Reduce the attack surface or keep the compatibility:

Lessons of sudo-rs and run0 transition plans

FOSDEM 2026

Alexander Bokovoy, Sr. Principal Software Engineer, Red Hat
Alejandro López, Sr. Software Engineer, Red Hat

Reducing Attack Surface

Reducing the Attack Surface

run0

- run0 aims for a system without SUID
  - Lets the service manager to handle the process setup
- Bundled with systemd v256+
  - Written in C
  - Uses polkit for authorization decisions

Reducing the Attack Surface

run0

```
[abokovoy@kitinen ~]$ run0 echo "Foo bar is bar"
Foo bar is bar
```

```
polkit-agent-helper-1[633978]: pam_sss(polkit-1:auth): authentication success;
                               logname=abokovoy uid=1000 euid=0 tty= ruser=abokovoy rhost= user=abokovoy
systemd-logind[1728]: New session 36 of user root.
systemd[1]: Created slice user-0.slice - User Slice of UID 0.
systemd[1]: Starting user-runtime-dir@0.service - User Runtime Directory /run/user/0...
systemd[1]: Finished user-runtime-dir@0.service - User Runtime Directory /run/user/0.
systemd[1]: Starting user@0.service - User Manager for UID 0...
systemd-logind[1728]: New session 37 of user root.
(systemd)[634002]: pam_unix(systemd-user:session): session opened for user root(uid=0) by root(uid=0)
uresourced[2099]: Setting resources on user-0.slice (MemoryMin: 0, MemoryLow: 0, CPUWeight: 100, IOWeight: 100)
uresourced[2099]: Setting resources on user@0.service (MemoryMin: 0, MemoryLow: 0, CPUWeight: 100, IOWeight: 100)
systemd[1]: Started user@0.service - User Manager for UID 0.
systemd[634002]: Reached target default.target - Main User Target.
systemd[634002]: Startup finished in 267ms.
systemd[1]: Started session-36.scope - Session 36 of User root.
(echo)[633994]: pam_unix(systemd-run0:session): session opened for user root(uid=0) by root(uid=0)
systemd[1]: Started run-p633948-i4826191.service - [run0] /usr/sbin/echo "Foo bar is bar".
systemd[1]: run-p633948-i4826191.service: Deactivated successfully.
systemd[634002]: Created slice session.slice - User Core Session Slice.
systemd-logind[1728]: Session 36 logged out. Waiting for processes to exit.
systemd[634002]: Starting dbus-broker.service - D-Bus User Message Bus...
systemd[1]: session-36.scope: Deactivated successfully.
systemd-logind[1728]: Removed session 36.
systemd[1]: Stopping user@0.service - User Manager for UID 0...
```

Reducing the Attack Surface

sudo-rs

- Written in Rust
  - Memory safety
  - Thread safety
  - Error handling
  - Strong typing

Large-Scale Deployments

Large-Scale Deployments

Enterprise & HPC

- FreeIPA: known deployments have up to a million enrolled hosts, hundred thousands sudo rules
- Centralized Identity repository
  - LDAP storage instead of a local (possibly huge) database files that are not practical to handle.
    - Organizational issues at scale always bit technical issues at scale
- Audit
  - Who and when escalates privileges?
  - What do they do with them?
- Generic rules
  - sudo added support for regexes and it is starting to be used.

Enabling run0/polkit for Large-Scale Deployments

Missing Features

- **Centrally-Managed actions and rules**
  - Today polkit action definitions are always local files
    - XML-based action definitions with little to no generalization
  - Today polkit authorization rules are always local files
    - Written in JavaScript
- **Authentication context mismatches**
  - Polkit runs PAM stack separate from the user session initiating the request
    - Kerberos credentials aren’t visible anymore → cannot apply decisions based on what credentials are present

Enabling sudo-rs for Large-Scale Deployments

Missing Features

- **Centrally-Managed rules**
  - Rules from an LDAP, AD, FreeIPA store, etc.
- Improved Audit
  - Audit what the users do once the root privileges are acquired
- Wildcards and regexes
  - Wildcards in the sudoers files are supported with some constraints.
  - Regexes are not supported.
- Other features
  - Upstream's note about differences from sudo

Next steps

Reality

- Use IPC boundary to separate data access from privileged operations
  - PAM: most PAM modules talk to their daemons over UNIX domain sockets already
  - Identity: NSS interface plugins are usually small shims that talk over UNIX domain sockets already
  - sudo/SSSD integration already talks over UNIX domain socket, no direct LDAP access
  - shadow utils/SSSD integration already talks over UNIX domain socket, no direct LDAP access
- systemd provides Varlink API to allow daemons and clients communicate using JSON payload (e.g. userdb API)

Next steps

- sudo-rs: a *Feature Request* for sudo-rs to implement a plug-in interface to allow reading the sudoers file not only from the local file.
  - https://github.com/trifectatechfoundation/sudo-rs/issues/1421
  - Developers are willing to discuss possible solutions, varlink is one of them
- polkit: backend infrastructure to allow alternative implementations is already present
  - The work on polkit seem to stuck, maybe run0 popularity could boost it?
  - Potential to reuse sudo-rs policy engine here?

Questions / Discussion