Hashicorp Vault Artifactory Secrets Plugin

Overview

The HashiCorp Vault plugin talks to JFrog Artifactory server (5.0.0 or later) and will dynamically provision access tokens with specified scopes.

Page Contents

- Overview


```
   ----------------------------------------------------------------
This plugin is now being actively maintained by JFrog Inc. Please refer to [CONTRIBUTING.md](CONTRIBUTING.
md) for contributions and create github issues to ask for support.
----------------------------------------------------------------

![Build](https://github.com/idcmp/artifactory-secrets-plugin/workflows/Build/badge.svg)

# Vault Artifactory Secrets Plugin

This is a [HashiCorp Vault](https://www.vaultproject.io/) plugin which talks to JFrog Artifactory server
and will
dynamically provision access tokens with specified scopes. This backend can be mounted multiple times
to provide access to multiple Artifactory servers.

Using this plugin, you can limit the accidental exposure window of Artifactory tokens; useful for
continuous integration servers.

## Access Token Creation and Revoking

This backend creates access tokens in Artifactory using the admin credentials provided. Note that if you
provide non-admin credentials, then the "username" must match the username of the credential owner.

## What's Missing

* rotate the admin/config access_token when it's configured (if it's refreshable).

## Testing Locally

If you're compiling this yourself and want to do a local sanity test, you
can do something like:

```bash
terminal-1$ make
...

terminal-2$ export VAULT_ADDR=http://127.0.0.1:8200
terminal-2$ export VAULT_TOKEN=root
terminal-2$ make setup
...

terminal-2$ make artifactory &  # Runs netcat returning a static JSON response
terminal-2$ vault read artifactory/token/test
```

Installation

Using pre-built releases

You can find pre-built releases of the plugin [here][artreleases]. Once you have downloaded the latest
archive corresponding to your target OS, uncompress it to retrieve the `artifactory` binary file.

From Sources
```

If you prefer to build the plugin from sources, clone the GitHub repository locally and run the command `make build` from the root of the sources directory. Upon successful compilation, the resulting `artifactory` binary is stored in the `vault/plugins` directory.

Configuration

Copy the plugin binary into a location of your choice; this directory must be specified as the [`plugin_directory`][vaultdocplugindir] in the Vault configuration file:

```hcl
plugin_directory = "path/to/plugin/directory"
```

Start a Vault server with this configuration file:

```sh
$ vault server -config=path/to/vault/config.hcl
```

Once the server is started, register the plugin in the Vault server's [plugin catalog] [vaultdocplugincatalog]:

```sh
$ vault write sys/plugins/catalog/secret/artifactory \
    sha_256="$(sha256sum path/to/plugin/directory/artifactory | cut -d " " -f 1)" \
    command="artifactory"
```

You can now enable the Artifactory secrets plugin:

```sh
$ vault secrets enable artifactory
```

Usage

Artifactory

You will need the "admin" user's password (not an admin, but admin specifically).

1. Log into the Artifactory UI as "admin".
1. Under "Welcome, admin" (top right) go to "Edit Profile".
1. Unlock your user profile and get your API Key. Save your API Key as an environment variable `KEY`.

You will now create the Access Token that Vault will use to interact with Artifactory. In Artifactory 7.4+ this can be done in the UI Administration --> Identity and Access --> Access Tokens(Service: Artifactory, Expiry: Never Expires), otherwise use the REST API:

```
curl -XPOST -u admin:$KEY "https://artifactory.example.org/artifactory/api/security/token" \
    -dusername=admin \
    -dexpires_in=0 \
    "-dscope=member-of-groups:*"
```

Note that "username" must be "admin" otherwise you will not be able to specify different usernames for roles. Save the "access_token" from the JSON response as the environment variable `TOKEN`.

```
$ vault write artifactory/config/admin \
              url=https://artifactory.example.org/artifactory \
              access_token=$TOKEN
```

```
$ vault write artifactory/roles/jenkins \
                username="example-service-jenkins" \
                scope="api:* member-of-groups:ci-server" \  // for artifactory < 7.21.1
                default_ttl=1h max_ttl=3h

$ vault write artifactory/roles/jenkins \
                username="example-service-jenkins" \
                scope="applied-permissions/groups:automation " \  // scope for artifactory >= 7.21.1
                default_ttl=1h max_ttl=3h
```
Also supports grant_type=[Optional, default: "client_credentials"], and audience=[Optional, default: *@*]
see [JFrog documentation][artifactory-create-token].
Note : There are some changes in the **scopes** supported in artifactory request >7.21.
Please refer to the JFrog documentation for the same according to the artifactor version

```
$ vault list artifactory/roles
Keys
----
jenkins
```

```
$ vault read artifactory/token/jenkins
Key                 Value
---                 -----
lease_id            artifactory/token/jenkins/25jYH8DjUU548323zPWiSakh
lease_duration      1h
lease_renewable     true
access_token        adsdgbtybbeeyh...
role                jenkins
scope               api:* member-of-groups:ci-server
```

Issues

RTFACT-22477, proposing CIDR restrictions on the created access tokens.

[artreleases]: https://github.com/jfrog/artifactory-secrets-plugin/releases
[vaultdocplugindir]: https://www.vaultproject.io/docs/configuration/index.html#plugin_directory
[vaultdocplugincatalog]: https://www.vaultproject.io/docs/internals/plugins.html#plugin-catalog
[artifactory-create-token]: https://www.jfrog.com/confluence/display/JFROG
/Artifactory+REST+API#ArtifactoryRESTAPI-CreateToken.1