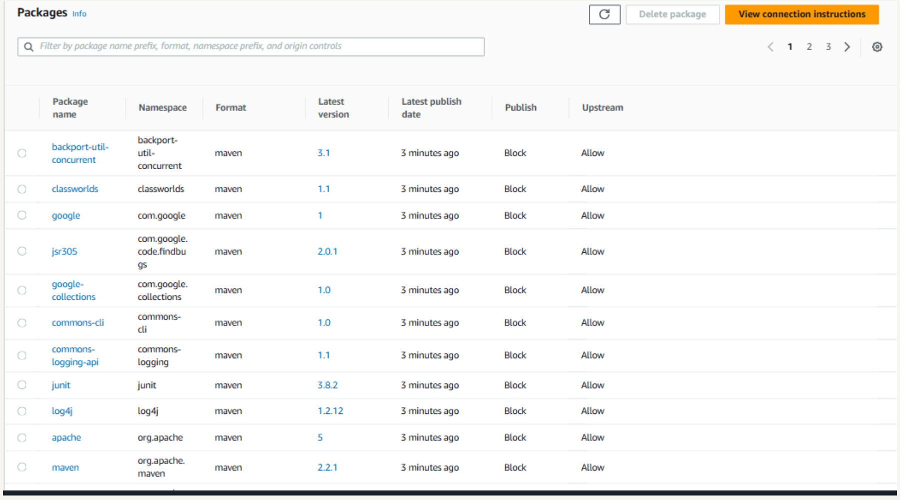
Secure Packages with CodeArtifact

Thabang Khasebe



Packages Info
Delete package
View connection instructions
Filter by package name prefix, format, namespace prefix, and origin controls
1 2 3

	Package name	Namespace	Format	Latest version	Latest publish date	Publish	Upstream
○	backport-util-concurrent	backport-util-concurrent	maven	3.1	3 minutes ago	Block	Allow
○	classworlds	classworlds	maven	1.1	3 minutes ago	Block	Allow
○	google	com.google	maven	1	3 minutes ago	Block	Allow
○	jsr305	com.google.code.findbugs	maven	2.0.1	3 minutes ago	Block	Allow
○	google-collections	com.google.collections	maven	1.0	3 minutes ago	Block	Allow
○	commons-cli	commons-cli	maven	1.0	3 minutes ago	Block	Allow
○	commons-logging-api	commons-logging	maven	1.1	3 minutes ago	Block	Allow
○	junit	junit	maven	3.8.2	3 minutes ago	Block	Allow
○	log4j	log4j	maven	1.2.12	3 minutes ago	Block	Allow
○	apache	org.apache	maven	5	3 minutes ago	Block	Allow
○	maven	org.apache.maven	maven	2.2.1	3 minutes ago	Block	Allow

Introducing Today's Project!

In this project, I will demonstrate how to use AWS CodeArtifact to securely store and manage my web app’s software packages. I'm doing this project to learn how to protect dependencies and improve the security of my development workflow.

Key tools and concepts

Services I used were AWS EC2, IAM, and CodeArtifact. Key concepts I learnt include setting up secure repositories, using IAM roles for access, configuring Maven with CodeArtifact, and compiling to verify dependency retrieval.

Project reflection

This project took me approximately 2 hours. The most challenging part was resolving the permission error with CodeArtifact. It was most rewarding to see my Maven project successfully compile and pull dependencies from a secure AWS repository.

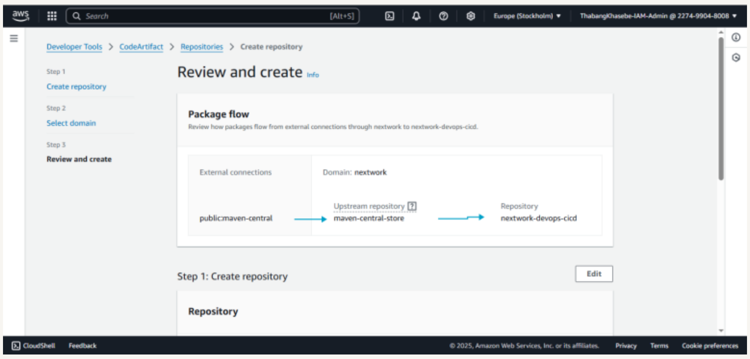
This project is part three of a series of DevOps projects where I'm building a CI/CD pipeline! I'll be working on the next project tomorrow.

CodeArtifact Repository

CodeArtifact is an AWS managed service that securely stores and manages software packages. Engineering teams use artifact repositories because they centralize dependencies, improve build consistency, and enhance security.

A domain is a collection of CodeArtifact repositories that share settings and permissions. My domain is the central place managing multiple repositories, making it easier to organize and control access across projects.

A CodeArtifact repository can have an upstream repository, meaning it fetches packages from another source if not found locally. My repository's upstream is Maven Central, the main public repository for Java packages.


aws
Search
[Alt+S]
Europe (Stockholm) ▾
ThabangKhasebe-IAM-Admin @ 2274-9904-8008 ▾
Developer Tools > CodeArtifact > Repositories > Create repository
Step 1
Create repository
Step 2
Select domain
Step 3
Review and create
Info
Edit
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Review and create

Package flow

Review how packages flow from external connections through nextwork to nextwork-devops-cicd.

External connections: public:maven-central

Domain: nextwork

Upstream repository: maven-central-store

Repository: nextwork-devops-cicd

Step 1: Create repository

Repository

CodeArtifact Security

Issue

To access CodeArtifact, we need an authorization token to authenticate securely. I ran into an error retrieving the token because my EC2 instance lacked the IAM policy needed to get and use the token.

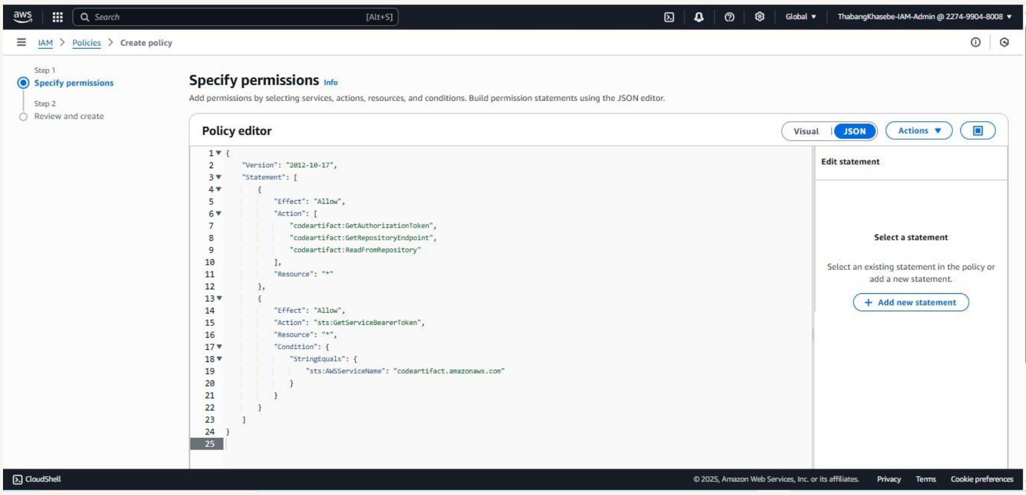
Resolution

To resolve the error with my security token, I created an IAM role with CodeArtifact permissions and attached it to my EC2 instance. This fixed the error because the instance gained the authorization needed to securely access CodeArtifact.

It's security best practice to use IAM roles because they provide temporary, managed permissions to resources without sharing long-term credentials, reducing the risk of leaks and ensuring secure, controlled access for services like EC2.

The JSON policy attached to my role

The JSON policy I set up grants permissions to access CodeArtifact repositories and fetch authorization tokens. These permissions are necessary so my EC2 instance can securely connect, download, and upload packages to the CodeArtifact repository.



Specify permissions Info

Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.

Policy editor

```
{
    "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"
                }
            }
        }
    ]
}
```

Select a statement

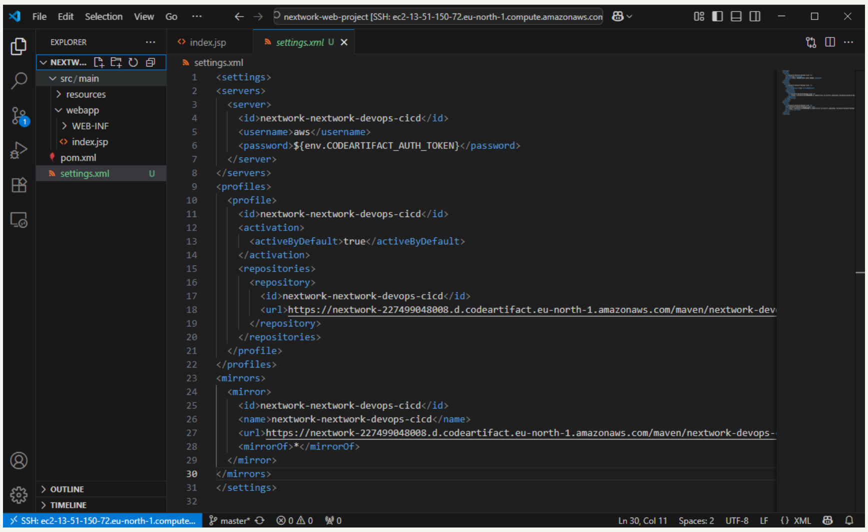
Select an existing statement in the policy or add a new statement.

Maven and CodeArtifact

To test the connection between Maven and CodeArtifact, I compiled my web app using settings.xml

The settings.xml file configures Maven to use CodeArtifact as a repository by specifying the repository URL and authentication details. This enables Maven to securely fetch and store project dependencies from CodeArtifact during builds.

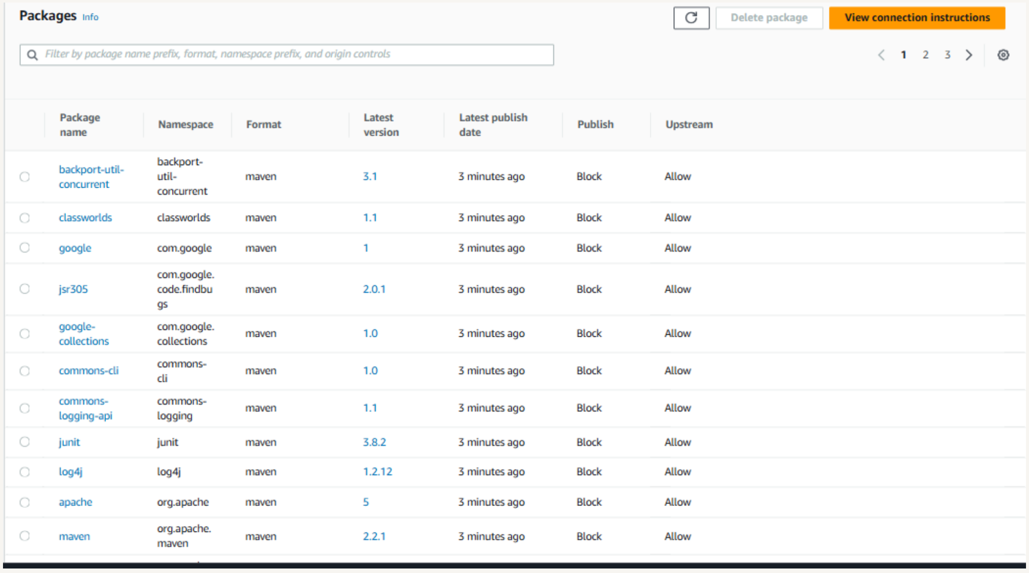
Compiling means transforming source code written in a programming language into executable code that a computer can run. In Java, it converts `.java` files into bytecode `.class` files that the Java Virtual Machine can execute.



```xml
<settings>
<servers>
  <server>
    <id>nextwork-nextwork-devops-cicd</id>
    <username>aws</username>
    <password>${env.CODEARTIFACT_AUTH_TOKEN}</password>
  </server>
</servers>
<profiles>
  <profile>
    <id>nextwork-nextwork-devops-cicd</id>
    <activation>
      <activeByDefault>true</activeByDefault>
    </activation>
    <repositories>
      <repository>
        <id>nextwork-nextwork-devops-cicd</id>
        <url>https://nextwork-227499048008.d.codeartifact.eu-north-1.amazonaws.com/maven/nextwork-dev
      </repository>
    </repositories>
  </profile>
</profiles>
<mirrors>
  <mirror>
    <id>nextwork-nextwork-devops-cicd</id>
    <name>nextwork-nextwork-devops-cicd</name>
    <url>https://nextwork-227499048008.d.codeartifact.eu-north-1.amazonaws.com/maven/nextwork-devops-
    <mirrorOf>*</mirrorOf>
  </mirror>
</mirrors>
</settings>
```

Verify Connection

After compiling, I checked my CodeArtifact repository. I noticed that the required dependencies for my Maven project were automatically stored there, confirming that the EC2 instance successfully pulled packages from CodeArtifact.


Packages
Info
Delete package
View connection instructions
Filter by package name prefix, format, namespace prefix, and origin controls
1
2
3

	Package name	Namespace	Format	Latest version	Latest publish date	Publish	Upstream
○	backport-util-concurrent	backport-util-concurrent	maven	3.1	3 minutes ago	Block	Allow
○	classworlds	classworlds	maven	1.1	3 minutes ago	Block	Allow
○	google	com.google	maven	1	3 minutes ago	Block	Allow
○	jsr305	com.google.code.findbugs	maven	2.0.1	3 minutes ago	Block	Allow
○	google-collections	com.google.collections	maven	1.0	3 minutes ago	Block	Allow
○	commons-cli	commons-cli	maven	1.0	3 minutes ago	Block	Allow
○	commons-logging-api	commons-logging	maven	1.1	3 minutes ago	Block	Allow
○	junit	junit	maven	3.8.2	3 minutes ago	Block	Allow
○	log4j	log4j	maven	1.2.12	3 minutes ago	Block	Allow
○	apache	org.apache	maven	5	3 minutes ago	Block	Allow
○	maven	org.apache.maven	maven	2.2.1	3 minutes ago	Block	Allow