LCI Advanced Workshop 2025: **Security in Slurm**

Josh Burks
HPC Systems Analyst
Arizona State University Research Computing
josh.burks@asu.edu

Learning Goals

- Understand Slurm’s built-in security mechanisms
- Recognize weaknesses in traditional HPC security models
- Explore built in options for hardening Slurm clusters
- Explore advanced tools (SPANK, containers, UserBasedFirewall)
- Identify ongoing risks beyond Slurm’s scope

Historical HPC Security Model

- “Hard exterior, soft interior”
  - External security: VPNs, bastion hosts, MFA
  - Internal security: basic POSIX permissions, SELinux often disabled
- Users don’t have full free roam, but inside is loosely controlled
- Onion model: layers exist, but mostly at the perimeter
- Malicious insiders (or compromised accounts) remain a threat

Unencrypted HPC Communications

- MPI and RDMA traffic are unencrypted
- Often bypass host firewalls (RDMA direct access)
- Mitigation:
  - VLAN separation for management vs. compute
  - UserBasedFirewall for cross-user traffic control

Slurm Security Layers

- Transport Security (TLS + certmgr)
- Job & Node Isolation (MCS, containers, /proc controls)
- Authentication & Authorization (PAM, Slurm Adopt, AdminLevels)
- Data Privacy & Audit Control (PrivateData)

PrivateData

- Restrict visibility of accounts, jobs, usage, reservations
- Protects sensitive data in shared accounting systems

```
#slurm.conf
PrivateData=jobs,accounts,events,jobs,nodes,partitions,rese
rvations,usage,users

#slurmdbd.conf
PrivateData=accounts,events,jobs,reservations,usage,users
```

Node Sharing & /proc Security

- Node sharing risks ⟶ job snooping
- /proc isolation with hidepid=1
  - Users may not access any /proc/<pid>/ directories but their own, protected against local eavesdroppers.
  - Reduces exposure of other users’ processes

```
# /etc/fstab
proc /proc proc hidpid=1 0 0
```

Job Container Tmpfs

- Isolated /tmp file systems
- Configurable job-level filesystem isolation
- If userA writes sensitive data to /tmp, normally userB could see this
- By bind-mounting /tmp to a private per-job mount, userB can only see userB’s /tmp

Job Container Tmpfs

```
#slurm.conf
JobContainerType=job_container/tmpfs
ProlögFlags=Contain


#job_container.conf
AutoBasePath=true
BasePath=/mnt/job_tmp # Real location where data lives
Dirs=/tmp,/dev/shm # Paths bind-mounted under the basepath
Shared=true # Required when using autofs
```

MCS (Multi-Category Security)

- Tenant isolation via categories
- Ensures jobs with different categories never share nodes
- Useful where data may be seen my members of a group, but not across groups on a shared system
  - mcs/none disables MCS labels and functionality.
  - mcs/account MCS labels equal the job's --account
  - mcs/group MCS labels equal to the job's user group
  - mcs/user MCS labels equal to the username of the job's --uid
  - mcs/label MCS labels are arbitrary strings

PAM Slurm Adopt

- PAM plugin that prevents users from sshing into nodes on which they don’t have a running job
- Prevents rogue or orphaned processes
- The user’s connection is “adopted” into the extern step cgroup of the job so that they cannot exceed cgroup limits
- All processes created by the user and the user’s connection are killed when the job ends

PAM Slurm Adopt

```
#/etc/pam.d/sshd
account     required       pam_nologin.so
account     include        password-auth
...
-account     required       pam_slurm_adopt.so \
                           action_generic_failure=deny \
                           action_adopt_failure=deny
```

PAM Slurm Adopt + SELinux

- PAM Slurm Adopt requires a custom SELinux module to work in an environment where SELinux is enabled

```
module pam_slurm_adopt 1.0;
require {
    …
}
allow sshd_t
…
```

TLS + certmgr + certgen

- New plugins in Slurm 25.05
- Encrypts all Slurm RPC traffic
- Protects job submissions, scheduling, accounting
- Uses s2n (signal-to-noise) github.com/aws/s2n-tls
    - dnf install s2n-tls-devel

TLS + certmgr + certgen

```
#slurm.conf

TLSType=tls/s2n

TLSParameters=ca_cert_file=/etc/pki/slurm_ca.pem

CertmgrType=certmgr/script

CertgenType=certgen/script


#slurmdbd.conf

TLSType=tls/s2n

TLSParameters=ca_cert_file=/etc/pki/slurm_ca.pem
```

TLS + certmgr + certgen

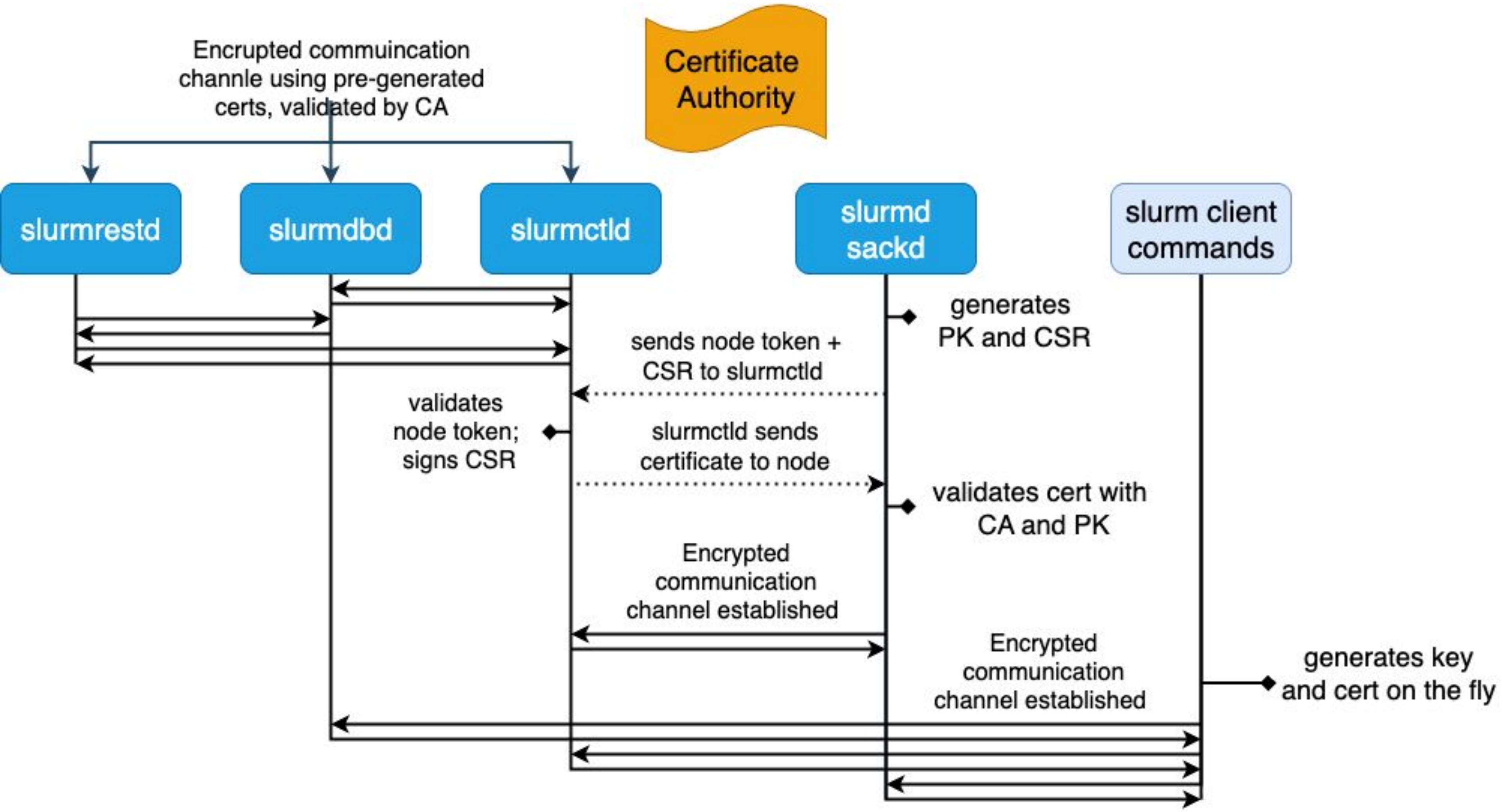
- certmgr plugin manages certificates for slurmd/sackd
- certgen plugin generates key/cert pairs on the fly for client commands

- slurmctld, slurmdbd, and slurmrestd all have unique certificates
- slurmd boot with a pre-shared token
- slurmd generates a private key
- slurmd sends token+certificate signing request to slurmctld
- slurmctld validates token is owned, and returns a certificate
- sackd follows the same process

Note:

- all certificates are signed by a common certificate authority
- slurmd must be started with –ca-cert-file

TLS + certmgr + certgen



Encrupted commuincation channle using pre-generated certs, validated by CA
Certificate Authority
slurmrestd
slurmdbd
slurmctld
slurmd sackd
slurm client commands
generates PK and CSR
sends node token + CSR to slurmctld
validates node token; signs CSR
slurmctld sends certificate to node
validates cert with CA and PK
Encrypted communication channel established
Encrypted communication channel established
generates key and cert on the fly

ASU Research Computing

Beyond Slurm: Network Security

- MPI & RDMA unencrypted
- Infiniband fabric controls: partition keys, SR-IOV
- VLAN separation (mgmt vs compute)
- UserBasedFirewall for user-to-user traffic control
  - github.com/mit-llsc/UserBasedFirewall

Firewalling & Ports

- Limit which hosts can connect to Slurm daemons
- Use firewalls to restrict RPC traffic
- Control SlurmctldPort and SlurmdPort
- Limit ephemeral ports srun can use
- CommunicationParameters=block_null_hash (new in 21.08.8)
- ,NoCtldInAddrAny,NoInAddrAny

Firewalling & Ports

- Limit which hosts can connect to Slurm daemons
- Use firewalls to restrict RPC traffic
- Control SlurmctldPort and SlurmdPort
- Limit ephemeral ports srun can use
- Block null hash (new in 21.08.8)

```
#slurm.conf
SlurmctldPort=6817
SlurmdPort=6818
SrunPortRange=61000-62000
CommunicationParameters=block_null_hash,NoCtldInAddrAny,NoInAddrAny
```

Prolog/Epilog for Security

- Use them to sanitize hosts between jobs
- Reset GPUs, wipe sensitive data
- Purge old mounts
- Ensure the job directory is private

SPANK Plugins

- Extend Slurm job launch with site-specific security policies
- Examples: enforce job isolation, restrict env vars, audit logging
    - github.com/BYUHPC/oodproxy

DOS & Abuse Prevention

- Rate-limit per-user RPC
- Limit number of jobs a user can submit
- Prevent sbatch fork-bombs

```
#slurm.conf
SlurmctldParameters=rl_enable
```

Staying Secure with Slurm

- Watch SchedMD advisories for CVEs
- Paid customers get early advisories
- Patch early, test often
- Security is an ongoing process

Putting It All Together

- TLS + certmgr -> encrypted daemon comms
- MCS + containers + hidepid -> isolate jobs and processes
- PAM + SPANK -> enforce user/job controls
- Prolog/Epilog sanitization -> clean nodes securely
- PrivateData + auditing -> protect accounting
- Network firewalls + VLANs + fabric keys -> protect backplane
- Fairshare enforcement and rate limiting RPC -> prevent abuse

Q & A

LCI workshop
on high performance clustered computing

ASU Research Computing