Policy-Constrained Delegated Access for HPC



November 15-20, 2026

Lahiru Jayathilake[1] (Speaker), Yasith Jayawardana[1], Dimuthu Wannipurage[1], Jim Basney[2], and Suresh Marru[1]

[1] Georgia Institute of Technology

[2] University of Illinois

Researchers

Workflows keep running

MFA at every step

Resource providers

Who ran each job

Limit each tool

Revoke access

THIS APPLIES TO



National HPC centers

Open to researchers nationwide

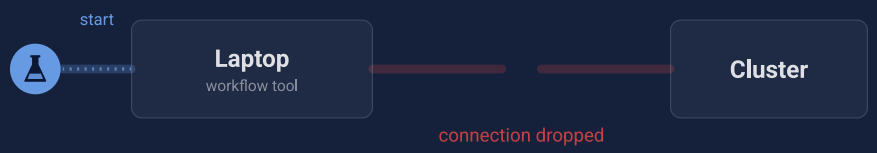


Institutional HPC clusters

For the institution's own users

Here is what researchers face today

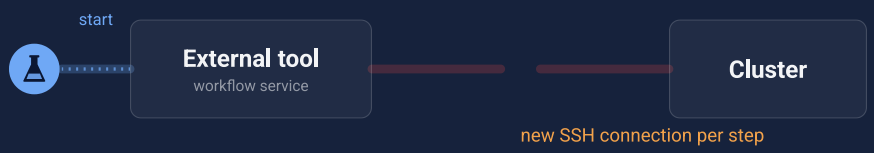
Workflow tool on a laptop, over SSH



- Step 1 complete
- Step 2 complete
- Step 3: connection dropped
- Reconnect: MFA required again

The workflow waits until the researcher approves MFA

External tool, using the researcher's own account



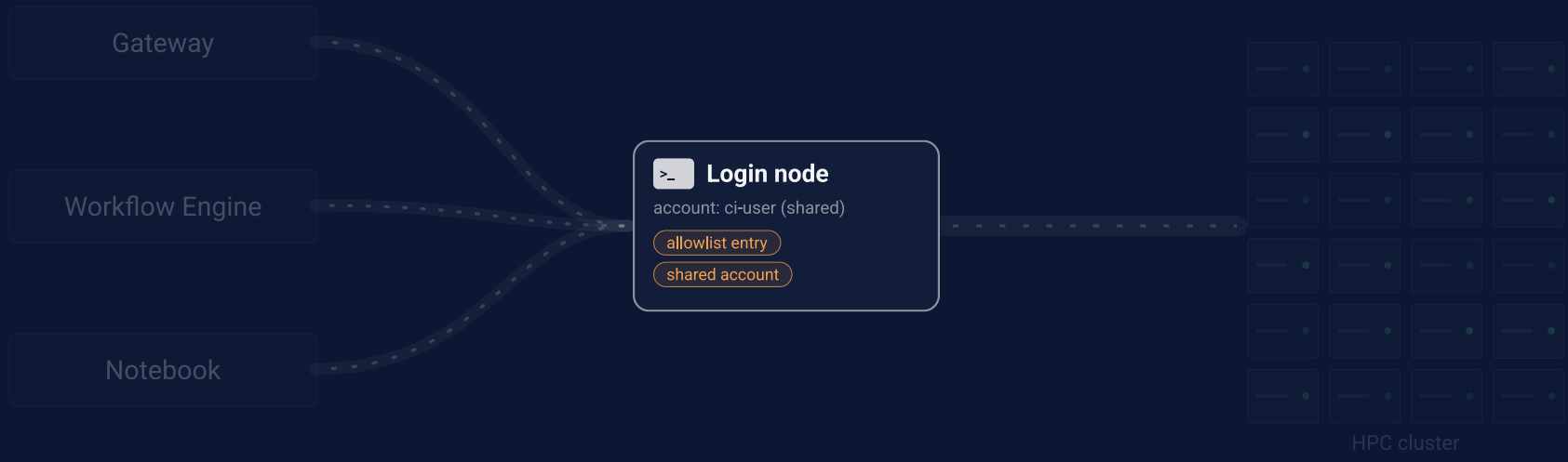
- Step 1: MFA approved
- Step 2: MFA approved
- Step 3: no one to approve
- Steps 4 and on: not started

Unattended automation stops at the first unanswered prompt

Researcher · running multi-step workflows over SSH

Special accounts per tool

On the login node, each tool must be allowlisted and given a shared account to operate with

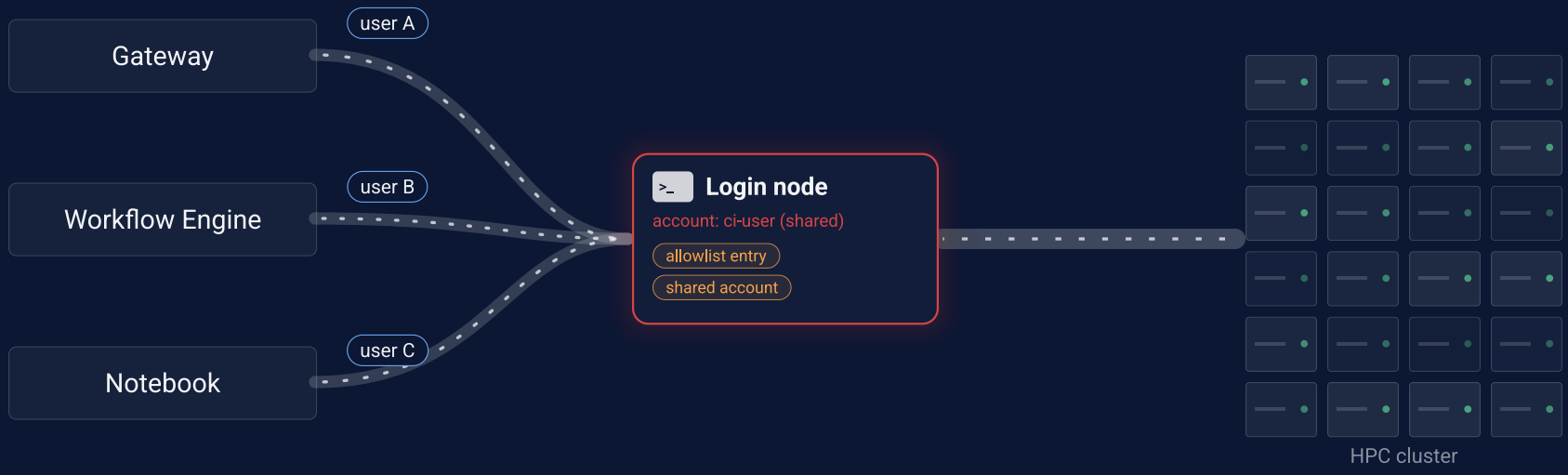


1 **Setup** | 2 Attribution | 3 Audit | 4 Revocation

Resource provider · tools reaching the cluster on shared accounts

Attribution is lost

Three people's work reaches the cluster as one account

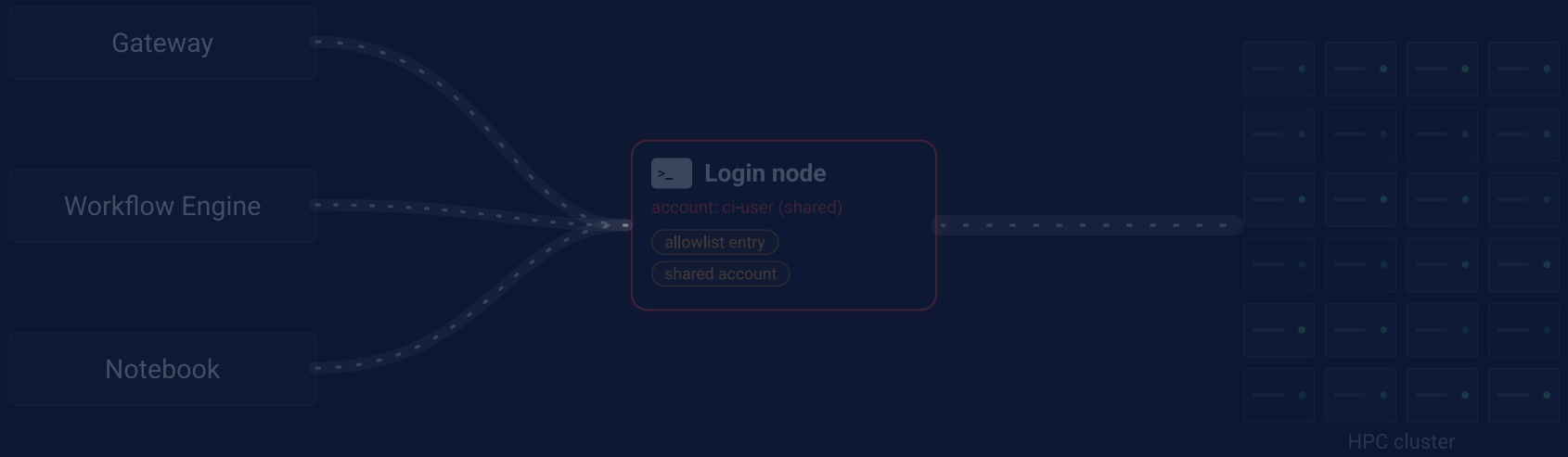


1 Setup | 2 **Attribution** | 3 Audit | 4 Revocation

Resource provider · tools reaching the cluster on shared accounts

Audit is flattened

The log records ci-user for every job. It does not show which user ran each job



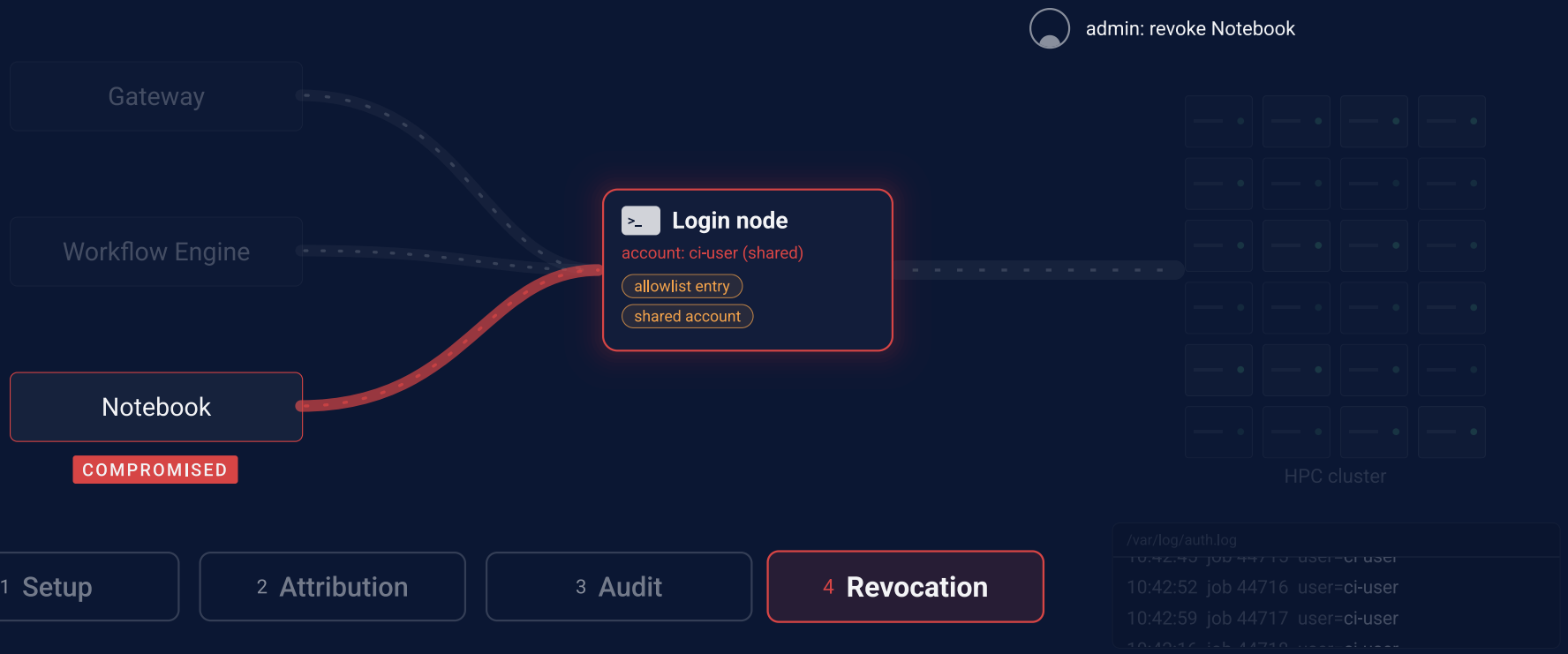
1 Setup | 2 Attribution | 3 Audit | 4 Revocation

```
/var/log/auth.log
10:42:17  job 44711  user=ci-user
10:42:24  job 44712  user=ci-user
10:42:31  job 44713  user=ci-user
```

Resource provider · tools reaching the cluster on shared accounts

Revocation is not possible

One tool is compromised, but revoking the shared key cuts off all three



1 Setup | 2 Attribution | 3 Audit | 4 Revocation

Resource provider · tools reaching the cluster on shared accounts

SOLUTION

Policy-constrained delegated access



Standard OpenSSH certificates

no new software, no protocol change



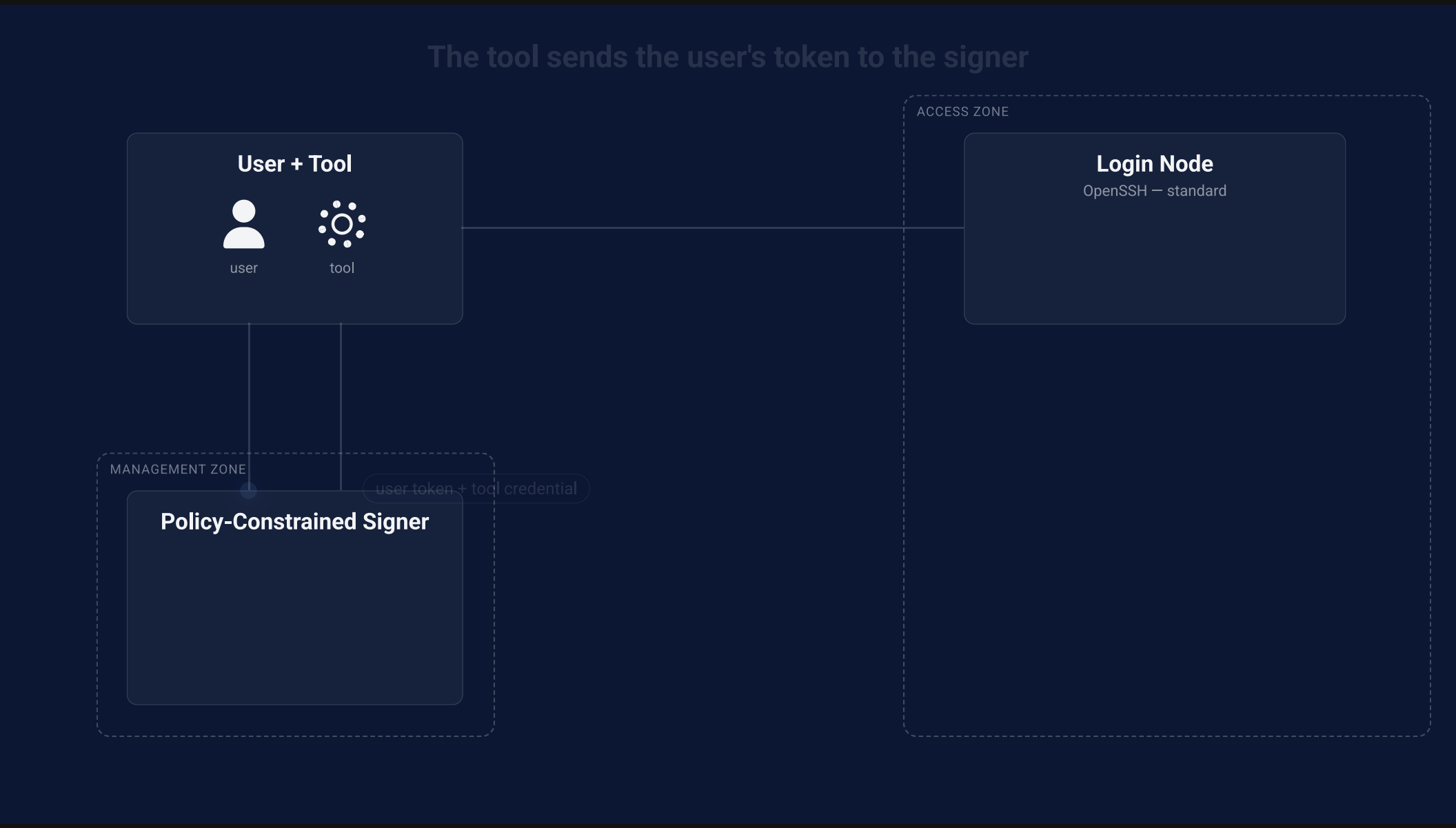
Policies set by each site

resource providers configure scope and privileges

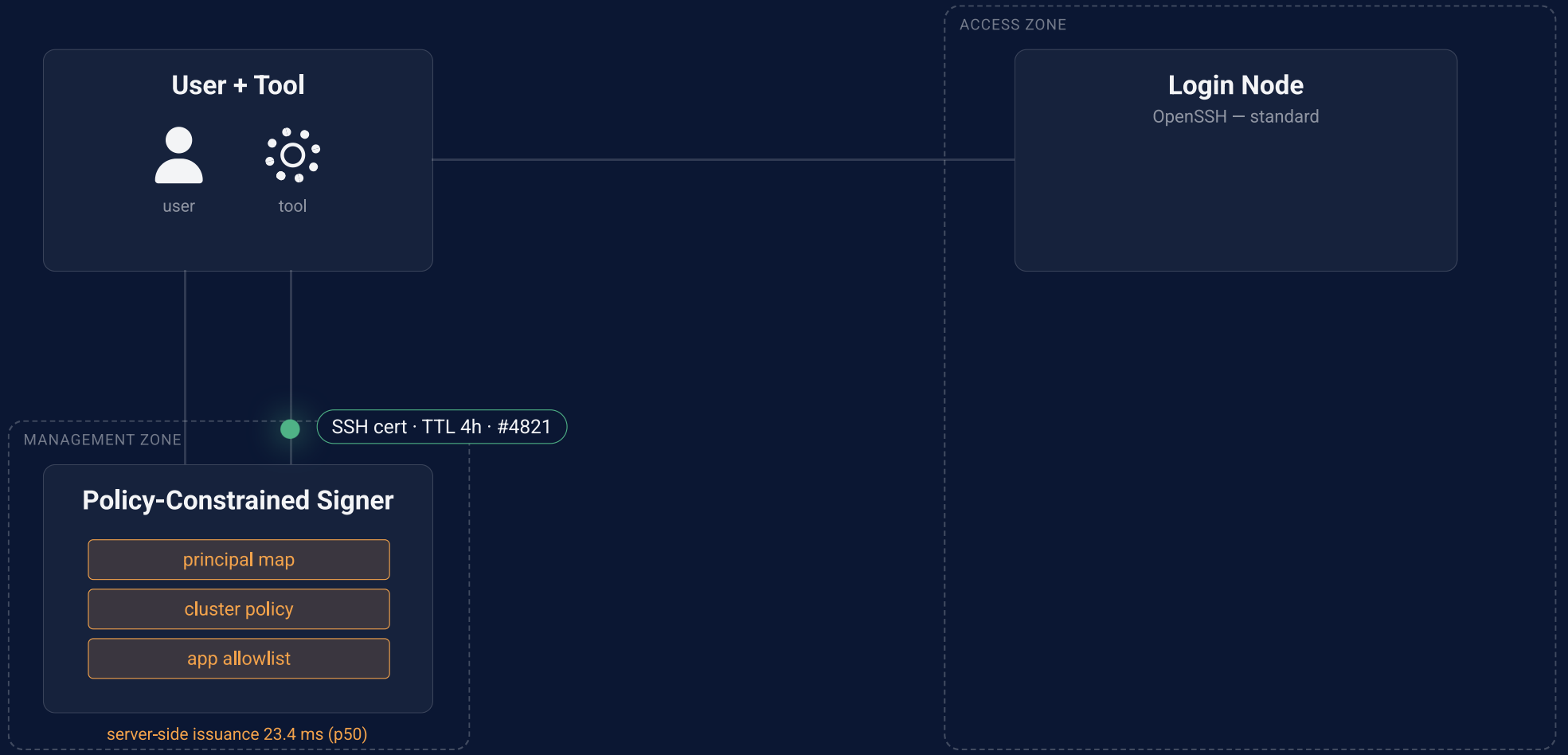


Revocable

any certificate, or a compromised app


The tool sends the user's token to the signer
ACCESS ZONE
User + Tool
user
tool
Login Node
OpenSSH — standard
MANAGEMENT ZONE
user token + tool credential
Policy-Constrained Signer

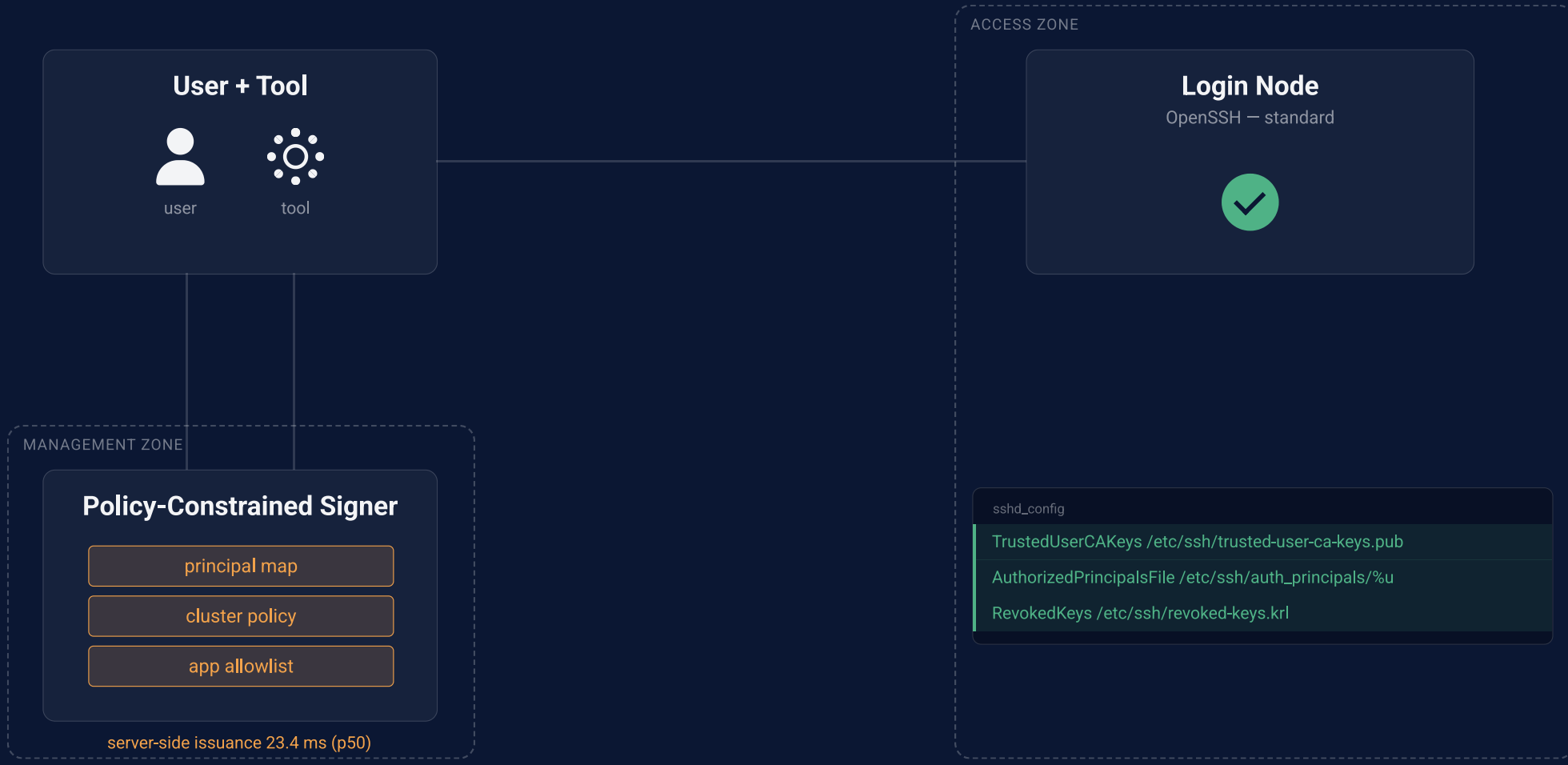
The certificate goes back to the tool, which logs in as the user



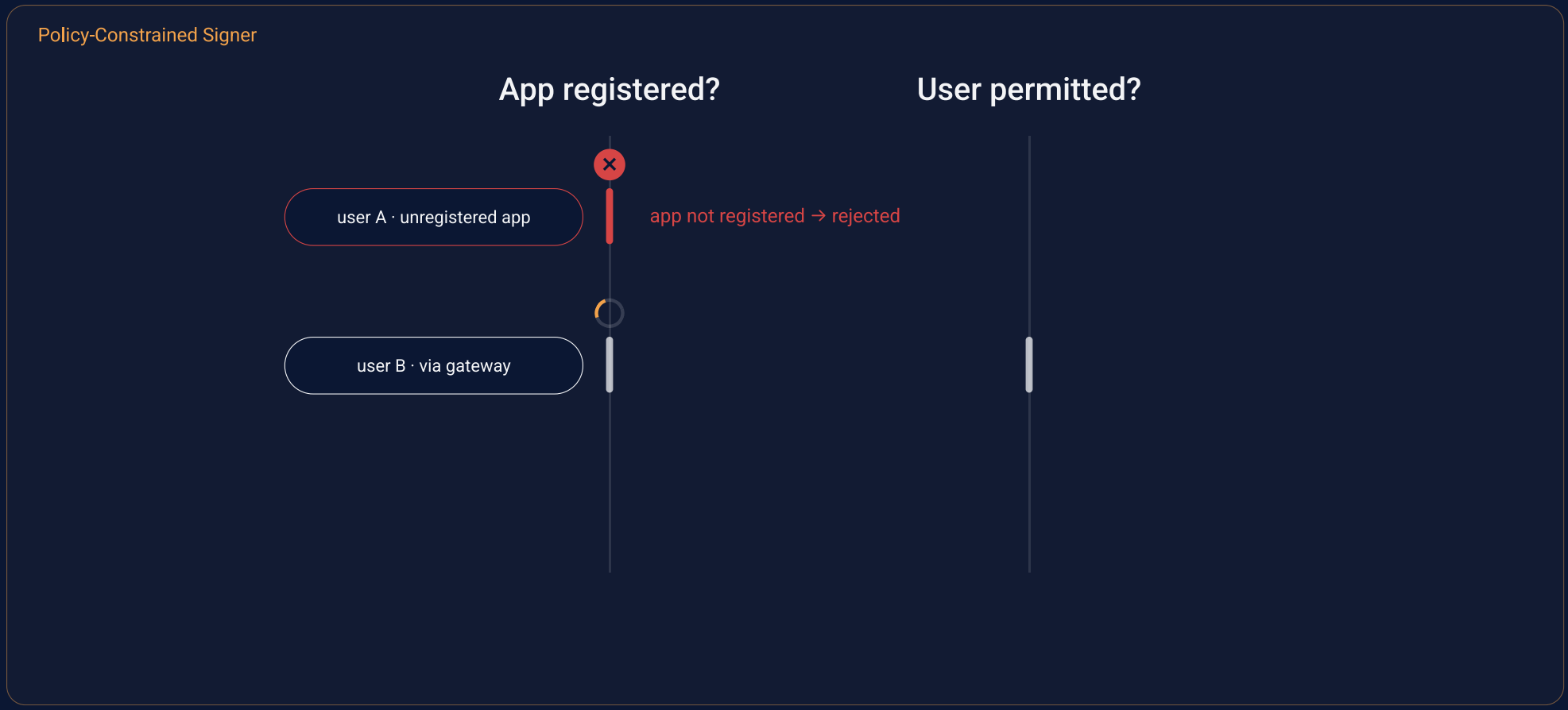
The certificate goes back to the tool, which logs in as the user



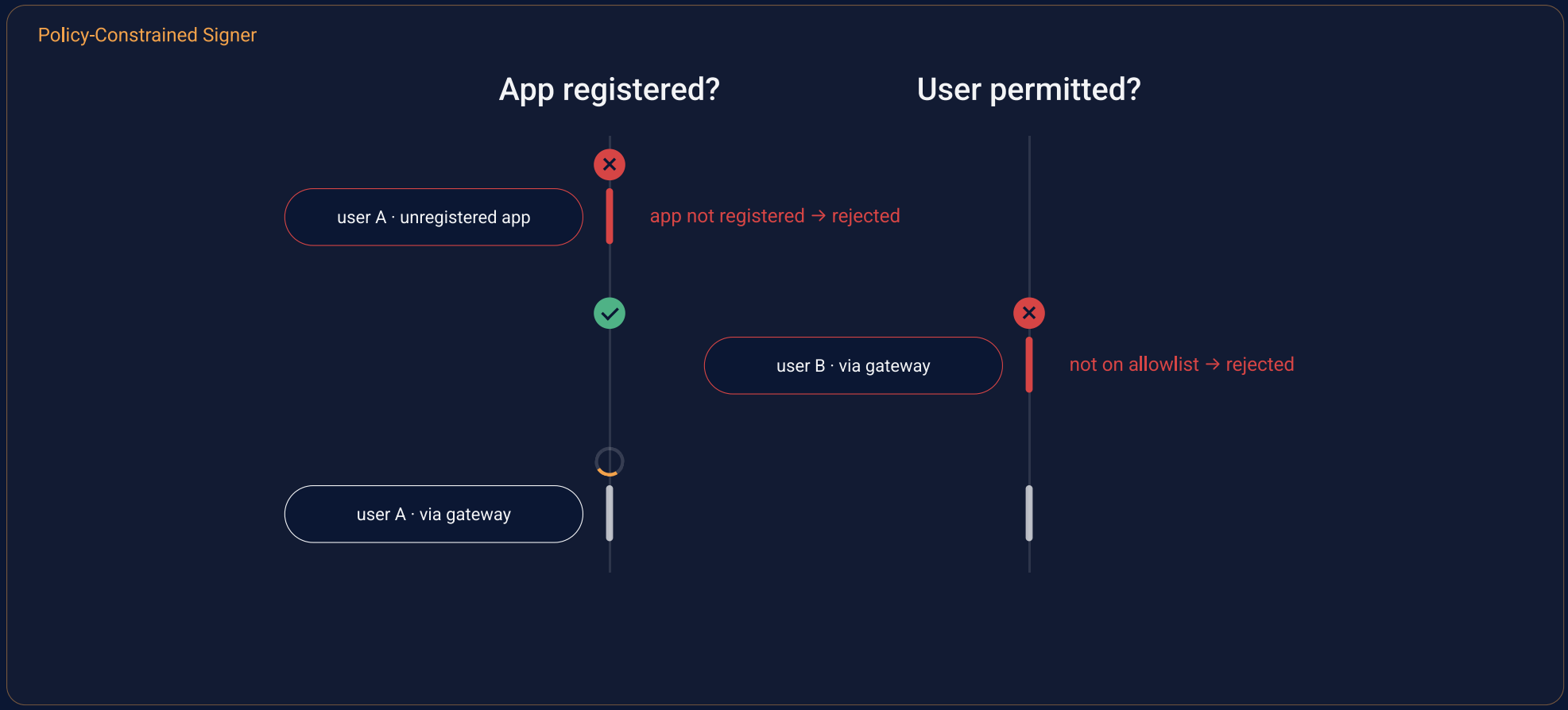
The login node enforces it with three standard sshd_config directives



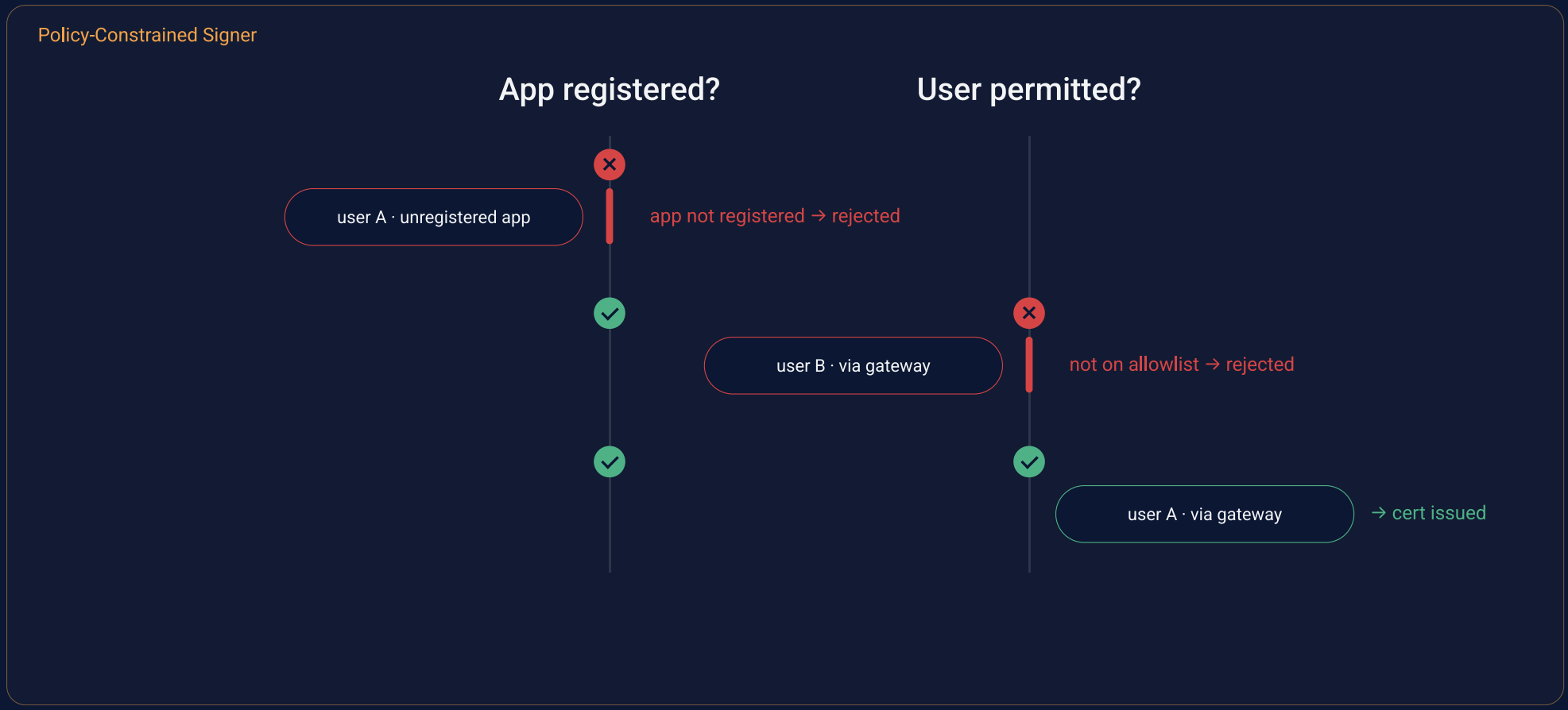
The signer checks **both the application and the user**



The signer checks **both the application and the user**



The signer checks **both the application and the user**



“Who actually ran this job?”

The certificate serial links the logs

Signer issuance log	
serial	#4821
identity	user A (federated ID)
account	usera
app	gateway
TTL	4 h

sshd log · login node	
serial	#4821
account	usera
accepted	certificate

SLURM accounting · sacct	
job	44716
user	usera

Jobs are attributed to the certificate's account in SLURM accounting

What can be stolen?

and what would it give an attacker



Identity token



Refresh token



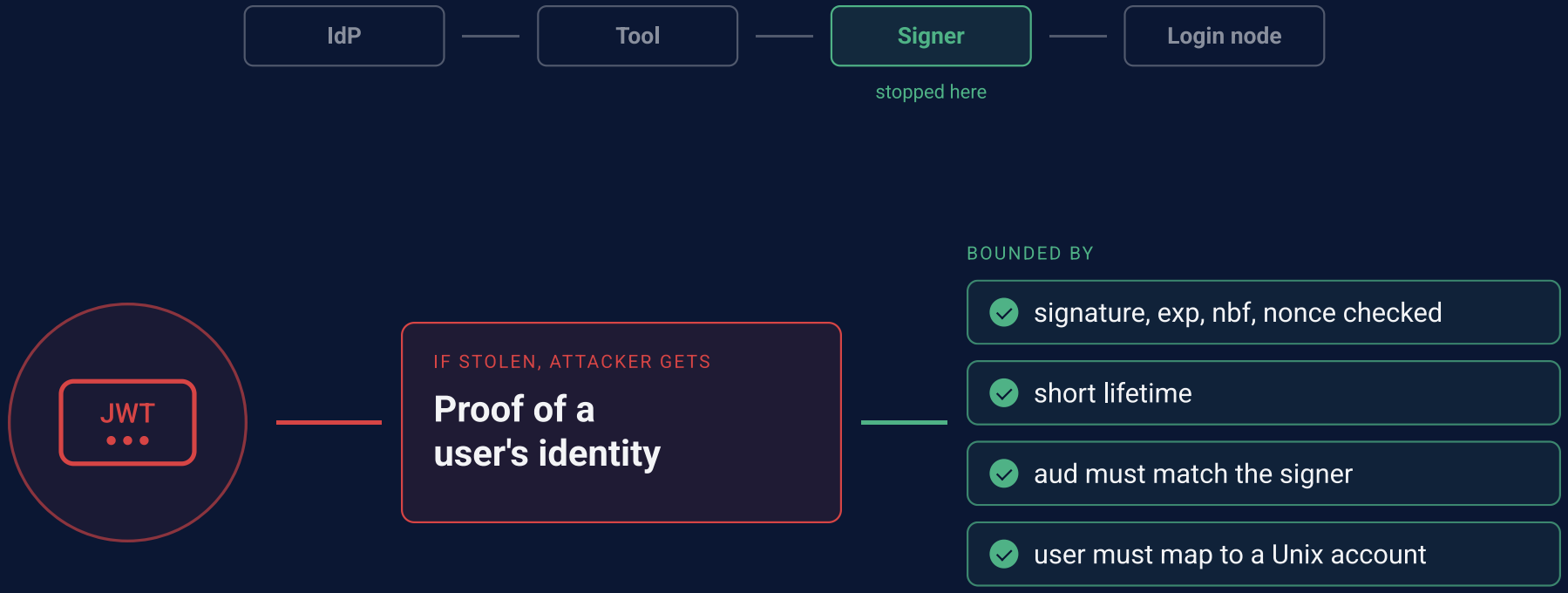
Tool credentials



Certificate + key



CA private key


IdP
Tool
Signer
Login node
stopped here
JWT
IF STOLEN, ATTACKER GETS
Proof of a user's identity
BOUNDED BY
signature, exp, nbf, nonce checked
short lifetime
aud must match the signer
user must map to a Unix account
1 Identity token
2 Refresh token
3 Tool credentials
4 Certificate + key
5 CA private key

IdP — Tool — Signer — Login node

stopped here



IF STOLEN, ATTACKER GETS

Ability to impersonate a user with a generated ID token

BOUNDED BY

- ✓ expires at its IdP lifetime
- ✓ revoke it at the IdP

1 Identity token | **2 Refresh token** | 3 Tool credentials | 4 Certificate + key | 5 CA private key

IdP — Tool — Signer — Login node

stopped here



IF STOLEN, ATTACKER GETS

Ability to impersonate the tool

BOUNDED BY

- only for registered tools
- only for registered clusters
- still needs a valid user token
- tool ID logged with each serial

1 Identity token | 2 Refresh token | 3 **Tool credentials** | 4 Certificate + key | 5 CA private key

IdP — Tool — Signer — **Login node**

stopped here



IF STOLEN, ATTACKER GETS

SSH access as one user

BOUNDED BY

- configurable bounded lifetime
- one site-mapped account
- site-set extensions + critical options
- revoke by serial (KRL)

1 Identity token | 2 Refresh token | 3 Tool credentials | 4 **Certificate + key** | 5 CA private key

IdP — Tool — Signer — Login node

stopped here



IF STOLEN, ATTACKER GETS

Sign certificates for any account

BOUNDED BY

- never leaves the secrets store
- rotate to the next CA key
- login nodes drop the old key

1 Identity token | 2 Refresh token | 3 Tool credentials | 4 Certificate + key | 5 **CA private key**

Certificate login takes about the same time as static-key login

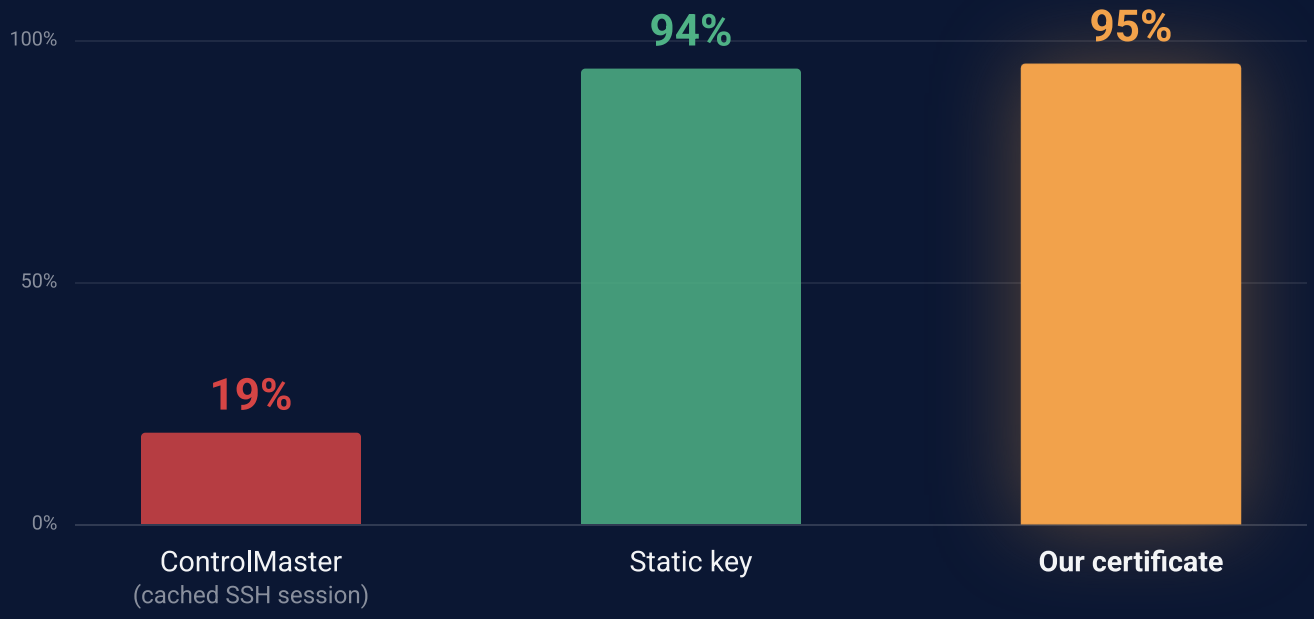
SSH login time, median (50 iterations)



Issuance adds 309 ms, mostly the network round-trip (~287 ms)

Workflow completion under connection drops (100 steps)

95% vs 19%

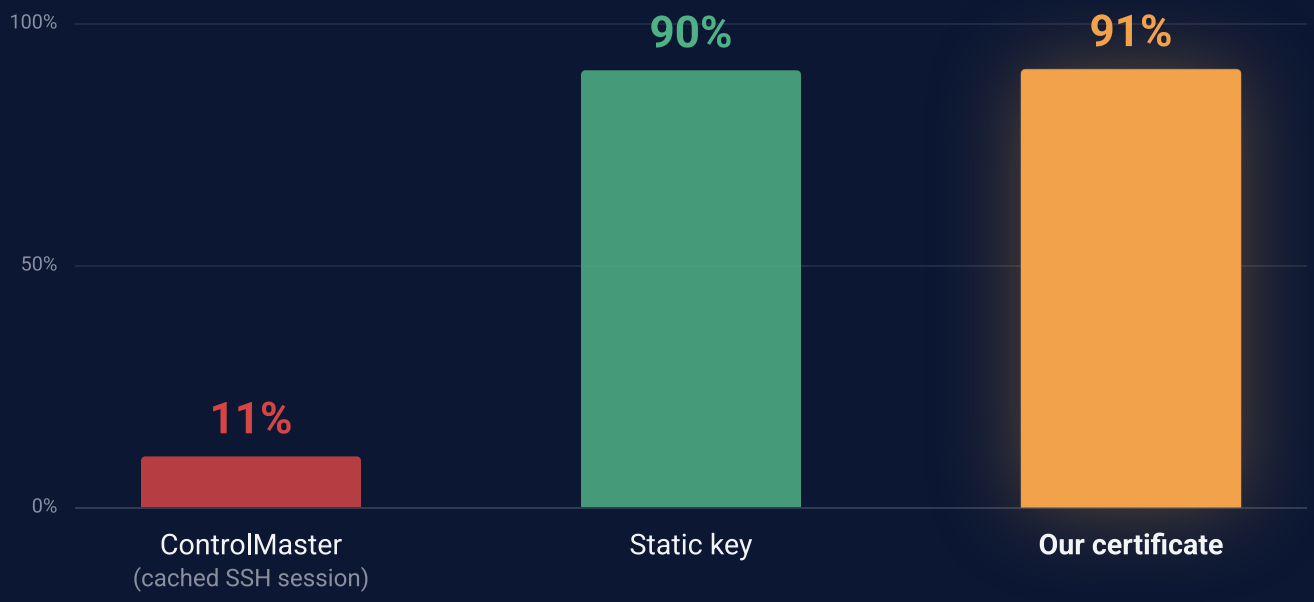


Drop rate: 5%

30 trials per configuration · drops simulated at the application layer

Certificates generated locally (ssh-keygen -s); signer issuance (~309 ms per step) measured separately

Workflow completion under connection drops (100 steps)

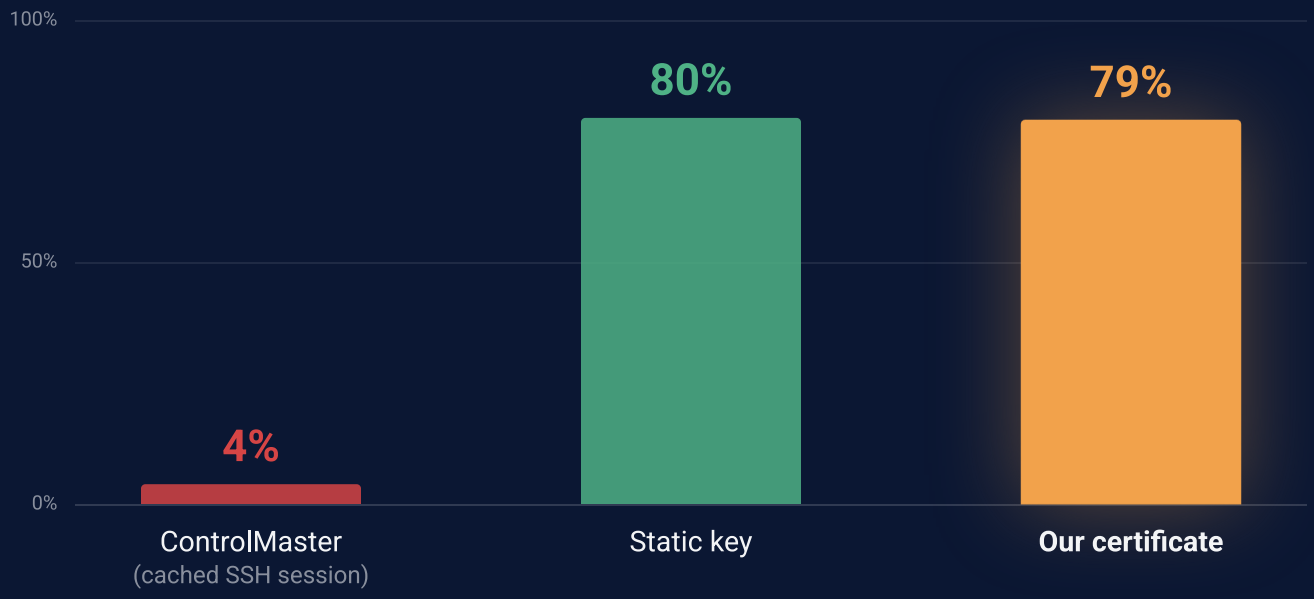


Drop rate: 10%

30 trials per configuration · drops simulated at the application layer

Certificates generated locally (ssh-keygen -s); signer issuance (~309 ms per step) measured separately

Workflow completion under connection drops (100 steps)



Drop rate: 20%

30 trials per configuration · drops simulated at the application layer

Certificates generated locally (ssh-keygen -s); signer issuance (~309 ms per step) measured separately

Policy-Constrained Delegated Access for HPC



Chicago, IL · November 15–20, 2026