```
cache:
  paths:
    - '/root/.m2/**/*'
```

Untuk alat lainnya, gunakan folder cache yang ditampilkan dalam tabel ini.

Alat	Direktori cache
<b>mvn</b>	<code>/root/.m2/**/*</code>
<b>gradle</b>	<code>/root/.gradle/caches/**/*</code>
<b>pip</b>	<code>/root/.cache/pip/**/*</code>
<b>npm</b>	<code>/root/.npm/**/*</code>
<b>nuget</b>	<code>/root/.nuget/**/*</code>
<b>yarn (classic)</b>	<code>/root/.cache/yarn/**/*</code>

# Pemantauan CodeArtifact

Pemantauan adalah bagian penting dari pemeliharaan keandalan, ketersediaan, dan kinerja CodeArtifact dan solusi AWS lain Anda. AWS menyediakan alat pemantauan berikut untuk memantau CodeArtifact, melaporkan jika ada yang salah, dan mengambil tindakan otomatis jika diperlukan:

- Anda dapat menggunakan Amazon EventBridge untuk mengotomatiskan AWS layanan Anda dan merespons secara otomatis peristiwa sistem, seperti masalah ketersediaan aplikasi atau perubahan sumber daya. Acara dari AWS layanan dikirimkan ke EventBridge dalam waktu dekat. Anda dapat menuliskan aturan sederhana untuk menunjukkan peristiwa mana yang sesuai kepentingan Anda, dan tindakan otomatis mana yang diambil ketika suatu peristiwa sesuai dengan suatu aturan. Untuk informasi selengkapnya, lihat [Panduan EventBridge Pengguna Amazon](#) dan [CodeArtifact format acara dan contoh](#).
- Anda dapat menggunakan CloudWatch metrik Amazon untuk melihat CodeArtifact penggunaan berdasarkan operasi. CloudWatch metrik mencakup semua permintaan yang dibuat CodeArtifact, dan permintaan ditampilkan berdasarkan akun. Anda dapat melihat metrik ini dalam CloudWatch metrik dengan menavigasi ke namespace Usage/By AWS Resource. AWS Untuk informasi selengkapnya, lihat [Menggunakan CloudWatch metrik](#) Amazon di Panduan CloudWatch Pengguna Amazon.

## Topik

- [Memantau CodeArtifact peristiwa](#)
- [Gunakan acara untuk memulai CodePipeline eksekusi](#)
- [Menggunakan peristiwa untuk menjalankan fungsi Lambda](#)

## Memantau CodeArtifact peristiwa

CodeArtifact terintegrasi dengan Amazon EventBridge, layanan yang mengotomatiskan dan merespons peristiwa, termasuk perubahan dalam repositori. CodeArtifact Anda dapat membuat aturan untuk acara dan mengonfigurasi apa yang terjadi ketika acara cocok dengan aturan. EventBridge Dulunya bernama CloudWatch Events.

Tindakan berikut dapat dipicu oleh suatu peristiwa:

- Memanggil fungsi AWS Lambda.
- Mengaktifkan mesin status AWS Step Functions.
- Memberi tahu topik Amazon SNS atau antrean Amazon SQS.
- Memulai alur di AWS CodePipeline.

CodeArtifact membuat acara ketika versi paket dibuat, dimodifikasi, atau dihapus. Berikut ini adalah contoh CodeArtifact peristiwa:

- Memublikasikan versi paket baru (misalnya, dengan menjalankan `npm publish`).
- Menambahkan aset baru ke versi paket yang ada (misalnya, dengan mendorong file JAR baru untuk paket Maven yang ada).
- Menyalin versi paket dari satu repositori ke repositori lainnya menggunakan `copy-package-versions`. Untuk informasi selengkapnya, lihat [Menyalin paket antar-repositori](#).
- Menghapus versi paket menggunakan `delete-package-versions`. Untuk informasi selengkapnya, lihat [Menghapus paket](#).
- Menghapus versi paket menggunakan `delete-package`. Satu acara akan dipublikasikan untuk setiap versi paket yang dihapus. Untuk informasi selengkapnya, lihat [Menghapus paket](#).
- Mempertahankan versi paket dalam repositori hilir ketika telah diambil dari repositori hulu. Untuk informasi selengkapnya, lihat [Bekerja dengan repositori hulu di CodeArtifact](#).
- Menelan versi paket dari repositori eksternal ke dalam repositori. CodeArtifact Untuk informasi selengkapnya, lihat [Connect CodeArtifact repositori ke repositori publik](#).

Peristiwa dikirim ke kedua akun yang memiliki domain dan akun yang mengelola repositori. Misalnya, anggaplah akun 111111111111 memiliki domain `my_domain`. Akun 222222222222 membuat repositori `my_domain` bernama `repo2`. Ketika versi paket baru diterbitkan ke `repo2`, kedua akun menerima EventBridge acara. Akun pemilik domain (111111111111) menerima peristiwa untuk semua repositori dalam domain. Jika satu akun memiliki domain dan repositori di dalamnya, hanya satu peristiwa yang dikirimkan.

Topik berikut menjelaskan format CodeArtifact acara. Mereka menunjukkan kepada Anda cara mengonfigurasi CodeArtifact acara, dan cara menggunakan acara dengan AWS layanan lain. Untuk informasi selengkapnya, lihat [Memulai Amazon EventBridge](#) di Panduan EventBridge Pengguna Amazon.

## CodeArtifact format acara dan contoh

Berikut ini adalah bidang acara dan deskripsi bersama dengan contoh CodeArtifact acara.


### CodeArtifact format acara

Semua CodeArtifact acara mencakup bidang-bidang berikut.

Bidang peristiwa	Deskripsi
versi	Versi format peristiwa. Saat ini hanya ada satu versi, 0.
id	Pengidentifikasi unik untuk peristiwa.
jenis-detail	Jenis peristiwa. Ini menentukan bidang dalam objek detail. Saat ini, yang didukung detail-type adalah CodeArtifact Package Version State Change.
sumber	Sumber peristiwa. Karena CodeArtifact, itu akan terjadiaws.codeartifact .
akun	ID AWS akun akun yang menerima acara tersebut.
Waktu	Waktu peristiwa dipicu.
wilayah	Wilayah tempat peristiwa tersebut dipicu.
sumber daya	Daftar berisi ARN paket yang berubah. Daftar berisi satu entri. Untuk informasi tentang format ARN paket, lihat <a href="#">Memberikan akses tulis ke paket</a> .
domainName	Domain berisi repositori yang berisi paket.
domainOwner	ID AWS akun pemilik domain.
repositoryName	Repositori yang berisi paket.



Bidang peristiwa	Deskripsi
RepositoryAdministrator	ID AWS akun administrator repositori.
packageFormat	Format paket yang memicu peristiwa.
packageNamespace	Namespace paket yang memicu peristiwa.
packageName	Nama paket yang memicu peristiwa.
packageVersion	Versi paket yang memicu peristiwa.
packageVersionState	Status versi paket ketika peristiwa dipicu. Kemungkinan nilai adalah <code>Unfinished</code> , <code>Published</code> , <code>Unlisted</code> , <code>Archived</code> , dan <code>Disposed</code> .
packageVersionRevision	Nilai yang secara unik mengidentifikasi status aset dan metadata versi paket ketika peristiwa dipicu. Jika versi paket dimodifikasi (misalnya , dengan menambahkan file JAR lain untuk paket Maven), <code>packageVersionRevision</code> berubah.
changes.assetsAdded	Jumlah aset yang ditambahkan ke paket yang memicu peristiwa. Contoh aset adalah file JAR Maven atau wheel Python.
changes.assetsRemoved	Jumlah aset yang dihapus dari paket yang memicu peristiwa.
changes.assetsUpdated	Jumlah aset yang diubah dalam paket yang memicu peristiwa tersebut.
changes.metadataUpdated	Nilai boolean yang diatur ke <code>true</code> jika peristiwa mencakup metadata tingkat paket yang dimodifikasi. Sebagai contoh, sebuah peristiwa mungkin memodifikasi file <code>pom.xml</code> Maven.

Bidang peristiwa	Deskripsi
changes.statusChanged	Nilai boolean yang diatur ke <code>true</code> jika <code>packageVersionStatus</code> peristiwa dimodifikasi (misalnya, jika <code>packageVersionStatus</code> berubah dari <code>Unfinished</code> menjadi <code>Published</code> ).
operationType	Menjelaskan jenis perubahan versi paket tingkat tinggi. Nilai yang mungkin adalah <code>Created</code> , <code>Updated</code> , dan <code>Deleted</code> .
sequenceNumber	<p>Integer yang menentukan nomor peristiwa untuk paket. Setiap peristiwa pada paket akan menambahkan <code>sequenceNumber</code> sehingga peristiwa dapat diatur secara berurutan. Peristiwa dapat menambahkan <code>sequenceNumber</code> dengan sembarang nomor integer.</p> <div data-bbox="829 993 1507 1308" style="border: 1px solid #add8e6; border-radius: 15px; padding: 10px;"><p> <b>Note</b></p><p>EventBridge peristiwa mungkin diterima di luar pesanan. <code>sequenceNumber</code> dapat digunakan untuk menentukan urutan aktual mereka.</p></div>
eventDeduplicationId	ID yang digunakan untuk membedakan peristiwa duplikat. EventBridge Dalam kasus yang jarang terjadi, EventBridge mungkin memicu aturan yang sama lebih dari sekali untuk satu acara atau waktu yang dijadwalkan. Atau, mungkin memanggil target yang sama lebih dari sekali untuk aturan terpicu yang diberikan.

## CodeArtifact contoh acara

Berikut ini adalah contoh CodeArtifact peristiwa yang mungkin dipicu ketika paket npm diterbitkan.

```
{
  "version": "0",
  "id": "73f03fec-a137-971e-6ac6-07c8ffffffff",
  "detail-type": "CodeArtifact Package Version State Change",
  "source": "aws.codeartifact",
  "account": "123456789012",
  "time": "2019-11-21T23:19:54Z",
  "region": "us-west-2",
  "resources": ["arn:aws:codeartifact:us-west-2:111122223333:package/my_domain/myrepo/npm//mypackage"],
  "detail": {
    "domainName": "my_domain",
    "domainOwner": "111122223333",
    "repositoryName": "myrepo",
    "repositoryAdministrator": "123456789012",
    "packageFormat": "npm",
    "packageNamespace": null,
    "packageName": "mypackage",
    "packageVersion": "1.0.0",
    "packageVersionState": "Published",
    "packageVersionRevision": "0E5DE26A4CD79FDF3EBC4924FFFFFFFF",
    "changes": {
      "assetsAdded": 1,
      "assetsRemoved": 0,
      "metadataUpdated": true,
      "assetsUpdated": 0,
      "statusChanged": true
    },
    "operationType": "Created",
    "sequenceNumber": 1,
    "eventDeduplicationId": "2mE00A2Ke07rWUTBXk3CAiQhdTXF4N94LNaT/ffffff="
  }
}
```

## Gunakan acara untuk memulai CodePipeline eksekusi

Contoh ini menunjukkan cara mengonfigurasi EventBridge aturan Amazon sehingga AWS CodePipeline eksekusi dimulai ketika versi paket dalam CodeArtifact repositori diterbitkan, dimodifikasi, atau dihapus.

Topik

- [Konfigurasi EventBridge izin](#)
- [Buat EventBridge aturan](#)
- [Buat target EventBridge aturan](#)

### Konfigurasi EventBridge izin

Anda harus menambahkan izin untuk digunakan EventBridge CodePipeline untuk menjalankan aturan yang Anda buat. Untuk menambahkan izin ini menggunakan AWS Command Line Interface (AWS CLI), ikuti langkah 1 di [Buat Aturan CloudWatch Acara untuk CodeCommit Sumber \(CLI\)](#) di Panduan Pengguna AWS CodePipeline.

### Buat EventBridge aturan

Untuk membuat aturan, gunakan perintah `put-rule` dengan parameter `--name` dan `--event-pattern`. Pola peristiwa menentukan nilai-nilai yang cocok terhadap isi dari setiap peristiwa. Target dipicu jika pola cocok dengan peristiwa. Misalnya, pola berikut cocok dengan CodeArtifact peristiwa dari `myrepo` repositori di domain `my_domain`

```
aws events put-rule --name MyCodeArtifactRepoRule --event-pattern \  
  '{"source":["aws.codeartifact"],"detail-type":["CodeArtifact Package Version State Change"],  
  "detail":{"domainName":["my_domain"],"domainOwner":  
  ["111122223333"],"repositoryName":["myrepo]}}'
```

### Buat target EventBridge aturan

Perintah berikut menambahkan target ke aturan sehingga ketika suatu peristiwa cocok dengan aturan, CodePipeline eksekusi dipicu. Untuk parameter `RoleArn`, tentukan Amazon Resource Name (ARN) peran yang dibuat sebelumnya dalam topik ini.

```
aws events put-targets --rule MyCodeArtifactRepoRule --targets \  
  [{"roleArn": "arn:aws:iam::111122223333:role/MyCodeArtifactRepoRuleTargetRole"}]
```

```
'Id=1,Arn=arn:aws:codepipeline:us-west-2:111122223333:pipeline-name,  
RoleArn=arn:aws:iam::123456789012:role/MyRole'
```

## Menggunakan peristiwa untuk menjalankan fungsi Lambda

Contoh ini menunjukkan cara mengonfigurasi EventBridge aturan yang memulai AWS Lambda fungsi saat versi paket dalam CodeArtifact repositori diterbitkan, dimodifikasi, atau dihapus.

Untuk informasi selengkapnya, lihat [Tutorial: Menjadwalkan AWS Lambda Fungsi Menggunakan EventBridge](#) di Panduan EventBridge Pengguna Amazon.

### Topik

- [Buat EventBridge aturan](#)
- [Buat target EventBridge aturan](#)
- [Konfigurasi EventBridge izin](#)

## Buat EventBridge aturan

Untuk membuat aturan yang memulai fungsi Lambda, gunakan perintah `put-rule` dengan opsi `--name` dan `--event-pattern`. Pola berikut menentukan paket npm dalam cakupan `@types` di repositori apa pun dalam domain `my_domain`.

```
aws events put-rule --name "MyCodeArtifactRepoRule" --event-pattern \  
  '{"source":["aws.codeartifact"],"detail-type":["CodeArtifact Package Version State  
Change"],  
  "detail":{"domainName":["my_domain"],"domainOwner":  
["111122223333"],"packageNamespace":["types"],"packageFormat":["npm"]}}'
```

## Buat target EventBridge aturan

Perintah berikut menambahkan target untuk aturan yang menjalankan fungsi Lambda ketika sebuah peristiwa cocok dengan aturan. Untuk parameter `arn`, tentukan Amazon Resource Name (ARN) fungsi Lambda.

```
aws events put-targets --rule MyCodeArtifactRepoRule --targets \  
  Id=1,Arn=arn:aws:lambda:us-west-2:111122223333:function:MyLambdaFunction
```

## Konfigurasi EventBridge izin

Gunakan perintah `add-permission` untuk memberikan izin pada aturan untuk memanggil fungsi Lambda. Untuk parameter `--source-arn`, tentukan ARN aturan yang Anda buat sebelumnya dalam contoh ini.

```
aws lambda add-permission --function-name MyLambdaFunction \<\  
  --statement-id my-statement-id --action 'lambda:InvokeFunction' \<\  
  --principal events.amazonaws.com \<\  
  --source-arn arn:aws:events:us-west-2:111122223333:rule/MyCodeArtifactRepoRule
```

# Keamanan di CodeArtifact

Keamanan cloud di AWS merupakan prioritas tertinggi. Sebagai pelanggan AWS, Anda mendapatkan manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara AWS dan Anda. [Model tanggung jawab bersama](#) menggambarkan ini sebagai keamanan dari cloud dan keamanan di dalam cloud:

- Keamanan cloud – AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan layanan-layanan AWS di dalam AWS Cloud. AWS juga memberikan Anda layanan yang dapat digunakan dengan aman. Auditor pihak ketiga melakukan pengujian dan verifikasi secara berkala terhadap efektivitas keamanan kami sebagai bagian dari [Program Kepatuhan AWS](#). Untuk mempelajari tentang program kepatuhan yang berlaku CodeArtifact, lihat [Layanan AWS dalam Lingkup menurut Program Kepatuhan](#).
- Keamanan di cloud – Tanggung jawab Anda ditentukan menurut layanan AWS yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain termasuk sensitivitas data Anda, persyaratan perusahaan Anda, serta hukum dan peraturan yang berlaku.

Dokumentasi ini akan membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan CodeArtifact. Topik berikut menunjukkan cara mengonfigurasi CodeArtifact untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga mempelajari cara menggunakan layanan AWS lain yang membantu untuk memantau dan mengamankan sumber daya CodeArtifact.

## Topik

- [Perlindungan data di AWS CodeArtifact](#)
- [Pemantauan CodeArtifact](#)
- [Validasi kepatuhan untuk AWS CodeArtifact](#)
- [AWS CodeArtifact otentikasi dan token](#)
- [Ketahanan di AWS CodeArtifact](#)
- [Keamanan infrastruktur di AWS CodeArtifact](#)
- [Serangan substitusi ketergantungan](#)
- [Identity and Access Management untuk AWS CodeArtifact](#)

# Perlindungan data di AWS CodeArtifact

[Model tanggung jawab AWS bersama model tanggung](#) berlaku untuk perlindungan data di AWS CodeArtifact. Sebagaimana diuraikan dalam model ini, AWS bertanggung jawab untuk memberikan perlindungan terhadap infrastruktur global yang menjalankan semua AWS Cloud. Anda harus bertanggung jawab untuk memelihara kendali terhadap konten yang di-hosting pada infrastruktur ini. Anda juga bertanggung jawab atas konfigurasi keamanan dan tugas manajemen untuk berbagai layanan Layanan AWS yang Anda gunakan. Untuk informasi selengkapnya tentang privasi data, silakan lihat [Pertanyaan Umum Privasi Data](#). Untuk informasi tentang perlindungan data di Eropa, silakan lihat postingan blog [Model Tanggung Jawab Bersama AWS dan GDPR](#) di Blog Keamanan AWS.

Untuk tujuan perlindungan data, sebaiknya Anda melindungi kredensial Akun AWS dan menyiapkan akun pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara seperti itu, 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:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk melakukan komunikasi dengan sumber daya AWS. Kami mensyaratkan TLS 1.2 dan merekomendasikan TLS 1.3.
- Siapkan API dan log aktivitas pengguna dengan AWS CloudTrail.
- Gunakan solusi enkripsi AWS, bersama dengan semua kontrol keamanan default dalam layanan Layanan AWS.
- Gunakan layanan keamanan terkelola lanjutan seperti Amazon Macie, yang membantu menemukan dan mengamankan data pribadi yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-2 ketika mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Untuk informasi selengkapnya tentang titik akhir FIPS yang tersedia, silakan lihat [Standar Pemrosesan Informasi Federal \(FIPS\) 140-2](#).

Sebaiknya Anda tidak memasukkan informasi rahasia atau sensitif, seperti alamat email pelanggan, ke dalam tanda atau bidang teks bebas seperti bidang Nama. Ini termasuk saat Anda bekerja dengan CodeArtifact atau lainnya Layanan AWS menggunakan konsol, APIAWS CLI, atau AWS SDK. Data apa pun yang Anda masukkan ke dalam tanda atau bidang teks bebas yang digunakan untuk nama dapat digunakan untuk log penagihan atau diagnostik. Saat Anda memberikan URL ke server



eksternal, sebaiknya jangan menyertakan informasi kredensial di URL untuk memvalidasi permintaan Anda ke server tersebut.

## Enkripsi data

Enkripsi adalah bagian penting dari CodeArtifact keamanan. Beberapa enkripsi, seperti untuk data dalam transit, disediakan secara default dan Anda tidak perlu melakukan apa pun. Enkripsi lain, seperti untuk data at rest, Anda dapat mengonfigurasi ketika Anda membuat proyek atau build.

- Enkripsi data saat istirahat - Semua aset yang disimpan CodeArtifact dienkripsi dengan menggunakan AWS KMS keys (kunci KMS). Ini mencakup semua aset dalam semua paket di semua repositori. Satu kunci KMS digunakan untuk setiap domain untuk mengenkripsi semua asetnya. Secara default, kunci KMS AWS terkelola digunakan, jadi Anda tidak perlu membuat kunci KMS. Jika mau, Anda dapat menggunakan kunci KMS yang dikelola pelanggan yang Anda buat dan konfigurasi. Untuk informasi selengkapnya, lihat [Creating keys](#) dan [AWS Key Management Service concepts](#) di Panduan Pengguna AWS Key Management Service. Anda dapat menentukan kunci KMS yang dikelola pelanggan saat membuat domain. Untuk informasi selengkapnya, lihat [Bekerja dengan domain di CodeArtifact](#).
- Enkripsi data dalam perjalanan - Semua komunikasi antara pelanggan dan antara CodeArtifact CodeArtifact dan dependensi hilirnya dilindungi menggunakan enkripsi TLS.

## Privasi lalu lintas

Anda dapat meningkatkan keamanan CodeArtifact domain Anda dan aset yang dikandungnya dengan mengonfigurasi CodeArtifact untuk menggunakan titik akhir antarmuka virtual private cloud (VPC). Anda tidak memerlukan gateway internet, perangkat NAT, atau virtual private gateway. Untuk informasi selengkapnya, lihat [Bekerja dengan Amazon VPC endpoint](#). Untuk informasi selengkapnya tentang AWS PrivateLink dan titik akhir VPC, lihat dan [AWS PrivateLink](#) Mengakses Layanan [AWS](#) Melalui PrivateLink

## Pemantauan CodeArtifact

Pemantauan adalah bagian penting dari pemeliharaan keandalan, ketersediaan, dan performa AWS CodeArtifact dan solusi AWS Anda. Anda harus mengumpulkan data pemantauan dari semua bagian solusi AWS Anda agar dapat lebih mudah melakukan debug kegagalan beberapa titik jika terjadi. AWS menyediakan beberapa alat untuk memantau sumber daya CodeArtifact Anda dan merespons potensi insiden:

## Topik

- [Mencatat panggilan API CodeArtifact dengan AWS CloudTrail](#)

## Mencatat panggilan API CodeArtifact dengan AWS CloudTrail

CodeArtifact terintegrasi dengan [AWS CloudTrail](#), layanan yang menyediakan catatan tindakan yang dilakukan oleh pengguna, peran, atau layanan AWS di CodeArtifact. CloudTrail merekam semua panggilan API untuk CodeArtifact sebagai kejadian, termasuk panggilan dari klien manajer paket.

Jika membuat jejak, Anda dapat mengaktifkan pengiriman berkelanjutan kejadian CloudTrail ke bucket Amazon Simple Storage Service (Amazon S3), termasuk kejadian untuk CodeArtifact. Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa terbaru dalam konsol CloudTrail di Riwayat peristiwa. Menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat ke CodeArtifact, alamat IP asal permintaan tersebut dibuat, siapa yang membuat permintaan, kapan permintaan dibuat, dan detail lainnya.

Untuk mempelajari selengkapnya tentang CloudTrail, lihat [Panduan Pengguna AWS CloudTrail](#).

### Informasi CodeArtifact di CloudTrail

CloudTrail diaktifkan pada akun AWS Anda saat Anda membuat akun tersebut. Ketika aktivitas terjadi di CodeArtifact, aktivitas tersebut dicatat dalam kejadian CloudTrail bersama kejadian layanan AWS lainnya di Riwayat kejadian. Anda dapat melihat, mencari, dan mengunduh peristiwa terbaru di akun AWS Anda. Untuk informasi selengkapnya, lihat [Melihat Peristiwa dengan Riwayat Peristiwa CloudTrail](#).

Untuk catatan berkelanjutan tentang kejadian di akun AWS Anda, termasuk kejadian untuk CodeArtifact, buat jejak. Sebuah Jejak mengaktifkan CloudTrail untuk mengirim berkas log ke bucket Amazon S3. Secara default, ketika Anda membuat jejak di konsol tersebut, jejak tersebut diterapkan ke semua Wilayah AWS. Jejak mencatat kejadian dari semua Wilayah di partisi AWS dan mengirimkan berkas log ke bucket Amazon S3 yang Anda tentukan. Anda juga dapat mengonfigurasi layanan AWS lainnya untuk menganalisis lebih lanjut dan menindaklanjuti data kejadian yang dikumpulkan di log CloudTrail. Untuk informasi lain, lihat topik berikut:

- [Membuat Jejak untuk Akun AWS Anda](#)
- [Layanan yang Didukung dan Integrasi CloudTrail](#)
- [Mengonfigurasi Notifikasi Amazon SNS untuk CloudTrail](#)

Saat pencatatan CloudTrail diaktifkan di akun AWS, panggilan API yang dibuat untuk tindakan CodeArtifact dilacak dalam file log CloudTrail, yang ditulis bersama catatan layanan AWS lainnya. CloudTrail menentukan kapan membuat dan menulis ke file baru berdasarkan periode waktu dan ukuran file.

Semua tindakan CodeArtifact dicatat oleh CloudTrail. Misalnya, panggilan ke `ListRepositories` (di dalam AWS CLI, `aws codeartifact list-repositories`), `CreateRepository` (`aws codeartifact create-repository`), dan `ListPackages` (`aws codeartifact list-packages`) Tindakan menghasilkan entri di berkas log CloudTrail, selain perintah klien. Perintah klien manajer paket biasanya membuat lebih dari satu permintaan HTTP ke server. Setiap permintaan menghasilkan log acara CloudTrail terpisah.

Pengiriman log CloudTrail lintas akun

Hingga tiga akun terpisah menerima log CloudTrail untuk satu panggilan API:

- Akun yang membuat permintaan—misalnya, akun yang memanggil `GetAuthorizationToken`.
- Akun administrator repositori—misalnya, akun yang mengelola repositori tempat `ListPackages` dipanggil.
- Akun pemilik domain—misalnya, akun yang memiliki domain yang berisi repositori tempat API dipanggil.

Untuk API seperti `ListRepositoriesInDomain` yang merupakan tindakan terhadap domain dan bukan repositori tertentu, hanya akun panggilan dan akun pemilik domain yang menerima log CloudTrail. Untuk API seperti `ListRepositories` yang tidak diotorisasi terhadap sumber daya apa pun, hanya akun pemanggil yang menerima log CloudTrail.

## Memahami entri file log CodeArtifact

File log CloudTrail berisi satu atau beberapa entri log. Setiap entri berisi beberapa peristiwa yang diformat JSON. Sebuah log acara mewakili satu permintaan dari sumber apa pun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. Entri log bukan jejak tumpukan yang dipesan dari panggilan API publik, sehingga tidak muncul dalam urutan tertentu.

Topik

- [Contoh: Entri log untuk memanggil API `GetAuthorizationToken`](#)
- [Contoh: Entri log untuk mengambil versi paket npm](#)

## Contoh: Entri log untuk memanggil API GetAuthorizationToken

Entri log yang dibuat oleh [GetAuthorizationToken](#) termasuk nama domain di bidang `requestParameters`.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:sts::123456789012:assumed-role/Console/example",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2018-12-11T13:31:37Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/Console",
        "accountId": "123456789012",
        "userName": "Console"
      }
    }
  },
  "eventTime": "2018-12-11T13:31:37Z",
  "eventSource": "codeartifact.amazonaws.com",
  "eventName": "GetAuthorizationToken",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "205.251.233.50",
  "userAgent": "aws-cli/1.16.37 Python/2.7.10 Darwin/16.7.0 boto3/1.12.27",
  "requestParameters": {
    "domainName": "example-domain"
    "domainOwner": "123456789012"
  },
  "responseElements": {
    "sessionToken": "HIDDEN_DUE_TO_SECURITY_REASONS"
  },
  "requestID": "6b342fc0-5bc8-402b-a7f1-ffffffffffffff",
  "eventID": "100fde01-32b8-4c2b-8379-ffffffffffffff",
  "readOnly": false,
  "eventType": "AwsApiCall",
}
```

```
"recipientAccountId": "123456789012"
}
```

Contoh: Entri log untuk mengambil versi paket npm

Permintaan yang dibuat oleh semua klien manajer paket, termasuk **npm** klien, memiliki data tambahan yang dicatat termasuk nama domain, nama repositori, dan nama paket direquestParametersBidang bidang. Jalur URL dan metode HTTP dicatat di bidang additionalEventData.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:sts::123456789012:assumed-role/Console/example",
    "accountId": "123456789012",
    "accessKeyId": "ASIAIJI0BJIBSREXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2018-12-17T02:05:16Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/Console",
        "accountId": "123456789012",
        "userName": "Console"
      }
    }
  },
  "eventTime": "2018-12-17T02:05:46Z",
  "eventSource": "codeartifact.amazonaws.com",
  "eventName": "ReadFromRepository",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "205.251.233.50",
  "userAgent": "npm/6.14.15 node/v12.22.9 linux x64 ci/custom",
  "requestParameters": {
    "domainName": "example-domain",
    "domainOwner": "123456789012",
    "repositoryName": "example-repo",
    "packageName": "lodash",
  }
}
```

```
    "packageFormat": "npm",
    "packageVersion": "4.17.20"
  },
  "responseElements": null,
  "additionalEventData": {
    "httpMethod": "GET",
    "requestUri": "/npm/lodash/-/lodash-4.17.20.tgz"
  },
  "requestID": "9f74b4f5-3607-4bb4-9229-fffffffffffffff",
  "eventID": "c74e40dd-8847-4058-a14d-fffffffffffffff",
  "readOnly": true,
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
}
```

## Validasi kepatuhan untuk AWS CodeArtifact

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan khusus, lihat [Layanan AWS di Scope oleh Program](#) Program Kepatuhan yang Anda minati. Untuk informasi umum, silakan lihat [Program Kepatuhan AWS](#) .

Anda bisa mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat [Mengunduh Laporan di AWS Artifact](#).

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, serta hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- [Panduan Quick Start Keamanan dan Kepatuhan](#) – Panduan Quick Start Keamanan dan Kepatuhan – Panduan deployment ini membahas pertimbangan arsitektur dan menyediakan langkah-langkah untuk melakukan deployment terhadap 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), Dewan Standar Keamanan Industri Kartu Pembayaran (PCI), dan Organisasi Internasional untuk Standardisasi (ISO)).
- [Mengevaluasi Sumber Daya dengan Aturan](#) di 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 pandangan komprehensif tentang status keamanan Anda di dalam AWS. Security Hub menggunakan kontrol keamanan untuk sumber daya AWS Anda dan untuk memeriksa kepatuhan 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 bagaimana Anda mengelola risiko dan kepatuhan terhadap regulasi dan standar industri.

## AWS CodeArtifact otentikasi dan token

CodeArtifact mengharuskan pengguna untuk mengautentikasi dengan layanan untuk mempublikasikan atau menggunakan versi paket. Anda harus mengautentikasi ke CodeArtifact layanan dengan membuat token otorisasi menggunakan kredensi Anda AWS . Untuk membuat token otorisasi, Anda harus memiliki izin yang benar. Untuk izin yang diperlukan untuk membuat token otorisasi, lihat `GetAuthorizationToken` entri di [AWSCodeArtifactreferensi izin](#) Untuk informasi lebih umum tentang CodeArtifact izin, lihat [Bagaimana AWS CodeArtifact bekerja dengan IAM](#).

[Untuk mengambil token otorisasi dari CodeArtifact, Anda harus memanggil API.](#)

[GetAuthorizationToken](#) Dengan menggunakan AWS CLI, Anda dapat menelepon `GetAuthorizationToken` dengan `get-authorization-token` perintah login or.

**Note**

Pengguna root tidak dapat menelepon `GetAuthorizationToken`.

- `aws codeartifact login`: Perintah ini memudahkan untuk mengkonfigurasi manajer paket umum untuk digunakan CodeArtifact dalam satu langkah. Memanggil `login` mengambil token dengan `GetAuthorizationToken` dan mengonfigurasi manajer paket Anda dengan token dan titik akhir CodeArtifact repositori yang benar. Manajer paket dukungan adalah sebagai berikut:
  - `dotnet`
  - `npm`
  - `nuget`
  - `pip`
  - `cepat`
  - `benang`
- `aws codeartifact get-authorization-token`: Untuk manajer paket yang tidak didukung oleh `login`, Anda dapat memanggil `get-authorization-token` secara langsung lalu mengonfigurasi manajer paket Anda dengan token seperti yang diperlukan, misalnya, dengan menambahkannya ke file konfigurasi atau menyimpannya di variabel lingkungan.

CodeArtifact token otorisasi berlaku untuk periode default 12 jam. Token dapat dikonfigurasi dengan masa pakai antara 15 menit dan 12 jam. Ketika masa pakai berakhir, Anda harus mengambil token lain. Masa pakai token dimulai setelah `login` atau `get-authorization-token` dipanggil.

Jika `login` atau `get-authorization-token` dipanggil saat mengasumsikan peran, Anda dapat mengonfigurasi masa pakai token menjadi sama dengan waktu yang tersisa dalam durasi sesi peran dengan menetapkan nilai `--duration-seconds` ke `0`. Jika tidak, masa pakai token independen dari durasi sesi maksimum peran. Misalnya, anggaplah Anda menelepon `sts assume-role` dan menentukan durasi sesi 15 menit, lalu panggil `login` untuk mengambil token CodeArtifact otorisasi. Dalam kasus ini, token berlaku untuk periode 12 jam penuh meskipun ini lebih lama dari durasi sesi 15 menit. Untuk informasi tentang mengontrol durasi sesi, lihat [Menggunakan IAM role](#) di Panduan Pengguna IAM.



## Token dibuat dengan perintah **login**

`aws codeartifact login`Perintah akan mengambil token dengan `GetAuthorizationToken` dan mengkonfigurasi manajer paket Anda dengan token dan titik akhir CodeArtifact repositori yang benar.

Tabel berikut menjelaskan parameter untuk perintah `login`.

Parameter	Wajib	Deskripsi
<code>--tool</code>	Ya	Manajer paket untuk autentikasi. Nilai yang mungkin adalah <code>dotnet</code> , <code>npm</code> , <code>nuget</code> , <code>pip</code> , <code>swift</code> dan <code>twine</code> .
<code>--domain</code>	Ya	Nama domain yang dimiliki repositori.
<code>--domain-owner</code>	Tidak	ID pemilik domain. Parameter ini diperlukan jika mengakses domain yang dimiliki oleh AWS akun yang tidak Anda autentikasi. Untuk informasi selengkapnya, lihat <a href="#">Domain lintas akun</a> .
<code>--repository</code>	Ya	Nama repositori untuk autentikasi.
<code>--duration-seconds</code>	Tidak	Waktu, dalam detik, bahwa informasi login valid. Nilai minimumnya adalah 900* dan nilai maksimumnya adalah 43200.
<code>--namespace</code>	Tidak	Mengaitkan namespace dengan alat repositori Anda.
<code>--dry-run</code>	Tidak	Hanya mencetak perintah yang akan dieksekusi untuk menghubungkan alat Anda dengan repositori Anda tanpa membuat perubahan pada konfigurasi Anda.

Parameter	Wajib	Deskripsi
		*Nilai 0 juga berlaku saat memanggil login saat mengasumsikan peran. Memanggil login dengan <code>--duration-seconds 0</code> menciptakan token dengan masa pakai sama dengan sisa waktu dalam durasi sesi peran yang diasumsikan.

Contoh berikut menunjukkan cara mengambil token otorisasi dengan perintah `login`.

```
aws codeartifact login \  
  --tool dotnet | npm | nuget | pip | swift | twine \  
  --domain my_domain \  
  --domain-owner 111122223333 \  
  --repository my_repo
```

Untuk panduan khusus tentang cara menggunakan perintah `login` dengan `npm`, lihat [Konfigurasi dan gunakan npm dengan CodeArtifact](#). Untuk Python, lihat [Menggunakan CodeArtifact dengan Python](#).

## Token dibuat dengan API `GetAuthorizationToken`

Anda dapat menelepon `get-authorization-token` untuk mengambil token otorisasi dari CodeArtifact

```
aws codeartifact get-authorization-token \  
  --domain my_domain \  
  --domain-owner 111122223333 \  
  --query authorizationToken \  
  --output text
```

Anda dapat mengubah berapa lama token berlaku menggunakan argumen `--duration-seconds`. Nilai minimumnya adalah 900 dan nilai maksimumnya adalah 43200. Contoh berikut membuat token yang akan bertahan selama 1 jam (3600 detik).

```
aws codeartifact get-authorization-token \  
  --domain my_domain \  
  --domain-owner 111122223333 \  
  --query authorizationToken \  
  --output text \  
  --duration-seconds 3600
```

```
--duration-seconds 3600
```

Jika memanggil `get-authorization-token` saat mengasumsikan peran, masa pakai token independen dari durasi sesi maksimum peran. Anda dapat mengonfigurasi token agar kedaluwarsa ketika durasi sesi peran yang diasumsikan berakhir dengan menetapkan `--duration-seconds 0`.

```
aws codeartifact get-authorization-token \  
  --domain my_domain \  
  --domain-owner 111122223333 \  
  --query authorizationToken \  
  --output text \  
  --duration-seconds 0
```

Lihat dokumentasi berikut ini untuk informasi selengkapnya:

- Untuk panduan tentang token dan variabel lingkungan, lihat [Teruskan token auth menggunakan variabel lingkungan](#).
- Untuk pengguna Python, lihat [Mengonfigurasi pip tanpa perintah login](#) atau [Konfigurasi dan gunakan benang dengan CodeArtifact](#).
- Untuk pengguna Maven, lihat [Gunakan CodeArtifact dengan Gradle](#) atau [Gunakan CodeArtifact dengan mvn](#).
- Untuk pengguna npm, lihat [Mengonfigurasi npm tanpa menggunakan perintah login](#).

## Teruskan token auth menggunakan variabel lingkungan

AWS CodeArtifact menggunakan token otorisasi yang dijual oleh `GetAuthorizationToken` API untuk mengautentikasi dan mengotorisasi permintaan dari alat build seperti Maven dan Gradle. Untuk informasi selengkapnya mengenai topik ini, lihat [Token dibuat dengan API `GetAuthorizationToken`](#).

Anda dapat menyimpan token autentikasi ini dalam variabel lingkungan yang dapat dibaca oleh alat build untuk mendapatkan token yang dibutuhkan untuk mengambil paket dari CodeArtifact repositori atau mempublikasikan paket ke dalamnya.

Untuk alasan keamanan, pendekatan ini lebih disukai untuk menyimpan token dalam file yang mungkin dibaca oleh pengguna atau proses lain, atau tidak sengaja masuk ke kontrol sumber.

1. Konfigurasi AWS kredensial Anda seperti yang dijelaskan dalam [Menginstal atau meningkatkan dan kemudian mengonfigurasi AWS CLI](#)

## 2. Atur variabel lingkungan CODEARTIFACT\_AUTH\_TOKEN:

### Note

Dalam beberapa skenario, Anda tidak perlu menyertakan argumen `--domain-owner`. Untuk informasi selengkapnya, lihat [Domain lintas akun](#).

- macOS atau Linux:

```
export CODEARTIFACT_AUTH_TOKEN=`aws codeartifact get-authorization-token --
domain my_domain --domain-owner 111122223333 --query authorizationToken --output
text`
```

- Windows (menggunakan shell perintah default):

```
for /f %i in ('aws codeartifact get-authorization-token --domain my_domain --
domain-owner 111122223333 --query authorizationToken --output text') do set
CODEARTIFACT_AUTH_TOKEN=%i
```

- Jendela PowerShell:

```
$env:CODEARTIFACT_AUTH_TOKEN = aws codeartifact get-authorization-token --
domain my_domain --domain-owner 111122223333 --query authorizationToken --output
text
```

## Mencabut token otorisasi CodeArtifact

Ketika pengguna yang diautentikasi membuat token untuk mengakses CodeArtifact sumber daya, token itu bertahan hingga periode akses yang dapat disesuaikan telah berakhir. Periode akses default adalah 12 jam. Dalam beberapa situasi, Anda mungkin ingin mencabut akses ke token sebelum periode akses telah kedaluwarsa. Anda dapat mencabut akses ke CodeArtifact sumber daya dengan mengikuti petunjuk ini.

Jika Anda membuat token akses menggunakan kredensial keamanan sementara, seperti peran yang diasumsikan atau akses pengguna gabungan, Anda dapat mencabut akses dengan memperbarui kebijakan IAM untuk menolak akses. Untuk informasi, lihat [Menonaktifkan Izin untuk Kredensial Keamanan Sementara](#) di Panduan Pengguna IAM.

Jika Anda menggunakan kredensial pengguna IAM jangka panjang untuk membuat token akses, Anda harus mengubah kebijakan pengguna untuk menolak akses, atau menghapus pengguna IAM. Untuk informasi selengkapnya, lihat [Changing Permissions for an IAM User](#) atau [Deleting an IAM User](#).

## Ketahanan diAWS CodeArtifact

Infrastruktur global AWS dibangun di sekitar Wilayah AWS dan Availability Zone. AWS Wilayah menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan jaringan berlatensi rendah, throughput yang tinggi, dan sangat redundan. AWS CodeArtifact beroperasi di beberapa Availability Zone dan menyimpan data artefak dan metadata di Amazon S3 dan Amazon DynamoDB. Data terenkripsi Anda disimpan secara berlebihan di berbagai fasilitas dan beberapa perangkat di setiap fasilitas, sehingga sangat tersedia dan sangat berdaya tahan.

Untuk informasi selengkapnya tentang Wilayah AWS dan Zona Ketersediaan, lihat [Infrastruktur Global AWS](#).

## Keamanan infrastruktur diAWS CodeArtifact

Sebagai layanan terkelola, AWS CodeArtifact dilindungi oleh AWS keamanan jaringan global. Untuk informasi tentang AWS layanan keamanan dan bagaimana AWS melindungi infrastruktur, lihat [AWS Keamanan Cloud](#). Untuk mendesain AWS lingkungan menggunakan praktik terbaik untuk keamanan infrastruktur, lihat [Perlindungan Infrastruktur](#) di Pilar Keamanan AWS Kerangka Kerja yang Diarsiteksikan.

Anda menggunakan panggilan API AWS yang dipublikasikan untuk mengakses CodeArtifact melalui jaringan. Klien harus mendukung hal berikut:

- Transport Layer Security (TLS). Kami membutuhkan TLS 1.2 dan merekomendasikan TLS 1.3.
- Suite cipher dengan kerahasiaan maju sempurna (PFS) seperti DHE (Ephemeral Diffie-Hellman) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Sebagian besar sistem modern seperti Java 7 dan sistem yang lebih baru mendukung mode ini.

Selain itu, permintaan harus ditandatangani menggunakan access key ID dan secret access key yang terkait dengan principal IAM. Atau Anda bisa menggunakan [AWS Security Token Service](#) (AWS STS) untuk membuat kredensial keamanan sementara guna menandatangani permintaan.

## Serangan substitusi ketergantungan

Manajer Package menyederhanakan proses pengemasan dan berbagi kode yang dapat digunakan kembali. Paket-paket ini mungkin merupakan paket pribadi yang dikembangkan oleh organisasi untuk digunakan dalam aplikasi mereka, atau mereka mungkin bersifat publik, biasanya paket sumber terbuka yang dikembangkan di luar organisasi dan didistribusikan oleh repositori paket publik. Saat meminta paket, pengembang mengandalkan manajer paket mereka untuk mengambil versi baru dependensi mereka. Serangan substitusi ketergantungan, juga dikenal sebagai serangan kebingungan dependensi, mengeksploitasi fakta bahwa manajer paket biasanya tidak memiliki cara untuk membedakan versi paket yang sah dari versi berbahaya.

Serangan substitusi ketergantungan termasuk dalam subset peretasan yang dikenal sebagai serangan rantai pasokan perangkat lunak. Serangan rantai pasokan perangkat lunak adalah serangan yang memanfaatkan kerentanan di mana saja dalam rantai pasokan perangkat lunak.

Serangan substitusi dependensi dapat menargetkan siapa saja yang menggunakan paket dan paket yang dikembangkan secara internal yang diambil dari repositori publik. Penyerang mengidentifikasi nama paket internal dan kemudian menempatkan kode berbahaya secara strategis dengan nama yang sama di repositori paket publik. Biasanya, kode berbahaya diterbitkan dalam paket dengan nomor versi tinggi. Manajer Package mengambil kode berbahaya dari umpan publik ini karena mereka percaya bahwa paket berbahaya adalah versi terbaru dari paket. Hal ini menyebabkan “kebingungan” atau “substitusi” antara paket yang diinginkan dan paket berbahaya, yang mengarah ke kode yang dikompromikan.

Untuk mencegah serangan substitusi ketergantungan, AWS CodeArtifact menyediakan kontrol asal paket. Kontrol asal Package adalah pengaturan yang mengontrol bagaimana paket dapat ditambahkan ke repositori Anda. Kontrol dikonfigurasi secara otomatis ketika versi paket pertama dari paket baru ditambahkan ke CodeArtifact repositori. Kontrol dapat memastikan versi paket tidak dapat dipublikasikan langsung ke repositori Anda dan dicerna dari sumber publik, melindungi Anda dari serangan substitusi ketergantungan. Untuk informasi selengkapnya tentang kontrol asal paket dan cara mengubahnya, lihat [Mengedit kontrol asal paket](#).

## Identity and Access Management untuk AWS CodeArtifact

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk menggunakan sumber daya. CodeArtifact IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

## Topik

- [Audiens](#)
- [Mengautentikasi dengan identitas](#)
- [Mengelola akses menggunakan kebijakan](#)
- [Bagaimana AWS CodeArtifact bekerja dengan IAM](#)
- [Contoh kebijakan berbasis identitas untuk AWS CodeArtifact](#)
- [Menggunakan tanda untuk mengontrol akses berbasis CodeArtifact peran](#)
- [AWSCodeArtifactreferensi izin](#)
- [Memecahkan masalah AWS CodeArtifact identitas dan akses](#)

## Audiens

Cara Anda menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan. CodeArtifact

**Pengguna layanan** — Jika Anda menggunakan CodeArtifact layanan untuk melakukan pekerjaan Anda, maka administrator Anda memberi Anda kredensi dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak CodeArtifact fitur untuk melakukan pekerjaan Anda, Anda mungkin memerlukan izin tambahan. Memahami cara akses dikelola dapat membantu Anda meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses fitur CodeArtifact, lihat [Memecahkan masalah AWS CodeArtifact identitas dan akses](#).

**Administrator layanan** — Jika Anda bertanggung jawab atas CodeArtifact sumber daya di perusahaan Anda, Anda mungkin memiliki akses penuh ke CodeArtifact. Tugas Anda adalah menentukan CodeArtifact fitur dan sumber daya mana yang harus diakses pengguna layanan Anda. Kemudian, Anda harus mengirimkan permintaan kepada administrator IAM Anda untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep Basic IAM. Untuk mempelajari lebih lanjut tentang bagaimana perusahaan Anda dapat menggunakan IAM CodeArtifact, lihat [Bagaimana AWS CodeArtifact bekerja dengan IAM](#).

**Administrator IAM** - Jika Anda seorang administrator IAM, Anda mungkin ingin mempelajari detail tentang cara menulis kebijakan untuk mengelola akses. CodeArtifact Untuk melihat contoh kebijakan CodeArtifact berbasis identitas yang dapat Anda gunakan di IAM, lihat. [Contoh kebijakan berbasis identitas untuk AWS CodeArtifact](#)

## Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensial identitas Anda. Anda harus diautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengasumsikan peran IAM.

Anda dapat masuk AWS sebagai identitas federasi dengan menggunakan kredensi yang disediakan melalui sumber identitas. AWS IAM Identity Center Pengguna (IAM Identity Center), autentikasi masuk tunggal perusahaan Anda, dan kredensi Google atau Facebook Anda adalah contoh identitas federasi. Saat Anda masuk sebagai identitas gabungan, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil peran.

Bergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal AWS akses. Untuk informasi selengkapnya tentang masuk AWS, lihat [Cara masuk ke Panduan AWS Sign-In Pengguna Anda Akun AWS](#).

Jika Anda mengakses AWS secara terprogram, AWS sediakan kit pengembangan perangkat lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensial Anda. Jika Anda tidak menggunakan AWS alat, Anda harus menandatangani permintaan sendiri. Untuk informasi selengkapnya tentang penggunaan metode yang disarankan untuk menandatangani permintaan sendiri, lihat [Menandatangani permintaan AWS API](#) di Panduan Pengguna IAM.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Misalnya, AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, lihat [Autentikasi multi-faktor](#) dalam Panduan Pengguna AWS IAM Identity Center dan [Menggunakan autentikasi multi-faktor \(MFA\) di AWS](#) dalam Panduan Pengguna IAM.

### Akun AWS pengguna root

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya di akun. Identitas ini disebut pengguna Akun AWS root dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari Anda. 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, mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensi sementara.

Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, direktori Pusat Identitas AWS Directory Service, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensial yang disediakan melalui sumber identitas. Ketika identitas federasi mengakses Akun AWS, mereka mengambil peran, dan peran memberikan kredensi sementara.

Untuk manajemen akses terpusat, kami sarankan Anda menggunakan AWS IAM Identity Center. Anda dapat membuat pengguna dan grup di Pusat Identitas IAM, atau Anda dapat menghubungkan dan menyinkronkan ke sekumpulan pengguna dan grup di sumber identitas Anda sendiri untuk digunakan di semua aplikasi Akun AWS dan aplikasi Anda. Untuk informasi tentang Pusat Identitas IAM, lihat [Apa itu Pusat Identitas IAM?](#) dalam Panduan Pengguna AWS IAM Identity Center .

## Pengguna dan grup IAM

[Pengguna IAM](#) adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, sebaiknya andalkan kredensial sementara daripada membuat pengguna IAM yang memiliki kredensial jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan khusus yang memerlukan kredensial jangka panjang dengan pengguna IAM, kami sarankan Anda merotasi kunci akses. Untuk informasi selengkapnya, lihat [Rotasikan kunci akses secara rutin untuk kasus penggunaan yang memerlukan kredensial jangka panjang](#) dalam Panduan Pengguna IAM.

[Grup IAM](#) adalah identitas yang menentukan kumpulan pengguna IAM. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin untuk beberapa pengguna sekaligus. Grup membuat izin lebih mudah dikelola untuk sekelompok besar pengguna. Misalnya, Anda dapat memiliki grup yang bernama IAMAdmins dan memberikan izin kepada grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk

mempelajari selengkapnya, silakan lihat [Kapan harus membuat pengguna IAM \(bukan peran\)](#) dalam Panduan Pengguna IAM.

## Peran IAM

[Peran IAM](#) adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Anda dapat mengambil peran IAM untuk sementara AWS Management Console dengan [beralih peran](#). Anda dapat mengambil peran dengan memanggil operasi AWS CLI atau AWS API atau dengan menggunakan URL kustom. Untuk informasi selengkapnya tentang cara menggunakan peran, lihat [Menggunakan peran IAM](#) dalam Panduan Pengguna IAM.

Peran IAM dengan kredensial sementara berguna dalam situasi berikut:

- Akses pengguna gabungan – Untuk menetapkan izin ke sebuah identitas gabungan, Anda dapat membuat peran dan menentukan izin untuk peran tersebut. Saat identitas terfederasi mengautentikasi, identitas tersebut akan dikaitkan dengan peran dan diberi izin yang ditentukan oleh peran tersebut. Untuk informasi tentang peran-peran untuk federasi, lihat [Membuat peran untuk Penyedia Identitas pihak ketiga](#) dalam Panduan Pengguna IAM. Jika Anda menggunakan Pusat Identitas IAM, Anda perlu mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM mengorelasikan izin yang diatur ke peran dalam IAM. Untuk informasi tentang rangkaian izin, lihat [Rangkaian 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 (pengguna utama tepercaya) dengan akun berbeda untuk mengakses sumber daya yang ada di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, dengan beberapa Layanan AWS, Anda dapat melampirkan kebijakan langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy). Untuk mempelajari perbedaan antara kebijakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Bagaimana peran IAM berbeda dari kebijakan berbasis sumber daya](#) dalam Panduan Pengguna IAM.
- Akses lintas layanan — Beberapa Layanan AWS menggunakan fitur lain Layanan AWS. Sebagai contoh, ketika Anda melakukan panggilan dalam suatu layanan, biasanya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Sebuah layanan mungkin melakukannya menggunakan izin prinsipal yang memanggil, menggunakan peran layanan, atau peran terkait layanan.

- Sesi akses teruskan (FAS) — Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Saat Anda menggunakan beberapa layanan, Anda mungkin melakukan tindakan yang kemudian memulai tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan saat membuat permintaan FAS, lihat [Teruskan sesi akses](#).
- Peran layanan – Peran layanan adalah [peran IAM](#) yang diambil oleh layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, memodifikasi, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat [Membuat peran untuk mendelegasikan izin ke Layanan AWS](#) dalam Panduan pengguna IAM.
- Peran terkait layanan — Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan dapat menggunakan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.
- Aplikasi yang berjalan di Amazon EC2 — Anda dapat menggunakan peran IAM untuk mengelola kredensi sementara untuk aplikasi yang berjalan pada instans EC2 dan membuat atau permintaan API. AWS CLI AWS Cara ini lebih dianjurkan daripada menyimpan kunci akses dalam instans EC2. Untuk menetapkan AWS peran ke instans EC2 dan membuatnya tersedia untuk semua aplikasinya, Anda membuat profil instance yang dilampirkan ke instance. 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 di 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 akses menggunakan kebijakan

Anda mengontrol akses AWS dengan membuat kebijakan dan melampirkannya ke AWS identitas atau sumber daya. Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika prinsipal (pengguna, pengguna root, atau sesi peran) membuat permintaan. Izin dalam kebijakan dapat menentukan permintaan yang diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk

informasi selengkapnya tentang struktur dan konten dokumen kebijakan JSON, lihat [Gambaran umum kebijakan JSON](#) dalam Panduan Pengguna IAM.

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, pengguna utama manakah yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan pada 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 operasi. Sebagai contoh, anggap saja Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan tersebut bisa mendapatkan informasi peran dari AWS Management Console, API AWS CLI, atau AWS API.

## Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan pengguna dan peran, di sumber daya mana, dan dengan ketentuan 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 dilampirkan ke beberapa pengguna, grup, dan peran dalam akun AWS. Kebijakan AWS terkelola mencakup kebijakan terkelola dan kebijakan yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan 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 berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya yang dilampiri kebijakan tersebut, kebijakan ini menentukan jenis tindakan yang dapat dilakukan oleh pengguna utama tertentu di sumber daya tersebut dan apa ketentuannya.

Anda harus [menentukan pengguna utama](#) dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan AWS terkelola dari IAM dalam kebijakan berbasis sumber daya.

## Daftar kontrol akses (ACL)

Daftar kontrol akses (ACL) mengendalikan pengguna utama mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL sama dengan kebijakan berbasis sumber daya, meskipun tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung ACL. Untuk mempelajari ACL selengkapnya, silakan lihat [Gambaran umum daftar kontrol akses \(ACL\)](#) di Panduan Developer Layanan Penyimpanan Ringkas Amazon.

## Tipe kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Tipe-tipe kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda berdasarkan tipe kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan di mana Anda menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas IAM (pengguna atau peran IAM). Anda dapat menetapkan batasan izin untuk suatu entitas. Izin yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas milik entitas dan batasan izinnya. Kebijakan berbasis sumber daya yang menentukan pengguna atau peran dalam bidang `Principal` tidak dibatasi oleh batasan izin. Penolakan secara eksplisit terhadap salah satu kebijakan ini akan mengesampingkan izin tersebut. 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 organisasi atau unit organisasi (OU) di. AWS Organizations AWS Organizations adalah layanan untuk mengelompokkan dan mengelola secara terpusat beberapa Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur di sebuah organisasi, maka Anda dapat menerapkan kebijakan kontrol layanan (SCP) ke salah satu atau semua akun Anda. SCP membatasi izin untuk entitas di akun anggota, termasuk masing-masing. Pengguna root akun AWS Untuk informasi selengkapnya tentang Organisasi dan SCP, lihat [Cara kerja SCP](#) dalam Panduan Pengguna AWS Organizations .
- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda teruskan sebagai parameter saat Anda membuat sesi sementara secara terprogram untuk peran atau pengguna gabungan. Izin

sesi yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas pengguna atau peran dan kebijakan sesi. Izin juga bisa datang dari kebijakan berbasis sumber daya. Penolakan eksplisit dalam salah satu kebijakan ini membatalkan izin. Untuk informasi selengkapnya, lihat [Kebijakan sesi](#) dalam Panduan Pengguna IAM.

## Beberapa jenis kebijakan

Ketika beberapa jenis kebijakan berlaku untuk sebuah permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat [Logika evaluasi kebijakan](#) di Panduan Pengguna IAM.

## Bagaimana AWS CodeArtifact bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses CodeArtifact, pelajari fitur IAM yang tersedia untuk digunakan. CodeArtifact

Fitur IAM yang dapat Anda gunakan AWS CodeArtifact

Fitur IAM	CodeArtifact dukungan
<a href="#">Kebijakan berbasis identitas</a>	Ya
<a href="#">Kebijakan berbasis sumber daya</a>	Ya
<a href="#">Tindakan kebijakan</a>	Ya
<a href="#">Sumber daya kebijakan</a>	Ya
<a href="#">kunci-kunci persyaratan kebijakan (spesifik layanan)</a>	Tidak
<a href="#">ACL</a>	Tidak
<a href="#">ABAC (tanda dalam kebijakan)</a>	Parsial
<a href="#">Kredensial sementara</a>	Ya
<a href="#">Izin pengguna utama</a>	Ya
<a href="#">Peran layanan</a>	Tidak

Fitur IAM	CodeArtifact dukungan
<a href="#">Peran terkait layanan</a>	Tidak

Untuk mendapatkan tampilan tingkat tinggi tentang cara CodeArtifact dan AWS layanan lain bekerja dengan sebagian besar fitur IAM, lihat [AWS layanan yang bekerja dengan IAM di Panduan Pengguna IAM](#).

## Kebijakan berbasis identitas untuk CodeArtifact

Mendukung kebijakan berbasis identitas	Ya
----------------------------------------	----

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan pengguna dan peran, di sumber daya mana, dan dengan ketentuan apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat [Membuat kebijakan IAM](#) dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan tindakan dan sumber daya yang diizinkan atau ditolak, serta ketentuan terkait jenis tindakan yang diizinkan atau ditolak. Anda tidak dapat menentukan pengguna utama dalam kebijakan berbasis identitas karena kebijakan ini berlaku untuk pengguna atau peran yang dilampiri kebijakan. Untuk mempelajari semua elemen yang dapat digunakan dalam kebijakan JSON, lihat [Referensi elemen kebijakan JSON IAM](#) dalam Panduan Pengguna IAM.

### Contoh kebijakan berbasis identitas untuk CodeArtifact

Untuk melihat contoh kebijakan CodeArtifact berbasis identitas, lihat. [Contoh kebijakan berbasis identitas untuk AWS CodeArtifact](#)

## Kebijakan berbasis sumber daya dalam CodeArtifact

Mendukung kebijakan berbasis sumber daya	Ya
------------------------------------------	----

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan



kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya yang dilampiri kebijakan tersebut, kebijakan ini menentukan jenis tindakan yang dapat dilakukan oleh pengguna utama tertentu di sumber daya tersebut dan apa ketentuannya. Anda harus [menentukan pengguna utama](#) dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan seluruh akun atau entitas IAM di akun lain sebagai pengguna utama dalam kebijakan berbasis sumber daya. Menambahkan pengguna utama lintas akun ke kebijakan berbasis sumber daya bagian dari membangun hubungan kepercayaan. Ketika prinsipal dan sumber daya berbeda Akun AWS, administrator IAM di akun tepercaya juga harus memberikan izin entitas utama (pengguna atau peran) untuk mengakses sumber daya. Izin diberikan dengan melampirkan kebijakan berbasis identitas ke entitas tersebut. Namun, jika kebijakan berbasis sumber daya memberikan akses kepada pengguna utama dalam akun yang sama, kebijakan berbasis identitas lainnya tidak diperlukan. Untuk informasi selengkapnya, lihat [Perbedaan peran IAM dengan kebijakan berbasis sumber daya](#) di Panduan Pengguna IAM.

## Tindakan kebijakan untuk CodeArtifact

Mendukung tindakan kebijakan	Ya
------------------------------	----

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, pengguna utama mana yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama dengan operasi AWS API terkait. Ada beberapa pengecualian, misalnya tindakan hanya izin yang tidak memiliki operasi API yang cocok. Ada juga beberapa operasi yang memerlukan beberapa tindakan dalam suatu kebijakan. Tindakan tambahan ini disebut tindakan dependen.

Menyertakan tindakan dalam suatu kebijakan untuk memberikan izin melakukan operasi terkait.

Untuk melihat daftar CodeArtifact tindakan, lihat [Tindakan yang ditentukan oleh AWS CodeArtifact](#) dalam Referensi Otorisasi Layanan.



Tindakan kebijakan CodeArtifact menggunakan awalan berikut sebelum tindakan:

```
codeartifact
```

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan-tindakan tersebut dengan koma.

```
"Action": [  
  "codeartifact:action1",  
  "codeartifact:action2"  
]
```

Anda juga dapat menentukan beberapa tindakan menggunakan wildcard (\*). Sebagai contoh, untuk menentukan semua tindakan yang dimulai dengan kata `Describe`, sertakan tindakan berikut:

```
"Action": "codeartifact:Describe*"
```

Untuk melihat contoh kebijakan CodeArtifact berbasis identitas, lihat. [Contoh kebijakan berbasis identitas untuk AWS CodeArtifact](#)

## Sumber daya kebijakan untuk CodeArtifact

Mendukung sumber daya kebijakan	Ya
---------------------------------	----

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, pengguna utama mana yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek atau beberapa objek yang menjadi target penerapan tindakan. Pernyataan harus menyertakan elemen `Resource` atau `NotResource`. Praktik terbaiknya, tentukan sumber daya menggunakan [Amazon Resource Name \(ARN\)](#). Anda dapat melakukan ini untuk tindakan yang mendukung jenis sumber daya tertentu, yang dikenal sebagai izin tingkat sumber daya.

Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, misalnya operasi pencantuman, gunakan wildcard (\*) untuk mengindikasikan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

```
"Resource": "*"
```

Untuk melihat daftar jenis CodeArtifact sumber daya dan ARNnya, lihat [Sumber daya yang ditentukan oleh AWS CodeArtifact](#) dalam Referensi Otorisasi Layanan. Untuk mempelajari tindakan mana yang dapat Anda tentukan ARN dari setiap sumber daya, lihat [Tindakan yang ditentukan oleh](#). AWS CodeArtifact Untuk melihat contoh menentukan ARN CodeArtifact sumber daya dalam kebijakan, lihat [AWSCodeArtifactsumber daya dan operasi](#)

## Kunci kondisi kebijakan untuk CodeArtifact

Mendukung kunci kondisi kebijakan spesifik layanan	Tidak
----------------------------------------------------	-------

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, pengguna utama mana yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Elemen `Condition` (atau blok `Condition`) memungkinkan Anda menentukan kondisi di mana suatu pernyataan akan diterapkan. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi kondisional yang menggunakan [operator kondisi](#), misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta.

Jika Anda menentukan beberapa elemen `Condition` dalam satu pernyataan, atau beberapa kunci dalam satu elemen `Condition`, AWS akan mengevaluasinya dengan menggunakan operasi AND logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS mengevaluasi kondisi menggunakan OR operasi logis. Semua kondisi harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan kondisi. Misalnya, Anda dapat memberikan izin kepada pengguna IAM untuk mengakses sumber daya hanya jika izin tersebut mempunyai tag yang sesuai dengan nama pengguna IAM mereka. Untuk informasi selengkapnya, lihat [Elemen kebijakan IAM: variabel dan tag](#) di Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi khusus layanan. Untuk melihat semua kunci kondisi AWS global, lihat [kunci konteks kondisi AWS global](#) di Panduan Pengguna IAM.

### Note

AWS CodeArtifact tidak mendukung Kunci Konteks Kondisi AWS Global berikut:

- [Perujuk](#)
- [UserAgent](#)

Untuk melihat daftar kunci CodeArtifact kondisi, lihat [Kunci kondisi untuk AWS CodeArtifact](#) dalam Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat Anda gunakan kunci kondisi, lihat [Tindakan yang ditentukan oleh AWS CodeArtifact](#).

Untuk melihat contoh kebijakan CodeArtifact berbasis identitas, lihat. [Contoh kebijakan berbasis identitas untuk AWS CodeArtifact](#)

## ACL di CodeArtifact

Mendukung ACL

Tidak

Daftar kontrol akses (ACL) mengontrol pengguna utama (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL sama dengan kebijakan berbasis sumber daya, meskipun tidak menggunakan format dokumen kebijakan JSON.

## ABAC dengan CodeArtifact

Mendukung ABAC (tanda dalam kebijakan)

Parsial

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang menentukan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke entitas IAM (pengguna atau peran) dan ke banyak AWS sumber daya. Pemberian tanda ke entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian, rancanglah kebijakan ABAC untuk mengizinkan operasi saat 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 di mana pengelolaan kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tag, berikan informasi tentang tag di [elemen kondisi](#) 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 hanya untuk beberapa jenis sumber daya, nilainya adalah Parsial.

Untuk informasi selengkapnya tentang ABAC, lihat [Apa itu ABAC?](#) di Panduan Pengguna IAM. Untuk melihat tutorial terkait langkah-langkah penyiapan ABAC, lihat [Menggunakan kontrol akses berbasis atribut \(ABAC\)](#) di Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang penandaan CodeArtifact sumber daya, termasuk contoh kebijakan berbasis identitas untuk membatasi akses ke sumber daya berdasarkan tag pada sumber daya tersebut, lihat [Menggunakan tanda untuk mengontrol akses berbasis CodeArtifact peran](#)

## Menggunakan kredensial sementara dengan CodeArtifact

Mendukung kredensial sementara	Ya
--------------------------------	----

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensial sementara. Untuk informasi tambahan, termasuk yang Layanan AWS bekerja dengan kredensi sementara, lihat [Layanan AWS yang bekerja dengan IAM di Panduan Pengguna IAM](#).

Anda menggunakan kredensi sementara jika Anda masuk AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Misalnya, ketika Anda mengakses AWS menggunakan tautan masuk tunggal (SSO) perusahaan Anda, proses tersebut secara otomatis membuat kredensial sementara. Anda juga akan membuat kredensial sementara secara otomatis saat masuk ke konsol sebagai pengguna dan kemudian beralih peran. Untuk informasi selengkapnya tentang cara beralih peran, lihat [Beralih 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 merekomendasikan agar Anda secara dinamis menghasilkan kredensi sementara alih-alih menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, lihat [Kredensial keamanan sementara di IAM](#).

## Izin utama lintas layanan untuk CodeArtifact

Mendukung sesi akses maju (FAS)	Ya
---------------------------------	----

Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Saat Anda menggunakan beberapa layanan, Anda mungkin melakukan tindakan yang kemudian memulai tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan saat membuat permintaan FAS, lihat [Teruskan sesi akses](#).

Ada dua tindakan CodeArtifact API yang mengharuskan prinsipal pemanggil memiliki izin di layanan lain:

1. `GetAuthorizationToken` membutuhkan `sts:GetServiceBearerToken` bersama dengan `codeartifact:GetAuthorizationToken`.
2. `CreateDomain`, saat menyediakan kunci enkripsi non-default, membutuhkan keduanya `kms:DescribeKey` dan `kms:CreateGrant` pada kunci KMS bersama dengan `codeartifact:CreateDomain`

Untuk informasi selengkapnya tentang izin dan sumber daya yang diperlukan untuk tindakan CodeArtifact, lihat [AWS CodeArtifact referensi izin](#).

## Peran layanan untuk CodeArtifact

Mendukung peran layanan

Tidak

Peran layanan adalah sebuah [peran IAM](#) yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat [Membuat peran untuk mendelegasikan izin ke Layanan AWS](#) dalam Panduan pengguna IAM.

### Warning

Mengubah izin untuk peran layanan dapat merusak CodeArtifact fungsionalitas. Edit peran layanan hanya jika CodeArtifact memberikan panduan untuk melakukannya.

## Peran terkait layanan untuk CodeArtifact

Mendukung peran terkait layanan

Tidak

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan dapat menggunakan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Untuk detail tentang pembuatan atau pengelolaan peran terkait layanan, lihat [Layanan AWS yang berfungsi dengan IAM](#). Temukan sebuah layanan dalam tabel yang memiliki Yes di kolom Peran terkait layanan. Pilih tautan Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

## Contoh kebijakan berbasis identitas untuk AWS CodeArtifact

Secara default, pengguna dan peran tidak memiliki izin untuk membuat atau memodifikasi CodeArtifact sumber daya. Mereka juga tidak dapat melakukan tugas dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS API. Untuk memberikan izin kepada pengguna untuk melakukan tindakan pada sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat menjalankan peran.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM menggunakan contoh dokumen kebijakan JSON ini, lihat [Membuat kebijakan IAM](#) dalam Panduan Pengguna IAM.

Untuk detail tentang tindakan dan jenis sumber daya yang ditentukan oleh CodeArtifact, termasuk format ARN untuk setiap jenis sumber daya, lihat [Kunci tindakan, sumber daya, dan kondisi AWS CodeArtifact di Referensi Otorisasi Layanan](#).

### Topik

- [Praktik terbaik kebijakan](#)
- [Menggunakan konsol CodeArtifact](#)
- [Kebijakan AWS yang dikelola \(telah ditentukan sebelumnya\) untuk AWS CodeArtifact](#)
- [Memungkinkan pengguna untuk melihat izin mereka sendiri](#)
- [Memungkinkan pengguna mendapatkan informasi tentang repositori dan domain](#)
- [Memungkinkan pengguna mendapatkan informasi tentang domain tertentu](#)

- [Memungkinkan pengguna mendapatkan informasi tentang repositori tertentu](#)
- [Membatasi durasi token otorisasi](#)

## Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus CodeArtifact sumber daya di akun Anda. Tindakan ini dikenai biaya untuk Akun AWS Anda. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit — Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola AWS pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat [kebijakan yang dikelola AWS](#) atau [kebijakan yang dikelola AWS untuk fungsi pekerjaan](#) 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 melakukan ini dengan menentukan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, juga dikenal sebagai izin hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk menerapkan izin, lihat [Kebijakan dan izin di IAM](#) di Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan jika digunakan melalui yang spesifik Layanan AWS, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat [Elemen kebijakan JSON IAM: Syarat](#) di Panduan Pengguna IAM.
- Menggunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda guna memastikan izin yang aman dan berfungsi – 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 Analizer Akses IAM](#) di Panduan Pengguna IAM.

- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Anda, Akun AWS aktifkan MFA untuk keamanan tambahan. Untuk mewajibkan MFA saat operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat [Mengonfigurasi akses API yang dilindungi MFA](#) di Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat [Praktik terbaik keamanan di IAM](#) di Panduan Pengguna IAM.

## Menggunakan konsol CodeArtifact

Untuk mengakses AWS CodeArtifact konsol, Anda harus memiliki set izin minimum. Izin ini harus memungkinkan Anda untuk membuat daftar dan melihat detail tentang CodeArtifact sumber daya di Anda Akun AWS. Jika Anda membuat kebijakan berbasis identitas yang lebih ketat daripada izin minimum yang diperlukan, konsol tidak akan berfungsi sebagaimana mestinya untuk entitas (pengguna atau peran) dengan kebijakan tersebut.

Anda tidak perlu mengizinkan izin konsol minimum untuk pengguna yang melakukan panggilan hanya ke AWS CLI atau AWS API. Sebaliknya, izinkan akses hanya ke tindakan yang cocok dengan operasi API yang coba dilakukan.

Untuk memastikan bahwa pengguna dan peran masih dapat menggunakan CodeArtifact konsol, lampirkan juga kebijakan `AWSCodeArtifactAdminAccess` atau `AWSCodeArtifactReadOnlyAccess` AWS terkelola ke entitas. Untuk informasi selengkapnya, lihat [Menambahkan izin ke pengguna](#) di Panduan Pengguna IAM.

## Kebijakan AWS yang dikelola (telah ditentukan sebelumnya) untuk AWS CodeArtifact

AWS mengatasi banyak kasus penggunaan umum dengan menyediakan kebijakan IAM mandiri yang dibuat dan dikelola oleh AWS. Kebijakan AWS terkelola ini memberikan izin yang diperlukan untuk kasus penggunaan umum sehingga Anda dapat menghindari keharusan menyelidiki izin apa yang diperlukan. Untuk informasi selengkapnya, lihat [Kebijakan yang Dikelola AWS](#) dalam Panduan Pengguna IAM.

Kebijakan AWS terkelola berikut, yang dapat Anda lampirkan ke pengguna di akun Anda, khusus untuk AWS CodeArtifact.

- `AWSCodeArtifactAdminAccess`— Menyediakan akses penuh untuk CodeArtifact menyertakan izin untuk mengelola domain. CodeArtifact



```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "codeartifact:*"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sts:GetServiceBearerToken",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "sts:AWSServiceName": "codeartifact.amazonaws.com"
        }
      }
    }
  ]
}
```

- **AWSCodeArtifactReadOnlyAccess**— Menyediakan akses hanya-baca ke CodeArtifact

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "codeartifact:Describe*",
        "codeartifact:Get*",
        "codeartifact:List*",
        "codeartifact:ReadFromRepository"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sts:GetServiceBearerToken",
      "Resource": "*"
    }
  ]
}
```

```
    "Condition": {
      "StringEquals": {
        "sts:AWSServiceName": "codeartifact.amazonaws.com"
      }
    }
  ]
}
```

Untuk membuat dan mengelola peran CodeArtifact layanan, Anda juga harus melampirkan kebijakan AWS terkelola bernama `IAMFullAccess`.

Anda juga dapat membuat kebijakan IAM kustom Anda sendiri untuk mengizinkan izin CodeArtifact tindakan dan sumber daya. Anda dapat melampirkan kebijakan-kebijakan kustom ini ke pengguna IAM atau grup yang memerlukan izin-izin tersebut.

## Memungkinkan pengguna untuk 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 ini di konsol atau menggunakan API atau secara terprogram. `AWS CLI` `AWS`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam: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": "*"
  }
]
```

## Memungkinkan pengguna mendapatkan informasi tentang repositori dan domain

Kebijakan berikut memungkinkan pengguna atau peran IAM untuk membuat daftar dan mendeskripsikan semua jenis CodeArtifact sumber daya, termasuk domain, repositori, paket, dan aset. Kebijakan ini juga mencakup `codeartifact:ReadFromRepository` izin, yang memungkinkan prinsipal untuk mengambil paket dari CodeArtifact repositori. Kebijakan ini tidak mengizinkan membuat domain atau repositori baru dan tidak mengizinkan publikasi paket baru.

Izin `codeartifact:GetAuthorizationToken` dan `sts:GetServiceBearerToken` diperlukan untuk memanggil API `GetAuthorizationToken`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codeartifact:List*",
        "codeartifact:Describe*",
        "codeartifact:Get*",
        "codeartifact:Read*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sts:GetServiceBearerToken",
```

```
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "sts:AWSServiceName": "codeartifact.amazonaws.com"
      }
    }
  }
]
```

## Memungkinkan pengguna mendapatkan informasi tentang domain tertentu

Berikut ini adalah contoh kebijakan izin yang memungkinkan pengguna mencantumkan domain hanya di wilayah us-east-2 untuk akun 123456789012 untuk semua domain yang dimulai dengan nama my.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codeartifact:ListDomains",
      "Resource": "arn:aws:codeartifact:us-east-2:123456789012:domain/my*"
    }
  ]
}
```

## Memungkinkan pengguna mendapatkan informasi tentang repositori tertentu

Berikut ini adalah contoh kebijakan izin yang memungkinkan pengguna mendapatkan informasi tentang repositori yang diakhiri dengan test, termasuk informasi tentang paket di dalamnya. Pengguna tidak akan dapat memublikasikan, membuat, atau menghapus sumber daya.

Izin `codeartifact:GetAuthorizationToken` dan `sts:GetServiceBearerToken` diperlukan untuk memanggil API `GetAuthorizationToken`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```

    "Action": [
      "codeartifact:List*",
      "codeartifact:Describe*",
      "codeartifact:Get*",
      "codeartifact:Read*"
    ],
    "Resource": "arn:aws:codeartifact:*:*:repository/**/*test"
  },
  {
    "Effect": "Allow",
    "Action": [
      "codeartifact:List*",
      "codeartifact:Describe*"
    ],
    "Resource": "arn:aws:codeartifact:*:*:package/**/*test/**/*/*"
  },
  {
    "Effect": "Allow",
    "Action": "sts:GetServiceBearerToken",
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "sts:AWSServiceName": "codeartifact.amazonaws.com"
      }
    }
  },
  {
    "Effect": "Allow",
    "Action": "codeartifact:GetAuthorizationToken",
    "Resource": "*"
  }
]
}

```

## Membatasi durasi token otorisasi

Pengguna harus mengautentikasi CodeArtifact dengan token otorisasi untuk menerbitkan atau menggunakan versi paket. Token otorisasi hanya berlaku selama masa pakai yang dikonfigurasi. Token memiliki masa pakai default 12 jam. Untuk informasi selengkapnya tentang token otorisasi, lihat [AWS CodeArtifact otentikasi dan token](#).

Saat mengambil token, pengguna dapat mengonfigurasi masa pakai token. Nilai yang valid untuk masa pakai token otorisasi adalah 0, dan angka berapa pun antara 900 (15 menit) dan 43200 (12

jam). Nilai 0 akan membuat token dengan durasi yang sama dengan kredensial sementara peran pengguna.

Administrator dapat membatasi nilai yang valid untuk masa pakai otorisasi token dengan kunci syarat `sts:DurationSeconds` dalam kebijakan izin yang dilampirkan ke pengguna atau grup. Jika pengguna mencoba membuat token otorisasi dengan masa pakai di luar nilai yang valid, pembuatan token akan gagal.

Contoh kebijakan berikut membatasi kemungkinan durasi token otorisasi yang dibuat oleh CodeArtifact pengguna.

Contoh kebijakan: Batasi masa pakai token hingga tepat 12 jam (43200 detik)

Dengan kebijakan ini, pengguna hanya akan dapat membuat token otorisasi dengan masa pakai 12 jam.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codeartifact:*",
      "Resource": "*"
    },
    {
      "Sid": "sts",
      "Effect": "Allow",
      "Action": "sts:GetServiceBearerToken",
      "Resource": "*",
      "Condition": {
        "NumericEquals": {
          "sts:DurationSeconds": 43200
        },
        "StringEquals": {
          "sts:AWSServiceName": "codeartifact.amazonaws.com"
        }
      }
    }
  ]
}
```

Contoh kebijakan: Batasi masa pakai token antara 15 menit dan 1 jam, atau sama dengan periode kredensi sementara pengguna

Dengan kebijakan ini, pengguna akan dapat membuat token yang berlaku antara 15 menit dan 1 jam. Pengguna juga akan dapat membuat token yang berlaku selama durasi kredensial sementara peran mereka dengan menentukan 0 untuk `--durationSeconds`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "codeartifact:*",
      "Resource": "*"
    },
    {
      "Sid": "sts",
      "Effect": "Allow",
      "Action": "sts:GetServiceBearerToken",
      "Resource": "*",
      "Condition": {
        "NumericLessThanEquals": {
          "sts:DurationSeconds": 3600
        },
        "StringEquals": {
          "sts:AWSServiceName": "codeartifact.amazonaws.com"
        }
      }
    }
  ]
}
```

## Menggunakan tanda untuk mengontrol akses berbasis CodeArtifact peran

Syarat dalam pernyataan kebijakan pengguna IAM adalah bagian dari sintaks yang Anda gunakan untuk menentukan izin sumber daya yang diperlukan oleh tindakan. CodeArtifact Menggunakan tanda dalam kondisi adalah salah satu cara untuk mengontrol akses ke sumber daya dan permintaan. Untuk informasi tentang menandai CodeArtifact sumber daya, lihat [Penandaan pada sumber daya](#). Topik ini membahas kontrol akses berbasis tanda.

Saat merancang kebijakan IAM, Anda mungkin menetapkan izin terperinci dengan memberikan akses ke sumber daya tertentu. Saat jumlah sumber daya yang Anda kelola bertambah, tugas ini menjadi lebih sulit. Menandai sumber daya dan menggunakan tanda dalam kondisi pernyataan kebijakan dapat mempermudah tugas ini. Anda memberikan akses secara massal ke sumber daya dengan tag tertentu. Kemudian Anda menerapkan tag ini berulang kali ke sumber daya yang relevan, selama pembuatan atau yang lebih baru.

Tag dapat dilampirkan ke sumber daya atau diteruskan atas permintaan ke layanan yang mendukung penandaan. Di CodeArtifact, sumber daya dapat memiliki tag, dan beberapa tindakan dapat mencakup tag. Saat membuat kebijakan IAM, Anda dapat menggunakan kunci kondisi tag untuk mengontrol:

- Pengguna yang dapat melakukan tindakan pada sumber daya domain atau repositori, berdasarkan tanda yang telah dimiliki.
- Tanda apa yang dapat diteruskan dalam permintaan tindakan.
- Apakah kunci tag tertentu dapat digunakan dalam permintaan.

Untuk sintaks dan semantik kunci syarat tanda yang lengkap, lihat [Controlling Access Using Tags](#) dalam Panduan Pengguna IAM.

#### Important

Saat menggunakan tag pada sumber daya untuk membatasi tindakan, tag harus berada di sumber daya tempat tindakan beroperasi. Misalnya, untuk menolak `DescribeRepository` izin dengan tag, tag harus ada di setiap repositori dan bukan domain. Lihat [AWSCodeArtifactreferensi izin](#) daftar tindakan di CodeArtifact dan sumber daya mana mereka beroperasi.

## Contoh kontrol akses berbasis peran

Contoh berikut ini menunjukkan cara menentukan syarat tanda dalam kebijakan untuk CodeArtifact pengguna.

Example 1: Batasi tindakan berdasarkan tanda dalam permintaan

Kebijakan pengguna `AWSCodeArtifactAdminAccess` terkelola memberi pengguna izin tak terbatas untuk melakukan CodeArtifact tindakan pada sumber daya apa pun.



Kebijakan berikut membatasi kekuatan ini dan menolak izin pengguna yang tidak sah untuk membuat repositori kecuali permintaan berisi tanda tertentu. Untuk melakukan itu, kebijakan menolak tindakan `CreateRepository` jika permintaan tidak menentukan tanda bernama `costcenter` dengan salah satu nilai 1 atau 2. Administrator pelanggan harus melampirkan kebijakan IAM ini kepada pengguna IAM yang tidak sah, selain kebijakan pengguna terkelola.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "codeartifact:CreateRepository",
      "Resource": "*",
      "Condition": {
        "Null": {
          "aws:RequestTag/costcenter": "true"
        }
      }
    },
    {
      "Effect": "Deny",
      "Action": "codeartifact:CreateRepository",
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringNotEquals": {
          "aws:RequestTag/costcenter": [
            "1",
            "2"
          ]
        }
      }
    }
  ]
}
```

#### Example 2: Batasi tindakan berdasarkan tanda sumber daya

Kebijakan pengguna `AWSCodeArtifactAdminAccess` terkelola memberi pengguna izin tak terbatas untuk melakukan CodeArtifact tindakan pada sumber daya apa pun.

Kebijakan berikut membatasi kekuatan ini dan menolak izin pengguna yang tidak sah untuk melakukan tindakan pada repositori dalam domain tertentu. Untuk melakukan itu, kebijakan menolak

beberapa tindakan jika sumber daya memiliki tanda bernama Key1 dengan salah satu nilai Value1 atau Value2. (Kunci syarat `aws:ResourceTag` digunakan untuk mengontrol akses ke sumber daya berdasarkan tanda pada sumber daya tersebut.) Administrator pelanggan harus melampirkan kebijakan IAM ini kepada pengguna IAM yang tidak sah, selain kebijakan pengguna terkelola.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "codeartifact:TagResource",
        "codeartifact:UntagResource",
        "codeartifact:DescribeDomain",
        "codeartifact:DescribeRepository",
        "codeartifact:PutDomainPermissionsPolicy",
        "codeartifact:PutRepositoryPermissionsPolicy",
        "codeartifact:ListRepositoriesInDomain",
        "codeartifact:UpdateRepository",
        "codeartifact:ReadFromRepository",
        "codeartifact:ListPackages",
        "codeartifact:ListTagsForResource"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Key1": ["Value1", "Value2"]
        }
      }
    }
  ]
}
```

### Example 3: Izinkan tindakan berdasarkan tanda sumber daya

Kebijakan berikut memberikan pengguna izin untuk melakukan tindakan pada, dan mendapatkan informasi tentang, repositori dan paket. CodeArtifact

Untuk melakukan itu, kebijakan memungkinkan tindakan tertentu jika repositori memiliki tanda bernama Key1 dengan nilai Value1. (Kunci syarat `aws:RequestTag` digunakan untuk mengontrol tanda yang dapat diteruskan dalam permintaan IAM.) Syarat `aws:TagKeys` memastikan kunci tanda peka huruf besar dan kecil. Kebijakan ini berguna bagi pengguna IAM yang tidak memiliki kebijakan

pengguna `AWSCodeArtifactAdminAccess` terkelola terlampir. Kebijakan terkelola memberi pengguna izin tak terbatas untuk melakukan CodeArtifact tindakan terhadap sumber daya apa pun.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codeartifact:UpdateRepository",
        "codeartifact:DeleteRepository",
        "codeartifact:ListPackages"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Key1": "Value1"
        }
      }
    }
  ]
}
```

#### Example 4: Izinkan tindakan berdasarkan tanda dalam permintaan

Kebijakan berikut memberikan pengguna izin untuk membuat repositori di domain tertentu di CodeArtifact

Untuk melakukan itu, kebijakan memungkinkan tindakan `CreateRepository` dan `TagResource` jika API membuat sumber daya dalam permintaan menentukan tanda bernama `Key1` dengan nilai `Value1`. (Kunci syarat `aws:RequestTag` digunakan untuk mengontrol tanda yang dapat diteruskan dalam permintaan IAM.) Syarat `aws:TagKeys` memastikan kunci tanda peka huruf besar dan kecil. Kebijakan ini berguna bagi pengguna IAM yang tidak memiliki kebijakan pengguna `AWSCodeArtifactAdminAccess` terkelola terlampir. Kebijakan terkelola memberi pengguna izin tak terbatas untuk melakukan CodeArtifact tindakan terhadap sumber daya apa pun.

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

```

    "codeartifact:CreateRepository",
    "codeartifact:TagResource"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "aws:RequestTag/Key1": "Value1"
    }
  }
}
]
}

```

## AWSCodeArtifactreferensi izin

### AWSCodeArtifact sumber daya dan operasi

Dalam AWSCodeArtifact, sumber daya utama adalah domain. Dalam sebuah kebijakan, Anda menggunakan Amazon Resource Name (ARN) untuk mengidentifikasi sumber daya tempat kebijakan berlaku. Repositori juga sumber daya dan memiliki ARN yang terkait. Untuk informasi selengkapnya, lihat [Amazon Resource Names \(ARN\)](#) di Referensi Umum Amazon Web Services

Jenis sumber daya	Format ARN
Domain	<code>arn:aws:codeartifact: <i>region-ID</i> :<i>account-ID</i> :domain/<i>my_domain</i></code>
Repositori	<code>arn:aws:codeartifact: <i>region-ID</i> :<i>account-ID</i> :repository/ <i>my_domain</i> /<i>my_repo</i></code>
Paket dengan namespace	<code>arn:aws:codeartifact: <i>region-ID</i> :<i>account-ID</i> :package/ <i>my_domain</i> /<i>my_repo</i>/<i>package-format</i> /<i>namespace</i> /<i>package-name</i></code>
Paket tanpa namespace	<code>arn:aws:codeartifact: <i>region-ID</i> :<i>account-ID</i> :package/ <i>my_domain</i> /<i>my_repo</i>/<i>package-format</i> //<i>package-name</i></code>
Semua sumber daya CodeArtifact	<code>arn:aws:codeartifact:*</code>

Jenis sumber daya	Format ARN
Semua CodeArtifact sumber daya yang dimiliki oleh akun yang ditentukan di Wilayah AWS yang ditentukan	<code>arn:aws:codeartifact: <i>region-ID</i> :<i>account-ID</i> :*</code>

ARN sumber daya yang Anda tentukan bergantung pada tindakan atau tindakan yang Anda inginkan untuk mengontrol akses.

Anda dapat mengindikasikan domain tertentu (*myDomain*) dalam pernyataan Anda dengan menggunakan ARN seperti berikut ini.

```
"Resource": "arn:aws:codeartifact:us-east-2:123456789012:domain/myDomain"
```

Anda dapat mengindikasikan repositori tertentu (*MyRepo*) dalam pernyataan Anda dengan menggunakan ARN seperti berikut ini.

```
"Resource": "arn:aws:codeartifact:us-east-2:123456789012:domain/myDomain/myRepo"
```

Untuk menentukan beberapa sumber daya dalam satu pernyataan, pisahkan ARN dengan koma. Pernyataan berikut berlaku untuk semua paket dan repositori dalam domain tertentu.

```
"Resource": [  
  "arn:aws:codeartifact:us-east-2:123456789012:domain/myDomain",  
  "arn:aws:codeartifact:us-east-2:123456789012:repository/myDomain/*",  
  "arn:aws:codeartifact:us-east-2:123456789012:package/myDomain/*"  
]
```

### Note

Banyak AWS layanan memperlakukan tanda titik dua (:) atau garis miring (/) sebagai karakter yang sama di ARN. Namun, CodeArtifact menggunakan sumber daya yang sama persis dalam pola dan aturan sumber daya. Pastikan untuk menggunakan karakter yang benar saat membuat pola peristiwa sehingga cocok dengan sintaks ARN di sumber daya.

## AWSCodeArtifactOperasi dan izin API

Anda dapat menggunakan tabel berikut sebagai referensi ketika Anda menyiapkan kontrol akses dan menulis kebijakan izin yang dapat Anda lampirkan ke identitas IAM (kebijakan berbasis identitas).

Anda dapat menggunakan kunci ketentuan AWS seluruh di AWS CodeArtifact kebijakan Anda untuk menyatakan ketentuan. Untuk daftarnya, lihat [IAM JSON Policy Elements Reference](#) dalam Panduan Pengguna IAM.

Anda menentukan tindakan di bidang `Action` kebijakan. Untuk menentukan tindakan, gunakan prefiks `codeartifact:` diikuti dengan nama operasi API (misalnya `codeartifact:CreateDomain` dan `codeartifact:AssociateExternalConnection`). Untuk menetapkan beberapa tindakan dalam satu pernyataan, pisahkan dengan koma (misalnya, `"Action": [ "codeartifact:CreateDomain", "codeartifact:AssociateExternalConnection" ]`).

### Menggunakan karakter wildcard

Anda menentukan ARN, dengan atau tanpa karakter wildcard (\*), sebagai nilai sumber daya dalam bidang `Resource` kebijakan. Anda bisa menggunakan wildcard untuk menentukan beberapa tindakan atau sumber daya. Misalnya, `codeartifact:*` menentukan semua CodeArtifact tindakan dan `codeartifact:Describe*` menentukan semua CodeArtifact tindakan yang dimulai dengan kata. `Describe`

## Memecahkan masalah AWS CodeArtifact identitas dan akses

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan CodeArtifact dan IAM.

### Topik

- [Saya tidak berwenang untuk melakukan tindakan di CodeArtifact](#)
- [Saya ingin mengizinkan orang di luar saya Akun AWS untuk mengakses CodeArtifact sumber daya saya](#)

### Saya tidak berwenang untuk melakukan tindakan di CodeArtifact

Jika Anda menerima pesan kesalahan bahwa Anda tidak memiliki otorisasi untuk melakukan tindakan, kebijakan Anda harus diperbarui agar Anda dapat melakukan tindakan tersebut.

Contoh kesalahan berikut terjadi ketika pengguna IAM `mateojackson` mencoba menggunakan konsol untuk melihat detail tentang suatu sumber daya `my-example-widget` rekaan, tetapi tidak memiliki izin `codeartifact:GetWidget` rekaan.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
codeartifact:GetWidget on resource: my-example-widget
```

Dalam hal ini, kebijakan untuk pengguna `mateojackson` harus diperbarui untuk mengizinkan akses ke sumber daya `my-example-widget` dengan menggunakan tindakan `codeartifact:GetWidget`.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya ingin mengizinkan orang di luar saya Akun AWS untuk mengakses CodeArtifact sumber daya saya

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau pengguna di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa saja yang dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis sumber daya atau daftar kontrol akses (ACL), Anda dapat menggunakan kebijakan tersebut untuk memberi pengguna akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa hal berikut:

- Untuk mempelajari apakah CodeArtifact mendukung fitur-fitur ini, lihat [Bagaimana AWS CodeArtifact bekerja dengan IAM](#).
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda di seluruh sumber daya Akun AWS yang Anda miliki, lihat [Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki](#) di Panduan Pengguna IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda kepada pihak ketiga Akun AWS, lihat [Menyediakan akses yang Akun AWS dimiliki oleh pihak ketiga](#) dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat [Memberikan akses kepada pengguna eksternal yang sah \(federasi identitas\)](#) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara penggunaan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Perbedaan antara peran IAM dan kebijakan berbasis sumber daya](#) di Panduan Pengguna IAM.

# Bekerja dengan Amazon VPC endpoint

Anda dapat mengonfigurasi CodeArtifact untuk menggunakan antarmuka virtual private cloud (VPC) endpoint untuk meningkatkan keamanan VPC Anda.

Penggunaan titik akhir VPC AWS PrivateLink, layanan yang memungkinkan Anda mengakses CodeArtifact API melalui alamat IP pribadi. AWS PrivateLink membatasi semua lalu lintas jaringan antara VPC Anda CodeArtifact dan ke jaringan AWS. Jika menggunakan VPC endpoint antarmuka, Anda juga tidak memerlukan gateway internet, perangkat NAT, atau virtual private gateway. Untuk informasi selengkapnya, lihat [VPC Endpoint](#) dalam Panduan Pengguna Amazon Virtual Private Cloud.

## Important

- Titik akhir VPC tidak mendukung permintaan lintas Wilayah.AWS Pastikan Anda membuat titik akhir di AWS Wilayah yang sama tempat Anda berencana untuk mengeluarkan panggilan API. CodeArtifact
- Titik akhir VPC hanya mendukung DNS yang disediakan Amazon melalui Amazon Route 53. Jika Anda ingin menggunakan DNS Anda sendiri, Anda dapat menggunakan penerusan DNS bersyarat. Untuk informasi selengkapnya, lihat [Set Opsi DHCP](#) dalam Panduan Pengguna Amazon Virtual Private Cloud.
- Grup keamanan yang dilampirkan ke VPC endpoint harus mengizinkan koneksi masuk pada port 443 dari subnet privat VPC.

## Topik

- [Buat titik akhir VPC untuk CodeArtifact](#)
- [Buat titik akhir gateway Amazon S3](#)
- [Gunakan CodeArtifact dari VPC](#)
- [Membuat kebijakan titik akhir VPC untuk CodeArtifact](#)



## Buat titik akhir VPC untuk CodeArtifact

Untuk membuat titik akhir virtual private cloud (VPC) CodeArtifact, gunakan perintah Amazon EC2. `create-vpc-endpoint` AWS CLI Untuk informasi selengkapnya, lihat [Interface VPC Endpoints \(AWS PrivateLink\)](#) dalam Panduan Pengguna Amazon Virtual Private Cloud.

Dua titik akhir VPC diperlukan sehingga semua permintaan CodeArtifact berada di jaringan. AWS Titik akhir pertama digunakan untuk memanggil CodeArtifact API (misalnya, `GetAuthorizationToken` dan `CreateRepository`).

```
com.amazonaws.region.codeartifact.api
```

Endpoint kedua digunakan untuk mengakses CodeArtifact repositori menggunakan manajer paket dan alat build (misalnya, `npm` dan `Gradle`).

```
com.amazonaws.region.codeartifact.repositories
```

Perintah berikut membuat titik akhir untuk mengakses CodeArtifact repositori.

```
aws ec2 create-vpc-endpoint --vpc-id vpcid --vpc-endpoint-type Interface \  
--service-name com.amazonaws.region.codeartifact.api --subnet-ids subnetid \  
--security-group-ids groupid --no-private-dns-enabled
```

Perintah berikut membuat titik akhir untuk mengakses manajer paket dan alat pembangun.

```
aws ec2 create-vpc-endpoint --vpc-id vpcid --vpc-endpoint-type Interface \  
--service-name com.amazonaws.region.codeartifact.repositories --subnet-ids subnetid \  
--security-group-ids groupid --private-dns-enabled
```

### Note

Saat membuat titik akhir `codeartifact.repositories`, Anda harus membuat nama host DNS privat menggunakan opsi `--private-dns-enabled`. Namun, karena beberapa nama host DNS privat saat ini tidak didukung untuk titik akhir `codeartifact.api` dan `codeartifact.repositories`, gunakan opsi `--no-private-dns-enabled` untuk `codeartifact.api`. Jika Anda tidak dapat atau tidak ingin membuat nama host DNS pribadi saat membuat `codeartifact.repositories` titik akhir, Anda harus mengikuti langkah konfigurasi tambahan untuk menggunakan pengelola

paket Anda dari VPC CodeArtifact . Untuk informasi selengkapnya, lihat [Gunakan `codeartifact.repositories` titik akhir tanpa DNS pribadi](#).

Setelah membuat titik akhir VPC, Anda mungkin perlu melakukan lebih banyak konfigurasi dengan aturan grup keamanan untuk menggunakan titik akhir. CodeArtifact Untuk informasi selengkapnya tentang grup keamanan di Amazon VPC, lihat Grup [keamanan](#).

Jika Anda mengalami masalah saat terhubung CodeArtifact, Anda dapat menggunakan alat VPC Reachability Analyzer untuk men-debug masalah. Untuk informasi selengkapnya, lihat [Apa itu VPC Reachability Analyzer?](#)

## Buat titik akhir gateway Amazon S3

CodeArtifact menggunakan Amazon Simple Storage Service (Amazon S3) untuk menyimpan aset paket. Untuk menarik paket dari CodeArtifact, Anda harus membuat titik akhir gateway untuk Amazon S3. Saat proses build atau deployment Anda mengunduh paket CodeArtifact, paket harus mengakses CodeArtifact untuk mendapatkan metadata paket dan Amazon S3 untuk mengunduh aset paket (misalnya, file Maven). `.jar`

### Note

Endpoint Amazon S3 tidak diperlukan saat menggunakan format paket Python atau Swift.

Untuk membuat titik akhir gateway Amazon S3 CodeArtifact, gunakan perintah Amazon EC2. `create-vpc-endpoint` AWS CLI Saat membuat titik akhir, Anda harus memilih tabel rute untuk VPC Anda. Untuk informasi selengkapnya, lihat [Gateway VPC Endpoints](#) dalam Panduan Pengguna Amazon Virtual Private Cloud.

Perintah berikut membuat titik akhir Amazon S3.

```
aws ec2 create-vpc-endpoint --vpc-id vpcid --service-name com.amazonaws.region.s3 \  
--route-table-ids routetableid
```

## Izin bucket Amazon S3 minimum untuk AWS CodeArtifact

titik akhir gateway Amazon S3 menggunakan dokumen kebijakan IAM untuk membatasi akses ke layanan. Agar hanya mengizinkan izin bucket Amazon S3 minimum CodeArtifact, batasi akses ke

bucket Amazon S3 yang CodeArtifact digunakan saat Anda membuat dokumen kebijakan IAM untuk titik akhir.

Tabel berikut menjelaskan bucket Amazon S3 yang harus Anda referensikan dalam kebijakan Anda untuk mengizinkan akses CodeArtifact di setiap wilayah.

Wilayah	ARN Bucket Amazon S3
us-east-1	arn:aws:s3:::assets-193858265520-us-east-1
us-east-2	arn:aws:s3:::assets-250872398865-us-east-2
us-west-2	arn:aws:s3:::assets-787052242323-us-west-2
eu-west-1	arn:aws:s3:::assets-438097961670-eu-west-1
eu-west-2	arn:aws:s3:::assets-247805302724-eu-west-2
eu-west-3	arn:aws:s3:::assets-762466490029-eu-west-3
eu-north-1	arn:aws:s3:::assets-611884512288-eu-north-1
eu-south-1	arn:aws:s3:::assets-484130244270-eu-south-1
eu-central-1	arn:aws:s3:::assets-769407342218-eu-central-1
ap-northeast-1	arn:aws:s3:::assets-660291247815-ap-northeast-1
ap-southeast-1	arn:aws:s3:::assets-421485864821-ap-southeast-1
ap-southeast-2	arn:aws:s3:::assets-860415559748-ap-southeast-2
ap-south-1	arn:aws:s3:::assets-681137435769-ap-south-1

Anda dapat menggunakan `aws codeartifact describe-domain` perintah untuk mengambil bucket Amazon S3 yang digunakan oleh CodeArtifact domain.

```
aws codeartifact describe-domain --domain mydomain
```

```
{
  "domain": {
    "name": "mydomain",
    "owner": "111122223333",
    "arn": "arn:aws:codeartifact:us-west-2:111122223333:domain/mydomain",
    "status": "Active",
    "createdTime": 1583075193.861,
    "encryptionKey": "arn:aws:kms:us-west-2:111122223333:key/a73que8sq-ba...",
    "repositoryCount": 13,
    "assetSizeBytes": 513830295,
    "s3BucketArn": "arn:aws:s3:::assets-787052242323-us-west-2"
  }
}
```

## Contoh

Contoh berikut menggambarkan cara menyediakan akses ke bucket Amazon S3 yang diperlukan CodeArtifact untuk operasi di wilayah tersebut. `us-east-1` Untuk wilayah lain, perbarui entri Resource dengan ARN izin yang benar untuk wilayah Anda berdasarkan tabel di atas.

```
{
  "Statement": [
    {
      "Sid": "Access-to-specific-bucket-only",
      "Principal": "*",
      "Action": [
        "s3:GetObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::assets-193858265520-us-east-1/*"]
    }
  ]
}
```

## Gunakan CodeArtifact dari VPC

Untuk memanggil CodeArtifact API menggunakan AWS CLI atau SDK menggunakan Titik Akhir VPC, Anda harus mengganti titik akhir default yang digunakan oleh CodeArtifact Ikuti instruksi [Konfigurasi AWS CLI untuk menggunakan titik `codeartifact.api` akhir](#) untuk mendapatkan nama host VPC Endpoint dan konfigurasi CLI dengannya.

Jika Anda tidak dapat atau tidak ingin mengaktifkan DNS pribadi pada titik akhir `com.amazonaws.region.codeartifact.repositories` VPC yang Anda buat [Buat titik akhir VPC untuk CodeArtifact](#), Anda harus menggunakan konfigurasi yang berbeda untuk titik akhir repositori yang akan digunakan dari VPC. CodeArtifact Ikuti petunjuk [Gunakan `codeartifact.repositories` titik akhir tanpa DNS pribadi](#) untuk mengonfigurasi CodeArtifact jika `com.amazonaws.region.codeartifact.repositories` titik akhir tidak mengaktifkan DNS pribadi.

### Konfigurasi AWS CLI untuk menggunakan titik `codeartifact.api` akhir

Gunakan petunjuk berikut untuk mengganti CodeArtifact nama host default dengan nama host yang digunakan oleh titik akhir `com.amazonaws.region.codeartifact.api` VPC.

1. Jalankan perintah berikut untuk menemukan VPC endpoint yang akan digunakan untuk mengganti nama host.


```
$ aws ec2 describe-vpc-endpoints --filters Name=service-name,Values=com.amazonaws.region.codeartifact.api \
--query 'VpcEndpoints[*].DnsEntries[*].DnsName'
```

Output-nya akan terlihat seperti berikut.

```
[
  [
    "vpce-0743fe535b883ffff-76ddffff.api.codeartifact.us-west-2.vpce.amazonaws.com",
    "vpce-0743fe535b883ffff-76edffff-us-west-2a.api.codeartifact.us-west-2.vpce.amazonaws.com"
  ]
]
```

Dalam contoh ini, Anda dapat menggunakan salah satu nama host untuk mengganti titik akhir `com.amazonaws.region.codeartifact.api`.

2. Jika Anda menggunakan CodeArtifact AWS CLI, gunakan `codeartifact login` perintah untuk mengganti CodeArtifact nama host default dengan titik akhir Amazon VPC dengan meneruskan titik akhir ke parameter. `--endpoint-url` Lihat contoh berikut ini.

 Warning

`loginPerintah` tidak mendukung Maven atau Gradle. Untuk mengonfigurasi manajer paket tersebut, lihat [Menggunakan CodeArtifact dengan Maven](#).

```
aws codeartifact login --tool npm --domain mydomain --domain-owner 111122223333 --  
repository myrepo --endpoint-url VPC_endpoint
```

Ganti *VPC\_Endpoint* dengan *endpoint* Amazon VPC Anda, diawali dengan. `https://`  
Lihat contoh endpoint berikut.

```
https://vpce-0743fe535b883ffff-76ddffff.api.codeartifact.us-  
west-2.vpce.amazonaws.com
```

Jika Anda menggunakan SDK, baca dokumentasi SDK untuk mempelajari cara mengganti nama host. Cara melakukannya bervariasi menurut bahasa yang Anda gunakan.

## Gunakan `codeartifact.repositories` titik akhir tanpa DNS pribadi

Jika Anda tidak dapat atau tidak ingin mengaktifkan DNS pribadi pada titik akhir `com.amazonaws.region.codeartifact.repositories` VPC yang Anda buat [Buat titik akhir VPC untuk CodeArtifact](#), Anda harus mengikuti petunjuk ini untuk mengonfigurasi manajer paket Anda dengan URL yang benar. CodeArtifact

1. Jalankan perintah berikut untuk menemukan VPC endpoint yang akan digunakan untuk mengganti nama host.

```
$ aws ec2 describe-vpc-endpoints --filters Name=service-  
name,Values=com.amazonaws.region.codeartifact.repositories \
```

```
--query 'VpcEndpoints[*].DnsEntries[*].DnsName'
```

Output-nya akan terlihat seperti berikut.

```
[
  [
    "vpce-0743fe535b883ffff-76ddffff.d.codeartifact.us-west-2.vpce.amazonaws.com"
  ]
]
```

2. Perbarui jalur titik akhir VPC untuk menyertakan format paket, nama CodeArtifact domain Anda, dan CodeArtifact nama repositori. Lihat contoh berikut ini.

```
https://vpce-0743fe535b883ffff-76ddffff.d.codeartifact.us-  
west-2.vpce.amazonaws.com/format/d/domain_name-domain_owner/repo_name
```

Ganti bidang berikut dari titik akhir contoh.

- *Format*: Ganti dengan format CodeArtifact paket yang valid, misalnya, npm ataupun pi.
- *domain\_name*: Ganti dengan CodeArtifact domain yang berisi CodeArtifact repositori yang menghosting paket Anda.
- *domain\_owner*: Ganti dengan ID pemilik CodeArtifact domain, misalnya, 111122223333
- *repo\_name*: Ganti dengan CodeArtifact repositori yang menghosting paket Anda.

URL berikut adalah contoh titik akhir repositori npm.

```
https://vpce-0dc4daf7fca331ed6-et36qa1d.d.codeartifact.us-  
west-2.vpce.amazonaws.com/npm/d/domainName-111122223333/repoName
```

3. Konfigurasi pengelola paket Anda untuk menggunakan titik akhir VPC yang diperbarui dari langkah sebelumnya. Anda harus mengkonfigurasi manajer paket tanpa menggunakan CodeArtifact login perintah. Untuk petunjuk konfigurasi untuk setiap format paket, lihat dokumentasi berikut.

- npm: [Mengonfigurasi npm tanpa menggunakan perintah login](#)
- nuget: [Konfigurasi nuget atau dotnet tanpa perintah login](#)
- pip: [Mengonfigurasi pip tanpa perintah login](#)
- benang: [Konfigurasi dan gunakan benang dengan CodeArtifact](#)

- Gradle: [Gunakan CodeArtifact dengan Gradle](#)
- mvn: [Gunakan CodeArtifact dengan mvn](#)

## Membuat kebijakan titik akhir VPC untuk CodeArtifact

Untuk membuat kebijakan titik akhir VPC CodeArtifact, tentukan yang berikut ini:

- Prinsipal yang dapat melakukan tindakan.
- Tindakan yang dapat dilakukan.
- Sumber daya yang dapat memiliki tindakan yang dilakukan pada mereka.

Contoh kebijakan berikut menetapkan bahwa prinsipal di akun 123456789012 dapat memanggil API dan mengambil paket dari repositori. `GetAuthorizationToken` CodeArtifact

```
{
  "Statement": [
    {
      "Action": [
        "codeartifact:GetAuthorizationToken",
        "codeartifact:GetRepositoryEndpoint",
        "codeartifact:ReadFromRepository",
        "sts:GetServiceBearerToken"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      }
    }
  ]
}
```



# Menciptakan CodeArtifact sumber daya dengan AWS CloudFormation

CodeArtifact terintegrasi dengan AWS CloudFormation, layanan yang membantu Anda memodelkan dan mengatur AWS sumber daya Anda sehingga Anda dapat menghabiskan lebih sedikit waktu untuk membuat dan mengelola sumber daya dan infrastruktur Anda. Anda membuat templat yang menjelaskan semua sumber daya AWS yang Anda inginkan (seperti domain atau repositori), dan AWS CloudFormation mengurus penyediaan dan konfigurasi sumber daya tersebut untuk Anda.

Ketika Anda menggunakan AWS CloudFormation, Anda dapat menggunakan kembali template Anda untuk mengatur CodeArtifact sumber daya Anda secara konsisten dan berulang kali. Cukup jelaskan sumber daya Anda sekali dan kemudian sediakan sumber daya yang sama berulang-ulang dalam beberapa akun dan Wilayah AWS.

## CodeArtifact dan AWS CloudFormation template

Untuk menyediakan dan mengonfigurasi sumber daya untuk CodeArtifact dan layanan terkait, Anda harus memahami [AWS CloudFormation templat](#). Templat adalah file teks dengan format JSON atau YAML. Templat ini menjelaskan sumber daya yang ingin Anda sediakan di tumpukan AWS CloudFormation Anda. Jika Anda tidak terbiasa dengan JSON atau YAML, Anda dapat menggunakan AWS CloudFormation Designer untuk membantu Anda memulai dengan templat AWS CloudFormation. Untuk informasi selengkapnya, lihat [Apa itu AWS CloudFormation Designer?](#) dalam AWS CloudFormation User Guide.

CodeArtifact mendukung pembuatan domain dan repositori di AWS CloudFormation. Untuk informasi selengkapnya, termasuk contoh template JSON dan YAMAL untuk domain dan repositori, lihat dan.

[AWS::CodeArtifact::DomainAWS::CodeArtifact::Repository](#)

## Mencegah penghapusan sumber daya CodeArtifact

CodeArtifact repositori berisi dependensi aplikasi kritis yang mungkin tidak mudah dibuat ulang jika hilang. Untuk melindungi CodeArtifact sumber daya dari penghapusan yang tidak disengaja saat mengelola CodeArtifact sumber daya dengan CloudFormation, sertakan `UpdateRetainPolicy` atribut `DeletionPolicy` dan dengan nilai `Retain` pada semua domain dan repositori. Ini akan mencegah penghapusan jika sumber daya dihapus dari template tumpukan, atau seluruh tumpukan

dihapus secara tidak sengaja. Cuplikan YAMAL berikut menunjukkan domain dasar dan repositori dengan atribut ini:

```
Resources:
  MyCodeArtifactDomain:
    Type: 'AWS::CodeArtifact::Domain'
    DeletionPolicy: Retain
    UpdateReplacePolicy: Retain
    Properties:
      DomainName: "my-domain"

  MyCodeArtifactRepository:
    Type: 'AWS::CodeArtifact::Repository'
    DeletionPolicy: Retain
    UpdateReplacePolicy: Retain
    Properties:
      RepositoryName: "my-repo"
      DomainName: !GetAtt MyCodeArtifactDomain.Name
```

Untuk informasi selengkapnya tentang atribut ini, lihat [DeletionPolicy](#) dan [UpdateReplacePolicy](#) di Panduan AWS CloudFormation Pengguna.

## Pelajari selengkapnya tentang AWS CloudFormation

Untuk mempelajari selengkapnya tentang AWS CloudFormation, lihat sumber daya berikut:

- [AWS CloudFormation](#)
- [Panduan Pengguna AWS CloudFormation](#)
- [AWS CloudFormation Panduan Pengguna Antarmuka Baris Perintah](#)

# Pemecahan Masalah AWS CodeArtifact

Informasi berikut dapat membantu Anda memecahkan masalah umum. CodeArtifact

Untuk informasi tentang pemecahan masalah khusus format, lihat topik berikut:

- [Pemecahan masalah Maven](#)
- [Pemecahan masalah cepat](#)

## Saya tidak dapat melihat notifikasi

**Masalah:** Saat Anda berada di konsol Alat Developer dan memilih Notifikasi di bagian Pengaturan, Anda melihat kesalahan izin.

**Kemungkinan perbaikan:** Meskipun notifikasi adalah fitur konsol Alat Pengembang, saat ini CodeArtifact tidak mendukung pemberitahuan. Tak satu pun dari kebijakan terkelola untuk CodeArtifact menyertakan izin yang memungkinkan pengguna untuk melihat atau mengelola notifikasi. Jika Anda menggunakan layanan lain di konsol Alat Developer, dan layanan tersebut mendukung notifikasi, kebijakan terkelola untuk layanan tersebut mencakup izin yang diperlukan untuk melihat dan mengelola notifikasi untuk layanan tersebut.

# Penandaan pada sumber daya

Tag adalah label atribut khusus yang Anda atau AWS tetapkan ke sumber daya AWS. Setiap tanda AWS memiliki dua bagian:

- Sebuah kunci tag (misalnya, `CostCenter`, `Environment`, `Project`, atau `Secret`). Kunci tanda peka terhadap huruf besar dan kecil.
- Bidang opsional yang dikenal sebagai nilai tag (misalnya, `111122223333`, `Production`, atau nama tim). Mengabaikan nilai tag sama dengan menggunakan rangkaian kosong. Seperti kunci tanda, nilai tanda peka huruf besar dan kecil.

Bersama-sama ini dikenal sebagai pasangan nilai-kunci.

Tanda membantu Anda mengidentifikasi dan mengatur sumber daya AWS. Banyak tag memberikan support pada layanan AWS, sehingga Anda dapat menetapkan tag yang sama ke sumber daya dari berbagai layanan untuk menunjukkan bahwa sumber daya tersebut terkait. Contohnya, Anda dapat menetapkan tanda yang sama ke repositori yang Anda tetapkan ke proyek AWS CodeBuild.

Untuk tips dan praktik terbaik dalam menggunakan tag, lihat Whitepaper [Praktik Terbaik untuk Menandai AWS Sumber Daya](#).

Anda dapat menandai jenis sumber daya berikut di CodeArtifact:

- [Menandai repositori di CodeArtifact](#)
- [Menandai domain di CodeArtifact](#)

Anda dapat menggunakan konsol, CodeArtifact API/AWS CLI, atau AWS SDK untuk:

- Menambahkan tanda ke domain atau repositori saat Anda membuatnya\*.
- Menambahkan, mengelola, dan menghapus tanda untuk domain atau repositori.

\* Anda tidak dapat menambahkan tanda ke domain atau repositori saat membuatnya di konsol.

Selain mengidentifikasi, mengatur, dan melacak sumber daya Anda dengan tanda, Anda dapat menggunakan tanda dalam kebijakan IAM untuk membantu mengontrol siapa yang dapat melihat dan berinteraksi dengan sumber daya Anda. Untuk contoh kebijakan akses berbasis tag, lihat [Menggunakan tanda untuk mengontrol akses berbasis CodeArtifact peran](#).

## CodeArtifact alokasi biaya dengan tag

Anda dapat menggunakan tag untuk mengalokasikan penyimpanan dan meminta biaya. CodeArtifact

### Mengalokasikan biaya penyimpanan data di CodeArtifact

Biaya penyimpanan data terkait dengan domain, oleh karena itu untuk mengalokasikan biaya CodeArtifact penyimpanan Anda, Anda dapat menggunakan tag apa pun yang diterapkan ke domain Anda. Untuk informasi tentang menambahkan tag ke domain, lihat [Menandai domain di CodeArtifact](#).

### Mengalokasikan biaya permintaan di CodeArtifact

Sebagian besar penggunaan permintaan terkait dengan repositori, oleh karena itu untuk mengalokasikan biaya CodeArtifact permintaan Anda, Anda dapat menggunakan tag apa pun yang diterapkan ke repositori Anda. Untuk informasi tentang menambahkan tag ke repositori, lihat [Menandai repositori di CodeArtifact](#)

Beberapa jenis permintaan dikaitkan dengan domain daripada repositori, sehingga penggunaan permintaan dan biaya yang terkait dengan permintaan akan dialokasikan ke tag pada domain. Cara terbaik untuk menentukan apakah jenis permintaan dikaitkan dengan domain atau repositori adalah dengan menggunakan [Tindakan yang ditentukan oleh AWS CodeArtifact](#) tabel dalam Referensi Otorisasi Layanan. Temukan jenis permintaan di kolom Tindakan, dan lihat nilai di kolom Jenis sumber daya yang sesuai. Jika jenis sumber daya adalah domain, permintaan jenis itu akan ditagih ke domain. Jika jenis sumber daya adalah repositori atau paket, permintaan jenis itu akan ditagih ke repositori. Beberapa tindakan menunjukkan kedua jenis sumber daya, untuk tindakan tersebut sumber daya yang ditagih bergantung pada nilai apa yang diteruskan dalam permintaan.

## Kuota diAWSCodeArtifact

Tabel berikut menjelaskan kuota sumber daya diCodeArtifact. Untuk melihat kuota sumber daya beserta daftar endpoint layananCodeArtifact, lihat [kuotaAWS layanan](#) di bagian Referensi Umum Amazon Web.

Anda dapat [meminta peningkatan kuota layanan](#) untuk kuotaCodeArtifact sumber daya berikut. Untuk informasi lebih lanjut tentang permintaan peningkatan Service Quotas, lihat [KuotaAWS Layanan](#).

Nama	Default	Dapat Disesuaikan	Deskripsi
Ukuran file aset	Setiap Wilayah yang didukung: 5 Gigabyte	<a href="#">Ya</a>	Ukuran file maksimum per aset.
Aset per versi paket	Setiap Wilayah yang didukung: 150	Tidak	Jumlah maksimum aset per versi paket.
CopyPackageVersionspermintaan per detik	Setiap Wilayah yang didukung: 5	<a href="#">Ya</a>	Jumlah maksimum panggilan yang dapat dilakukanCopyPackageVersions per detik.
Hulu langsung per repositori	Setiap Wilayah yang didukung: 10	Tidak	Jumlah maksimum repositori hulu langsung per repositori.
Domain perAWS akun	Setiap Wilayah yang didukung: 10	<a href="#">Ya</a>	Jumlah maksimum domain yang dapat dibuat perAWS akun.
GetAuthorizationTokenpermintaan per detik	Setiap Wilayah yang didukung: 40	<a href="#">Ya</a>	Jumlah maksimum token otorisasi yang diambil per detik.

Nama	Default	Dapat Disesan	Deskripsi
GetPackageVersionAssetpermintaan per detik	Setiap Wilayah yang didukung: 50	<a href="#">Ya</a>	Jumlah maksimum panggilan yang dapat dilakukanGetPackag eVersionAsset per detik.
ListPackageVersionAssetspermintaan per detik	Setiap Wilayah yang didukung: 200	<a href="#">Ya</a>	Jumlah maksimum panggilan yang dapat dilakukanListPacka geVersionAssets per detik.
ListPackageVersionspermintaan per detik	Setiap Wilayah yang didukung: 200	<a href="#">Ya</a>	Jumlah maksimum panggilan yang dapat dilakukanListPacka geVersions per detik.
ListPackagespermintaan per detik	Setiap Wilayah yang didukung: 200	<a href="#">Ya</a>	Jumlah maksimum panggilan yang dapat dilakukanListPackages per detik.
PublishPackageVersionpermintaan per detik	Setiap Wilayah yang didukung: 10	<a href="#">Ya</a>	Jumlah maksimum panggilan yang dapat dilakukanPublishPa ckageVersion per detik.
Permintaan baca per detik dari satuAWS akun	Setiap Wilayah yang didukung: 800	<a href="#">Ya</a>	Jumlah maksimum permintaan baca dari satuAWS akun per detik.
Repositori per domain	Setiap Wilayah yang didukung: 1.000	<a href="#">Ya</a>	Jumlah maksimum repositori yang dapat dibuat per domain.

Nama	Default	Dapat Disesan	Deskripsi
Permintaan per detik menggunakan satu token autentikasi	Setiap Wilayah yang didukung: 1.200	Tidak	Jumlah maksimum permintaan per detik menggunakan satu token autentikasi.
Permintaan tanpa token autentikasi per alamat IP	Setiap Wilayah yang didukung: 600	Tidak	Jumlah maksimum permintaan per detik tanpa token autentikasi dari satu alamat IP.
Repositori hulu dicari	Setiap Wilayah yang didukung: 25	Tidak	Jumlah maksimum repositori hulu dicari saat menyelesaikan paket.
Permintaan per detik dari satu AWS akun	Setiap Wilayah yang didukung: 100	<a href="#">Ya</a>	Jumlah maksimum permintaan tulis dari satu AWS akun per detik.



# AWS CodeArtifact riwayat dokumen panduan pengguna

Tabel berikut menjelaskan perubahan penting pada dokumentasi untuk CodeArtifact.

Perubahan	Deskripsi	Tanggal
<a href="#">Menambahkan manajer paket valid tambahan ke dokumentasi tentang aws codeartifact login perintah.</a>	Ditambahkan dotnetnuget,, dan swift ke daftar manajer paket yang valid untuk digunakan dengan aws codeartifact login perintah. Untuk informasi selengkapnya, lihat <a href="#">AWS CodeArtifact otentikasi dan token</a> .	Februari 18, 2024
<a href="#">Menambahkan entri ke dokumentasi pemecahan masalah Swift tentang Xcode yang tergantung pada mesin CI</a>	Informasi tambahan, termasuk solusi, tentang masalah yang dapat menyebabkan Xcode bertahan di mesin CI karena permintaan gantungan kunci untuk kata sandi. Untuk informasi selengkapnya, lihat <a href="#">Xcode hang pada mesin CI karena permintaan gantungan kunci untuk kata sandi</a> .	Februari 6, 2024
<a href="#">Menambahkan informasi tentang pemecahan masalah waktu pemasangan paket npm lambat dengan npm 8.x atau lebih tinggi</a>	Menambahkan informasi tentang mengatasi waktu pemasangan paket npm yang lambat CodeArtifact, yang dapat menyebabkan waktu pembuatan lambat. Untuk informasi selengkapnya, lihat <a href="#">Memecahkan masalah</a>	Desember 29, 2023

---

<a href="#">Informasi terbaru tentang aset paket Python dan perilaku metadata di CodeArtifact</a>	<a href="#">pemasangan lambat dengan npm 8.x atau lebih tinggi.</a> Informasi terbaru tentang bagaimana CodeArtifact repositori mempertahankan dan menyegarkan aset dan metadata versi paket Python. Untuk informasi selengkapnya, lihat <a href="#">Meminta paket Python dari upstream dan koneksi eksternal.</a>	14 Desember 2023
<a href="#">Dokumentasi yang direorganisasi tentang pemantauan CodeArtifact</a>	Menata ulang informasi tentang pemantauan CodeArtifact peristiwa, dan menambahkan informasi tentang melihat CodeArtifact permintaan dengan CloudWatch metrik Amazon. Untuk informasi selengkapnya, lihat <a href="#">Pemantauan CodeArtifact.</a>	14 Desember 2023
<a href="#">Menambahkan informasi lebih lanjut tentang mengelola CodeArtifact sumber daya dengan AWS CloudFormation</a>	Menambahkan referensi dan tautan ke dokumentasi tentang mengelola CodeArtifact sumber daya dengan CloudFormation, termasuk bagian tentang mencegah penghapusan CodeArtifact sumber daya yang dikelola. CloudFormation Untuk informasi selengkapnya, lihat <a href="#">Mencegah penghapusan sumber daya CodeArtifact .</a>	Desember 7, 2023

[Menambahkan dokumentasi yang merinci CodeArtifact dukungan dari Toko Kunci AWS KMS Eksternal \(XKS\)](#)

Menambahkan bagian dengan informasi CodeArtifact tentang dukungan kunci KMS, termasuk menggunakan kunci XKS dengan CodeArtifact. Untuk informasi selengkapnya, lihat [Jenis AWS KMS kunci yang didukung di CodeArtifact](#).

31 Oktober 2023

[Diperbarui dokumentasi pemecahan masalah baru yang ada dan ditambahkan](#)

Menambahkan topik pemecahan masalah Maven dan menyertakan tautan ke dokumentasi pemecahan masalah Swift dan Maven dalam topik pemecahan masalah umum. Untuk informasi selengkapnya, lihat [Pemecahan Masalah AWS CodeArtifact](#).

28 September 2023

[Dokumentasi yang diperbarui untuk menyertakan perintah Swift Package Manager publish](#)

Swift 5.9 memperkenalkan swift package-registry publish perintah untuk membuat dan menerbitkan paket Swift ke repositori paket. Diperbarui dokumentasi Swift untuk menyertakan instruksi untuk menggunakan perintah itu. Untuk informasi selengkapnya, lihat [Menggunakan CodeArtifact dengan Swift](#).

25 September 2023

[Ditambahkan dokumentasi untuk mengkonfigurasi CodeArtifact dengan Swift](#)

CodeArtifact sekarang mendukung paket Swift. Menambahkan dokumentasi dengan panduan tentang mengkonfigurasi Swift untuk menggunakan CodeArtifact repositori. Untuk informasi selengkapnya, lihat [Menggunakan CodeArtifact dengan Swift](#).

20 September 2023

[Menambahkan panduan tentang cara CodeArtifact menangani versi paket Python yang ditarik](#)

Menambahkan dokumentasi dengan informasi tentang cara mengetahui apakah versi paket Python ditarik, cara CodeArtifact menangani versi paket yang ditarik, dan jawaban atas pertanyaan umum. Untuk informasi selengkapnya, lihat [Versi paket yang ditarik](#).

2 Agustus 2023

[Memperbaiki perintah baris perintah yang salah dalam dokumentasi Yarn](#)

Memperbaiki perintah baris perintah yang salah yang mengambil token CodeArtifact otorisasi dan menyimpannya dalam variabel lingkungan dalam dokumentasi [Yarn](#).

Juli 20, 2023

<a href="#">Penambahan kecil dan perbaikan bug kecil untuk dokumentasi Python</a>	Menambahkan informasi pip dan benang dalam dokumentasi masing-masing dan mengoreksi apa yang terjadi saat menggunakan codeartifact login perintah dengan benang. Lihat informasi yang lebih lengkap di <a href="#">Konfigurasi dan gunakan pip dengan CodeArtifact</a> dan <a href="#">Konfigurasi dan gunakan benang dengan CodeArtifact</a> .	14 Juli 2023
<a href="#">Memperbaiki perintah dotnet yang salah dalam dokumentasi CodeBuild</a>	Memperbaiki dotnet add package perintah dalam <a href="#">Menggunakan NuGet paket di CodeBuild</a> dokumentasi.	13 Juli 2023
<a href="#">Diperbarui AWS CodeArtifact dan AWS Identity and Access Management dokumentasi</a>	Merombak IAM dalam CodeArtifact dokumentasi untuk menambah kejelasan dan konsistensi dengan dokumentasi untuk layanan lain. AWS Lihat <a href="#">Identity and Access Management untuk AWS CodeArtifact</a> .	24 Mei 2023
<a href="#">Menambahkan informasi tentang versi paket Python yang ditarik</a>	Menambahkan informasi tentang cara CodeArtifact mempertahankan metadata versi paket Python yang ditarik, Untuk informasi selengkapnya, lihat. <a href="#">Versi paket yang ditarik</a>	11 April 2023

[Menambahkan informasi tentang dukungan Clojure](#)

Menambahkan informasi tentang dukungan Clojure, termasuk mengelola dependensi untuk proyek Clojure. Untuk informasi selengkapnya, lihat [Gunakan CodeArtifact dengan deps.edn](#).

21 Maret 2023

[Menambahkan informasi tentang penerbitan paket generik](#)

Menambahkan informasi tentang paket generik dan cara mempublikasikan dan mengunduh konten paket dengan AWS CLI. Lihat informasi selengkapnya di [Menggunakan CodeArtifact dengan paket generik](#), [Menerbitkan dan mengonsumsi paket generik](#), dan [Perintah yang didukung untuk paket generik](#).

10 Maret 2023

[Menambahkan informasi tentang batas ukuran aset untuk penerbitan](#)

Menambahkan bagian ke Package publishing untuk menjelaskan batas ukuran aset untuk penerbitan.

21 Juni 2022

[Memfaktorkan ulang dokumentasi koneksi eksternal](#)

Memindahkan dokumentasi koneksi eksternal dan mengatur ulang untuk fokus pada tujuan akhir pengguna, yaitu menghubungkan repositori mereka ke CodeArtifact repositori paket publik. Juga menambahkan lebih banyak panduan dan informasi seputar metode yang berbeda untuk mencapai tujuan itu. Untuk informasi selengkapnya, lihat [Connect CodeArtifact repositori ke repositori publik](#).

9 Mei 2022

[Memperbarui informasi CodeArtifact acara untuk Amazon CloudWatch Events](#)

Menambahkan informasi lebih lanjut ke account bidang dan menambahkan repository Administrator bidang. Untuk informasi selengkapnya, lihat [CodeArtifact format acara dan contoh](#).

7 Maret 2022

[Menambahkan instruksi konfigurasi untuk menggunakan CodeArtifact dari VPC tanpa DNS pribadi](#)

Jika Anda tidak dapat atau tidak ingin mengaktifkan DNS pribadi di titik akhir `codeartifact.repositories` VPC Anda, Anda harus menggunakan konfigurasi yang berbeda untuk titik akhir repositori yang akan digunakan dari VPC. CodeArtifact Untuk informasi selengkapnya, lihat [Gunakan `codeartifact.repositories` titik akhir tanpa DNS pribadi](#).

8 Februari 2022

[Ditambahkan dokumentasi mendalam untuk memperbarui status versi paket](#)

Memperluas dokumentasi status versi paket pembaruan ke topiknya sendiri. Menambahkan dokumentasi untuk memperbarui status versi paket, termasuk izin IAM yang diperlukan, AWS CLI perintah contoh untuk berbagai skenario, dan kemungkinan kesalahan. Untuk informasi selengkapnya, lihat [Memperbarui status versi paket](#).

1 September 2021



[Memperbarui dokumentasi versi paket salin dengan informasi izin yang lebih mendalam](#)

Menambahkan informasi lebih lanjut tentang IAM yang diperlukan dan izin kebijakan berbasis sumber daya untuk memanggil `aws codeartifact copy-package-versions` perintah untuk menyalin versi paket dari satu repositori ke repositori lain dalam domain yang sama di CodeArtifact. Seiring dengan informasi lebih lanjut, sekarang ada contoh kebijakan berbasis sumber daya yang diperlukan untuk repositori sumber dan tujuan. Untuk informasi selengkapnya, lihat [Izin IAM yang diperlukan untuk menyalin paket](#).

25 Agustus 2021

[Dokumentasi yang diperbarui untuk menjalankan build Gradle di IntelliJ IDEA](#)

Memperbarui dokumentasi untuk menjalankan build Gradle di IntelliJ IDEA dengan langkah-langkah untuk mengonfigurasi Gradle untuk mengambil plugin. CodeArtifact Juga menambahkan opsi untuk membuat token CodeArtifact otorisasi baru untuk setiap proses baru dengan panggilan `aws codeartifact get-authorization-token`. Untuk informasi selengkapnya, lihat [Menjalankan build Gradle di IntelliJ IDEA](#).

23 Agustus 2021

[Menambahkan dokumentasi untuk mengkonfigurasi dan menggunakan Yarn dengan AWS CodeArtifact](#)

Menambahkan dokumentasi untuk mengkonfigurasi dan menggunakan Yarn 1.X dan Yarn 2.X untuk mengelola paket npm dengan CodeArtifact. Untuk informasi selengkapnya, lihat [Konfigurasi dan gunakan Yarn dengan CodeArtifact](#).

30 Juli 2021

[AWS CodeArtifact sekarang mendukung NuGet paket](#)

CodeArtifact pengguna sekarang dapat mempublikasikan dan menggunakan NuGet paket. Menambahkan dokumentasi untuk mengkonfigurasi dan menggunakan alat Visual Studio dan baris NuGet perintah seperti nuget dan dotnet dengan CodeArtifact repositori. Untuk informasi selengkapnya, lihat [Menggunakan CodeArtifact dengan NuGet](#).

19 November 2020

[Menandai sumber daya di AWS CodeArtifact](#)

Menambahkan dokumentasi tentang penandaan repositori dan domain di AWS CodeArtifact. Lihat [Penandaan pada sumber daya](#).

30 Oktober 2020

<a href="#">CodeArtifact sekarang mendukung AWS CloudFormation</a>	CodeArtifact pengguna sekarang dapat menggunakan AWS CloudFormation template untuk membuat CodeArtifact repositori dan domain. Lihat <a href="#">Menciptakan CodeArtifact sumber daya dengan AWS CloudFormation</a> untuk informasi selengkapnya dan memulai.	8 Oktober 2020
<a href="#">Tambahkan informasi tentang membuat titik akhir gateway Amazon S3 untuk digunakan dengan CodeArtifact Amazon VPC</a>	Menambahkan informasi tentang membuat titik akhir gateway Amazon S3 dengan perintah Amazon EC2. AWS CLI Dokumentasi ini juga berisi informasi tentang izin khusus yang CodeArtifact perlu digunakan dengan lingkungan Amazon VPC. Lihat <a href="#">Buat titik akhir gateway Amazon S3</a> .	12 Agustus 2020
<a href="#">Menerbitkan artefak Maven dengan curl dan menerbitkan artefak Maven pihak ketiga</a>	Menambahkan panduan untuk <a href="#">Publikasi dengan curl</a> dan <a href="#">Memublikasikan artefak pihak ketiga</a> .	10 Agustus 2020
<a href="#">Rilis Ketersediaan Umum (GA)</a>	Versi awal Panduan CodeArtifact Pengguna.	10 Juni 2020

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.