Fed’ing SSH: Some Recent Work

- Outline: Problems – Solutions – SSH certs – DeiC’s solution
- SSH is here to stay – ***how to leverage web-based fed IDs there?*** Gap.
- T&I Incubator aut22: Workshops (RI and webfed people) identified ***actual*** problems and ***existing*** solutions in this space. Some findings:
  - SIs *widely* ‘borrow’ PIs’ private keys to avoid too-cumbersome **onboarding**.
  - Non-trivial for admins to keep track of what public keys to **offboard** when.
  - SSH and webfed largely separate communities, co-op. would be beneficial:
    - ***Expanding scope of webfed*** + solving such SSH security, usability and ***scalability*** issues.
  - Solutions for web fed’ing SSH already exist, both $ and **community OS**.
- OS solution teams formed group, agreed on co-op’ing to improve.

Mikkel Hald, DeiC; mikkel.hald@deic.dk

6 Community OS Solutions for Federating SSH

	DAASI FedSSH	DEIC SSH certs	KIT SSH OIDC	SURF PAM WebSSO	JISC Moonshot	STFC SSH OIDC
Key sharing mitigated?	✅	✅	✅	✅	✅	
Client requirements	Vanilla	Vanilla	mccli+oidc-agent	Vanilla	Moonshot	
Server requirements	Smart shell	Vanilla	PAM module+MC	PAM module	Moonshot	
Supported platforms	Interactive	All	All	Interactive	All	
Delegation	✅	✅	✅	❌	✅	
Provisioning	Possible	Possible	✅	❌	✅	
Revocation	✅	Short TTL	✅	✅	✅	
MFA possible?	✅	✅	✅	✅	✅	

Leveraging **Std. SSH** to the full: What ***SSH certs*** are and what they can do

- Like X509: a pubkey + extra info (nbl. expiry), signed by a trusted CA.
- ***Eliminate*** (poorly scaling) per-user pubkeys management on server:
  - User logs in presenting a SSH cert; server trusts its pubkey if signed by CA.
  - Server only needs the pubkeys of trusted CAs (may trust more than 1). **Easy.**
  - SSH certs contain expiry (set by CA), i.e. are ***auto-expiring per-user pubkeys***.
  - User ID part of cert – so trivial coupling of SSH session and user ID.
- Convey user ID and rights to SSH server ***front-channel***:
  - If user info in cert, no need for backend integrations; ***easy mlple-orgs sharing.***
  - Srv. could JIT-update (incl. create) local user account from cert per SSH login.
- Easy for CAs to issue ***based on a web SSO token*** (next slide):
  - Essentially converting a web token to a SSH holder-of-key token (the cert).
- Part of ***standard SSH*** server and client software for 10+ years.

Webfed'ing it: DeiC's SSH CA

- If user has no valid SSH cert in her terminal, she needs to visit the CA:
    - ***The CA is an OIDC RP*** – she logs in using her federated institutional account.
    - The CA in the browser generates ***a terminal ssh command with a token*** in it.
    - User copies and executes command in terminal, thereby retrieving cert from CA containing expiry, user ID and perhaps VO group info and info from OIDC (or SAML) token (**e.g. assurance**). CA issued cert on pubkey revealed to it in user's ssh call.
- Some benefits easily achieved with SSH certs and CA – in summary:
    - ***No special client-side requirements*** but term+ssh-client (scalability, usability).
    - No need for other credentials than user's institutional login (scale, usability).
    - SSH ***access tied to institutional web credentials*** – far less likely shared by PIs than private keys (security).
    - ***No per-user pubkeys on server*** (security, scalability); negligible sshd config.
    - No need for VO backend(s); easy sharing of srv. among orgs (CAs) (scalability).
    - SSH server admin offloads IDM to IdP, AuthZ to fed VOs (the CAs) (scalability).

Warning: NO DEMO

Time for Questions and Discussion

- Open-source SSH teams group website:
  - https://github.com/FederatedSSH
- DeiC's SSH CA on GitHub:
  - https://github.com/wayf-dk/ssh-certs-in-a-federated-world
- DeiC SSH CA people:
  - Mads Freek Petersen mads.freek.petersen@deic.dk
  - Tangui Coulouarn tangui.coulouarn@deic.dk
  - Mikkel Hald mikkel.hald@deic.dk