SSH Certificates in a Federated World

Tangui Coulouarn tangui.coulouarn@deic.dk
Mads Freek Petersen freek@wayf.dk

Typical flow with SSH (with public key)

- Generate a pair of keys
- Somehow find a way to tell the server what your public key is

The Problem (s)

- Key distribution, or how to tell the SSH server which users (with matching SSH public keys) to accept?
- How to revoke access?
- How to scale up / work with users from multiple origins?
- How to to tell users what the server(s)’ SSH host key(s) is/are to not rely on TOFU?

X.509 certificates
!=
SSH certificates

Not Invented Here!

- SmallStep - smallstep.com ("If you're not using SSH certificates you're doing SSH wrong")
- Teleport - goteleport.com
- HashiCorp Vault - www.vaultproject.io

Except for this using only standard ssh clients and servers

SSH certificates
!=
X-509 certificates

Agenda

- SSH Certificates 101
- What is needed
    - A SSH certificate authority
    - On a SSH server
    - On a SSH client

SSH Certificates 101

- A SSH certificate is a structure which contains a public key and some additional information signed by a SSH CA encoded according to rfc4251
- 2 types - user and host
- Only 1 level - i.e. only “root” keys that signs certificates
- Additional information
    - Principals - user names or host domain names
    - Validity period
    - Critical options
    - Extensions

SSH Certificate Authority

- A SSH Certificate Authority issue certificates based on
  - Knowledge of the user (principal / Key ID)
  - Policy (valid from - to)
  - Policy (extensions, critical options)

- The POC server is a go based http- and sshserver.

SSH server

- A SSH server trusts a SSH CA by
  - Adding it’s public key to the list of trusted SSH CAs in sshd_config:

```
TrustedUserCAKeys /path/to/file/with/list/of/public/keys/for/trusted/CAs
AuthorizedKeysFile none
```

SSH Client

This slide intentionally left blank

DEMO!

Step-by-step

- Go to the SSH CA webpage
- Login with your federated identity
- The SSH CA receives an assertion from your IdP
- The SSH CA creates a unix username from your eduPersonPrincipalName
- The SSH CA creates a token and uses that as a key to save your username in temporal map
- THE SSH CA creates a ssh command - with the token
- You send the token using ssh to the SSH CAs ssh backend to let it create a SSH certificate based on your username and your public ssh key - that it gets via the ssh “login”
- The SSH CAs ssh backend writes the textual representation of the certificate to stdout so that it is available on your client
- The actual command redirects the output from the SSH CA to the certificate file
- You can now login to ssh servers that trust the SSH CA with the username in the certificate

We have created you as user:

madpe_dtu_dk@sshserver.lan

Go to

https://sshca.lan

to create a certificate

Not shown today

- Auto user generating and updating based on xtra information in the certificate
- A tiny sh client script that automates the pasting
- Host certificates