

# Security Features in systemd

NLUUG Najaarsconferentie 2014

Lennart Poettering  
lennart@poettering.net

November 2014

## Quick Recap of systemd Services

```
$ systemctl cat cups.service
# /usr/lib/systemd/system/cups.service
[Unit]
Description=CUPS Printing Service
After=network.target

[Service]
Type=notify
ExecStart=/usr/sbin/cupsd -f
PrivateTmp=true

[Install]
Also=cups.socket cups.path
WantedBy=printer.target
```

PrivateTmp=yes|no

`PrivateTmp=yes|no`

The service will get its own, private instance of `/tmp` and  
`/var/tmp`

PrivateTmp=yes|no

The service will get its own, private instance of /tmp and  
/var/tmp

Life-cycle of private /tmp and /var/tmp is bound to service  
runtime

PrivateTmp=yes|no

The service will get its own, private instance of /tmp and /var/tmp

Life-cycle of private /tmp and /var/tmp is bound to service runtime

/tmp and /var/tmp is not suitable for IPC across service boundaries if this is enabled!

`PrivateTmp=yes|no`

The service will get its own, private instance of `/tmp` and `/var/tmp`

Life-cycle of private `/tmp` and `/var/tmp` is bound to service runtime

`/tmp` and `/var/tmp` is not suitable for IPC across service boundaries if this is enabled!

Combine with `JoinNamespaceOf=`, to allow IPC with specific services anyway



PrivateDevices=yes|no

`PrivateDevices=yes|no`

The service will get its own, private instance of `/dev`

`PrivateDevices=yes|no`

The service will get its own, private instance of `/dev`

Only `/dev/null`, `/dev/random`, and similar pseudo-devices are available in the private `/dev`

`PrivateDevices=yes|no`

The service will get its own, private instance of `/dev`

Only `/dev/null`, `/dev/random`, and similar pseudo-devices are available in the private `/dev`

No access to real devices, such as hard disks (such as `/dev/sda` or `/dev/sdb7`)

PrivateNetwork=yes|no

`PrivateNetwork=yes|no`

The service will get its own, private network stack

`PrivateNetwork=yes|no`

The service will get its own, private network stack

Only a private loopback device (lo)

`PrivateNetwork=yes|no`

The service will get its own, private network stack

Only a private loopback device (lo)

No access to real devices, such as hard disks (more specifically, devices like `/dev/sda` or `/dev/sdb7`)



`PrivateNetwork=yes|no`

The service will get its own, private network stack

Only a private loopback device (lo)

No access to real devices, such as hard disks (more specifically, devices like `/dev/sda` or `/dev/sdb7`)

Combine with `JoinNamespaceOf=` for extra namespace magic!

ProtectSystem=yes|no|full

`ProtectSystem=yes|no|full`

The service will only get read-only access to `/usr` (as well as `/etc`, in case of `full`).

`ProtectSystem=yes|no|full`

The service will only get read-only access to `/usr` (as well as `/etc`, in case of `full`).

Undoable if the `CAP_SYS_ADMIN` capability is granted to the service (see below)

ProtectHome=yes|no|read-only

`ProtectHome=yes|no|read-only`

The service will not get any access to `/home` (or only read-only access, in case of `full`).

```
ReadOnlyDirectories=  
InaccessibleDirectories=
```

`ReadOnlyDirectories=`

`InaccessibleDirectories=`

Make specific directories read-only or inaccessible



`ReadOnlyDirectories=`

`InaccessibleDirectories=`

Make specific directories read-only or inaccessible

Manual versions of `ProtectSystem=` or `ProtectHome=`

MountFlags=slave

`MountFlags=slave`

Mounts and unmounts done by the service will not propagate to the rest of the system.

MountFlags=slave

Mounts and unmounts done by the service will not propagate to the rest of the system.

Implied by PrivateTmp=, PrivateDevices=, PrivateNetwork=,  
ProtectSystem=, ProtectHome=, ReadOnlyDirectories=,  
InaccessibleDirectories=.

CapabilityBoundingSet=

CapabilityBoundingSet=

Specify bounding set of capabilities, use it to run services with minimal capabilities.

CapabilityBoundingSet=

Specify bounding set of capabilities, use it to run services with minimal capabilities.

Namespace mounts/remounts can partially be undone unless CAP\_SYS\_ADMIN is dropped, hence consider using

CapabilityBoundingSet= when using PrivateTmp=,  
PrivateDevices=, PrivateNetwork=, ProtectSystem=,  
ProtectHome=, ReadOnlyDirectories=,  
InaccessibleDirectories=

...in particular when the service doesn't drop privileges on its own.

NoNewPrivileges=



`NoNewPrivileges=`

Disables UID, GID changes, acquiring of new capabilities and more.

NoNewPrivileges=

Disables UID, GID changes, acquiring of new capabilities and more.

setuid and setgid access mode bits on executable files and fcaps  
lose their power

DeviceAllow=

DeviceAllow=

Restrict access to specific device nodes

DeviceAllow=

Restrict access to specific device nodes

Example: DeviceAllow=/dev/sda5 rwm

DeviceAllow=

Restrict access to specific device nodes

Example: DeviceAllow=/dev/sda5 rwm

Example: DeviceAllow=char-alsa rw

SELinuxContext=

```
SELinuxContext=  
AppArmorProfile=
```



`SELinuxContext=`

`AppArmorProfile=`

Runs the service under a specific SELinux security context or AppArmor profile.

RestrictAddressFamilies=

`RestrictAddressFamilies=`

Restricts access to specific network socket families.

`RestrictAddressFamilies=`

Restricts access to specific network socket families.

Example: `RestrictAddressFamilies=AF_UNIX`

`RestrictAddressFamilies=`

Restricts access to specific network socket families.

Example: `RestrictAddressFamilies=AF_UNIX`

Example: `RestrictAddressFamilies=~AF_INET AF_INET6`

SystemCallArchitectures=

`SystemCallArchitectures=`  
Restricts access to system call architectures.

`SystemCallArchitectures=`

Restricts access to system call architectures.

Example: `RestrictAddressFamilies=x86 x86-64`



`SystemCallArchitectures=`

Restricts access to system call architectures.

Example: `RestrictAddressFamilies=x86 x86-64`

Example: `RestrictAddressFamilies=native`

SystemCallFilter=

SystemCallFilter=  
Limits access to specific system calls

User=, Group=, SupplementaryGroups=

User=, Group=, SupplementaryGroups=  
Runs a service under non-root user/group IDs.

User=, Group=, SupplementaryGroups=

Runs a service under non-root user/group IDs.

A lot of software does this on its own, use this for all other cases

`User=, Group=, SupplementaryGroups=`

Runs a service under non-root user/group IDs.

A lot of software does this on its own, use this for all other cases

Combine with `LimitNPROC=` for an effective `fork()` protection

LimitNFILE=0



LimitNFILE=0  
Disallow file creation

Outlook: BusPolicy=

Outlook: BusPolicy=

Restrict which bus names a service can access or even see

Outlook: BusPolicy=

Restrict which bus names a service can access or even see  
kdbus!

RootDirectory=

RootDirectory=  
Good old chroot()

All of systemd's own long-running services now make use of these security features, as applicable.

All of systemd's own long-running services now make use of these security features, as applicable.



Some of these features are now used by many Fedora packages by default.

Some of these features are now used by many Fedora packages by default.

Please help adding more of these security features to the various services by default.

Some of these features are now used by many Fedora packages by default.

Please help adding more of these security features to the various services by default.

In the distributions, upstream, and locally on your systems.

systemd

<http://0pointer.de/blog/projects/security.html>

<http://www.freedesktop.org/wiki/Software/systemd>

[git://anongit.freedesktop.org/systemd](https://anongit.freedesktop.org/systemd)

#systemd on [irc.freenode.org](https://freenode.org)