Predict | Protect | Prevent

ARCON|PAM

ARCON Integration with Jenkins

Table of Contents

1 Overview

Jenkins is an open-source automation server that is free to download. It aids continuous deployment and execution by automating the aspects of product creation related to building, testing, and deploying.

This document covers the ARCON PAM integration with Jenkins and how Jenkins is being used to automate the CI process.

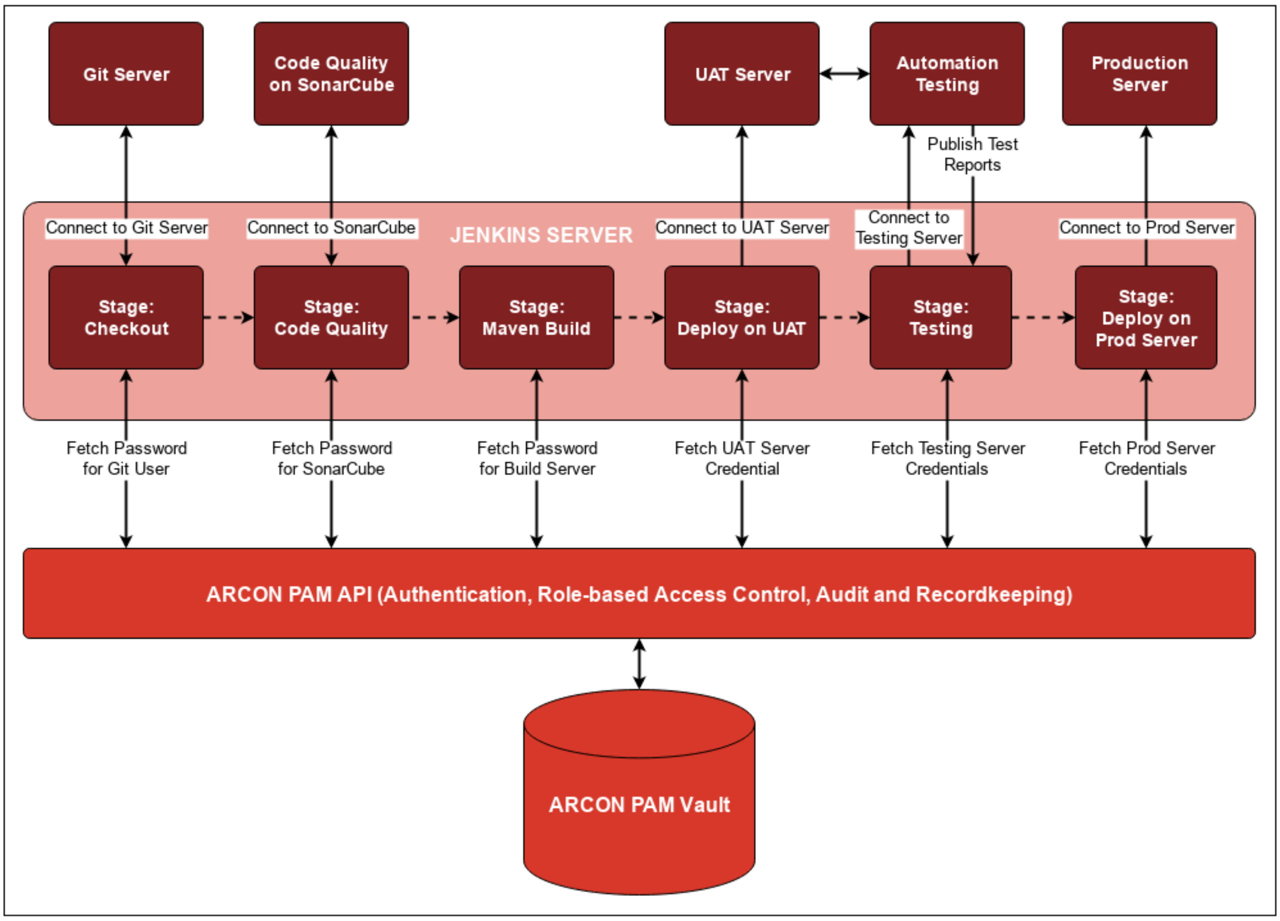
The ARCON PAM integration with Jenkins process contains several phases, which are shown as follows:

1. Configuration in ARCON PAM:
   a. API User Registration
   b. Register Machines

2. Configuration in Jenkins:
   a. Install PAM Plugin
   b. Add Build steps to get credentials of Target Machines from PAM
   c. Using SSH credentials obtained from a plugin in a pipeline script

2 Architecture

The following diagram represents the architecture of ARCON PAM integration with Jenkins:


Git Server
Code Quality on SonarCube
UAT Server
Automation Testing
Production Server
Publish Test Reports
Connect to Git Server
Connect to SonarCube
JENKINS SERVER
Connect to UAT Server
Connect to Testing Server
Connect to Prod Server
Stage: Checkout
Stage: Code Quality
Stage: Maven Build
Stage: Deploy on UAT
Stage: Testing
Stage: Deploy on Prod Server
Fetch Password for Git User
Fetch Password for SonarCube
Fetch Password for Build Server
Fetch UAT Server Credential
Fetch Testing Server Credentials
Fetch Prod Server Credentials
ARCON PAM API (Authentication, Role-based Access Control, Audit and Recordkeeping)
ARCON PAM Vault

In the Jenkins pipeline, there are multiple stages that require credentials for applications or servers to connect and perform a privileged activity needed for the DevOps stream to flow.

At every stage, that requires a secret to be fetched from ARCON PAM Vault, a relevant call is made to ARCON PAM APIs in the pipeline. Every call is authenticated, authorized, and audited within the ARCON PAM.

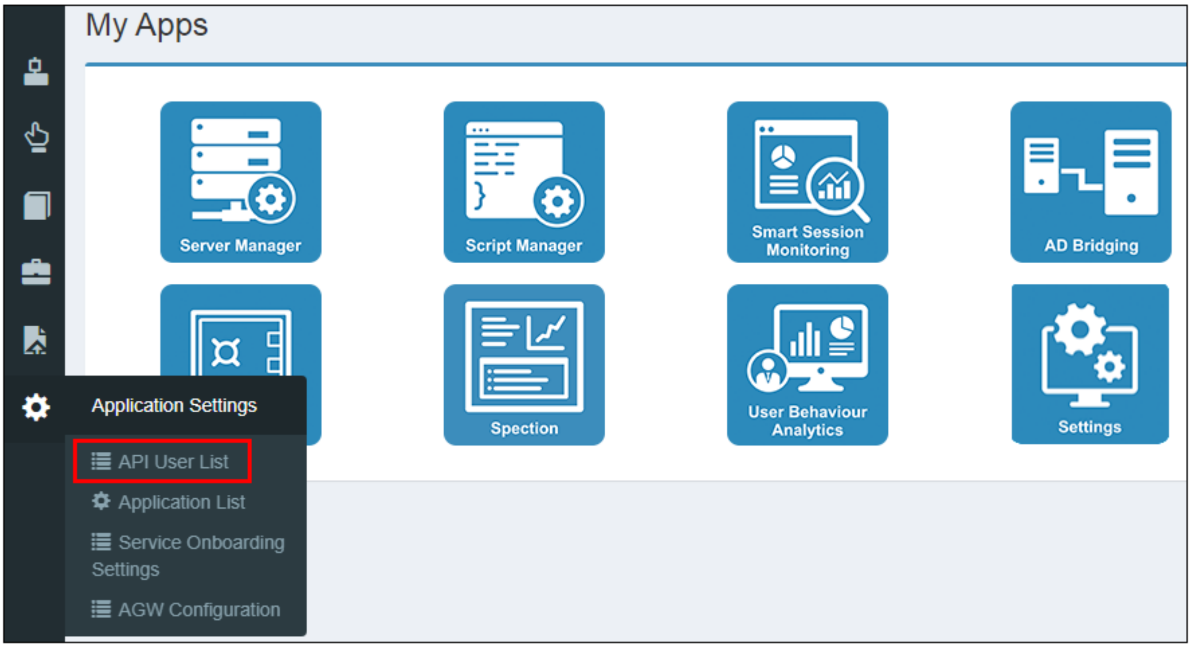
3 Configuration in ARCON PAM

3.1 API User Creation

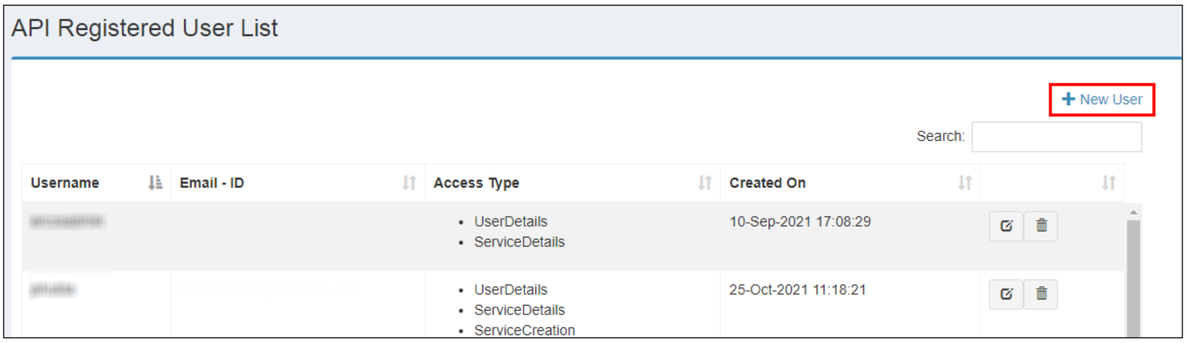
The API User Registration creates a user account in the ARCON PAM application, which will later on be used in Jenkins integration.

To create API User, perform the following steps:

1. From the ARCON PAM Application Setting, click on **API User List**:

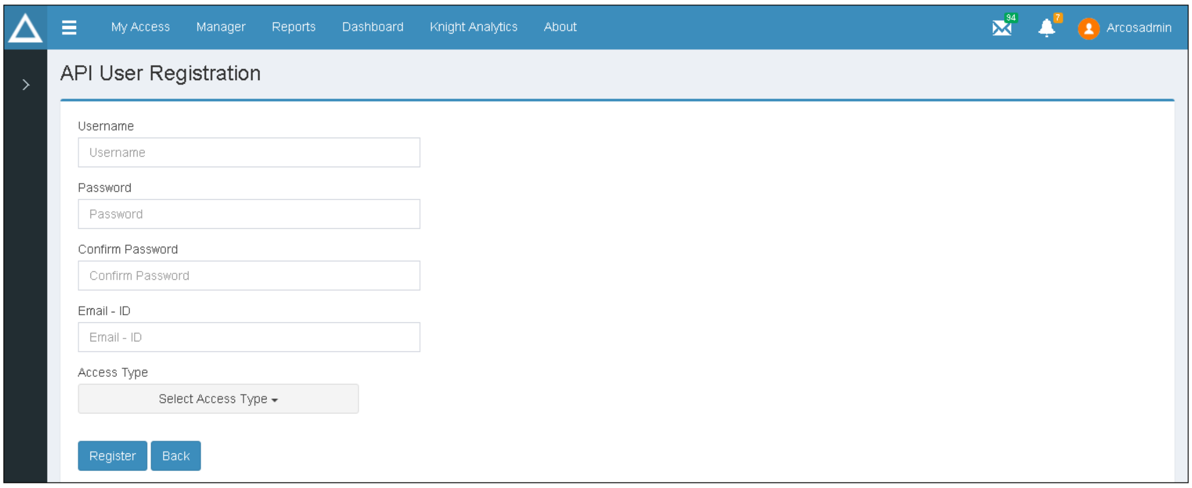


2. The API Registered User List appears. Click on the **+ New User** option:



Username	Email - ID	Access Type	Created On	
[redacted]	[redacted]	• UserDetails • ServiceDetails	10-Sep-2021 17:08:29	
[redacted]	[redacted]	• UserDetails • ServiceDetails • ServiceCreation	25-Oct-2021 11:18:21	

3. The API User Registration creation page will be displayed:



API User Registration

Username: Username

Password: Password

Confirm Password: Confirm Password

Email - ID: Email - ID

Access Type: Select Access Type

4. Refer to the following table to understand the field level description shown in the preceding screen:

Field Name	Description
User Name	The name of the user.
Password	The password of the user.
Confirm Password	Re-enter the password of the user.
Email ID	The email ID of the User.
Access Type	Select the access type **ServicePassword** from the dropdown, which is essential for Jenkins Configuration. for the user.

5. Once all the necessary fields are entered, click on **Register**. The API User will be created and listed in the API User List page.

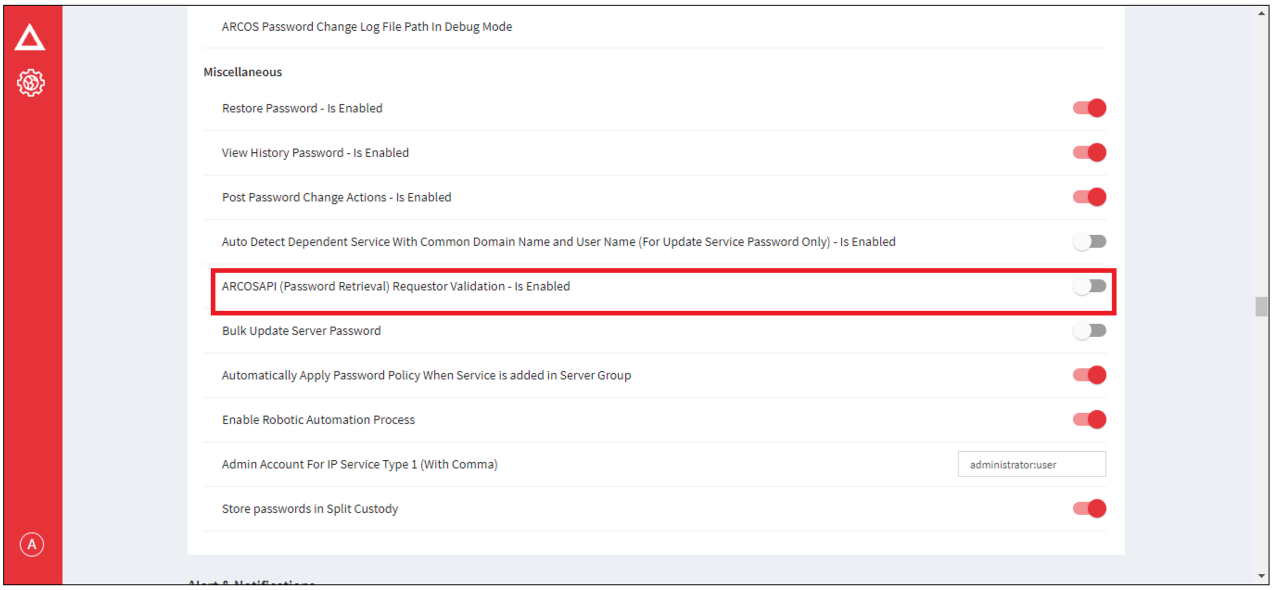
3.2 Registering Machines

Web API Registration helps you to register the user's machine IP address, where the user can view the password from the registered machine. This adds an additional layer of security as the requested passwords of target services created in ARCON PAM via API, will only be retrieved on the registered machine.

The Jenkins Server will need to be registered in this screen. Select services of the target devices that your Jenkins will need to connect with. Only then the passwords of these devices will be retrieved on Jenkins server via ARCON PAM Plugin.

For API Restriction to work, the **ARCOSAPI(Password Retrieval)RequestorValidator** configuration should be **enabled**. It works only if both the Requestor Machine and API Server are in the same Vlan/Subnet.

Path: **Settings** -> **Password** -> **Miscellaneous**



ARCOS Password Change Log File Path In Debug Mode

Miscellaneous

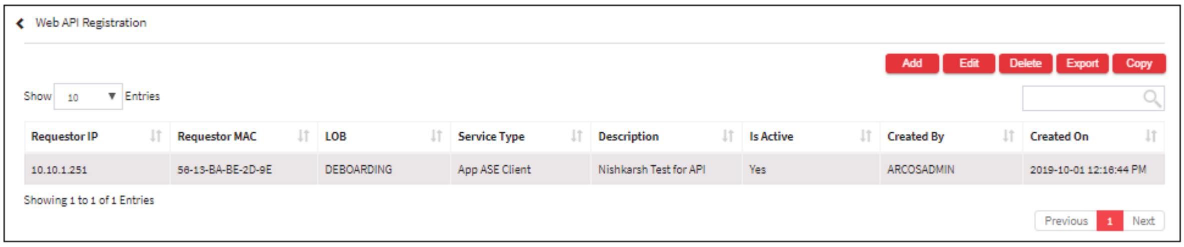
Restore Password - Is Enabled	●
View History Password - Is Enabled	●
Post Password Change Actions - Is Enabled	●
Auto Detect Dependent Service With Common Domain Name and User Name (For Update Service Password Only) - Is Enabled	○
ARCOSAPI (Password Retrieval) Requestor Validation - Is Enabled	○
Bulk Update Server Password	○
Automatically Apply Password Policy When Service is added in Server Group	●
Enable Robotic Automation Process	●
Admin Account For IP Service Type 1 (With Comma)	administrator:user
Store passwords in Split Custody	●

On enabling this configuration it will restrict Vault API Access and will allow only those API Request for which IP address and MAC address is registered in API registration.

To register user's machine, use the **Settings** → **API** → **Registered Machines** path and perform the below steps:

> The Administrator having **ARCON PAM Web API Registration** privileges in Server's Privileges will only be able to configure the Web API registration details.

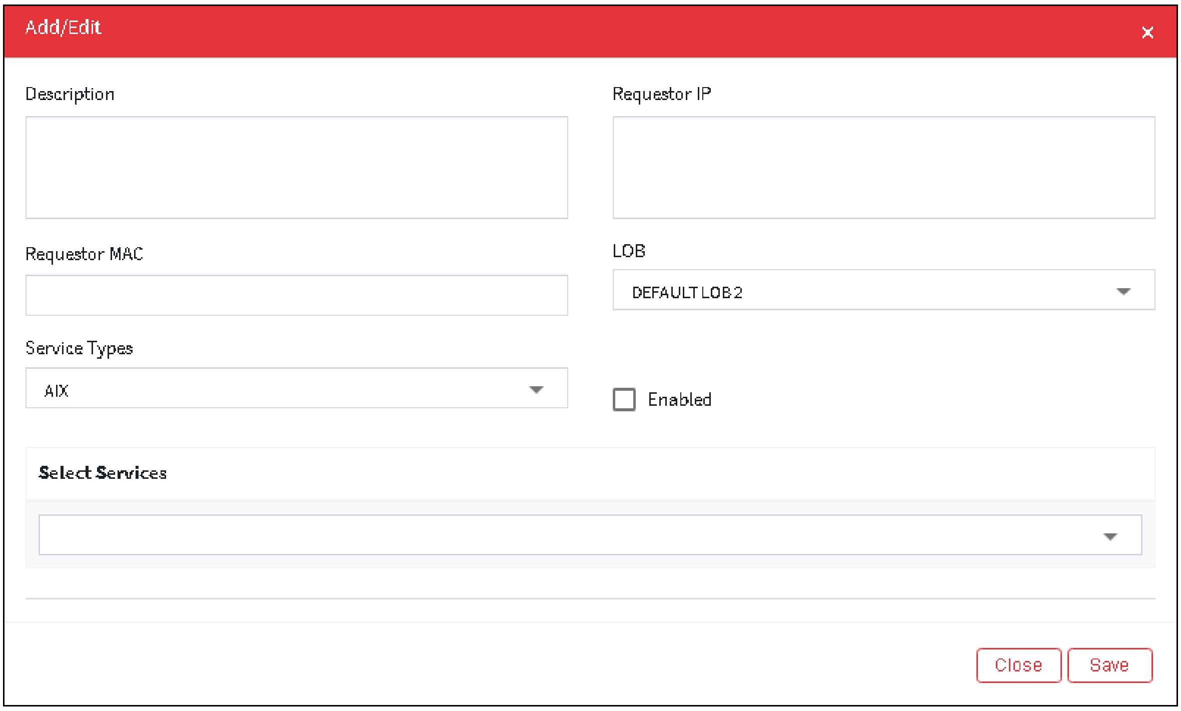
1. Select **Web API Registration** under **Registered Machines**:



Requestor IP	Requestor MAC	LOB	Service Type	Description	Is Active	Created By	Created On
10.10.1.251	56-13-BA-BE-2D-9E	DEBOARDING	App ASE Client	Nishkarsh Test for API	Yes	ARCOSADMIN	2019-10-01 12:16:44 PM

Showing 1 to 1 of 1 Entries

2. Click on the **Add** button to register a new Web API:



Add/Edit

Description

Requestor IP

Requestor MAC

LOB: DEFAULT LOB 2

Service Types: AIX

□ Enabled

Select Services

3. Refer to the below table to understand the fields displayed in the **Web API Registration** screen:

Field Name	Description
Description	Specify the description for registration.
Requestor IP	Specify the IP address of the requestor.
Requestor MAC	Specify the MAC address of the requestor.
LOB	Select the LOB from the dropdown.
Service Type	Select the service type from the dropdown.
Enabled	Check this option to enable the configuration.
Select Services	Based on the selection of Service Type, the list of services will be fetched. Select the services from the dropdown of which, the passwords can be retrieved for the machine.

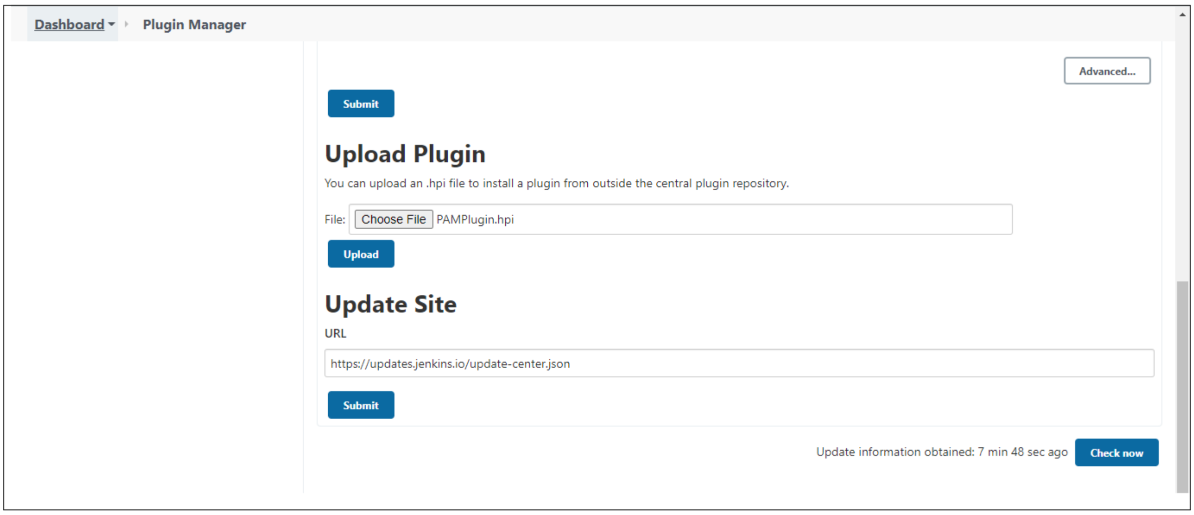
4. Enter the required details and click on **Save**.

4 Configuration in Jenkins

4.1 Plugin Installation

To install the ARCON PAM plugin, the user needs to perform the following steps:

1. From the Jenkins **Dashboard**, go to **Plugin Manager** and click on **Advanced**:



Upload Plugin

You can upload an .hpi file to install a plugin from outside the central plugin repository.

File: Choose File PAMPlugin.hpi

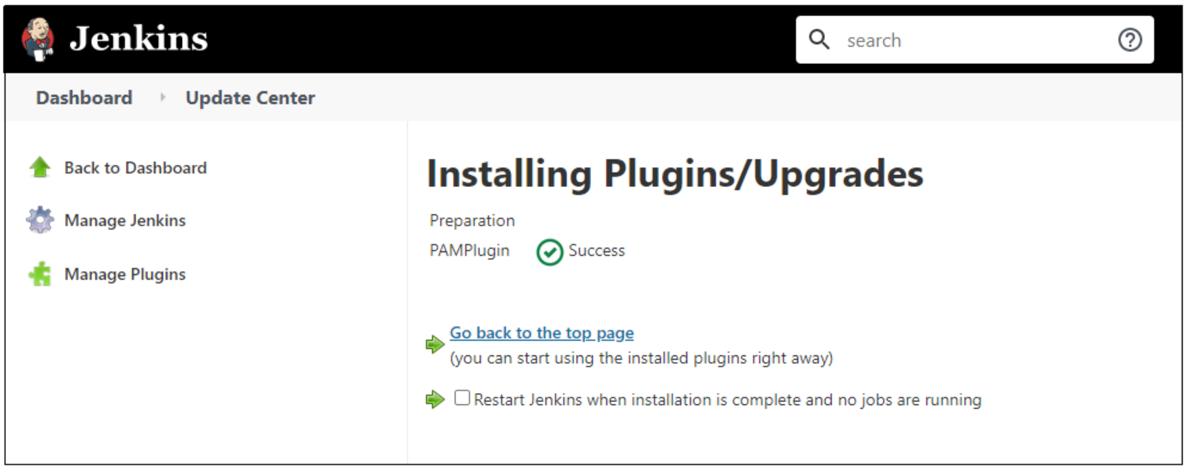
Update Site

URL

https://updates.jenkins.io/update-center.json

Update information obtained: 7 min 48 sec ago

2. From the **Upload Plugin** section, click on **Choose File** and select the **.hpi** file to upload.
3. The filename will appear next to the **Choose File** option. Click on **Upload**.
4. Once the Plugin is installed successfully, the following screen will be displayed:



Installing Plugins/Upgrades

Preparation

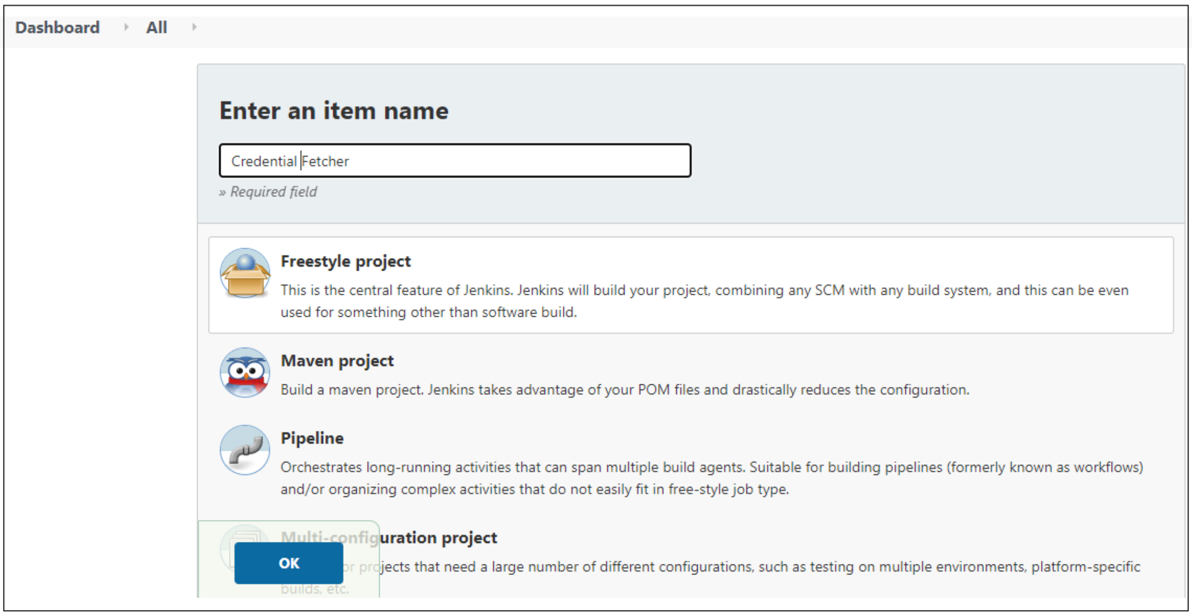
PAMPlugin ✓ Success

Go back to the top page
(you can start using the installed plugins right away)

□ Restart Jenkins when installation is complete and no jobs are running

4.2 Adding the User Credentials by Plugin

1. From the Jenkins **Dashboard**, go to **New Item** > **Freestyle project** and click on the **Build Environment** section:



Enter an item name

Credential Fetcher

» *Required field*

Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Maven project

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

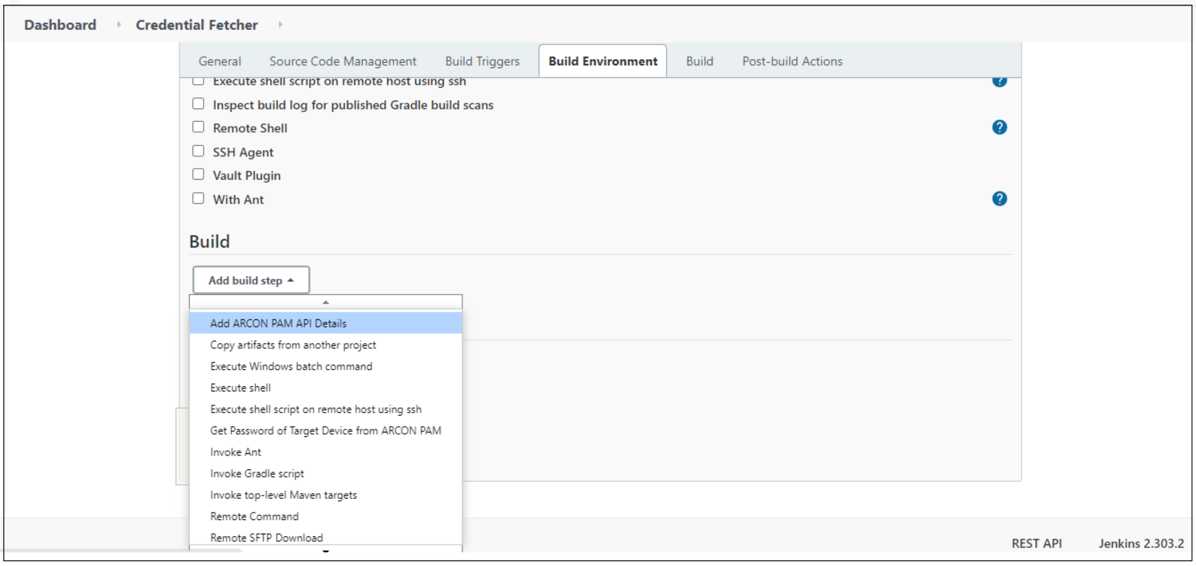
Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

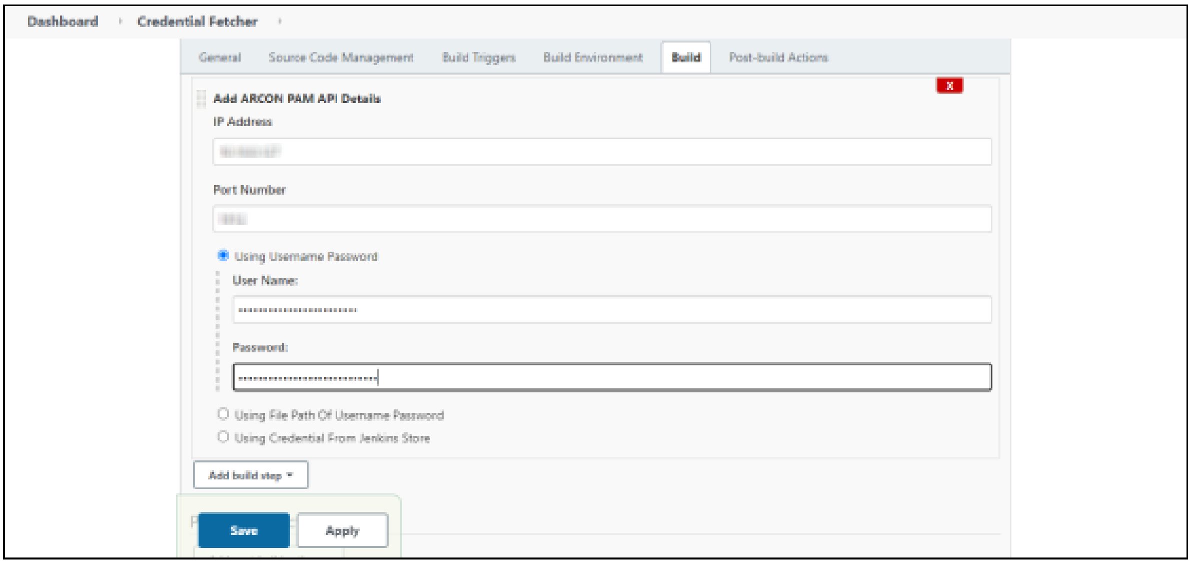
Multi-configuration project

...r projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

2. Open the **Build Environment** tab and go to the **Build** section.
3. Click on **Add Build Step** and select **Add ARCON PAM API Details** from the dropdown. This is the first mandatory build step in order to access the plugin features:


Dashboard
Credential Fetcher
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
□ Execute shell script on remote host using ssh
□ Inspect build log for published Gradle build scans
□ Remote Shell
□ SSH Agent
□ Vault Plugin
□ With Ant
Build
Add build step
Add ARCON PAM API Details
Copy artifacts from another project
Execute Windows batch command
Execute shell
Execute shell script on remote host using ssh
Get Password of Target Device from ARCON PAM
Invoke Ant
Invoke Gradle script
Invoke top-level Maven targets
Remote Command
Remote SFTP Download
REST API
Jenkins 2.303.2

4. The **Add ARCON PAM API Details** page will be displayed as below:



Add ARCON PAM API Details

IP Address

Port Number

● Using Username Password

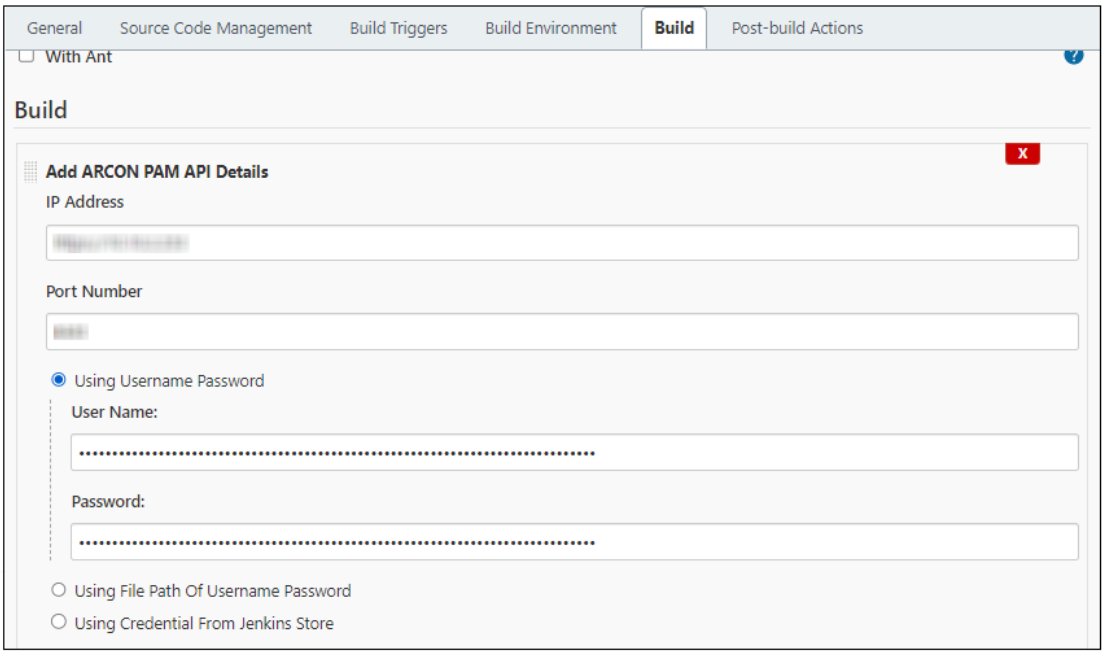
User Name: ••••••••••••••••••••••••

Password: ••••••••••••••••••••••••••••

○ Using File Path Of Username Password

○ Using Credential From Jenkins Store

5. Enter the applicable IP address and port number.
6. The username and password of the API User that we created during API User Creation in ARCON PAM will be used here. Select one of the user credentials methods from the following:
    a. **Using Username and Password**: This option can be used if the user running the job is given the API user credentials. Specify the username and password of API base64 encoded and click on **Save**.



□ With Ant

Build

Add ARCON PAM API Details

IP Address

Port Number

● Using Username Password

User Name: ••

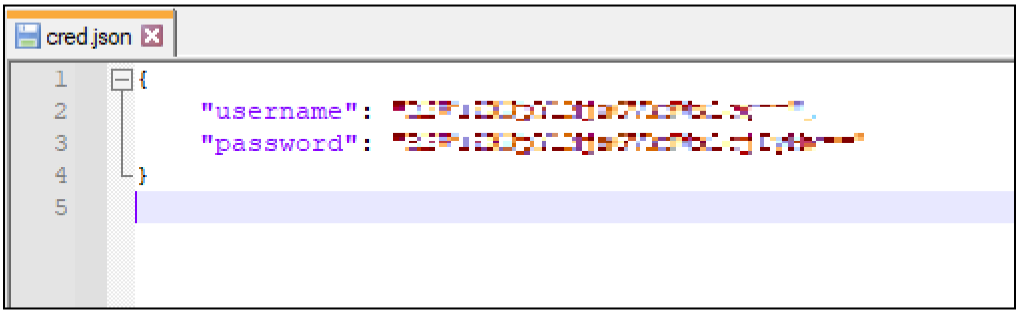
Password: ••

○ Using File Path Of Username Password

○ Using Credential From Jenkins Store

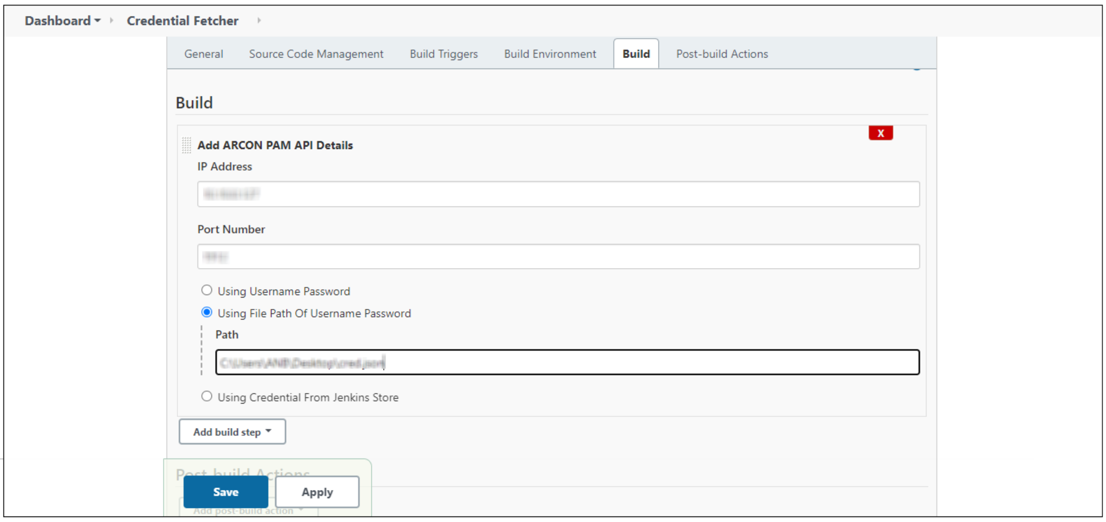
    b. **Using File path of Username and Password**: This option is preferred if the API Cred administrator and Job Administrator are separate entities and credentials are not stored within

Jenkins.
Add the path of the API user credential JSON file:



```
{
    "username": [redacted]
    "password": [redacted]
}
```

Specify the path in the **Path** field of **Using File Path of Username Password** option, and click on **Save**:



Build

Add ARCON PAM API Details

IP Address: [redacted]

Port Number: [redacted]

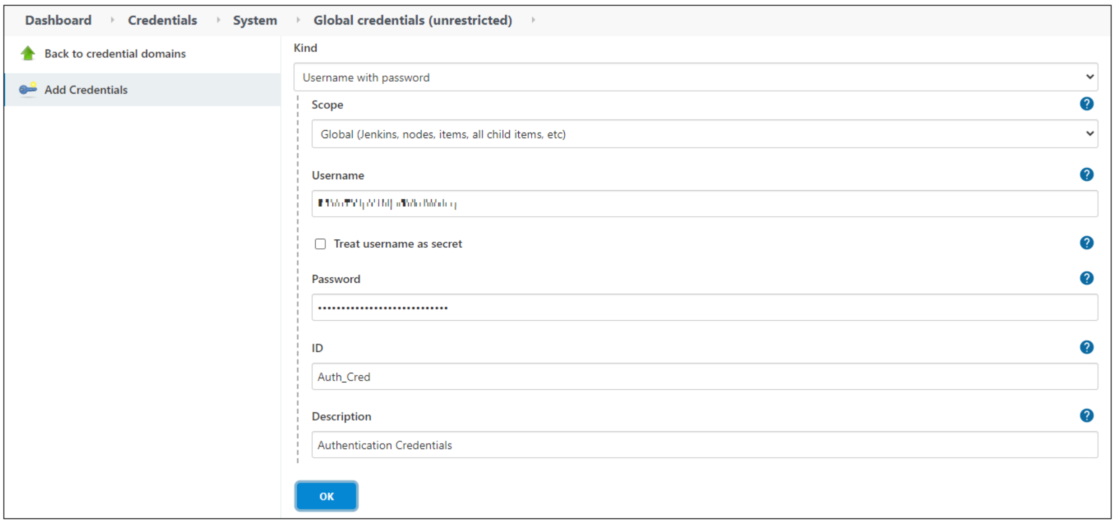
○ Using Username Password

● Using File Path Of Username Password

Path: [redacted]

○ Using Credential From Jenkins Store

c. **Using credentials from the Jenkins store**: The API Cred Administrator creates a username–password credential in the Jenkins Credential Store and shares the Credential ID with the Job Administrator.
Go to **Dashboard** > **Manage Jenkins** > **Manage Credentials** > **Global (Domain)** >**Add Credentials** to create credentials in the store as shown in the following screen:



Dashboard › Credentials › System › Global credentials (unrestricted) ›

Back to credential domains

Add Credentials

Kind

Username with password

Scope

Global (Jenkins, nodes, items, all child items, etc)

Username

[redacted]

□ Treat username as secret

Password

••••••••••••••••••••••••••••••••

ID

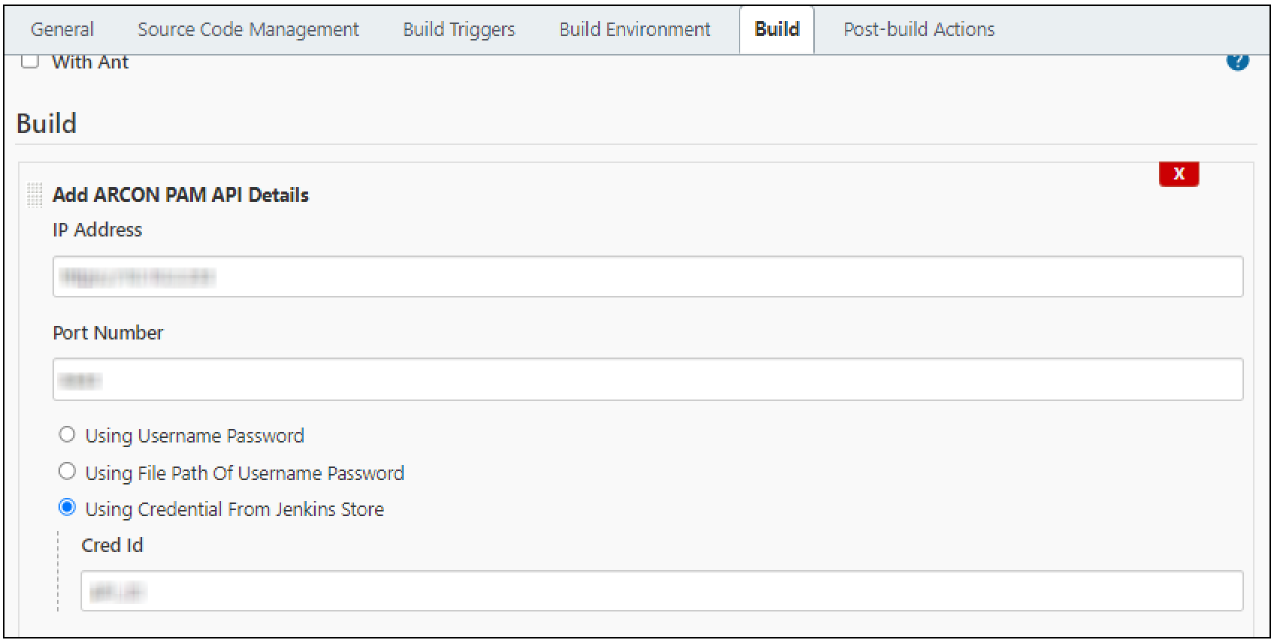
Auth_Cred

Description

Authentication Credentials

OK

Add the credential ID in the textbox, and click on **Save**:



General | Source Code Management | Build Triggers | Build Environment | Build | Post-build Actions

□ With Ant

Build

Add ARCON PAM API Details

IP Address

[redacted]

Port Number

[redacted]

○ Using Username Password

○ Using File Path Of Username Password

● Using Credential From Jenkins Store

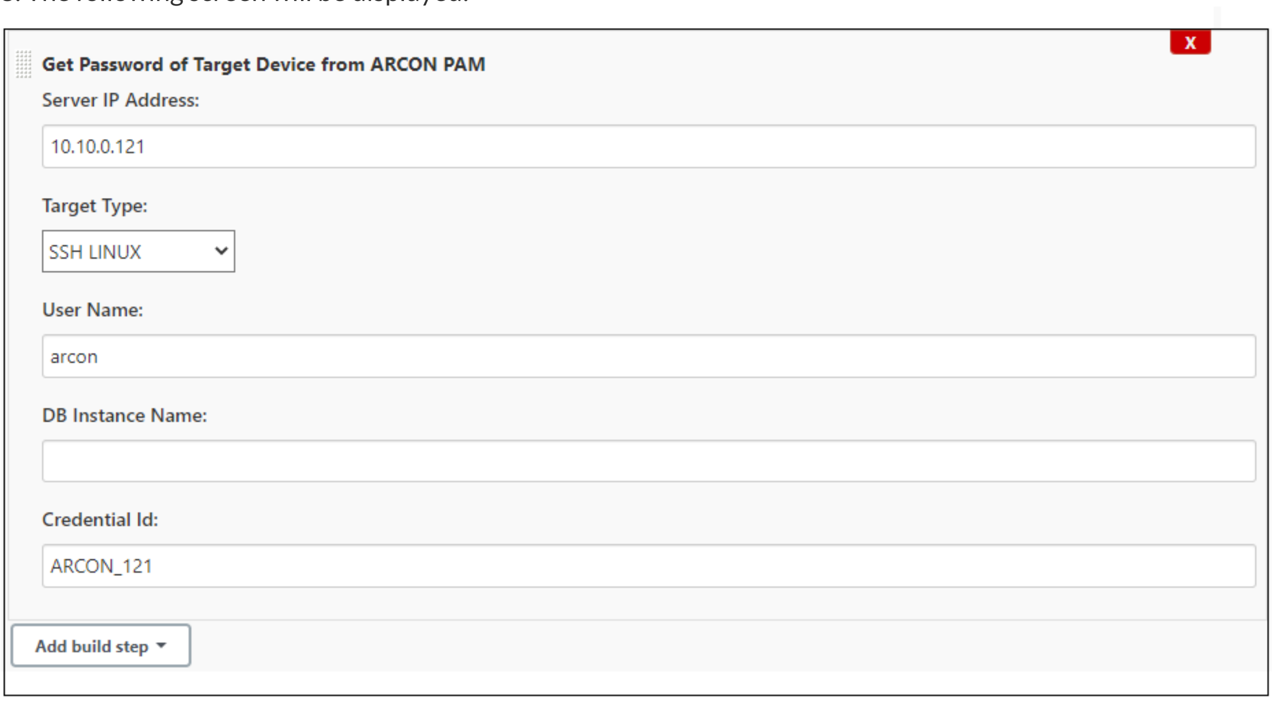
Cred Id

[redacted]

7. Add the **Get Password of Target Device from ARCON PAM** in the **Add build step** to retrieve the target device credentials from ARCON PAM.

You can retrieve passwords of multiple target devices by simply adding new build steps of **Get Password of Target Device from ARCON PAM**.

8. The following screen will be displayed:



Get Password of Target Device from ARCON PAM

Server IP Address: 10.10.0.121

Target Type: SSH LINUX

User Name: arcon

DB Instance Name:

Credential Id: ARCON_121

9. Refer to the following table to understand the field level description shown in the preceding screen:

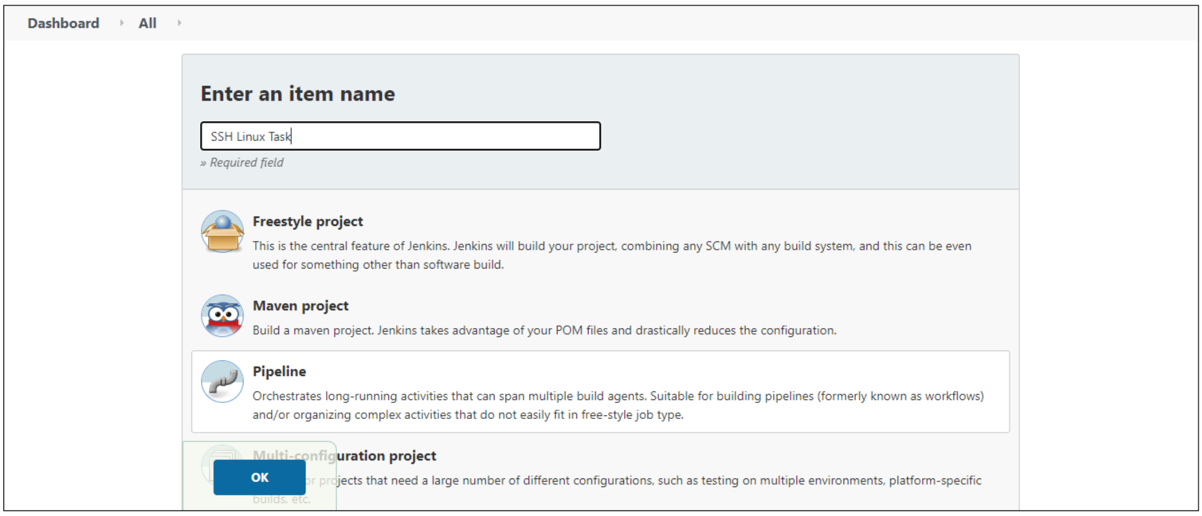
Field	Description
Server IP Address	IP of Target Service.
Target Type	Enter the type of Target Service Supported: Windows, SSH Linux, network devices, MsSql, Oracle, MySQL, SYBASE, Application, MongoDB, Postgres, DB2
User Name	Enter the Username of the Service.
DB Instance Name	Enter the instance name for Databases.
Credential ID	Specify the name of the Jenkins credential ID in which, the retrieved password will be stored and can be consumed further within Jenkins.

10. Once all the necessary fields are entered, click on **Save**.

4.3 Using SSH credentials obtained from a plugin in a pipeline script

Pipeline Job

1. Go to the **Dashboard** > **New Item** > **Pipeline**.
2. Enter an item name and click **OK** to continue.



Enter an item name

SSH Linux Task

» Required field

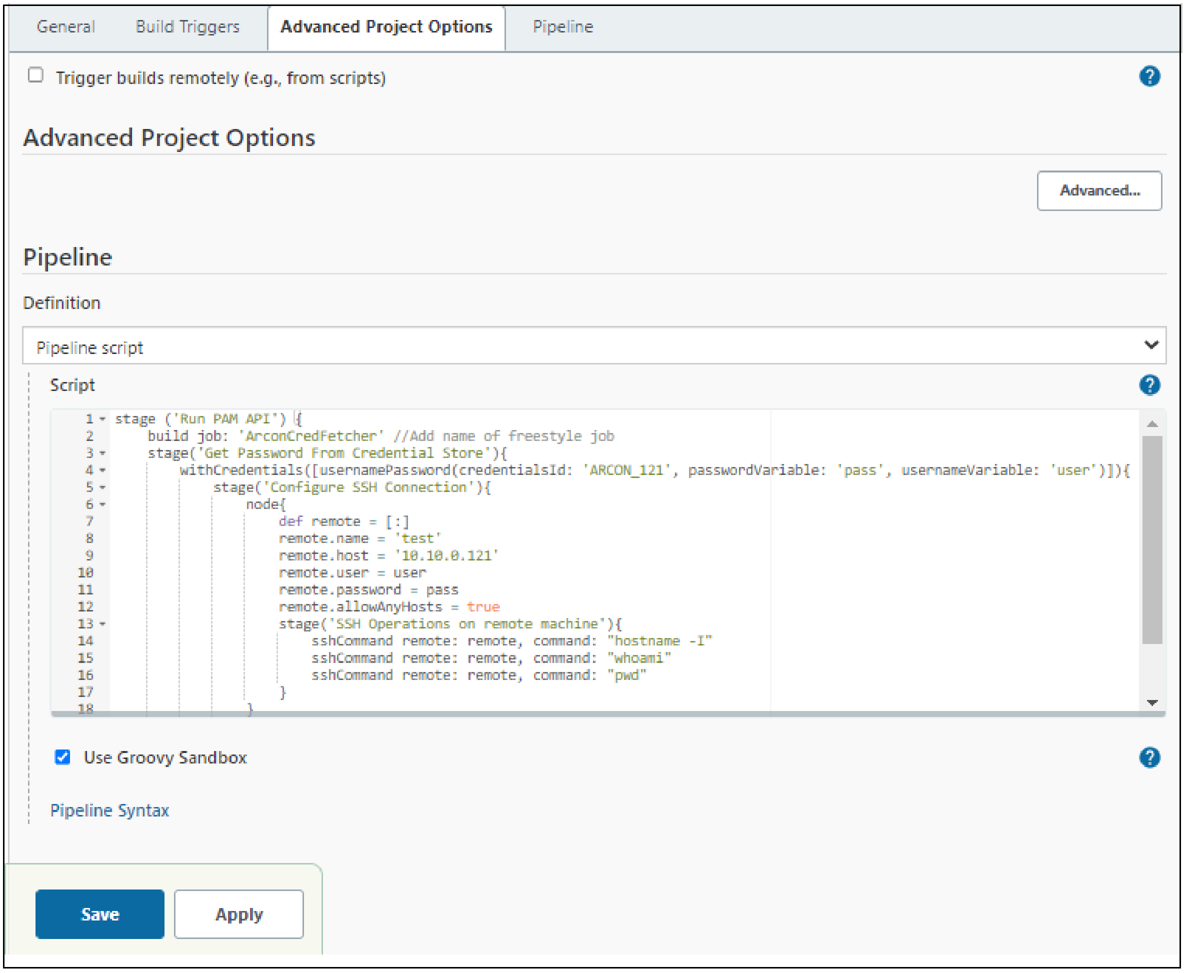
Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
...r projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

3. Run the build job where you retrieve the credentials from ARCON PAM and click **Save**.



□ Trigger builds remotely (e.g., from scripts)

Advanced Project Options

Pipeline

Definition

Pipeline script

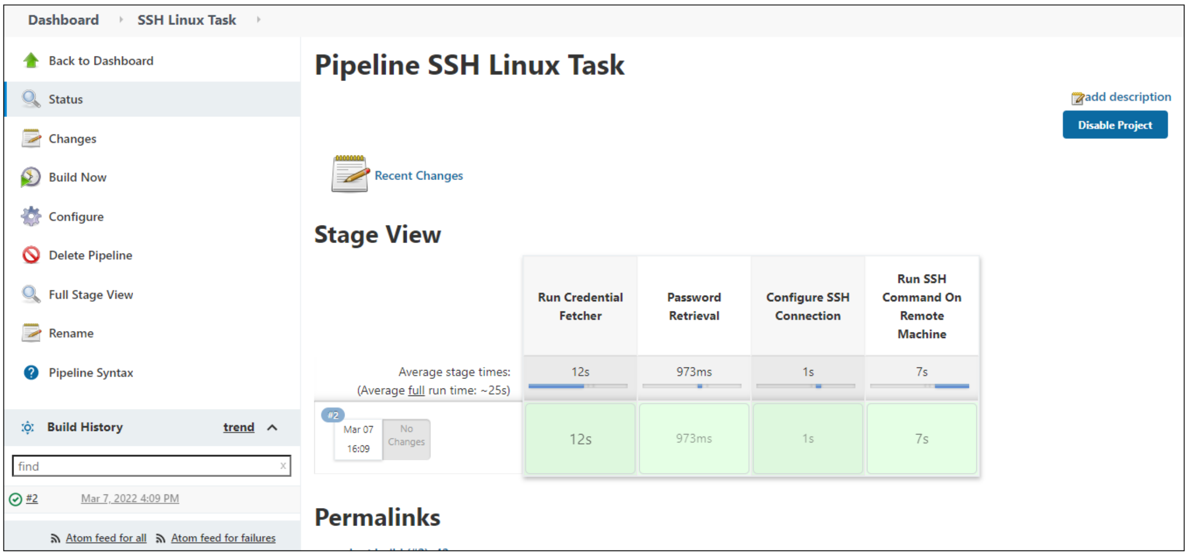
Script

```
stage ('Run PAM API') {
    build job: 'ArconCredFetcher' //Add name of freestyle job
    stage('Get Password From Credential Store'){
        withCredentials([usernamePassword(credentialsId: 'ARCON_121', passwordVariable: 'pass', usernameVariable: 'user')]){
            stage('Configure SSH Connection'){
                node{
                    def remote = [:]
                    remote.name = 'test'
                    remote.host = '10.10.0.121'
                    remote.user = user
                    remote.password = pass
                    remote.allowAnyHosts = true
                    stage('SSH Operations on remote machine'){
                        sshCommand remote: remote, command: "hostname -I"
                        sshCommand remote: remote, command: "whoami"
                        sshCommand remote: remote, command: "pwd"
                    }
                }
```

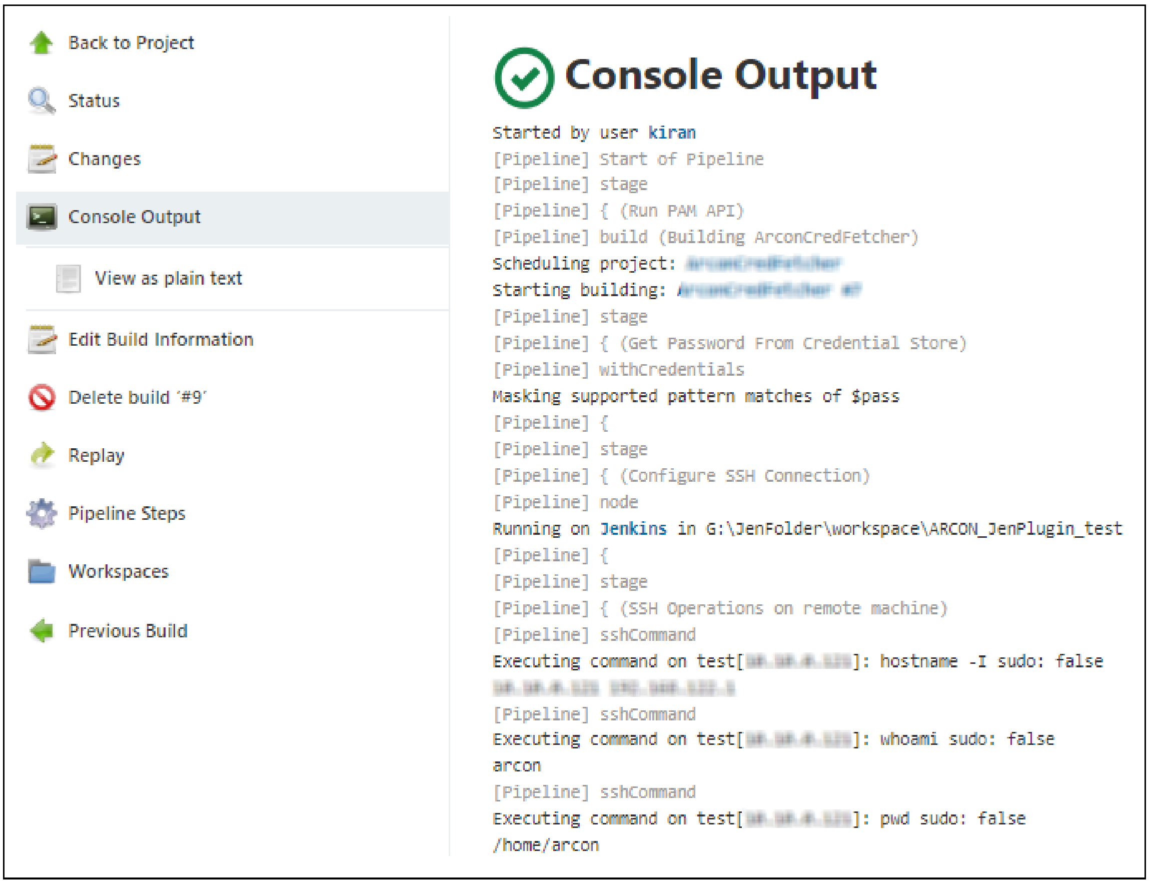
☑ Use Groovy Sandbox

Pipeline SSH Linux Task

1. To check the status and stage view, navigate to **Dashboard** > **SSH Linux Task** > **Status**. The status will be displayed:



2. To check the Console Output, navigate to **Dashboard** > **SSH Linux Task** > **Console Output**:



```
Started by user kiran
[Pipeline] Start of Pipeline
[Pipeline] stage
[Pipeline] { (Run PAM API)
[Pipeline] build (Building ArconCredFetcher)
Scheduling project: [redacted]
Starting building: [redacted]
[Pipeline] stage
[Pipeline] { (Get Password From Credential Store)
[Pipeline] withCredentials
Masking supported pattern matches of $pass
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Configure SSH Connection)
[Pipeline] node
Running on Jenkins in G:\JenFolder\workspace\ARCON_JenPlugin_test
[Pipeline] {
[Pipeline] stage
[Pipeline] { (SSH Operations on remote machine)
[Pipeline] sshCommand
Executing command on test[[redacted]]: hostname -I sudo: false
[redacted]
[Pipeline] sshCommand
Executing command on test[[redacted]]: whoami sudo: false
arcon
[Pipeline] sshCommand
Executing command on test[[redacted]]: pwd sudo: false
/home/arcon
```

*Similar Integrations are available for GitLab, Bitbucket, Chef, etc.

Privileged Access Management Suite

arcon

Predict | Protect | Prevent