General Security Settings

Overview

The JFrog Platform security model offers protection at several levels. It allows you to do the following:

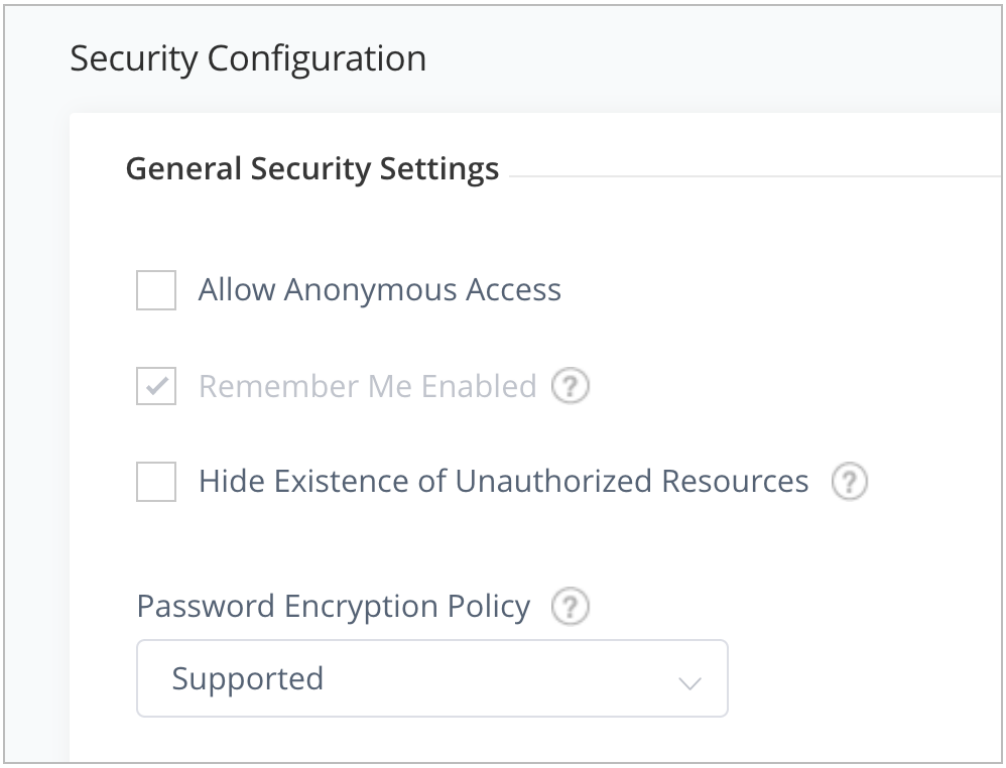
- Assign role-based or user-based permissions to areas in your repositories (called Permission Targets)
- Allow sub-administrators for Permission Targets
- Configure LDAP out-of-the-box
- Prevent clear text in Maven's `settings.xml` file
- Inspect security definitions for a single artifact or folder and more.

The security is based on Spring Security and can be extended and customized.

Page Contents

General Configuration

There are several system-wide settings to control access to different resources located under **Security | Settings** in the **Administration** module



Allow Anonymous Access

The detailed and flexible permission-based system allows you to control users' access to different features and artifacts.

You gain an additional layer of security with the concept of "Anonymous Access" which controls the features and artifacts available to a user who has not logged in. This is done through an "Anonymous User" which comes as a built-in user with a default set of permissions.

Anonymous access may be switched on (default) or off using the **Allow Anonymous Access** setting under **Security** in the **Administration** module**.** You can modify the set of permissions assigned to the "Anonymous User" just like you would for any other user, and this requires that **Allow Anonymous Access** is enabled.

Hide Existence of Unauthorized Resources

When a user tries to access a resource for which he is not authorized, the default behavior is to indicate that the resource exists but is protected.

For example, an anonymous request will result in a request for authentication (401), and a request by an unauthorized authenticated user will simply be denied (403).

You can configure to return a 404 (instead of 403) - Not Found response in these cases by setting **Hide Existence of Unauthorized Resources** under **Security | Settings** in the **Administration** module.

Alternatively, update Artifactory Configuration Descriptors#GlobalConfigurationDescriptor to set this parameter globally (for all virtual repositories) or for specific repo:

```
<security>
    <hideUnauthorizedResources>false</hideUnauthorizedResources>
</security>
<virtualRepositories>
        <virtualRepository>
                <hideUnauthorizedResources>false</hideUnauthorizedResources>
        </virtualRepository>
</virtualRepositories>
```

Password Encryption Policy

Artifactory provides a unique solution to support encrypted passwords through the **Password Encryption Policy** setting as follows:

Supported	Artifactory can receive requests with an encrypted password but will also accept requests with a non-encrypted password (default)

Unsupported	Artifactory will reject requests with encrypted password
Required	Artifactory requires an encrypted password for every authenticated request

For more details on why Artifactory allows you to enforce password encryption please refer to Centrally Secure Passwords.

Disable Basic Authentication Method

When using an external authentication method such as LDAP, SAML, etc, the basic authentication method can be disabled for internal users.

Note the following if the feature is enabled:

- Basic authentication is disabled for both UI and REST API.
- All forms of automations, scripts, plugins using basic authentication are also disabled.

Before disabling, a few steps have to be taken to ensure that the external users have the necessary permissions and prevent a system lockout.

Perform the following steps:

Step 1 You must first configure an external authentication method.

Step 2 After configuring the external authentication method, a user group needs to be created for the external users, and users need to be provided with Admin permissions. For more information, see Users and Groups.

Step 3 Once the group is created, you can choose to disable the basic authentication method.

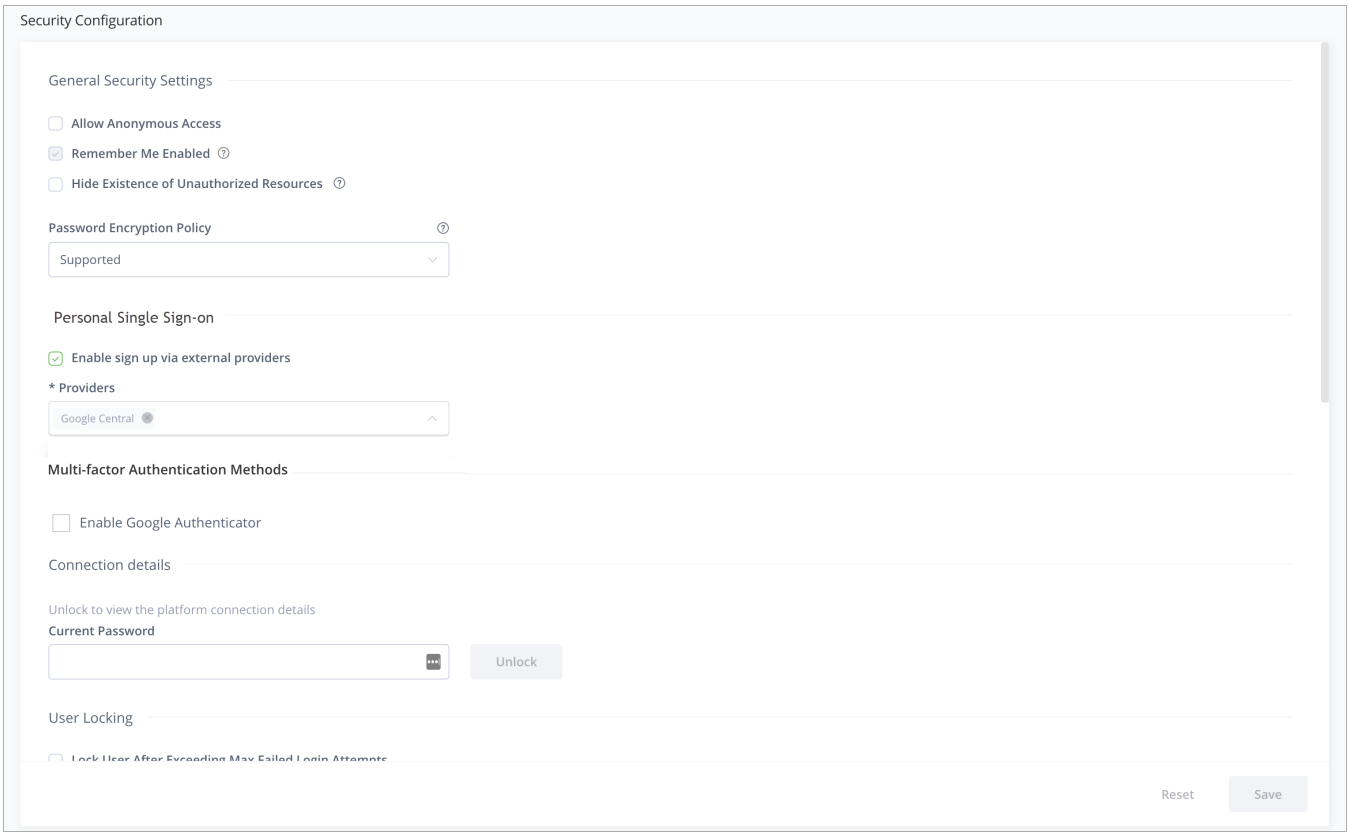
Take note, these steps are important in case an external system admin is not configured before basic authentication is disabled, the system will be left without administrators. If this occurs, and the system is locked out contact JFrog support for assistance.

Multi-factor Authentication

Multi-factor authentication (MFA) enables a higher level of security when accessing JFrog applications. When enabled, in addition to their user credentials, users will have to authenticate with a one-time password (OTP) generated by an additional authentication factor - the Google Authenticator application. This ensures that in the case where a users' credentials have been compromised, the multi-factor authentication method will prevent malicious users from gaining access to JFrog applications.

Enabling Multi-factor Authentication

To enable multi-factor authentication for users in the JFrog Platform, navigate to **Administration module | Security | Settings | Multi-factor Authentication |** and check the **Enable Google Authenticator** checkbox.



Multi-factor authentication is applied to all users of the JFrog Platform application.

Follow the steps of logging in using MFA after it is enabled.

Resetting Enrollment

In the case a user's ability to log in using multi-factor authentication is compromised, an Administrator should reset the user's enrollment status by performing the following steps:

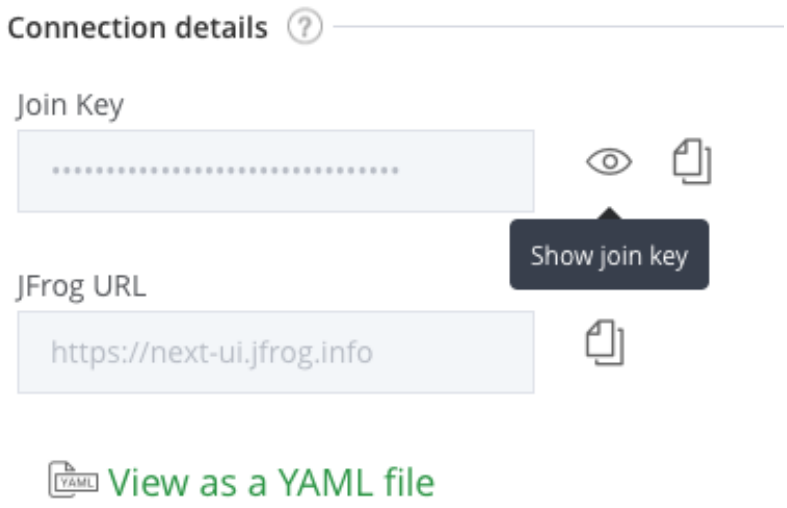
1. Navigate to **Administration module | Identify and Access | Users**
2. Select the specific user and in the **Edit** screen, select **Reset MFA Enrollment.**
   The user will have to follow the enrollment steps again to authenticate.

If an Administrator loses the capability of signing in using the multi-factor authentication method, the Administrator should bootstrap a new Administrator or an existing Administrator with new credentials to reset the multi-factor authentication for the bootstrapped Administrator. For more information see, Recreating the Default Admin User.

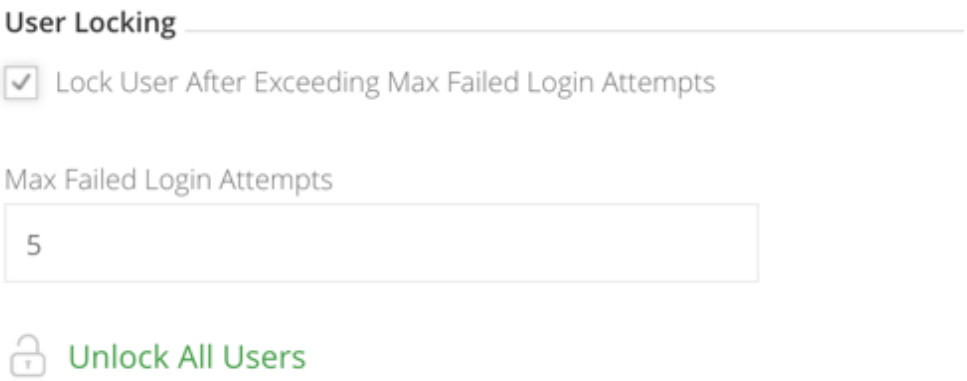
Viewing the Join Key

JFrog join.key feature establishes trust between the JFrog services based on the AES-128 bit symmetric encryption. This feature is an alternative to the basic authentication trust method. For more information, see Creating Trust between Services.

You can view the JPD join key.



User Lock and Login Suspension



User account locking and temporary login suspension are two mechanisms employed to prevent identity theft via brute force attack.

Temporary Login Suspension

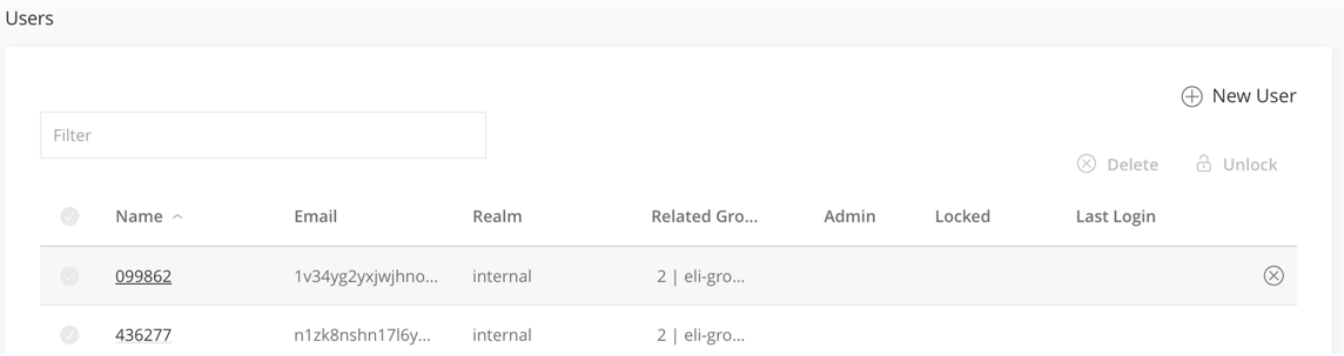
Temporary login suspension means that when a login attempt fails due to incorrect authentication credentials being used, the system will temporarily suspend that user's account for a brief period of time during which additional login attempts will be ignored. If login attempts fail repeatedly, the suspension period is increased each time until it reaches a maximum of 60 seconds.

User Account Locking

In addition to temporary login suspension, you can lock a user's account after a specified number of failed login attempts. This is enabled by selecting the "Lock User After Exceeding Max Failed Login Attempts" check box, and specifying the **Max Failed Login Attempts** field. Users who get locked out of their account because they have exceeded the maximum number of failed login attempts allowed (as specified in **Max Failed Login Attempts)** must have an administrator access to unlock their account.

Unlocking User Accounts

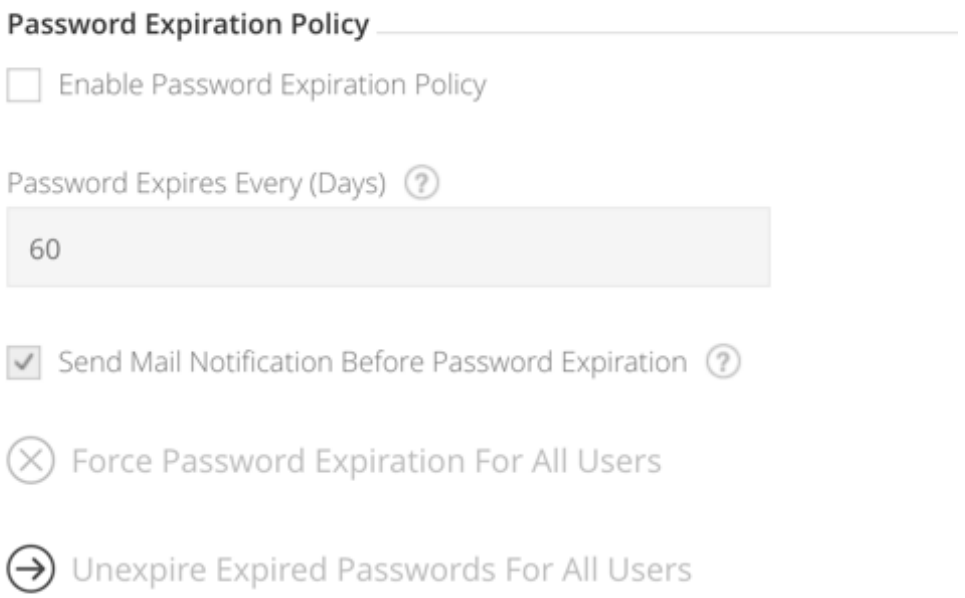
An administrator can unlock all locked-out users by clicking **Unlock All Users** in the **Security | Settings** page where user locking is configured. An administrator can also unlock a specific user or a group of users in **Identity and Access | Users**.



Through the REST API, an administrator can unlock a single user, a group of users, or all locked-out users at once.

Password Expiration Policy

An admin user can enforce a password expiration policy to force all users to change their passwords at regular intervals. When the password expiration policy is enforced, users who are not within the specified time interval will have their accounts locked until they change their password.



Enable Password Expiration Policy	When selected, password expiration policy is enabled.
Password Expires Every (Days)	Specifies how frequently all users must change their password.
Send Mail Notification Before Password Expiration	When selected, users receive an email notification a few days before their password expires.
Force Password Expiration For All Users	Forces all passwords to expire. All users will have to change their password at next login.
Unexpire Expired Password for All Users	Removes the password expiry status for all users

---

Managing API Keys

As an admin user, you can revoke all the API keys currently defined in the system under **Artifactory** | **Security | Settings** in the **Administration** module.

To revoke all API keys in the system, click **Remove API Keys for All Users**.

> Once you revoke an API key, any REST API calls using that API key will no longer work. The user will have to create new API key and update any scripts that use it.

Passwords Encryption

Different configuration files may include password information stored in plain text.

To keep passwords secure, you may choose to encrypt them as described in Key Encryption.

Hardening Security for Secrets

A a set of encrypted parameters (secrets) is used to connect to external resources such as the different databases it uses. While these secrets may be stored in the configuration file, this poses a risk of their being exposed.

To keep secrets safe from exposure, you may pre-load secrets from a temporary file when you startup the system. Once the system has read and successfully used the secrets, the file is deleted.

The snippet below shows an example of the parameters you could include in this temporary file. These are the parameters Artifactory uses to connect to a PostgreSQL database.

```
type=postgresql
driver=org.postgresql.Driver
url=jdbc:postgresql://postgresql:5432/artifactory
username=artifactory
password=JE2cyPQtEmJovMbxwEGrghre9EXcu4ANtTtPu9Lk3s15UPs73M
```

While we recommend only including sensitive information such as encrypted connection strings, this file may contain any of the database configuration parameters, and any parameters specified (including environment variables and system properties) will override the corresponding ones in the database configuration file.

To load parameters using this mechanism, place them in the following temporary file before your startup Artifactory:

`$JFROG_HOME/artifactory/var/etc/artifactory/.secrets/.temp.db.properties`

> **Execute on every restart of Artifactory**
>
> Since the temporary file is deleted when Artifactory starts, you need to replace the temporary file each time you restart Artifactory.