ParticleOS, from Fedora to Feast: Stirring Traditional Distros into Immutable Delights

Luca Boccassi, Linux Systems Group, Microsoft

Agenda

- Motivations - what is an immutable system anyway
- Cookware - what tools do we use to build one
- Ingredients - what do we build it from
- Recipes - how do we put it together
- Kitchen - where do we build it
- End result - what sort of image and workflow we get out of it
- Yum yum!

Glossary

- TPM: Trusted Platform Module
- UEFI Secure Boot: firmware-based payload signature verification
- UKI: Unified Kernel Image, kernel + initrd + cmdline + … in a signed PE
- EFI Addons: sidecars for UKIs with additional options, files, …
- Hermetic-usr: OS vendor tree self-contained or can regenerate from /usr/
  - Hint: sysusers.d for system users/groups, tmpfiles.d for /var/ or /etc/ dirs/files
- DDI: Discoverable Disk Image, GPT self-described disk
- dm-verity: kernel device mapper driver for cryptographically verified block devices
- IPE: Integrity Policy Enforcement, kernel LSM for code signing

What is an immutable system anyway

- A system is either immutable or it is not
  - Vendor tree (/usr/) a little bit writable? It's not!
  - Can just run this rpm-ostree command to pull in some packages? It's not!
  - Can just build a new snapshot and reboot into it? It's not!
- Immutable means immutable, with a chain of trust. I.E.: the threat model is local execution after privilege escalation attempting to change the system.
- Translation: if you use a kernel-verified signed dm-verity volume(s) without locally available private keys, then you have an immutable system, otherwise it's just a sparkling package manager
- ParticleOS gives you the tools and the recipes to achieve this out of the box

mkosi to the rescue

- The swiss army knife of image building from the systemd project
- Pure Python3 implementation, no dependencies
- INI-style configuration files, composable/drop-ins style
- Native support for all the new fancy systemd tools
- Builds system images with bells and whistles
  - UKIs, DDIs, extensions, portable images, etc.
- Development builds for local workflows, booting containers/VMs with zeroconf
- Production builds with support for various signing workflows
  - OpenSSL provider/engine for inline signing with hardware tokens
  - Offline signing for multi-staged builds

Choose your Destiny

- Currently Fedora, Arch, SUSE and Debian have recipes in ParticleOS
- Any distribution supported by mkosi can be trivially added
- The recipes broadly speaking cover three distinct areas:
  - mkosi boilerplate and glue
  - List(s) of packages to install - often per-release as packages tend to change and be incompatible
  - Workarounds for distributions that do not support hermetic-usr and other modern standards out of the box
- If mkosi does not support the distribution (or more precisely, the package manager), then work is more involved for a new port
  - Needs one or two Python3 modules to implement the internal install/update/etc APIs
  - Still doable! Plz send PR kkthxbye

OBS - SUSE Open Build Service

- Build system for Linux distributions by SUSE
  - Available for open source developers at https://build.opensuse.org
- Rebuilds on git push or dependency tree changes
- Native support for mkosi added in 2022
- Can build UKIs or DDIs (full system images) or portable images or extensions
- Signing keys handled by build service, not accessible by developers
  - Same key management guarantees for random OSS developer and openSUSE builds
  - No custom code/scripting/management of any kind of private keys, all automated
  - Multi-stage build, hashes sent to signing service, builder gets signatures back and starts over
- Published on CDN with PGP-signed manifest compatible with sysupdate.d
- system:systemd OBS project builds GNOME/x86_64 DDI images based on F42, F43, F44, Deb13, Deb14
  - Server arm64 images for Deb13/Deb14
- Users can fork images, modify the recipe and get automated rebuilds and publishing, signed with a key scoped to their own user

particleos-fedora

Watch

Set Labels

Assign someone

No description set

Edit

1 derived packages

Download package

Checkout Package

Create Badge

Source Files

Show 25 entries

Search...

Filename	Size	Changed	Actions
_service	326 Bytes	4 months ago	
_service:obs_scm:mkosi.conf	3.74 KB	about 3 hours ago	
_service:obs_scm:particleos.obscpio	148 KB	about 2 hours ago	
_service:obs_scm:particleos.obsinfo	112 Bytes	about 2 hours ago	

page 1 of 1 (4 records)

First Previous Next Last

Add local files

Add an empty file or service

Latest Revision

Luca Boccassi (bluca) committed about 2 hours ago (revision 46)

```
Service triggered by the 'mkosi-obs-workflow' token via Github
push 7ec4154 on obs.
```

Files Changed Browse Source

Build Results

Refresh

Show 29 excluded/disabled results

particleos-fedora

fedora_42_images

x86_64 succeeded

fedora_43_images

x86_64 succeeded

fedora_44_images

x86_64 succeeded

End result

- Erofs and signed dm-verity for /usr/ with Fedora/Arch/SUSE/Debian
- ESP with signed systemd-boot and UKI
- UKI with profile preconfigured for IPE code integrity enforcement
- On first boot systemd-repart formats root, home and swap with LUKS2 encrypted with TPM2
- Has systemd-sysupdate preconfigured to update systemd-boot, UKI and DDI as new builds appear on OBS with A/B partitioning scheme
  - With boot counting and assessment for fallback on failure
- Uses systemd-homed for user and home area management
- GNOME or KDE with Flatpak + Flathub preconfigured for desktop flavours
- Portable Services or containers for additional software on servers

Chain of trust

- UEFI Secure Boot automated self-enrollment with OBS signing keys
  - Still have the MSFT 3rd Party CAs to avoid bricking laptops that need to load signed OPROMs
- CPU verifies firmware (e.g.: Intel BootGuard)
- Firmware verifies systemd-boot
- sd-boot verifies UKI and addons via firmware
- Kernel verifies vendor tree as signed dm-verity via UEFI DB/MOK keyrings
- IPE LSM verifies loaded binaries and libraries originate from signed dm-verity
- End result is an immutable system cryptographically verified by a chain of trust from hardware to userspace binaries
  - TODO: interpreted scripts integrity verification via AT_EXECVE_CHECK
  - TODO: making GNOME IPE-friendly

Demos

- Code integrity with IPE on Fedora
- GNOME on Debian

Thank you

Questions?