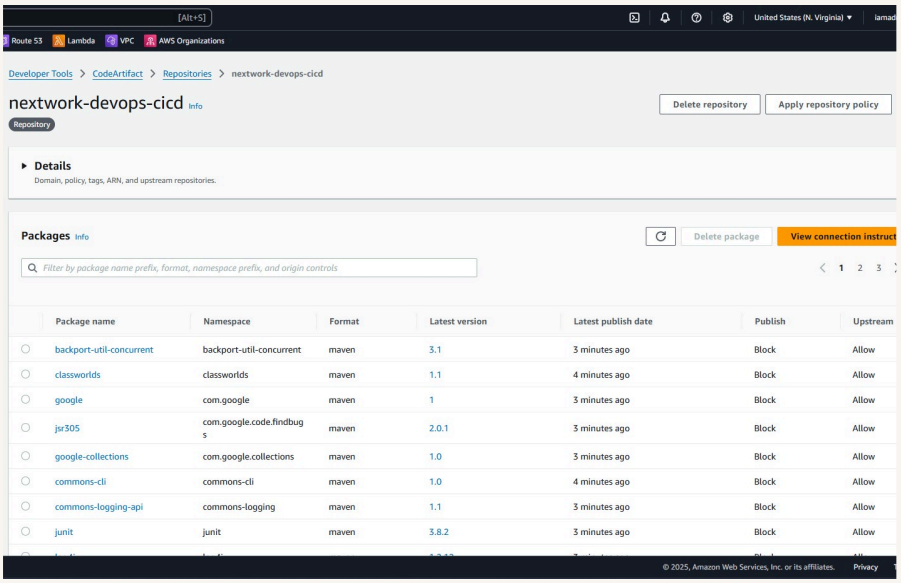
nextwork.org

Secure Packages with CodeArtifact

Gursimran Singh



Gursimran Singh
NextWork Student

Introducing Today's Project!

In this project, I will demonstrate how to use AWS CodeArtifact to manage project dependencies. I'm doing this project to learn how to securely connect a web app to CodeArtifact using IAM roles, and even upload my own custom packages!

Key tools and concepts

Services I used were AWS CodeArtifact, IAM, EC2, and Maven. Key concepts I learnt include managing dependencies securely with CodeArtifact, using IAM roles and policies for fine-grained access control, and integrating Maven with a private repo.

Project reflection

This project took me approximately 2 hours. The most challenging part was configuring IAM roles correctly. It was most rewarding to see the dependencies appear in CodeArtifact after compiling, confirming a successful and secure setup.

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 to continue strengthening my cloud and automation skills.

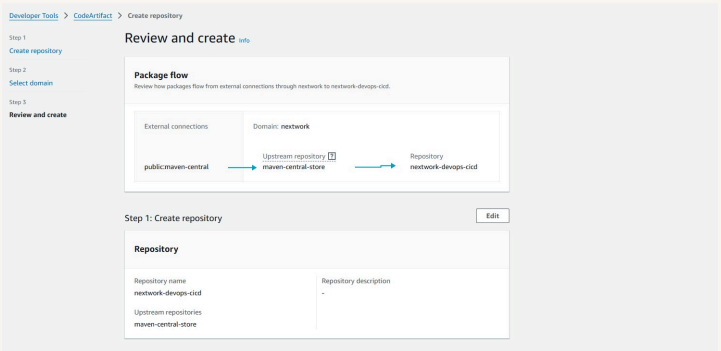
Gursimran Singh
NextWork Student

CodeArtifact Repository

CodeArtifact is a fully managed artifact repository service by AWS. Engineering teams use artifact repositories because they offer security, reliability, and control—making it safer and easier to manage and share package dependencies across projects.

A domain is a grouping mechanism in AWS CodeArtifact that helps manage multiple repositories under a single namespace, enabling better access control and dependency sharing across teams. My domain is used to centrally manage all related repositories.

A CodeArtifact repository can have an upstream repository, which means it can pull packages from another source if they aren't already stored locally. My repository's upstream repository is Maven Central, a common source for Java dependencies.



Gursimran Singh
NextWork Student

CodeArtifact Security

Issue

To access CodeArtifact, we need an authorization token that verifies our identity and grants temporary access. I ran into an error when retrieving a token because my EC2 instance didn’t have the correct IAM permissions at first.

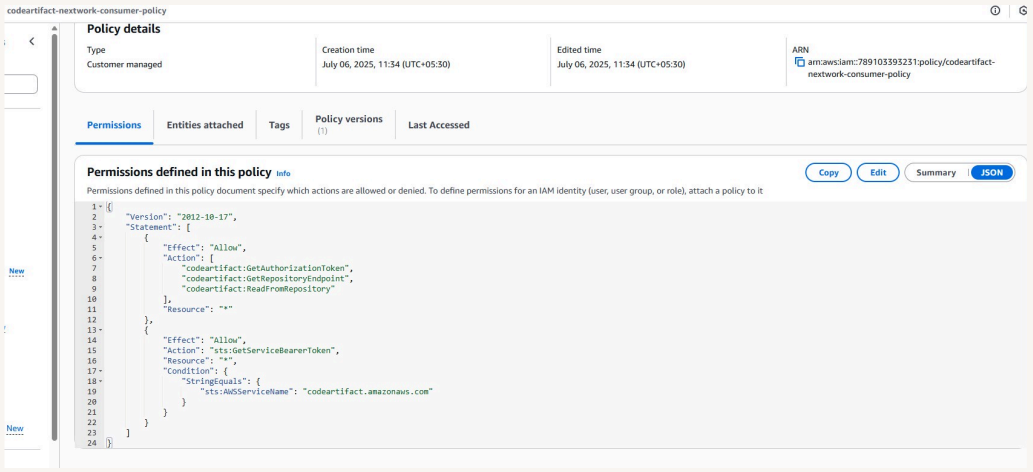
Resolution

To resolve the error with my security token, I attached the correct IAM role with CodeArtifact permissions to my EC2 instance. This resolved the error because the instance then had the necessary rights to access the repository securely.

It's security best practice to use IAM roles because they provide temporary, automatically rotated credentials, minimizing the risk of leaked access keys. Roles also support least-privilege access, helping secure resources effectively.

The JSON policy attached to my role

The JSON policy grants access to get a token, repo endpoint, and read from CodeArtifact. It also allows STS to issue a bearer token. These are needed for Maven on EC2 to fetch packages securely from CodeArtifact.



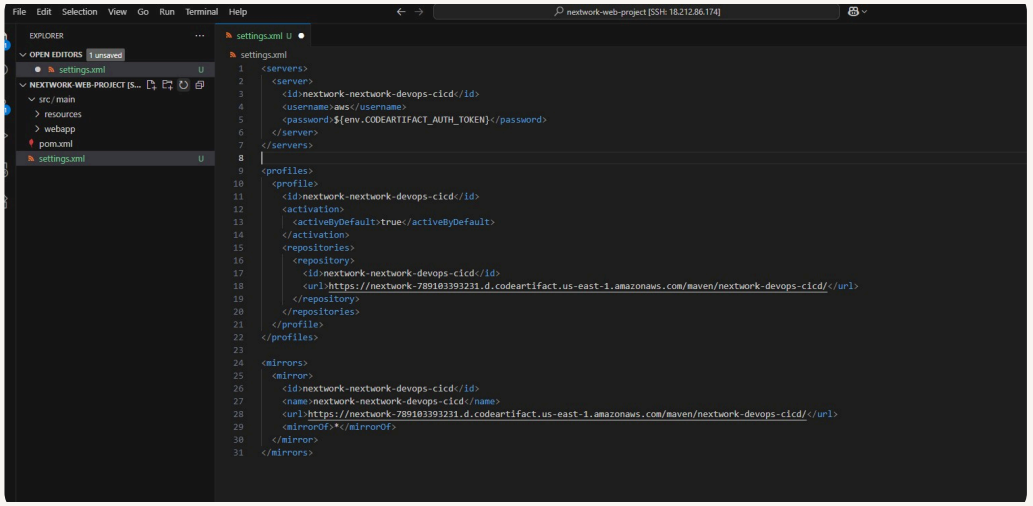
Gursimran Singh
NextWork Student

Maven and CodeArtifact

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

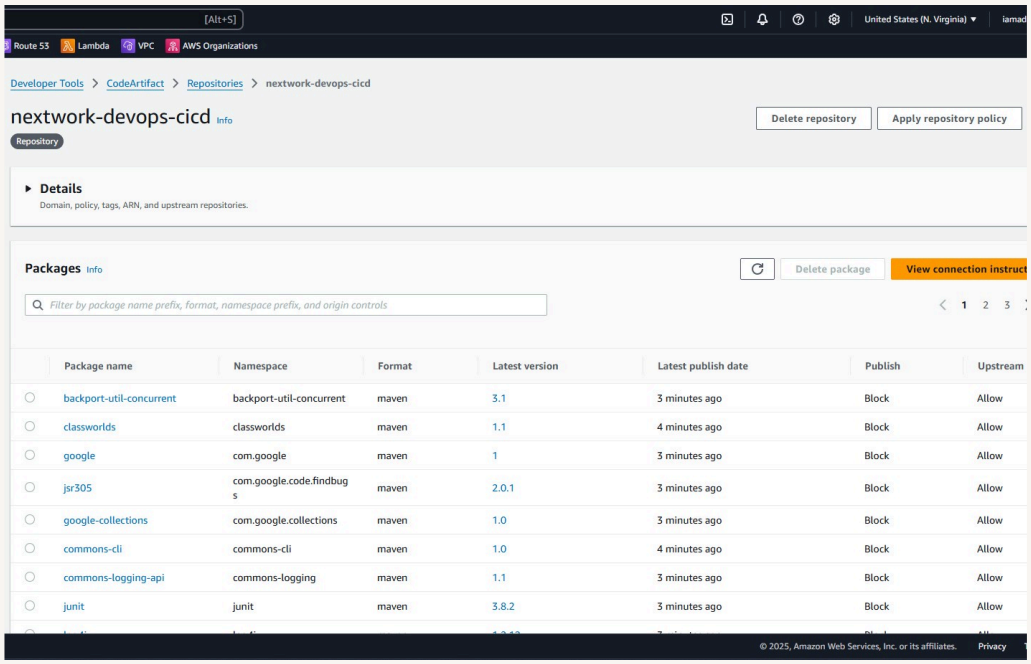
The `settings.xml` file configures Maven to authenticate with CodeArtifact by including the repository URL and authorization token. This allows Maven to securely access and download dependencies directly from the CodeArtifact repository.

Compiling means converting your source code into bytecode that the Java Virtual Machine can run. When Maven compiles with CodeArtifact, it checks for dependencies in CodeArtifact first, fetches missing ones from Maven Central, caches them, and builds



Verify Connection

After compiling, I checked my CodeArtifact repository. I noticed that the Java dependencies listed in my `pom.xml` were now stored there, meaning Maven successfully retrieved and cached them from the upstream repository.



The place to learn & showcase your skills

Check out nextwork.org for more projects