Systemd Stage 1

Lessons Learned

Will Fancher - ElvishJerricco

What is Stage 1?

- Unique NixOS terminology. Equivalent to initrd, initramfs.
  - Temporal, rather than files / archives.
- When the kernel starts, stage 1 is the first userspace it executes.
- Stage 1 is when userspace gets to find and prepare the operating system.
  - Separate from the boot loader because significantly more complicated things can be done.
- Mount the root file system, and other core file systems.
  - E.g. /nix/store, /etc
- Once done, the operating system can start.
  - switch_root command moves PID 1 into a new root directory.
  - NixOS activation also has to take place.

Scripted Stage 1

- nixos/modules/system/boot/stage-1-init.sh
- fileSystems.<path>
- boot.initrd.preDeviceCommands
- boot.initrd.preLVMCommands
- boot.initrd.postDeviceCommands
- boot.initrd.postMountCommands

So what’s the problem?

- Well, it’s one of those little things…
- boot.initrd.luks.devices.<name>.preLVM
- It’s serial; imperative
- It’s a lot of custom shell code

Systemd

- PID 1
- Bring up applications and services
  - Manages the processes, mountpoints, and devices that constitute a functioning operating system.
- Based on declarative “units”
  - Services, devices, mounts.
- Arbitrary dependencies and ordering

Systemd Stage 1

- systemd has a suite of configurations and tools for stage 1 included
- Declarative
  - Arbitrary dependencies
- Parallel
- Many additional tools…

Rescue and debug shells

- Useful tools like systemctl and journalctl
- Journal survives to stage 2
- ‘systemctl default’ to try boot again

systemd-networkd

- More reliable
- Declarative
- Arbitrarily complex network configurations

systemd-ask-password

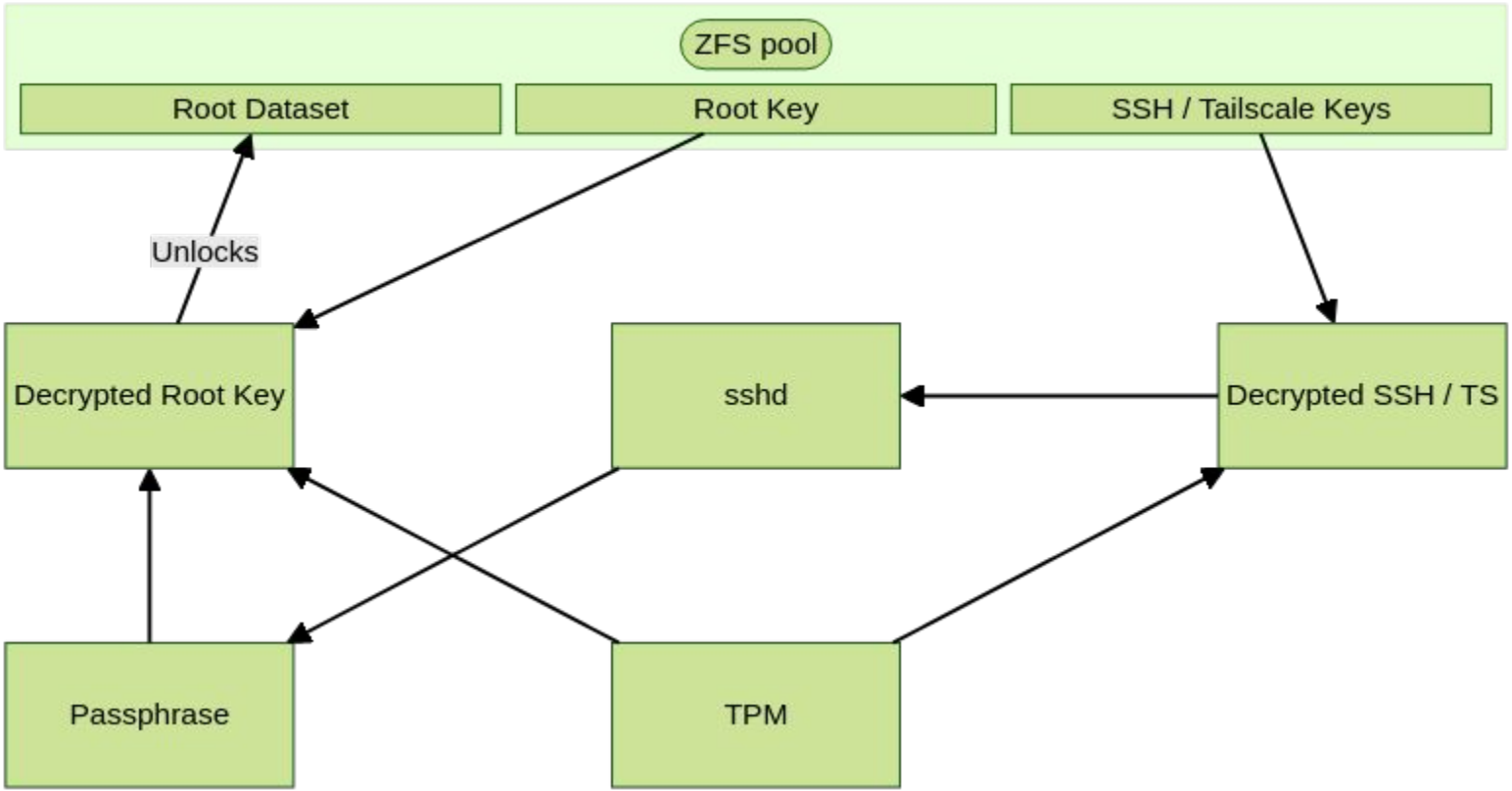
- Scripted initrd uses a number of unpleasant mechanisms to ask for passwords
- systemd-ask-password is a common interface
- Anything that needs a password entry just runs systemd-ask-password
- Anything that can provide password entry responds to the protocol
- Plymouth graphical password prompt for free

TPM2, FIDO2, YubiKey integration

- Automated or 2FA disk decryption built into systemd-cryptsetup
- Complements UEFI Secure Boot / Lanzaboote

An Example

- Root file system on an encrypted ZFS dataset (*)
- Unlocked with a keyfile stored on a LUKS volume
  - That LUKS volume is on a zvol on the same pool
- That LUKS volume is unlocked with combination of TPM2 and passphrase
- That passphrase is entered over SSH
- Over Tailscale VPN
- The SSH host keys and Tailscale state are stored on another LUKS volume
  - Also on a zvol on the same pool
- Which is unlocked automatically by the TPM2
  - Shared with stage 2 fairly securely while still requiring a passphrase for the root dataset



https://github.com/ElvishJerricco/stage1-tpm-tailscale

Roadmap

- 22.05 - Experimental Availability
  - boot.initrd.systemd.enable = true;
- 23.11 - Stable
- 24.05 - Default
  - Compatibility detection will fallback to scripted stage 1
- 24.11 - Remove scripted stage 1 networking
- 25.05(?) - Remove scripted stage 1

Small Things

- Stage 1 is only a few seconds for most systems
- Just one option caught my interest
- Massive collaboration with many community members