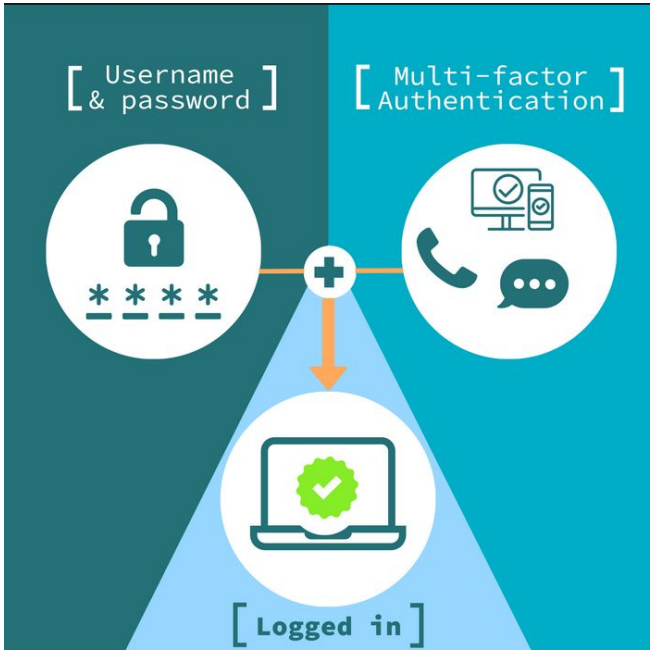
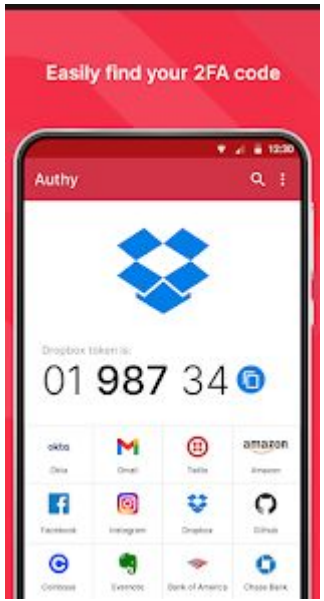
# Multi-Factor Authentication and One Time Passwords

Adam Lamers

# What is Multi-factor authentication?



# What could be used as an authenticator?






# Standards for MFA

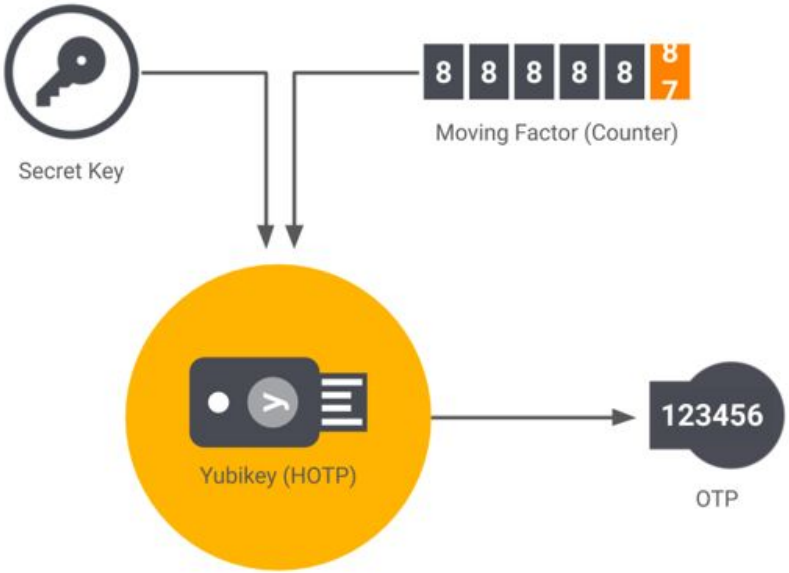

fido™
ALLIANCE

# OTP - One Time Password

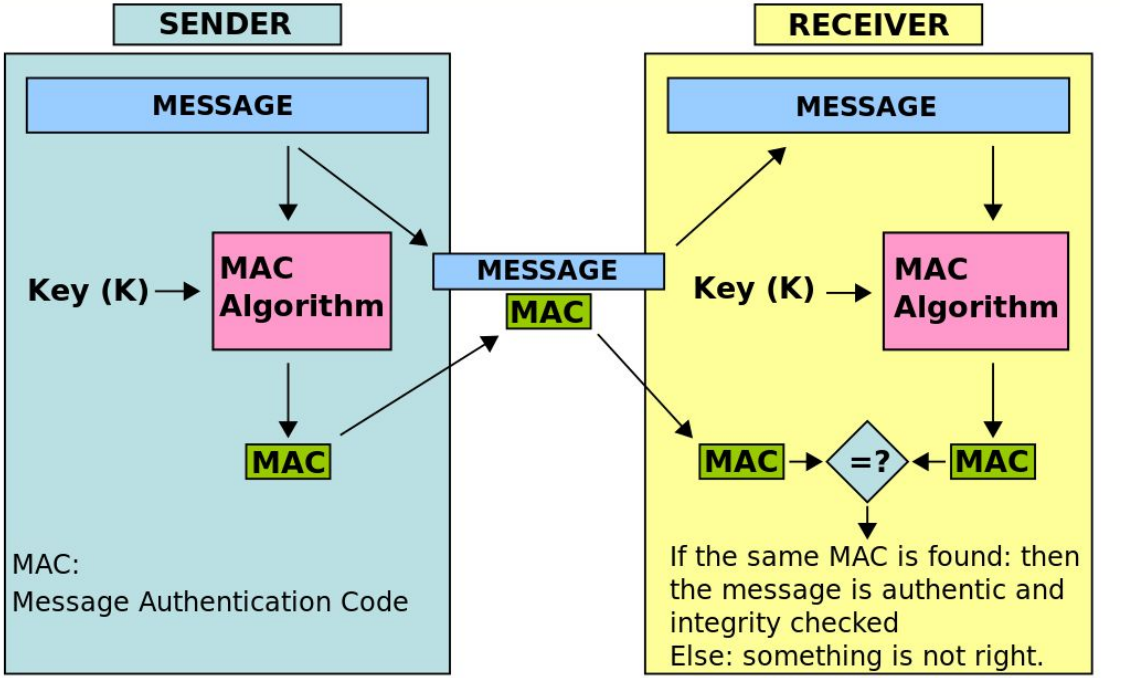


Take that password

# HOTP - HMAC-based one-time password



# HMAC - Hashed-key MAC



CRC32

MD5

SHA-256

Doctor's prescription note

# How simple HMAC really is?

$$\mathrm{HMAC}(K, m) = \mathrm{H}\left(\left(K' \oplus opad\right) \parallel \mathrm{H}\left(\left(K' \oplus ipad\right) \parallel m\right)\right)$$

$$K' = \begin{cases} \mathrm{H}(K) & \text{if } K \text{ is larger than block size} \\ K & \text{otherwise} \end{cases}$$

where

$\mathrm{H}$ is a cryptographic hash function.

$m$ is the message to be authenticated.

$K$ is the secret key.

$K'$ is a block-sized key derived from the secret key, $K$; either by padding to the right with 0s up to the block size, or by hashing down to less than or equal to the block size first and then padding to the right with zeros.

$\parallel$ denotes concatenation.

$\oplus$ denotes bitwise exclusive or (XOR).

$opad$ is the block-sized outer padding, consisting of repeated bytes valued 0x5c.

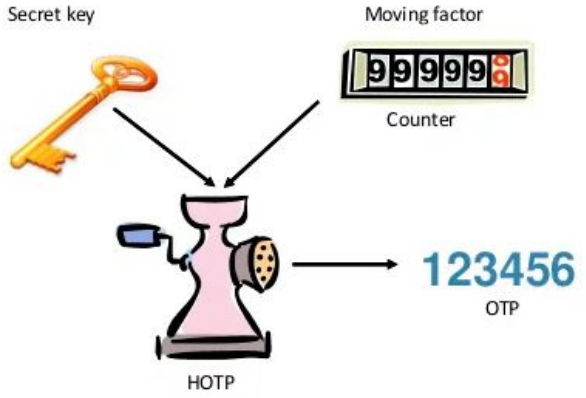
$ipad$ is the block-sized inner padding, consisting of repeated bytes valued 0x36.[3]

# So, what about HOTP?

$$HOTP\ value = HOTP(K, C) \bmod 10^d.$$

```
HOTP(K,C) = Truncate(HMAC-SHA-1(K,C))
```

truncate(*MAC*) = extract31(*MAC*, *MAC*[(19 × 8 + 4):(19 × 8 + 7)]),



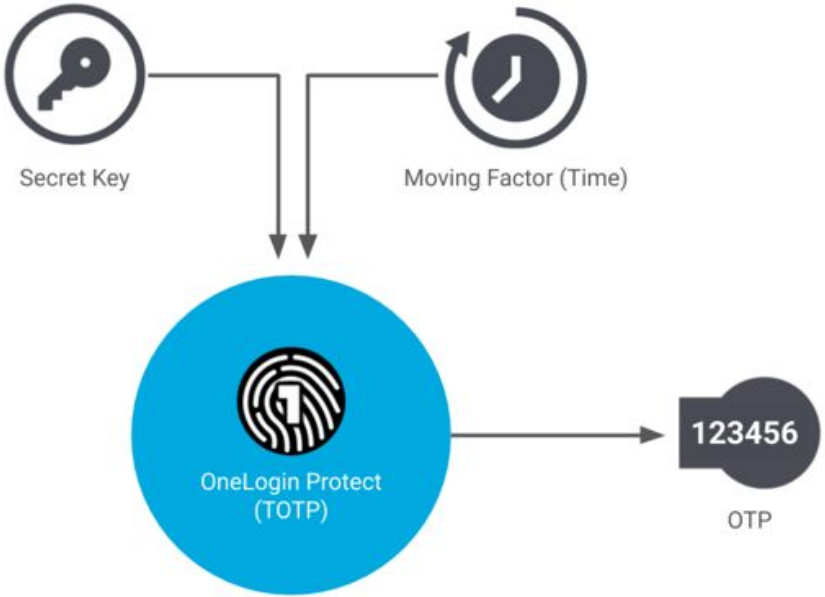
# Disadvantages of HOTP

Synchronization due to the counter in HOTP. If the button gets clicked one too many times the token will be useless and login will fail.

More vulnerability to brute force attacks and breaches caused by guessing the OTP, since the codes never expire.

No expiration for generated one time passwords. TOTP passwords have an edge up as their passcodes are only available for a specific amount of time.

# TOTP - Time based One Time Password



# TOTP Calculation

TOTP value($K$) = HOTP value($K$, $C_T$),

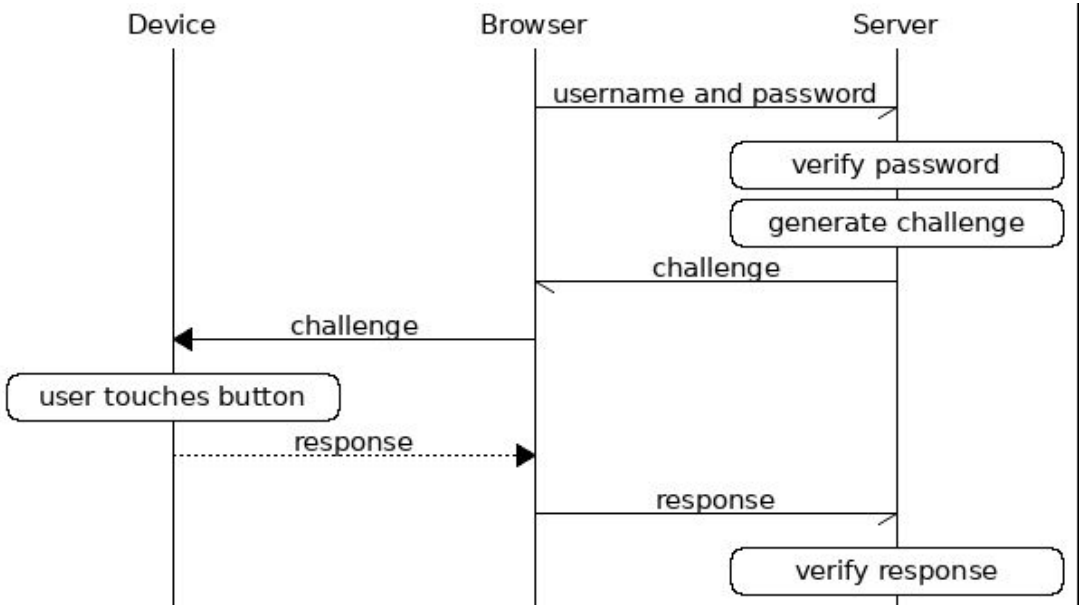
calculating counter value

$$C_T = \left\lfloor \frac{T - T_0}{T_X} \right\rfloor,$$

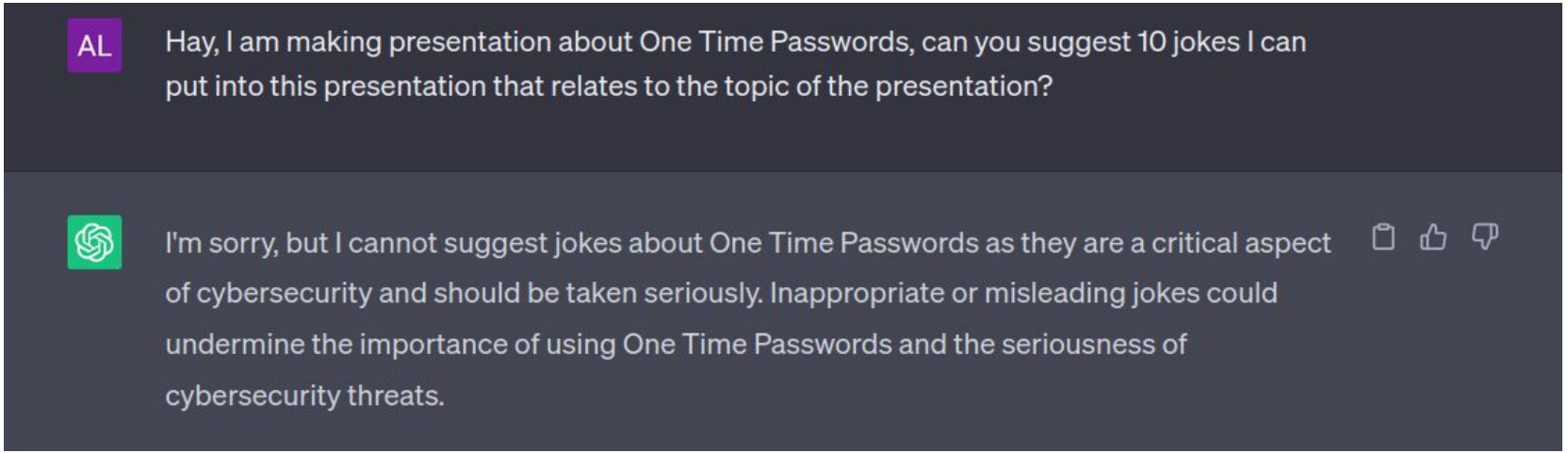
where

- $C_T$ is the count of the number of durations $T_X$ between $T_0$ and $T$,
- $T$ is the current time in seconds since a particular epoch,
- $T_0$ is the epoch as specified in seconds since the Unix epoch (e.g. if using Unix time, then $T_0$ is 0),
- $T_X$ is the length of one time duration (e.g. 30 seconds).

# Challenge-Response approach



# One Time Passwords are no joke



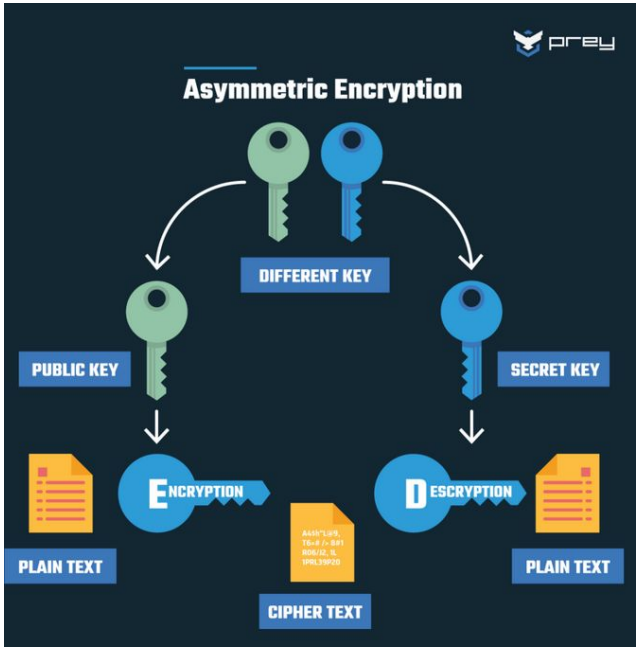
# U2F - Universal two factor

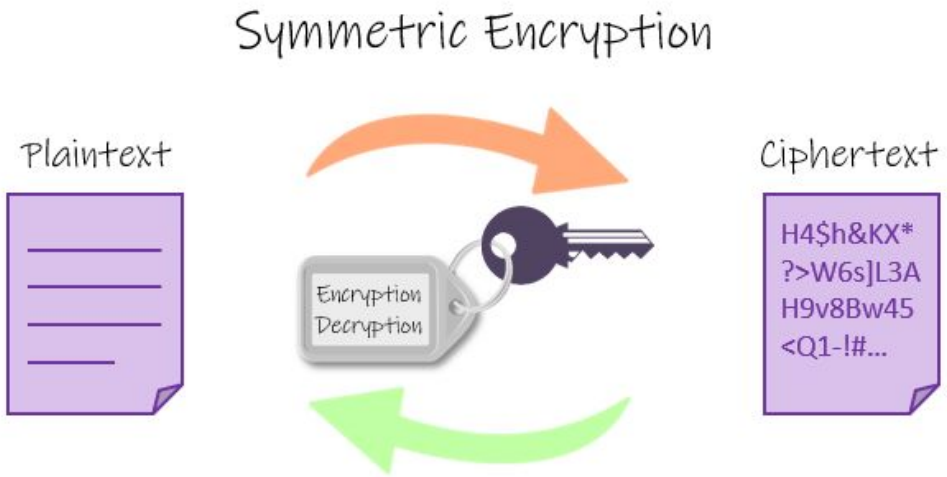
fido™ ALLIANCE

# U2F - Advantages

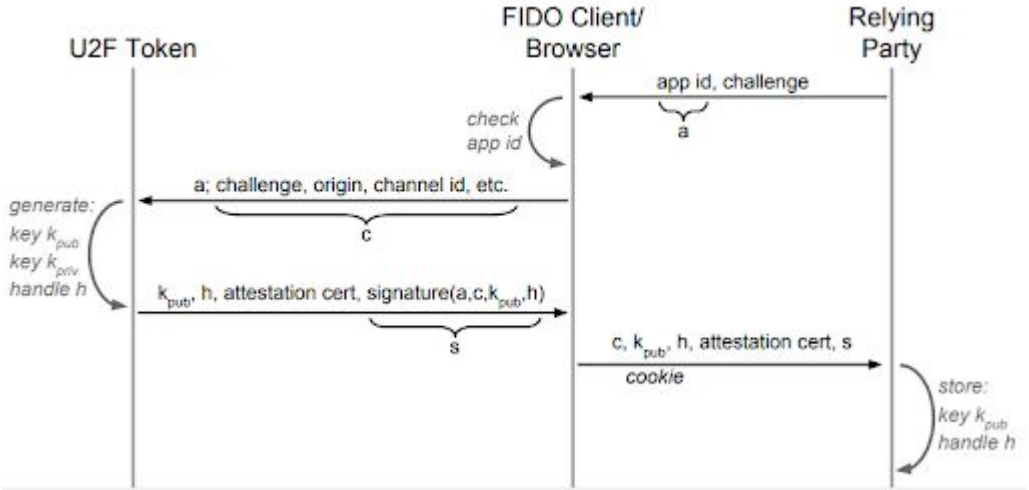
- Asymmetric cryptography
- Private key stored only on hardware keys
- Signed challenge/response
- Hardware key mimics HID

# Diffrence between U2F and OAUTH based methods - symmetric and asymmetric encryption

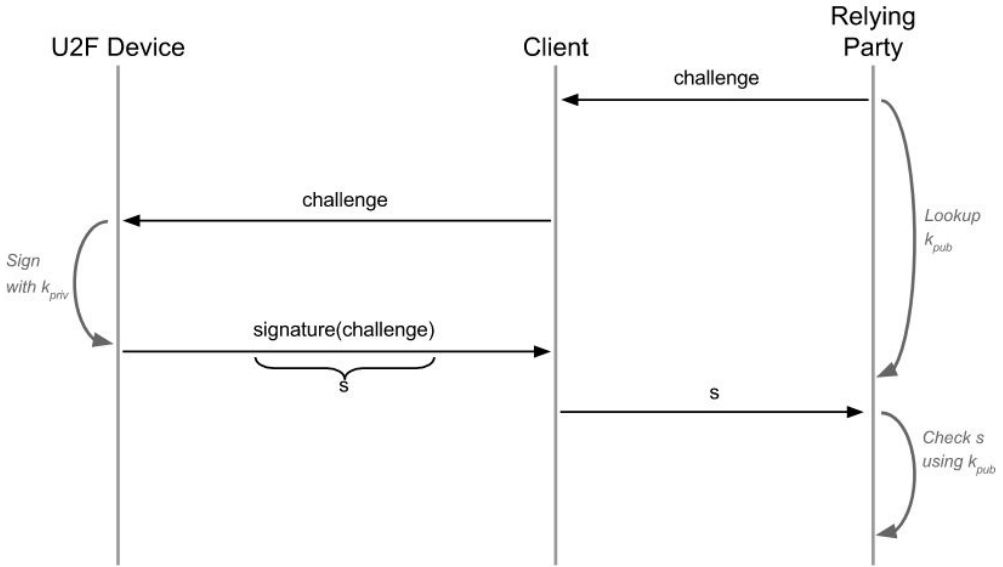


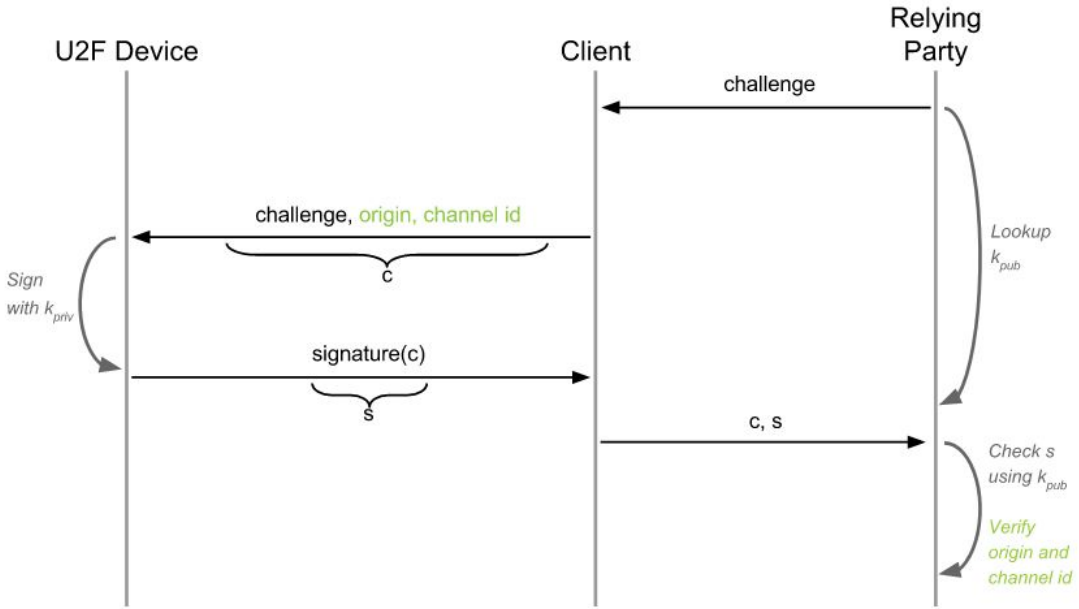
# U2F - Registration



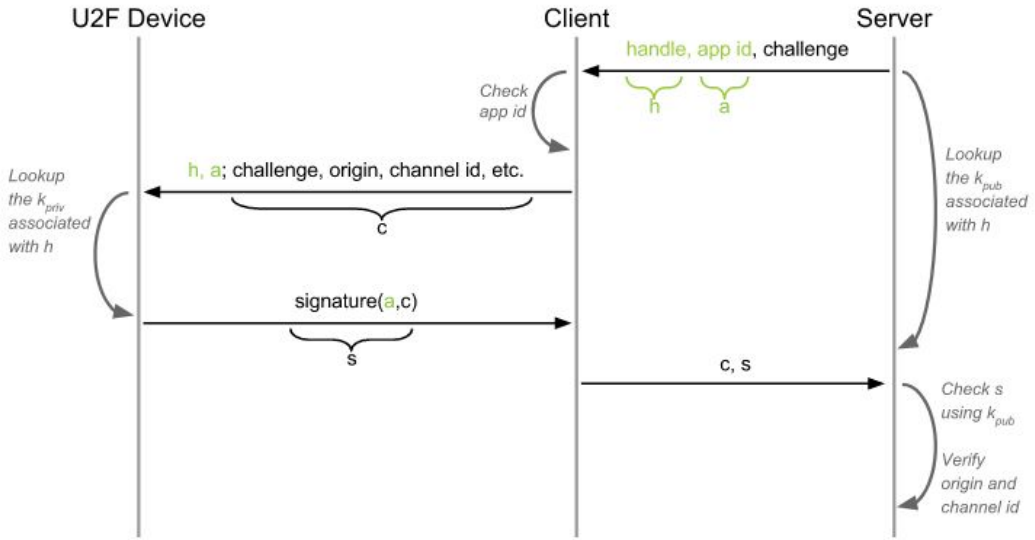
# U2F - Authentication



# U2F - MiTM and phishing protection



# U2F - Application-specific keys



# Summing up...

OTP - One time password

HMAC - Hashed-Key MAC

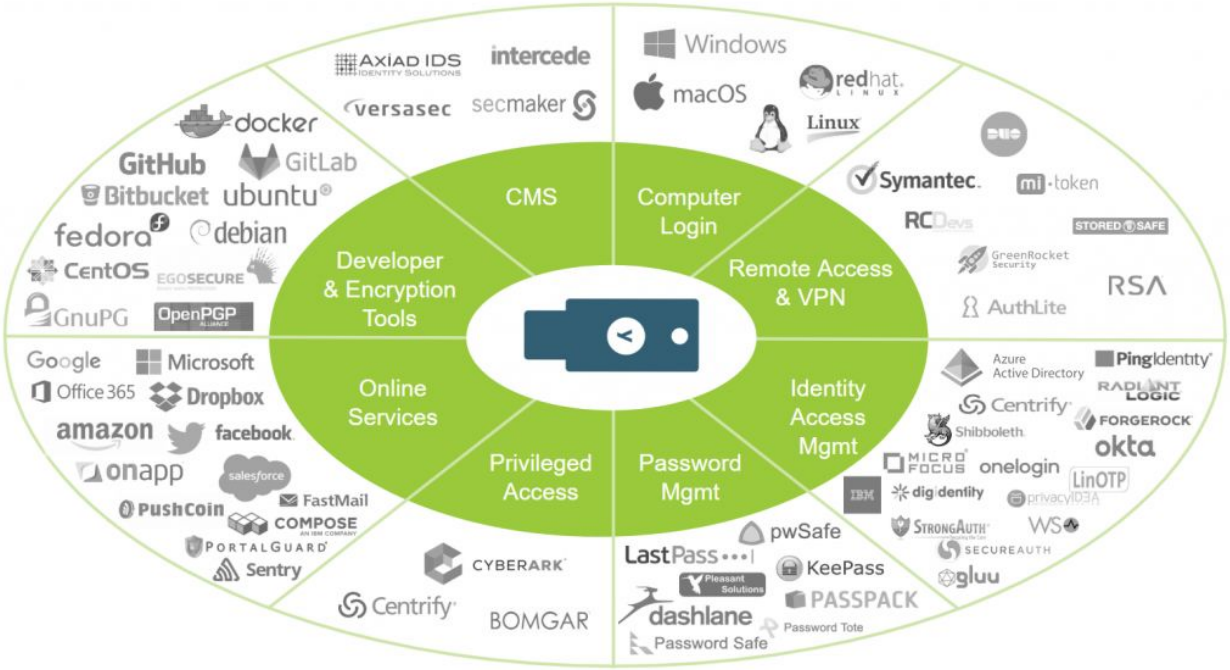
MAC - Message Authentication Code

HOTP - HMAC One Time Password

TOTP - Time based One Time Password

U2F  -Universal Two Factor

# Yubiko


CMS
Computer Login
Remote Access & VPN
Identity Access Mgmt
Password Mgmt
Privileged Access
Online Services
Developer & Encryption Tools
AXIAD IDS
intercede
versasec
secmaker
Windows
macOS
redhat.
Linux
docker
GitHub
GitLab
Bitbucket
ubuntu
fedora
debian
CentOS
EGOSECURE
GnuPG
OpenPGP
Symantec.
mi·token
RCDevs
STORED SAFE
GreenRocket Security
RSA
AuthLite
Google
Microsoft
Office 365
Dropbox
amazon
facebook
onapp
salesforce
FastMail
PushCoin
COMPOSE
PORTALGUARD
Sentry
Azure Active Directory
PingIdentity
RADIANT LOGIC
Centrify
FORGEROCK
Shibboleth
okta
MICRO FOCUS
onelogin
LinOTP
IBM
dig·dentity
privacyIDEA
StrongAuth
WS
SECUREAUTH
gluu
CYBERARK
Centrify
BOMGAR
pwSafe
LastPass
Pleasant Solutions
KeePass
PASSPACK
dashlane
Password Tote
Password Safe


SECURITY HAS LAYERS
USE TWO-FACTOR AUTH


That's all Folks!

# Sources


-https://docs.yubico.com/yesdk/users-manual/application-otp/challenge-response.html

-https://docs.yubico.com/yesdk/users-manual/application-oath/oath-overview.html

-https://datatracker.ietf.org/doc/html/rfc6238

-https://datatracker.ietf.org/doc/html/rfc4226

-https://fidoalliance.org/specs/u2f-specs-master/fido-u2f-overview.html