Stress-Free OS Upgrades

Using systemd-nspawn and ZFS to trim cruft and minimize downtime

OS upgrades are dirty.

- They stop services
- They edit files
- They alter databases
- No "undo"
- ...often littering files everywhere

The Goal

- What if you could upgrade the OS...
  - ...leaving core services running?
  - ...leaving all known-good files intact?
  - ...without modifying a single database record?
- Just apply your settings to a shiny, new OS image...
  - ...and reboot into it!
- If anything goes wrong...
  - *just reboot back!*

Key Terms

ZFS

- Cross-platform fs + volume manager + RAID
- Compare `btrfs`, LVM, and hardware RAID systems

ZFS Dataset

- A logical group of files and directories
- Like formatted partition, but not restricted to specific section of the disk

Boot Environment

- A special dataset whose mountpoint is `/`
- Designed to be bootable (contains kernel/initrd)

Key Terms

Container

- Linux feature allowing process and resource isolation
- Between an old-school `chroot(2)` and a full-fledged VM
- Compare FreeBSD jails or Solaris zones

`systemd-nspawn`

- "Boots" a filesystem like a VM, but without emulation
- Takes a path with a systemd OS

Caveats

`systemd-nspawn` is...

- *not* a VM...
  - Boots guest from local directory, not a virtual partition
  - Must use same kernel as host
- *not* a Docker container...
  - Boots guest from local directory, not a system of overlays
  - Filesystem is directly mutable, *not* layered
  - No `CMD` or `ENTRYPOINT` —guest OS manages its own services

How It Works

From a running system...

- Fetch and import (`zfs recv`) BE for new OS
- Mount new BE
- Pre-boot migration
- Spin up nspawn container for new BE
- Transfer settings from host to container
  - Container is mutable; services run simultaneously
  - Copy configuration and data across and verify live
- Reboot!
- Worst case?
  - Reboot to original BE and triage what went wrong

In Action

```
# Fetch and import:
zfs recv zroot/ROOT/next-os
# Mount:
zfs set mountpoint=/mnt/next zroot/ROOT/0m-next
zfs mount zroot/ROOT/next
# Pre-boot migration:
cp /etc/hostid /etc/hostname /etc/resolv.conf /mnt/next/etc
# Start container:
systemd-nspawn --boot --network-veth --directory=/mnt/next --machine=0m-next
```

Demo

Implementation Details

- Changeset: set of scripts, logic, and validations for each minor version
- Only upgrades to successive versions
  - To upgrade 2.5 to 2.7 requires `2.5→2.6` followed by `2.6→2.7`
  - Manageable support matrix
  - Release of 2.8 only requires one changeset: `2.7→2.8`

Implementation Details

- Written in Python by Rust fans
  - Static typing & analysis: `typing`, `vulture`, `pyright` (later `ty`)
  - `Result` type:

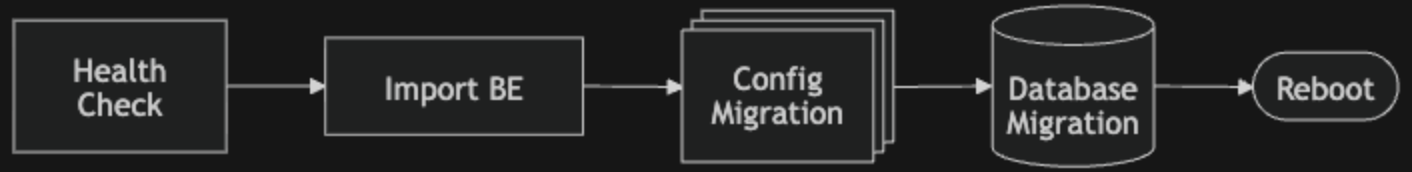
```
match read_file(file):
    case Ok(content): process(content)
    case Err(e): handle_failed_copy(file, e)
```

  - Automated formatting via `black`
  - PEX - self-extracting, self-contained Python application archives

```
$ upgrader --be-name 0m-next
```

  - reasonable at the time; Rust preferred if greenfield/rewrite

The Process



Goal accomplished ... with some notes

- Implemented for 3 deployed releases, maintained for the next
- Some irregular installations still failed...
  - *BUT* were resolved with a reboot, not hours of unscheduled downtime
  - *Post-mortem* to find cause, patch `upgrader`, retry!

Honorable Mention

"Atomic" distros — OSTree / `rpm-ostree`

- Cruft isn't cleaned, just paved over.

Immutable / Composable distros — `Nix` et al.

- Rollbacks require "building" previous configuration

But ultimately...

- We already had an OS
- We wanted a drop-in replacement for upgrades

Links

Boot Environments

- Boot Environments
- `beadm` at Oracle & FreeBSD

`systemd-nspawn`

- `systemd-nspawn` @ freedesktop.org
- More `systemd-nspawn` (Debian)

Questions?

Thank You