Overview

using two or more independent authentication keys. Usually one of those is a password assigned when creating the account. it is a one time code, even if it is captured, it cannot be re-used.

The idea is to use a series of pseudo random numbers derived from a seed value. When you log in you fetch one of the values in the series. The one you fetch is a function of the current time and the seed value. You use that value for the the second MFA authentication key.

Practical matters

When you are first negotiating with the remote party what you will use to generate the :abbr:`TOTP (Time based One Time Password)`. You need to agree on a seed value. This is a secret held by the two parties. If you lose that, you can no longer supply the :abbr:`TOTP (Time based One Time Password)`. So it is important to take care to back up that piece of information securely.

Another factor is timing. The two parties need to be synchronized with respect to time otherwise the :abbr:`TOTP (Time based One Time Password)` will come out different. Failures can be due to corruption of seed value or due to times being out of sync.

Implementations I have tried successfully:

Ente Auth

This is an app that encapsulates :abbr:`TOTP (Time based One Time Password)` (RFC 6238). It is available under Ubuntu 24.04 Linux.

KeePassXC

This is a key database application that I have used extensively.

It includes good functionality for maintaining :abbr:`TOTP (Time based One Time Password)` secrets alongside your normal passords. It can generate one time codes for accomplishing MFA logins. It can generate QR codes so you can easilly share the secret with apps on other devices.

You just right click on your password entry and select what you want.

keepassxc.org

Aegis Authenticator

This an app available for android devices which I have tried and found that its one time codes agree with the other two apps suggested above. Aegis Authenticor jurisdictional [1] points seem acceptable.

Aegis Authenticator google play app

Site for checking TOTP codes

You put in your secret key and press the save button. It emites one time codes as time ticks away. Its output should match the tool you are using.

TOTP test site

1 **Aegis Authenticator Jurisd]iction and Data Privacy** : Aegis does not appear to be operated by a traditional corporation based in a single jurisdiction, but rather maintained by developers on GitHub.

Local Storage: The app operates entirely offline and does not

require an internet connection, meaning tokens are stored locally on the user's device.

Data Collection: Aegis does not collect, transmit, or store any

personal data.

Safety Audit: Being open source, the codebase is publicly

auditable.