Multi-Factor Authentication

**Overview**

Passwords are presently the primary method by which users authenticate themselves to computer systems. However, passwords are proving less and less capable of protecting systems from abuse. Stronger methods, notably multifactor authentication (MFA)—which combines something you know (e.g., a personal identification number, or PIN), something you have (e.g., a token), and/or something you are (e.g., a fingerprint)—are increasingly required.

Authentication in cyberspace is the process of verifying user identity prior to granting access to specific computer, network, or Internet services and resources. The user password is the form of authentication that remains the primary means of user identification. Passwords can be very convenient, requiring little more than memory and typing to apply them. Yet, as nearly every computer and security professional will attest, passwords are a notoriously weak form of authentication, they can be compromised at any point in the authentication process. Since passwords alone no longer provide adequate authentication for many types of information, the use of multiple factors for network access might be recommended. The benefits of multifactor authentication are that hackers (or insiders) have to break (that is, gain unauthorized access to systems protected by) not one but many authentication devices.

Requiring re-authentication before performing high security actions further limits the damage an attacker can do even if they manage to compromise a login session.

**What is Multi -Factor Authentication?**

Multi-factor authentication allows you to add additional credentials to your account to make it more secure. Multi -factor authentication is an authentication method that requires two or more independent pieces of information to establish identity and privileges. Two-factor authentication is stronger and more rigorous than traditional password authentication that only requires one factor (the user’s password).

**Benefits**

• Greatly enhances security by requiring two or more independent pieces of information for authentication

• Reduces the risk posed by weak user passwords that are easily cracked

**User Prerequisites**

User account with secret key should be created.

**How Does Two-Factor Authentication Work?**

The authentication service consists of two components:

* Generation of temporary token
* Authentication with secret key and temporary token

With two-factor authentication, users must enter a valid temporary passcode to gain access. A passcode consists of the following:

* The user’s personal key
* A temporary token generated from TokenGenerator

If the key is correct and the token is correct and current, the user is authenticated.

**Generation of temporary token:**

**Step1:** Login to TokenGenerator web application with user credentials

When a user tries to login, the spring security checks the configuration file for the user credentials, if matched, the user is authenticated, else, the application prompts an error message.

**Step2:** Enter key and click generate button

On clicking the generate button, application loads the user's logged in secret key from a stored text file() and encrypt it with the user entered key using AES encryption algorithm which returns an encrypted value as the token, this generated token with a current timestamp is stored in another text file(generatedtokens.txt) and is returned to the user.

**Authentication:**

Enter userName, password and passcode into MultiFactorAuthentication web application's login form

The spring security checks the configuration file for user credentials, if they don't match, the system prompts an error message saying "Invalid Username/Password", if matched, application loads the generated token details from the text file(generatedtokens.txt) which was created in the previous step(token generator), validates token generated timestamp with the current time. If the time difference is greater than 30 seconds, then token expires, displays an error message saying "Token Expired". If the time difference is less than 30 seconds, then the application will decrypt user entered token with user entered key using AES encryption algorithm and loads the user secret key from text file(usersecretkey.txt), then validates it against decrypted token, if matched, user is authenticated, else, application prompts an error message saying "Invalid Token".