SELinux

Related articles

Security
AppArmor
TOMOYO Linux

Security-Enhanced Linux (SELinux) is a Linux feature that provides a variety of security policies, including U.S. Department of Defense style Mandatory Access Control (MAC), through the use of Linux Security Modules (LSM) in the Linux kernel. It is not a Linux distribution, but rather a set of modifications that can be applied to Unix-like operating systems, such as Linux and BSD.

Running SELinux under a Linux distribution requires three things: An SELinux enabled kernel, SELinux Userspace tools and libraries, and SELinux Policies (mostly based on the Reference Policy). Some common Linux programs will also need to be patched/compiled with SELinux features.

Contents

Current status in Arch Linux

SELinux is not officially supported (see [1] (https://lists.archlinux.org/pipermail/arch-general/2013-October/034352.html)[2] (https://lists.archlinux.org/pipermail/arch-general/2017-February/043149.html)). The status of unofficial support is:

Name	Status	Available at
SELinux enabled kernel	Implemented for `linux` (https://www.archlinux.org/packages/?name=linux), `linux-zen` (https://www.archlinux.org/packages/?name=linux-zen) and `linux-hardened` (https://www.archlinux.org/packages/?name=linux-hardened)	Available in official repositories since 4.18.8 (https://git.archlinux.org/svntogit/packages.git/commit/?id=c4660 9a4b0325c363455264844091b71de01eddc).
SELinux Userspace tools and libraries	Implemented in AUR: https://aur.archlinux.org/packages/?O=0&K=selinux	Work is done at https://github.com/archlinuxhardened/selinux
SELinux Policy	Work in progress, using Reference Policy (https://github.com/SELinuxProject/refpolicy) as upstream	Upstream: https://github.com/SELinuxProject/refpolicy (since release 20170805 the policy has integrated support for systemd and single-/usr/bin directory)

Summary of changes in AUR as compared to official core packages:

Name	Status and comments
linux	Need following kernel parameters at boot: `selinux=1 security=selinux`
linux-hardened	Need following kernel parameters at boot: `selinux=1 security=selinux`
coreutils	Need a rebuild with `--with-selinux` flag to link with libselinux
cronie	Need a rebuild with `--with-selinux` flag
dbus	Need a rebuild with `--enable-libaudit` and `--enable-selinux` flags
findutils	Need a rebuild with libselinux installed to enable SELinux-specific options
iproute2	Need a rebuild with `--with-selinux` flag
logrotate	Need a rebuild with `--with-selinux` flag
openssh	Need a rebuild with `--with-selinux` flag
pam	Need a rebuild with `--enable-selinux` flag for Linux-PAM ; Need a patch for pam_unix2, which only removes a function already implemented in a recent versions of libselinux
pambase	Configuration changes to add pam_selinux.so to `/etc/pam.d/system-login`
psmisc	Need a rebuild with `--with-selinux` flag
shadow	Need a rebuild with `--with-selinux` flags
sudo	Need a rebuild with `--with-selinux` flag
systemd	Need a rebuild with `--enable-audit` and `--enable-selinux` flags
util-linux	Need a rebuild with `--with-selinux` flag

All of the other SELinux-related packages may be included without changes nor risks.

Concepts: Mandatory Access Controls

Note: This section is meant for beginners. If you know what SELinux does and how it works, feel free to skip ahead to the installation.

Before you enable SELinux, it is worth understanding what it does. Simply and succinctly, SELinux enforces *Mandatory Access Controls (MACs)* on Linux. In contrast to SELinux, the traditional user/group/rwx permissions are a form of *Discretionary Access Control (DAC)*. MACs are different from DACs because security policy and its execution are completely separated.

An example would be the use of the *sudo* command. When DACs are enforced, sudo allows temporary privilege escalation to root, giving the process so spawned unrestricted systemwide access. However, when using MACs, if the security administrator deems the process to have access only to a certain set of files, then no matter what the kind of privilege escalation used, unless the security policy itself is changed, the process will remain constrained to simply that set of files. So if *sudo* is tried on a machine with SELinux running in order for a process to gain access to files its policy does not allow, it will fail.

Another set of examples are the traditional (-rwxr-xr-x) type permissions given to files. When under DAC, these are user-modifiable. However, under MAC, a security administrator can choose to freeze the permissions of a certain file by which it would become impossible for any user to change these permissions until the policy regarding that file is changed.

As you may imagine, this is particularly useful for processes which have the potential to be compromised, i.e. web servers and the like. If DACs are used, then there is a particularly good chance of havoc being wreaked by a compromised program which has access to privilege escalation.

For further information, visit **wikipedia:Mandatory access control**.

Installing SELinux

Package description

All SELinux related packages belong to the *selinux* group in the AUR.

SELinux aware system utilities

`coreutils-selinux (https://aur.archlinux.org/packages/coreutils-selinux/)` AUR
: Modified coreutils package compiled with SELinux support enabled. It replaces the `coreutils (https://www.archlinux.org/packages/?name=coreutils)` package

`cronie-selinux (https://aur.archlinux.org/packages/cronie-selinux/)` AUR
: Fedora fork of Vixie cron with SELinux enabled. It replaces the `cronie (https://www.archlinux.org/packages/?name=cronie)` package.

`dbus-selinux (https://aur.archlinux.org/packages/dbus-selinux/)` AUR
: An SELinux aware version of **D-Bus**. It replaces the `dbus (https://www.archlinux.org/packages/?name=dbus)` package.

findutils-selinux (https://aur.archlinux.org/packages/findutils-selinux/) AUR
: Patched findutils package compiled with SELinux support to make searching of files with specified security context possible. It replaces the findutils (https://www.archlinux.org/packages/?name=findutils) package.

iproute2-selinux (https://aur.archlinux.org/packages/iproute2-selinux/) AUR
: iproute2 package compiled with SELinux support; for example, it adds the `-Z` option to `ss` . It replaces the iproute2 (https://www.archlinux.org/packages/?name=iproute2) package.

logrotate-selinux (https://aur.archlinux.org/packages/logrotate-selinux/) AUR
: Logrotate package compiled with SELinux support. It replaces the logrotate (https://www.archlinux.org/packages/?name=logrotate) package.

openssh-selinux (https://aur.archlinux.org/packages/openssh-selinux/) AUR
: OpenSSH package compiled with SELinux support to set security context for user sessions. It replaces the openssh (https://www.archlinux.org/packages/?name=openssh) package.

pam-selinux (https://aur.archlinux.org/packages/pam-selinux/) AUR and pambase-selinux (https://aur.archlinux.org/packages/pambase-selinux/) AUR
: PAM package with pam_selinux.so. and the underlying base package. They replace the pam (https://www.archlinux.org/packages/?name=pam) and pambase (https://www.archlinux.org/packages/?name=pambase) packages respectively.

psmisc-selinux (https://aur.archlinux.org/packages/psmisc-selinux/) AUR
: Psmisc package compiled with SELinux support; for example, it adds the `-Z` option to `killall` . It replaces the psmisc (https://www.archlinux.org/packages/?name=psmisc) package.

shadow-selinux (https://aur.archlinux.org/packages/shadow-selinux/) AUR
: Shadow package compiled with SELinux support; contains a modified `/etc/pam.d/login` file to set correct security context for user after login. It replaces the shadow (https://www.archlinux.org/packages/?name=shadow) package.

sudo-selinux (https://aur.archlinux.org/packages/sudo-selinux/) AUR
: Modified sudo package compiled with SELinux support which sets the security context correctly. It replaces the sudo (https://www.archlinux.org/packages/?name=sudo) package.

systemd-selinux (https://aur.archlinux.org/packages/systemd-selinux/) AUR
: An SELinux aware version of Systemd. It replaces the systemd (https://www.archlinux.org/packages/?name=systemd) package.

util-linux-selinux (https://aur.archlinux.org/packages/util-linux-selinux/) AUR
: Modified util-linux package compiled with SELinux support enabled. It replaces the util-linux (https://www.archlinux.org/packages/?name=util-linux) package.

SELinux userspace utilities

`checkpolicy (https://aur.archlinux.org/packages/checkpolicy/)` AUR
: Tools to build SELinux policy

`mcstrans (https://aur.archlinux.org/packages/mcstrans/)` AUR
: Daemon which is used by libselinux to translate MCS labels

`libselinux (https://aur.archlinux.org/packages/libselinux/)` AUR
: Library for security-aware applications. Python bindings needed for *semanage* and *setools* now included.

`libsemanage (https://aur.archlinux.org/packages/libsemanage/)` AUR
: Library for policy management. Python bindings needed for *semanage* and *setools* now included.

`libsepol (https://aur.archlinux.org/packages/libsepol/)` AUR
: Library for binary policy manipulation.

`policycoreutils (https://aur.archlinux.org/packages/policycoreutils/)` AUR
: SELinux core utils such as newrole, setfiles, etc.

`restorecond (https://aur.archlinux.org/packages/restorecond/)` AUR
: Daemon which maintains the label of some files

`secilc (https://aur.archlinux.org/packages/secilc/)` AUR
: Compiler for SELinux policies written in CIL (Common Intermediate Language)

`selinux-dbus-config (https://aur.archlinux.org/packages/selinux-dbus-config/)` AUR
: DBus service which allows managing SELinux configuration

`selinux-gui (https://aur.archlinux.org/packages/selinux-gui/)` AUR
: SELinux GUI tools (system-config-selinux)

`selinux-python (https://aur.archlinux.org/packages/selinux-python/)` AUR **and** `selinux-python2 (https://aur.archlinux.org/packages/selinux-python2/)` AUR
: SELinux python tools and libraries (semanage, sepolgen, sepolicy, etc.)

`selinux-sandbox (https://aur.archlinux.org/packages/selinux-sandbox/)` AUR
: Sandboxing tool for SELinux

`semodule-utils (https://aur.archlinux.org/packages/semodule-utils/)` AUR
: Tools to handle SELinux modules when building a policy

SELinux policy packages

`selinux-refpolicy-src (https://aur.archlinux.org/packages/selinux-refpolicy-src/)` AUR
: Reference policy sources

`selinux-refpolicy-git (https://aur.archlinux.org/packages/selinux-refpolicy-git/)` AUR
: Reference policy git master (**https://github.com/SELinuxProject/refpolicy**) built

with configuration specific for Arch Linux

selinux-refpolicy-arch (https://aur.archlinux.org/packages/selinux-refpolicy-arch/) AUR
Precompiled modular Reference policy with headers and documentation but without sources. Development Arch Linux Refpolicy patches included, which fixes issues related to path labeling and systemd support. These patches are also sent to Reference Policy maintainers and their inclusion in **selinux-refpolicy-arch (https://aur.archlinux.org/packages/selinux-refpolicy-arch/)** AUR is mainly a way to perform updates between Refpolicy releases.

Other SELinux tools

setools (https://aur.archlinux.org/packages/setools/) AUR
CLI and GUI tools to manage SELinux

selinux-alpm-hook (https://aur.archlinux.org/packages/selinux-alpm-hook/) AUR
pacman hook to label files accordingly to SELinux policy when installing and updating packages

Installation

There are two methods to install the requisite SELinux packages.

Via AUR

- First, install SELinux userspace tools and libraries, in this order (because of the dependencies): **libsepol (https://aur.archlinux.org/packages/libsepol/)** AUR, **libselinux (https://aur.archlinux.org/packages/libselinux/)** AUR, **secilc (https://aur.archlinux.org/packages/secilc/)** AUR, **checkpolicy (https://aur.archlinux.org/packages/checkpolicy/)** AUR, **setools (https://aur.archlinux.org/packages/setools/)** AUR, **libsemanage (https://aur.archlinux.org/packages/libsemanage/)** AUR, **semodule-utils (https://aur.archlinux.org/packages/semodule-utils/)** AUR, **policycoreutils (https://aur.archlinux.org/packages/policycoreutils/)** AUR, **selinux-python (https://aur.archlinux.org/packages/selinux-python/)** AUR (which depends on **python-ipy (https://www.archlinux.org/packages/?name=python-ipy)**), **mcstrans (https://aur.archlinux.org/packages/mcstrans/)** AUR and **restorecond (https://aur.archlinux.org/packages/restorecond/)** AUR.
- Then install **pambase-selinux (https://aur.archlinux.org/packages/pambase-selinux/)** AUR and **pam-selinux (https://aur.archlinux.org/packages/pam-selinux/)** AUR and make sure you can login again after the installation completed, because files in `/etc/pam.d/` got removed and created when **pambase (https://www.archlinux.org/packages/?name=pambase)** got replaced with **pambase-selinux (https://aur.archlinux.org/packages/pambase-selinux/)** AUR.
- Next you can recompile some core packages by installing: **coreutils-selinux (https://aur.archlinux.org/packages/coreutils-selinux/)** AUR, **findutils-selinux (https://aur.archlinux.org/packages/findutils-selinux/)** AUR, **iproute2-selinux (https://aur.archlinux.org/packages/iproute2-selinux/)** AUR, **logrotate-selinux (https://aur.archlinux.org/packages/logrotate-selinux/)** AUR, **openssh-selinux (https://aur.archlinux.org/packages/openssh-selinux/)** AUR, **psmisc-selinux (https://aur.archlin**

ux.org/packages/psmisc-selinux/)[AUR], shadow-selinux (https://aur.archlinux.org/packages/shadow-selinux/)[AUR], cronie-selinux (https://aur.archlinux.org/packages/cronie-selinux/)[AUR]

- Next, backup your `/etc/sudoers` file. Install sudo-selinux (https://aur.archlinux.org/packages/sudo-selinux/)[AUR] and restore your `/etc/sudoers` (it is overridden when this package is installed as a replacement of sudo (https://www.archlinux.org/packages/?name=sudo)).
- Next come util-linux and systemd. Because of a cyclic makedepends between these two packages which will not be fixed (FS#39767 (https://bugs.archlinux.org/task/39767)), you need to build the source package systemd-selinux (https://aur.archlinux.org/packages/systemd-selinux/)[AUR], install systemd-libs-selinux (https://aur.archlinux.org/packages/systemd-libs-selinux/)[AUR], build and install util-linux-selinux (https://aur.archlinux.org/packages/util-linux-selinux/)[AUR] (with libutil-linux-selinux (https://aur.archlinux.org/packages/libutil-linux-selinux/)[AUR]) and rebuild and install systemd-selinux (https://aur.archlinux.org/packages/systemd-selinux/)[AUR].
- Next, install dbus-selinux (https://aur.archlinux.org/packages/dbus-selinux/)[AUR].
- Next, install selinux-alpm-hook (https://aur.archlinux.org/packages/selinux-alpm-hook/)[AUR] in order to run restorecon every time pacman installs a package.

After all these steps, you can install a SELinux kernel (like linux-selinux (https://aur.archlinux.org/packages/linux-selinux/)[AUR]) and a policy (like selinux-refpolicy-arch (https://aur.archlinux.org/packages/selinux-refpolicy-arch/)[AUR] or selinux-refpolicy-git (https://aur.archlinux.org/packages/selinux-refpolicy-git/)[AUR]).

Using the GitHub repository

All packages are maintained at https://github.com/archlinuxhardened/selinux . This repository also contains a script named `build_and_install_all.sh` which builds and installs (or updates) all packages in the needed order. Here is an example of a way this script can be used in a user shell to install all packages (with downloading the GPG keys which are used to verify the source tarballs of the package):

```
git clone https://github.com/archlinuxhardened/selinux
cd selinux
./recv_gpg_keys.sh
./build_and_install_all.sh
```

Of course, it is possible to modify the content of `build_and_install_all.sh` before running it, for example if you already have SELinux support in your kernel.

Changing boot loader configuration

If you have installed a new kernel, make sure that you update your bootloader accordingly to boot on it. Moreover you may need to add `security=selinux selinux=1` to the kernel command line. More precisely, if the kernel configuration does not set `CONFIG_DEFAULT_SECURITY_SELINUX`, `security=selinux` is needed, and if it contains `CONFIG_SECURITY_SELINUX_BOOTPARAM=y` `CONFIG_SECURITY_SELINUX_BOOTPARAM_VALUE=0`, `selinux=1` is needed.

GRUB

Add `security=selinux selinux=1` to `GRUB_CMDLINE_LINUX_DEFAULT` variable in `/etc/default/grub` Run the following command:

```
# grub-mkconfig -o /boot/grub/grub.cfg
```

Syslinux

Change your syslinux.cfg file by adding:

```
/boot/syslinux/syslinux.cfg
```

```
LABEL arch-selinux
        LINUX ../vmlinuz-linux-selinux
        APPEND root=/dev/sda2 ro security=selinux selinux=1
        INITRD ../initramfs-linux-selinux.img
```

at the end. Change "linux-selinux" to whatever kernel you are using.

systemd-boot

Create a new loader entry, for example in `/boot/loader/entries/arch-selinux.conf` :

```
/boot/loader/entries/arch-selinux.conf
```

```
title Arch Linux SELinux
linux /vmlinuz-linux-selinux
initrd /initramfs-linux-selinux.img
options root=/dev/sda2 ro selinux=1 security=selinux
```

Checking PAM

A correctly set-up **PAM** is important to get the proper security context after login. Check for the presence of the following lines in `/etc/pam.d/system-login` :

```
# pam_selinux.so close should be the first session rule
session         required        pam_selinux.so close
```

```
# pam_selinux.so open should only be followed by sessions to be executed in the user context
session         required        pam_selinux.so open
```

Installing a policy

Warning: The reference policy as given by **SELinuxProject (https://github.com/SELinuxProject/refpolicy/wiki)** is not very good for Arch Linux, as before release 20170805 almost no file were labelled correctly. The major problems were:

- `/lib` and `/usr/lib` were considered different (and also `/bin` , `/sbin` , `/usr/bin` and `/usr/sbin`). This introduced some instability when applying labels to the whole system, as files in these folders might be seen with 2 (or 4) different labels.

- systemd was not yet supported (C. PeBenito, main developer of the refpolicy, announced its willingness to work on it in its github repository in October 2014, http://oss.tresys.com/pipermail/refpolicy/2014-October/007430.html)

Since refpolicy release 20170805 these two points have been addressed, but most people submitting patches to improve the policy use an other distribution (Debian, Gentoo, RHEL, etc.). Therefore the compatibility with Arch Linux packages is not perfect (for example the policy may not support the most recent features of a program).

Policies are the mainstay of SELinux. They are what govern its behaviour. The only policy currently available in the AUR is the Reference Policy. In order to install it, you should use the source files, which may be got from the package **selinux-refpolicy-src (https://aur.archlinux.org/packages/selinux-refpolicy-src/)**[AUR] or by downloading the latest release on **https://github.com/SELinuxProject/refpolicy/wiki/DownloadRelease#current-release**. When using the AUR package, navigate to `/etc/selinux/refpolicy/src/policy` and run the following commands:

```
# make bare
# make conf
# make install
```

to install the reference policy as it is. Those who know how to write SELinux policies can tweak them to their heart's content before running the commands written above. The command takes a while to do its job and taxes one core of your system completely, so do not worry. Just sit back and let the command run for as long as it takes.

To load the reference policy run:

```
# make load
```

Then, make the file `/etc/selinux/config` with the following contents (Only works if you used the defaults as mentioned above. If you decided to change the name of the policy, you need to tweak the file):

/etc/selinux/config

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#                   Set this value once you know for sure that SELinux is configured the way yo
u like it and that your system is ready for deployment
#       permissive - SELinux prints warnings instead of enforcing.
#                    Use this to customise your SELinux policies and booleans prior to deploym
ent. Recommended during policy development.
#       disabled - No SELinux policy is loaded.
#                  This is not a recommended setting, for it may cause problems with file label
ling
SELINUX=permissive
# SELINUXTYPE= takes the name of SELinux policy to
# be used. Current options are:
#       refpolicy (vanilla reference policy)
#       <custompolicy> - Substitute <custompolicy> with the name of any custom policy you choo
se to load
SELINUXTYPE=refpolicy
```

Now, you may reboot. After rebooting, run:

```
# restorecon -r /
```

to label your filesystem.

Now, make a file `requiredmod.te` with the contents:

requiredmod.te

```
module requiredmod 1.0;

require {
        type devpts_t;
        type kernel_t;
        type device_t;
        type var_run_t;
        type udev_t;
        type hugetlbfs_t;
        type udev_tbl_t;
        type tmpfs_t;
        class sock_file write;
        class unix_stream_socket { read write ioctl };
        class capability2 block_suspend;
        class dir { write add_name };
        class filesystem associate;
}

#============= devpts_t ==============
allow devpts_t device_t:filesystem associate;

#============= hugetlbfs_t ==============
allow hugetlbfs_t device_t:filesystem associate;

#============= kernel_t ==============
allow kernel_t self:capability2 block_suspend;

#============= tmpfs_t ==============
allow tmpfs_t device_t:filesystem associate;

#============= udev_t ==============
allow udev_t kernel_t:unix_stream_socket { read write ioctl };
allow udev_t udev_tbl_t:dir { write add_name };
allow udev_t var_run_t:sock_file write;
```

and run the following commands:

```
# checkmodule -m -o requiredmod.mod requiredmod.te
# semodule_package -o requiredmod.pp -m requiredmod.mod
# semodule -i requiredmod.pp
```

This is required to remove a few messages from `/var/log/audit/audit.log` which are a nuisance to deal with in the reference policy. This is an ugly hack and it should be made very clear that the policy so installed simply patches the reference policy in order to hide the effects of incorrect labelling.

Testing in a Vagrant virtual machine

It is possible to use **Vagrant** to provision a virtual Arch Linux machine with SELinux configured. This is a convenient way to test an Arch Linux system running SELinux without modifying a current system. Here are commands which can be used to achieve this:

```
git clone https://github.com/archlinuxhardened/selinux
cd selinux/_vagrant
vagrant up
vagrant ssh
```

Post-installation steps

You can check that SELinux is working with `sestatus`. You should get something like:

```
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             refpolicy
Current mode:                   permissive
Mode from config file:          permissive
Policy MLS status:              disabled
Policy deny_unknown status:     allowed
Max kernel policy version:      28
```

To maintain correct context, you can use *restorecond*:

```
# systemctl enable restorecond
```

To switch to enforcing mode without rebooting, you can use:

```
# echo 1 > /sys/fs/selinux/enforce
```

Swapfiles

If you have a swap file instead of a swap partition, issue the following commands in order to set the appropriate security context:

```
# semanage fcontext -a -t swapfile_t "/path/to/swapfile"
# restorecon /path/to/swapfile
```

Working with SELinux

SELinux defines security using a different mechanism than traditional Unix access controls. The best way to understand it is by example. For example, the SELinux security context of the apache homepage looks like the following:

```
$ls -lZ /var/www/html/index.html
-rw-r--r--  username username system_u:object_r:httpd_sys_content_t /var/www/html/index.html
```

The first three and the last columns should be familiar to any (Arch) Linux user. The fourth column is new and has the format:

```
user:role:type[:level]
```

To explain:

1. **User:** The SELinux user identity. This can be associated to one or more roles that the SELinux user is allowed to use.
2. **Role:** The SELinux role. This can be associated to one or more types the SELinux user is allowed to access.
3. **Type:** When a type is associated with a process, it defines what processes (or domains) the SELinux user (the subject) can access. When a type is associated with an object, it defines what access permissions the SELinux user has to that object.
4. **Level:** This optional field can also be know as a range and is only present if the policy supports MCS or MLS.

This is important in case you wish to understand how to build your own policies, for these are the basic building blocks of SELinux. However, for most purposes, there is no need to, for the reference policy is sufficiently mature. However, if you are a power user or someone with very specific needs, then it might be ideal for you to learn how to make your own SELinux policies.

This (http://www.fosteringlinux.com/tag/selinux/) is a great series of articles for someone seeking to understand how to work with SELinux.

Troubleshooting

The place to look for SELinux errors is the **systemd journal**. In order to see SELinux messages related to the label `system_u:system_r:policykit_t:s0` (for example), you would need to run:

```
# journalctl _SELINUX_CONTEXT=system_u:system_r:policykit_t:s0
```

Useful tools

There are some tools/commands that can greatly help with SELinux.

restorecon
: Restores the context of a file/directory (or recursively with `-R`) based on any policy rules

chcon
: Change the context on a specific file

Reporting issues

Please report issues on GitHub: **https://github.com/archlinuxhardened/selinux/issues**

See also

- **Security Enhanced Linux**
- **Gentoo SELinux Handbook (https://wiki.gentoo.org/wiki/SELinux)**
- **Fedora Project's SELinux Wiki (https://fedoraproject.org/wiki/SELinux)**
- **NSA's Official SELinux Homepage (https://www.nsa.gov/what-we-do/research/selinux/)**
- **SELinux Project Homepage (https://github.com/SELinuxProject)**

    - **Reference Policy Homepage (https://github.com/SELinuxProject/refpolicy/wiki)**
    - **SETools Homepage (https://github.com/SELinuxProject/setools/wiki)**
- **ArchLinux, SELinux and You (archived) (https://web.archive.org/web/20140816115906/http://jamesthebard.net/archlinux-selinux-and-you-a-trip-down-the-rabbit-hole/)**
