

# Auto-Generate Signed SSH Keys

## 1. Authentik Stores the User CA Private Key:

Upload `drheck_ca_user_key` (unencrypted or with a known passphrase) into Authentik under System -> Certificates.

## 2. Create an Authentik Python Policy/Execution Flow:

You create an API endpoint in Authentik (or a custom Policy/Flow) that accepts a POST request containing a public key and a username.

- \* Authentik validates the API request.

- \* Using its internal Python Expression Policies, Authentik signs the public key using the loaded `drheck_ca_user_key` and returns the signed certificate text.

## 3. Deploy `pam_exec.so` on the Desktop:

On your desktop machine, you add a line to `/etc/pam.d/system-auth` to execute a script upon successful login:

```
1 session optional pam_exec.so /usr/local/bin/mint-ssh-keys.sh
```

## 4. The `mint-ssh-keys.sh` Script:

Whenever any user logs in, this script runs under their local user context:

- \* Checks if they already have keys: If `~/.ssh/id_ed25519-cert.pub` already exists, it exits (doing nothing).

- \* Generates local keys: If they are logging in for the first time, it runs `ssh-keygen -t ed25519 -N "" -f ~/.ssh/id_ed25519` to generate a passwordless keypair.

- \* Requests a Signature from Authentik: It uses `curl` to send their new `id_ed25519.pub` to your Authentik API endpoint.

- \* Saves the Certificate: It saves the returned signed certificate to `~/.ssh/id_ed25519-cert.pub`.

---

Revision #1

Created 2026-07-14 03:57:06 UTC by admin

Updated 2026-07-14 03:59:11 UTC by admin