CLIP OS 5: Beta release

Timothée Ravier, Thibaut Sautereau

Agence nationale de la sécurité des systèmes d’information (ANSSI)

10 & 11 December 2019, Paris Open Source Summit
About the ANSSI

- *Agence nationale de la sécurité des systèmes d’information*
- French authority in the area of cyberdefence, network and information security
- Provides its expertise and technical assistance to government departments and businesses and plays an enhanced role in supporting operators of vital importance.
Linux distribution developed by the ANSSI
Initially only available internally
Now open source, mostly under the LGPL v2.1+
Code and issue tracker hosted on GitHub\(^1\):
  Version 4: available as reference and for upstream patch contribution
  Version 5: currently developed version, beta released in December 2019

\(^1\)https://github.com/CLIPOS
\(^2\)https://github.com/CLIPOS-Archive
CLIP OS?

Not yet another Linux distribution

- Not a generic/multi-purpose distribution

Targets three main use cases

- Mobile office workstation
- Remote administration workstation
- IPsec gateway
Hardened OS

- Based on Gentoo Hardened
- Hardened Linux kernel and confined services
- No interactive root account available:
  ⇒ "Unprivileged" admin, audit and update roles
- Automatic updates using A/B partition model (similar to Android 7+)
- Multilevel security:
  - Provide two isolated user environments
  - Controlled interactions between isolated environments
5.0 Alpha features & security
5.0 Alpha: Initial features

- Functional core (boot to command line shell)
- Strict split between:
  - Read Only: system executables, configuration and data
  - Read Write: runtime configuration, logs, user and application data
- Initial boot chain integrity:
  - Secure Boot (bootloader, initramfs, Linux kernel and its command line)
  - Read-only system partition protected by DM-Verity
- Initial hardware support: QEMU/KVM virtual machine
5.0 Beta features & security
5.0 Beta features & security / TPM 2.0 Support
TPM 2.0 Support

Goal:

- **Transparent** (no user interaction)
  - encryption of writable system state partition
TPM 2.0 Support

Implementation:
- Complements existing Secure Boot support and Boot Chain Integrity
TPM 2.0 Support

- **Seal the encryption key** and provide it at boot time if machine in known-good state:
  - Rely on **PCR 7**: records measure of Secure Boot state
  - Expected Secure Boot state ⇒ we booted a trusted EFI binary (kernel + initramfs + cmdline)
TPM 2.0 Support

- Using other PCRs is easy (e.g. PCR 0 to measure firmware integrity), but requires some care to handle updates
- Use Intel’s implementation of the TPM2 Software Stack, from the initramfs: tpm2-tss library via tpm2-tools binaries (may change)
5.0 Beta features & security / Update support
Update model

Goals:

- **Client side:**
  - **safe:** applied while the system is online and in use
  - **in-background:** happen transparently to the user
  - **atomic:** list only valid options during boot
  - **rollback:** temporary fallback to a working version
Update model

Goals:

- **Client side:**
  - **safe**: applied while the system is online and in use
  - **in-background**: happen transparently to the user
  - **atomic**: list only valid options during boot
  - **rollback**: temporary fallback to a working version

- **Server side:**
  - client identification and version reporting
  - update channels
Update model

Goals:

- **Client side:**
  - safe: applied while the system is online and in use
  - in-background: happen transparently to the user
  - atomic: list only valid options during boot
  - rollback: temporary fallback to a working version

- **Server side:**
  - client identification and version reporting
  - update channels

Threats:

- Compromised update server
- Active man-in-the-middle attacker
- Active local attacker
Update support: Client

CLIP OS system layout:

- UEFI boot only, following the Boot Loader Specification
- A/B partition setup using Logical Volumes for system Read-Only partitions (for example: Core)
- Single partition setup for stateful partitions
Update support: Client

Implementation:
- Download the latest Core partition and EFI binary from the update server
Update support: Client

Implementation:

- Download the latest Core partition and EFI binary from the update server
- Verify download integrity
Update support: Client

Implementation:

- Download the latest Core partition and EFI binary from the update server
- Verify download integrity
- Remove the EFI binary associated with previous and soon unavailable version
Update support: Client

Implementation:

- Download the latest Core partition and EFI binary from the update server
- Verify download integrity
- Remove the EFI binary associated with previous and soon unavailable version
- Install the Core partition in the currently unused Logical Volume or create a new one if only one exists
## Update support: Client

<table>
<thead>
<tr>
<th>Bootloader</th>
<th>EFI system partition</th>
<th>LVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EFI version N</td>
<td>Core version N</td>
</tr>
<tr>
<td></td>
<td>EFI version N + 1</td>
<td>Core version N + 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Core state</td>
</tr>
</tbody>
</table>

### Implementation:

- Download the latest Core partition and EFI binary from the update server
- Verify download integrity
- Remove the EFI binary associated with previous and soon unavailable version
- Install the Core partition in the currently unused Logical Volume or create a new one if only one exists
- Install the EFI binary with a name following the Boot Loader Specification
# Update support: Client

**Implementation:**

- Download the latest Core partition and EFI binary from the update server
- Verify download integrity
- Remove the EFI binary associated with previous and soon unavailable version
- Install the Core partition in the currently unused Logical Volume or create a new one if only one exists
- Install the EFI binary with a name following the Boot Loader Specification
- Reboot the system to automatically boot the new version
Update support: Server

Initial version:

- Static files served over HTTPS
- Versioned directory layout

```
https://update.clip-os.org/
  +-- dist
  |   +-- 5.0.0-alpha.2
  |   |   +-- clipos-core, clipos-core.sig
  |   |   +-- clipos-efiboot, clipos-efiboot.sig
  +-- update
      +-- v1
          +-- clipos
              +-- version
```
Update support: Server

Initial version:
- Static files served over HTTPS
- Versioned directory layout

https://update.clip-os.org/
  +-- dist
    |   +-- 5.0.0-alpha.2
    |   +-- clipos-core, clipos-core.sig
    |   +-- clipos-efiboot, clipos-efiboot.sig
  +-- update
    +-- v1
      +-- clipos
      +-- version

Planned:
- Client statistics and version reporting
- Channel support
Update support: Security

Implemented:

- Client in Rust
- HTTPS with TLS 1.2+ only
- Root CA pinning
- Payload signatures using minisign
- Runtime rollback resistance (payload version stored with signature)
Update support: Security

Implemented:

- Client in Rust
- HTTPS with TLS 1.2+ only
- Root CA pinning
- Payload signatures using minisign
- Runtime rollback resistance (payload version stored with signature)

Unaddressed issues:

- Offline rollback resistance
- Update signing key compromise
Update support: Planned improvements

- Reduce client privileges (unprivileged network processing, etc.)
- Incremental updates using *casync*
- Bootloader update
- Free disk space checks
5.0 Beta features & security / IPsec support
IPsec support

- Isolation using network namespaces
- IPsec access using XFRM interfaces (similar to Wireguard)
IPsec support

- Latest strongSwan release (5.8.1):
  - Strict compile time configuration
  - Strict default strongSwan configuration
  - Confined unprivileged strongSwan daemon

- IPsec DR conformity in progress:
  - All available compile time and runtime configuration changes applied
  - All items requiring code changes and code review postponed to 5.0 stable

- IPsec aware nftables based firewalling:
  - Currently static rules generated at install time
  - Dynamically generated / template based rules postponed to 5.0 stable
5.0 Beta features & security / Linux kernel maintenance
Set of hardening patches initially maintained by Daniel Micay, many of them extracted from grsecurity/PaX
Now maintained internally, in collaboration with Arch Linux
Tends to shrink due to upstreamization, but some features regularly require time-consuming adaptations
ASLR improvements, memory sanitizing, slab cookies, a bit more __ro_after_init, etc.
Patches merged upstream

Former out-of-tree patch sets merged and maintained in CLIP OS but now available upstream:

- Lockdown (in v5.4, as an LSM)
- STACKLEAK (since v4.20)
Running a recent kernel

Pros:

- Quickly benefit from new features
  - Kernel hardening (e.g. `init_on_free`, `STRUCTLEAK_BYREF_ALL`)
  - Security mechanisms (e.g. `dm_verity`, `nf_tables`)
- Receive more stable backports, especially security fixes
- Constant but easier (and less error-prone) work to keep in sync
  - As opposed to CLIP OS v4: massive work required once upon a time to jump from one LTS to another

Cons:

- "Stable" kernels are far from being stable (but neither are LTS ones)
  - We uncover bugs, either in new features or due to uncompromising combinations and configurations that nobody seems to use nor test
  - Several bugs reported to upstream, as well as missing backports
5.0 Beta features & security / Other features
Other features

- Virtual testbed using Vagrant:
  - Includes test support for updates and IPsec
- Initial admin & audit roles (available over SSH)
- X260 hardware profile
- etc.
Project infrastructure
Project infrastructure / Code review (Gerrit)
Code review (Gerrit)

Gerrit:
- Powerful, Git-based, code review web application
- Deployed at: review.clip-os.org
Project infrastructure / Continuous Integration (GitLab CI)
Continuous Integration (GitLab CI)

Why GitLab?

- Lots of features (Git LFS, container registry, artifact storage, etc.)
- Compatible with offline development environment requirements (DR/CD)
- Gerrit deployment now optional
- Good documentation, lots of high profile users
- GitLab CI integration
Continuous Integration (GitLab CI)

Why GitLab?

- Lots of features (Git LFS, container registry, artifact storage, etc.)
- Compatible with offline development environment requirements (DR/CD)
- Gerrit deployment now optional
- Good documentation, lots of high profile users
- GitLab CI integration

Why GitLab CI?

- Jobs described with simple YAML file & (Bash) scripts
- Container based:
  - mostly Docker for now
  - podman support in GitLab 12.6 (expected on 2019-12-22)
- Scheduler / worker split
Continuous Integration (GitLab CI)

Public CI with GitLab.com (gitlab.com/CLIPOS/ci):

- Weekly "from scratch" builds
  - Build Debian based work container
  - Build everything else from scratch
  - Takes approximately 2 hours 20 min

- Daily "incremental" builds
  - Re-use container image
  - Re-use SDKs from latest successful build
  - Re-use binary packages from latest successful build
  - Takes approximately 35 min

- Build results (artifacts) available at files.clip-os.org

- Now very easy to try the latest version of CLIP OS in QEMU:
  docs.clip-os.org/toolkit/quick-try.html
5.0 stable: Roadmap
Roadmap: 5.0 stable

- Confined user environments (GUI)
- Multilevel support (Vserver-like LSM)
- Automated installation using PXE
- Fix all remaining issues required for qualification
Conclusion

CLIP OS 5 Beta:
- All the building blocks to create an IPsec gateway are now available
  - IPsec DR compatibility in progress, planned for final 5.0
- All the building blocks to create a server are now available
  - Update, IPsec client, Remote administration over SSH, etc.

Focus is now on user environments (GUI) and multi-level support:
- Use case 1: Mobile office workstation
- Use case 2: Remote administration workstation
Thanks!

✉️ clipos@ssi.gouv.fr
🌐 Website: clip-os.org
🌐 Docs: docs.clip-os.org
🌐 Sources: github.com/CLIPOS
🌐 Bugs: github.com/CLIPOS/bugs