

SSH CA setup

OVERVIEW & ARCHITECTURE

- **Authentication Flow:** Incoming SSH clients authenticate using an OpenSSH user certificate signed by the server's User Certificate Authority (CA). Similarly, clients verify host identity using an OpenSSH host certificate signed by the Host CA.
- **LDAP & Authentik Integration:** Host **ul** derives user accounts from Authentik via LDAP (managed via SSSD). When a client presents an SSH certificate, `sshd` validates the certificate signature against the trusted CA public key and checks that one of the certificate's valid principals matches the requested LDAP login username. No per-user `~/.ssh/authorized_keys` file is required.
- **Key Storage & Backup:**
 - The User CA private key (`drheck_ca_user_key`) and Host CA private key (`drheck_ca_host_key`) must **never** reside on unencrypted client machines or internet-facing servers.
 - Store the offline CA private keys and passphrases securely in Vaultwarden under account `akadmin` (`admin@drheck.dev`).
 - Deploy only the CA public keys (`drheck_ca_user_key.pub` and `drheck_ca_host_key.pub`) to host **ul** and client machines.
- **Operational Separation:**
 - **Architecture & Foundational CA Setup:** Documented on this page.
 - **Day-to-Day Client & Host Onboarding:** See the dedicated operational runbook: [SSH Client and Host Onboarding Guide](#).

PHASE 1: CA GENERATION & VAULTWARDEN BACKUP

Generate the standard Ed25519 CA Keypairs on a secure offline or administrative workstation:

When prompted by `ssh-keygen`, enter a strong, unique passphrase to encrypt each CA private key.

```
# Generate the User CA keypair
ssh-keygen -t ed25519 -f drheck_ca_user_key -C "drheck-user-ca"
```

```
# Generate the Host CA keypair
ssh-keygen -t ed25519 -f drheck_ca_host_key -C "drheck-host-ca"
```

Storage in Vaultwarden:

1. Create a secure item in Vaultwarden under `akadmin` (`admin@drheck.dev`):
 - `drheck_ca_user_key` (multi-line private key in Notes field)
 - `drheck_ca_user_key_pw` (passphrase)
 - `drheck_ca_host_key` (multi-line private key in Notes field)
 - `drheck_ca_host_key_pw` (passphrase)
2. Remove the private CA keys from the unencrypted local disk of your workstation after verifying the Vaultwarden backup.

PHASE 2: MAIN SERVER CONFIGURATION (ul)

1. Deploy Trusted User CA Public Key

- Copy the public CA file `drheck_ca_user_key.pub` to host `ul` at `/etc/ssh/drheck_ca_user_key.pub`.
- Set secure permissions:

```
sudo chmod 644 /etc/ssh/drheck_ca_user_key.pub
sudo chown root:root /etc/ssh/drheck_ca_user_key.pub
```

2. Configure Host Certificate (Signed by Host CA)

1. Copy `ul`'s public host key `/etc/ssh/ssh_host_ed25519_key.pub` to your signing environment.
2. Sign the host key using the vaulted `drheck_ca_host_key` (defining all valid FQDNs and IP addresses as principals):

```
ssh-keygen -s drheck_ca_host_key -I "drheck-ul-host-cert" -h -n
ul,ul.drheck.dev,10.0.0.3,fd77:7777:5010:0:1337:f1ac:d00d:337 -V +52w
/etc/ssh/ssh_host_ed25519_key.pub
```

3. Place the resulting certificate at `/etc/ssh/ssh_host_ed25519_key-cert.pub`.
4. Set secure permissions:

```
sudo chmod 644 /etc/ssh/ssh_host_ed25519_key-cert.pub
sudo chown root:root /etc/ssh/ssh_host_ed25519_key-cert.pub
```

3. Configure OpenSSH Daemon on ul

- Create or edit `/etc/ssh/sshd_config.d/10-drheck-certs.conf`:

```
TrustedUserCAKeys /etc/ssh/drheck_ca_user_key.pub
HostCertificate /etc/ssh/ssh_host_ed25519_key-cert.pub
```

4. Validate and Reload SSHD on ul

- Validate the configuration syntax before reloading:

```
sudo sshd -t
```

- Reload the daemon:

```
sudo systemctl reload sshd
```

PHASE 3: AUDITING & KEY REVOCATION (KRL)

OpenSSH natively supports Key Revocation Lists (KRL) to block compromised keys or expired certificates before their expiration timestamp.

1. Key Inventory Architecture

Every key signed by the signing utilities is automatically archived under:

- `/etc/ssh/ca_inventory/hosts/<label>.pub` and `<label>-cert.pub`
- `/etc/ssh/ca_inventory/users/<label>.pub` and `<label>-cert.pub`
- `/etc/ssh/ca_inventory/manifest.tsv`

2. Listing Active & Revoked Keys

Run the inventory helper to view all issued certificates and their revocation state:

```
revoke-key --list
```

Example Output:

LABEL	TYPE	FINGERPRINT	STATUS
danknasty	host	SHA256:oj796Z0eYmi6hAm+pG	[ACTIVE]
u1	host	SHA256:1a4QDcX1p/QT4qbFdc	[ACTIVE]
borrowbook	user	SHA256:HPu6XQU1ARHnLWgMWo	[ACTIVE]
metropolis	user	SHA256:220ChYHL9QxAe57Noe	[ACTIVE]

3. Revoking a Key

To immediately revoke any key or certificate:

```
sudo revoke-key <label>
```

Actions performed automatically:

1. Locates the archived public key in `/etc/ssh/ca_inventory/`.
2. Appends the key hash into the OpenSSH binary KRL file at `/etc/ssh/revoked_keys`.
3. Ensures `RevokedKeys /etc/ssh/revoked_keys` is enabled in `/etc/ssh/sshd_config.d/10-drheck-certs.conf`.
4. Tests configuration syntax and reloads `sshd`.

PHASE 4: CLIENT-SIDE GLOBAL TRUST & ENFORCEMENT

1. Trusting the Host CA Across Clients

To trust all hosts bearing a DrHeck Host CA certificate without TOFU prompts, deploy this line to `/etc/ssh/ssh_known_hosts` (or `~/.ssh/known_hosts`):

```
@cert-authority * ssh-ed25519 <drheck_ca_host_key.pub_content>
```

2. Enforcing Certificates (Disabling Raw Keys)

Once all clients have valid user certificates, you can disable uncertified public keys while preserving an `authorized_keys` backdoor for LAN-only bootstrapping:

```
# /etc/ssh/sshd_config.d/10-drheck-certs.conf
AuthorizedKeysFile /dev/null

# Allow authorized_keys ONLY on the internal trusted network
Match Address 10.0.0.0/24,fd77:7777:5010::/64
    AuthorizedKeysFile .ssh/authorized_keys
```

OPERATIONAL RUNBOOK

For the step-by-step instructions on adding new machines and signing certificates, see:

- [SSH Client and Host Onboarding Guide](#)
-

Revision #7

Created 2026-07-14 03:56:07 UTC by admin

Updated 2026-09-03 12:54:48 UTC by luKe