Security Enhanced Linux (SELinux) - https://selinuxproject.org/

https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/using_selinux/index

- Provides an extra layer of security to resources in Linux systems
- Runs as a loadable kernel module, gets a pseudofilesystem mount point like /proc
- Restricts access by subjects (users, processes) to objects (files) by applying and using labels for them.
- Separates users, processes and objects into sandboxes, "confined" domains, and one sandbox for everything else (unconfined_t). i.e., as "targets" httpd and ntpd have their own sandboxes that are isolated from eachother
- If a targeted process tries to access resources outside it's confined domain, access is denied and it's logged.
- Provides mandatory access controls (MACs) to extend the basic Linux discretionary access controls (DACs)
- MAC-based checks happen AFTER the DAC-based checks
- Stores MAC permissions in extended attributes of file system, attaching SELinux "labels"
- An "access vector cache" (AVC) stores decisions made (allow/disallow access) to speed up performance during runtime

Side note: a special pain about SELinux is it doesn't install manpages (!) This will provide *some*: sepolicy manpage -a -p /usr/share/man/man8 after installing the policycoreutils-devel package. This issue has never been resolved for over a decade now.

Basic Concept

Type Enforcement (TE)

This is the foundation of SELinux. Things are given context labels, and rules to say how those things can interact.

- Context, applied to subject types (processes) and object types (files, resources) to define security relations between them. The "objects" can be devices, network interfaces, addresses, ports, sockets, (and many things defined in /proc)

- Rules dictate access control by specifying permissions between subject types (domains) and object types. They determine whether a subject with a specific context is allowed or denied access to objects based on their own contexts.

Security Levels: MCS and MLS (optional security enhancements)

These are optional mostly. If you don't change them they will simply be using default values (so not usually a concern).

- Multi-Category Security (MCS): Utilizes functional or departmental categories for access control within an organization.
- Multi-Level Security (MLS): Utilizes security clearance levels or sensitivity classifications (Bell-La Padula model)

Since these are mostly optional, they will be covered later in this document. These also get inspected after type enforcement.

SELinux Modes

- Multiple modes of SELinux functionality can be applied on a system:
    - permissive - permission is granted, but denials are logged to /var/log/messages (for testing)
    - enforcing - strictly enforces 'targeted' policy rules (default)
    - disabled - only basic DACs are used

Using **getenforce** simply reports current mode: enforcing, permissive, or disabled; **sestatus** gives more details:

```
# sestatus
   SELinux status:          enabled
   SELinuxfs mount:         SELinux
   Current mode:            enforcing      -- current mode of operation
   Mode from config file:   permissive     -- mode set by /etc/sysconfig/SELinux
```

To change temporarily, setenforce 0 (permissive) or setenforce 1 (enforce) -or- "echo 1 > /SELinux/enforce"
To set the persistent mode, edit the file /etc/SELinux/config (symlink to /etc/sysconfig/SELinux)

```
>>> cat /etc/selinux/config
# SELINUX= can take one of these three values:
#    enforcing - SELinux security policy is enforced.
#    permissive - SELinux prints warnings instead of enforcing.
#    disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these three values:
#    targeted - Targeted processes are protected,
#    minimum - Modification of targeted policy. Only selected processes are protected.
#    mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

Generally advice is don't disable SELinux. Some installs may call for it, but permissive mode should usually be enough.
If disabled, unless temporarily or to make changes to non-critical running items, you generally would restart after setting enforcing in /etc/sysconfig/SELinux so it will take effect after checking/relabling the system (same with switching from enforcing to disabled)
Switching between enforcing and permissive does not have that limitation (getenforce/ setenforce)
On startup, you can also switch the settings for SELinux in grub:

```
        kernel /vmlinuz-2.6.32-279.el6.x86_64 root=/dev/md3 SELinux=1 enforcing=0
```

SELinux=0 is disabled, 1 is enabled, and with the enforcing setting, permissive is 0 and 1 is enforcing

SELinux mode commands: setenforce, getenforce, sestatus

Viewing a file's SELinux context: Labels
Several commands take the -Z option to display SELinux contexts; output of ls shows us files/directories (objects):

```
$ ls -Z file1
        -rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
# ls -Z /var/www/html/file5
        -rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file5
ls -al -Z /var/www/html
        drwxr-xr-x. 2 apache root system_u:object_r:httpd_sys_content_t:s0 4096 Dec 23 20:47 .
```

These examples, the familiar user, group, DAC permissions and filename are shown, but adding the -Z option, we can see the context labels SELinux provides : a user (unconfined_u), a role (object_r), a type (user_home_t), and a level (s0).
- User labels: Non-privileged user = user_u ; Privileged user = root_u
- Role-based labels: Non-privileged and users = user_r, system_r
- Type/domain labels: 12 default protected daemons: httpd, ntpd, dhcpd, mysqld, named, nscd, portmap, postgres, snmp, squid, winbind, syslogd. All others (unless customized) get unconfined_t domain
- Levels (s0) are part of the MCS/MLS options I am saving till the end to keep this simple.

id -Z	Shows user's security context
ps -Z	Shows context for running processes (subjects and thier sandboxes/ domains/analogous to namespaces).
cp -Z, mv -Z	Maintains/preserves the security context when copying/moving files.
mkdir -Z	Sets the security context for newly created directories.
netstat -Z, ss -Z	Displays SELinux context information for network connections.

Using semanage to configure SELinux
Common options: -a, --add; -d, --delete; -m, --modify; -l, --list, -import and -export <filename> to input or output your configs

File context definitions	Add fcontext for all in /web	semanage fcontext -a -t httpd_sys_content_t "/web(/.*)?"
Network port type defs	Allow Apache to listen on tcp port 81	semanage port -a -t http_port_t -p tcp 81
Network interface type	List all interface definitions	semanage interface -l | grep eth*
Network node type	semanage node -a -t node_t -p ipv4 -M 255.255.255.0 192.168.1.0	
Manage policy modules	Install custom apache module	semanage module -a myapache

login - Manage login mappings; user - Manage confined users (roles and levels); boolean - Manage booleans; dontaudit - Disable/Enable dontaudit rules; ibpkey - infiniband pkey type definitions; ibendport - infiniband end port type definitions

Use-case example: You want your SSH host keys in /data/keys. You create the directory, move all the keys into the new home and change the sshd_config file to match the new mapping. When you attempt to use SSH, it fails.

```
    semanage fcontext -l | grep sshd     - outputs:
        /etc/ssh/primes regular file system_u:object_r:sshd_key_t:s0
        /etc/ssh/ssh_host.*_key regular file system_u:object_r:sshd_key_t:s0
        /etc/ssh/ssh_host.*_key.pub regular file system_u:object_r:sshd_key_t:s0
    semanage fcontext -a -t sshd_key_t   '/data/keys/*.*'
    restorecon -r /data/keys
```

Important! semanage only changes the policy: Use restorecon afterward to actually label the filesystem.
The tools chcon/chcat do NOT make persistent changes! These are only useful for temporary changes for testing.

Relabeling files and the filesystem
Restores filesystem to permissions(labels), according to what is specified in /etc/SELinux/targeted/policy/
Create hidden .autorelabel file at root of filesystem- gets init to relabel on startup

```
        touch / .autorelabel && reboot
```

If you don't want to reboot, using restorecon or fixfiles command will do the relabeling, - **however** - *existing processes may remain running in incorrect and insecure domains, and it will ask to empty /tmp/ since it can't relabel it, as root, temporary files that applications are relying upon are trashed.   Instead work* on specific directories or processes rather than the entire filesystem
- Use fixfiles to restore contexts of files by the package that installed them with '/sbin/fixfiles -R package_name'
- Or, use '/sbin/restorecon -Rv /directory/path' -R is recursive, use -n to looks for changed files but won't make changes

If a file is moved and restorecon is run on it, it will be given permissions of it's parent directory
Generally, if a file (object) does not have specific fc/te specified, it inherts that of the enclosing directory.

On a SysVinit machine, /usr/sbin/run_init ensures protected daemon isolation and sets up proper contexts for services during system startup. After making changes to SELinux settings or encountering processes running outside proper contexts, you would kill the parent process and then run /usr/sbin/run_init again for the affected service. (such as '/usr/sbin/run_init /etc/init.d/httpd')
On a systemd machine, after modifying SELinux settings for a process, run load_policy or restorecon, and restart it with a systemctl restart <service_name>.

If you make ANY changes to SELinux that are global (i.e. booleans or installing new SELinux binaries), you will have to restart all processes, so a reboot would be necessary

SELinux Booleans
Booleans conveniently permit runtime adjustments to SELinux without the need to modify or reload the policy, and activate specific functionalities on processes seamlessly.

semanage boolean -l *List of them with descriptions, if on or off, and the default value*

```
SELinux boolean              State  Default    Description
ftp_home_dir                 (off,   off)      Allow ftp to read and write files in the user home directories
xdm_sysadm_login             (off,   off)      Allow xdm logins as sysadm
xen_use_nfs                  (off,   off)      Allow xen to manage nfs files
ssh_chroot_rw_homedirs       (off,   off)      Allow ssh with chroot env to read and write files in the user home directories
postgresql_can_rsync         (off,   off)      Allow postgresql to use ssh and rsync for point-in-time recovery
authlogin_shadow             (off,   off)      Allow users login programs to access /etc/shadow.
httpd_can_network_relay      (off,   off)      Allow httpd to act as a relay
openvpn_enable_homedirs      (on,    on)       Allow openvpn to read home directories
```

getsebool -a | grep httpd *Provides a different view*

```
     httpd_builtin_scripting --> on
     httpd_can_network_connect --> off
```

To change a value with setsebool: setsebool httpd_enable_cgi off ---- Make persistent with -P (on and off can be 0 or 1)
Changing booleans persistently might not be done with semanage boolean, so use setsebool -P instead

Disabling Specific (Targeted) Policies While Running
We need to edit the items in the /booleans directory and toggle the boolean:
echo "1 1" > /SELinux/booleans/http_disable_trans
The file commit_pending_bools is monitored by SELinux to see if it needs to refresh the policies
echo "1" > /SELinux/commit_pending_bools
Restart the affected service: /sbin/service httpd restart
Apache is now running in the unconfined_t domain

MLS/MCS Levels Explained
The order of operations in SELinux is as follows:
1.) DAC (Discretionary Access Control, regular Linux permissions) are considered by the host OS
2.) TE (Type Enforcement, the most basic SELinux operation) is inspected
3.) MLS (Multi-Level Security sensitivity labels determine access control, i.e., secret, top secret)
4.) MCS (check Multi-Category Security for access control based on categories)

```
 ~ ] # semanage login -l
Login Name      SELinux User      MLS/MCS Range
_default__      unconfined_u      s0-s0:c0.c1023
root            unconfined_u      s0-s0:c0.c1023
system_u        system_u          s0-s0:c0.c1023
```

Login Name column lists Linux users, and the SELinux User column lists which SELinux user the Linux user is mapped to. For processes, the SELinux user limits which roles and levels are accessible. Finally, the ranges of MLS/MCS access are listed

An MLS range is a pair of levels, written as lowlevel-highlevel, or if the levels are identical- for example- s0-s0 is the same as s0. Each level is a sensitivity-category pair, with categories being optional. If there are categories, the level is written as sensitivity:category-set. If there are no categories, it is written as sensitivity.

A contiguous series of categories can be abbreviated, such as c0.c3 means c0,c1,c2,c3.
The /etc/SELinux/targeted/setrans.conf file maps levels (s0:c0) to human-readable form (ie. CompanyConfidential). This file needs to have changes to it made by semanage rather than manually edited.

MLS sensitivity levels range from s0 as the least to s15 as the most sensitive.
Default unconfigured SELinux has s0-s0:c0.c1023, with MLS level s0 authorized for all categories.
MCS has up to 1024 different categories: c0 through to c1023.

MLS is based on the Bell-La Padula MAC model, used in Labeled Security Protection Profile (LSPP) environments. You have to install the package [e.g., dnf install selinux-policy-mls], and configure MLS to be the default SELinux policy. This is still incomplete- you have to configure it specifically for your needs and it won't have what it needs for your programs- upstream SELinux Reference Policy can be built that is more inclusive but MLS isn't something you can just unbox and it's ready to go. A full discussion of it is our of the scope of this writing for that reason.

Star utility for backup (SELinux tar)
Tar does not archive security context labels. Star has it's own package: star-1.5a25-6.i386.rpm
star -xattr -H=exustar -c -f newarchive.star foldername/ ---extended attributes, -c create, -f for normal
star -xattr -x -f newarchive.star --- -x to extract

Logging
System calls are filtered through SELinux policy to see if allowed.
If not allowed, an avc:denied message is generated goes through auditd, which writes event to /var/log/audit/audit.log (config file is /etc/audit/auditd.conf) If SELinux is in enforcing, action stopped, if permissive it is allowed, but logged. For example, when a web browser asks Apache for /foo/index.html, a getattr /foo/index.html syscall is issued. If it has the wrong label, SELinux enforcing stops it there. You can also see the getattr referred to in the AVC alerts in audit.log

For a sample denial in logs, running 'grep AVC /var/log/audit/audit.log' you'll see something like this:

```
type=AVC msg=audit(1711932009.640:1125): avc: denied { open } for pid=7237 comm="httpd" path="/var/www/html/index.html" dev="dm-1"
ino=28668713 scontext=system_u:system_r:httpd_t:s0 tcontext=system_u:object_r:httpd_sys_content_t:s0 tclass=file
```

Deciphering the line above:
Type of message (AVC) followed by epoch timestamp, there has been an AVC denial on {open} operation.
Open was denied for PID 7237, where the command (comm) is related to httpd. The denied access occurred at path "/var/www/html/index.html" on device "dm-1", with inode number 28668713.
The source context (scontext) is system_u:system_r:httpd_t:s0, representing the httpd program. The target context (tcontext) is system_u:object_r:httpd_sys_content_t:s0, indicating the SELinux label associated with the target file.
Both contexts follow the SELinux label format (*_u, *_r, *_t for user, role, and type), with the source context representing the program (httpd) and the target context representing the type of item being accessed (httpd_sys_content_t).
In this case, httpd has access to files labeled as system_u:system_r:httpd_t:s0, but the file being accessed has a label of system_u:object_r:httpd_sys_content_t:s0, which doesn't match.
So if you run 'sealert -a /var/log/audit/audit.log', you'll see something like this.

```
SELinux is preventing httpd (httpd_t) from { open } access on the file index.html.
*****  Plugin httpd (72.4 confidence) suggests   ************
If you want to allow httpd to open index.html file
Then you need to change the file context to httpd_sys_content_t.
Do
# semanage fcontext -a -t httpd_sys_content_t '/var/www/html/index.html'
# restorecon -v '/var/www/html/index.html'
*****
Allowing Access:
Do
# semanage fcontext -a -t httpd_sys_content_t '/var/www/html/index.html'
# restorecon -v '/var/www/html/index.html'
Additional Information:
Source Context          system_u:system_r:httpd_t:s0
Target Context          system_u:object_r:httpd_sys_content_t:s0
Target Objects          /var/www/html/index.html [ file ]
```

AuditD Tools

```
# aureport -a
AVC Report
==================================================
#  date        time      comm  subj    syscall class   perm    obj     result   event
========================================================================
1. 02/16/2020 20:52:51  ?      (null)  0       (null)  (null)  (null)  unset    745
2. 02/16/2020 22:35:35  ?      (null)  0       (null)  (null)  (null)  unset    1391
3. 02/21/2020 10:29:41  httpd  system_u:system_r:httpd_t:s0 49 tcp_socket name_bind system_u:object_r:websm_port_t:s0 denied 1144
4. 02/21/2020 10:29:41  httpd  system_u:system_r:httpd_t:s0  49  tcp_socket name_bind    system_u:object_r:websm_port_t:s0    denied
```

Logged events in the past 3 days	aureport --start 04/08/2024 00:00:00 --end 04/11/2024 00:00:00
All executable file events	aureport -x
Summarize executable file events	aureport -x --summary
Failed events for all users	aureport -u --failed --summary -i
All failed login attempts per each system user	aureport --login --summary -i
All audit files that are queried and times they included	aureport -t
ausearch userID's file access events, make report	ausearch --start today --loginuid 1000 --raw | aureport -f --summary

aulast	List of last logged-in users with login times	aulast
aulastlog	Last login information of users	aulastlog
ausyscall	Converts between system call #'s and names	ausyscall --name 3 or ausyscall --number open
auvirt	List virtualization-related audit records	auvirt
auditctl	Control the kernel's audit system config	auditctl -a always,exit -F arch=b64 -S unlink
augenrules	Generates rules from text file for the audit framework	auegenrules /etc/audit/audit.rules
aureport	Generates summary reports from audit logs	aureport --summary
ausearch	Searches the audit logs for specific events	ausearch -ua 500
autrace	Traces execution of a program, capturing system calls	autrace /bin/ls

Troubleshooting Using sealert: Be Careful! Sometimes You Get Bad Advice
This example was used in Sander Van Vugt's videos for RHEL 7. One of the better examples of why you need to be careful.

```
[root@localhost]# ssh -p 2022 localhost
ssh: connect to host localhost port 2022: Connection refused
[root@localhost]# ssh -p 443 localhost
ssh: connect to host localhost port 443: Connection refused
[root@localhost]# lsof -l
        COMMAND    PID    USER   FD    TYPE    DEVICE    NODE    NAME
        sshd       3538   root   3u    IPv4    31495     TCP     192.168.4.172:5-192.168.4.1:59438 (ESTABLISHED)
        sshd       3538   root   8u    IPv6    31867     TCP     localhost:x11-ssh-offset (LISTEN)
        sshd       3538   root   9u    IPv4    31868     TCP     localhost:x11-ssh-offset (LISTEN)
[root@localhost]# grep AVC /var/log/audit/audit.log
type=AVC msg=audit(1425663361.745:487): avc: denied {name bind} for pid=4555 com="sshd" src=443 scontext-
system_u:system_r:sshd_t:s0-s0:c.c1023 tcontext-system_u:object_r:http_port_t:s0 tclass=tcp_socket

[root@localhost]# sealert -a /var/log/audit/audit.log
Mar 6 12:48:22 localhost dbus [868]: [system] Successfully activated service 'org.fedoraproject.Setroubleshootd'
Mar 6 12:40:23 localhost setroubleshoot: Plugin Exception restorecon_source
Mar 6 12:40:23 localhost setroubleshoot: SELinux is preventing /usr/sbin/sshd from name_bind access on the tcp_socket. For complete SELinux
messages. run sealert -l 88dc1625-8b9e-4a8f-ad9e-4412068fe9ac
Mar 6 12:48:23 localhost python: SELinux is preventing /usr/sbin/sshd from name_bind access on the tcp_socket.
*****      Plugin catchall (100. confidence) suggests      ********************
If you believe that sshd should be allowed name_bind access on the tcp_socket by default.
Then you should report this as a bug.
You can generate a local policy module to allow this access.
Do
allow this access for now by executing:
# grep sshd /var/log/audit/audit.log | audit2allow -M mypol
# semodule -l mypol.pp
Mar 6 12:40:23 localhost setroubleshoot: SELinux is preventing /usr/sbin/sshd from name_bind access on the tcp_socket For complete SELinux
messages. run sealert-1 88dc1d25-8b9e-4a8f-ad9e-4412068fe9ac
```

So there's this socket error. Some options are in "semanage port"
-a is add, -m is modify so for 443 its semanage -m -t sshd_t -p tcp 443
You can also list all port definitions and grep for port (semanage port -l | grep port) and you'll find "Allow sshd to listen on tcp port 8991
#semanage port -a -t ssh_port_t -p tcp 8991"

But think first: what sealert suggested is a blanket allow policy with module to allow all traffic of a particular type if we look at /var/log/messages, we sometimes get more data, but it's telling us to get more info by running sealert -l 188dc1d25-8b9e-4a8f-ad9e-4412068fe9ac, so we'll try that.

```
SELinux is preventing /usr/sbin/sshd from name bind access on the tcp_socket
*****      Plugin catchall (100. confidence) suggests      ********************
If you believe that sshd should be allowed name bind access on the tcp socket by default. Then you should report this as a bug.
You can generate a local policy module to allow this access.
Do
allow this access for now by executing:
# grep sshd /var/log/audit/audit.log | audit2allow -M mypol
# semodule -l mypol.pp
Additional Information:
Source Context                system_u:system_risshd_tise-se:ce.c1023
Target Context                 system u:object r:http_port_t:se
Target Objects                 [  tcp_socket  ]
Source                        sshd
Source Path                   /usr/sbin/sshd
Port                          443
Host                          localhost.localdomain
Source RPM Packages           openssh-server-6.4p1-8.el7.x86_64
Target RPM Packages
Policy RPM                    selinux-policy-3.12.1-153.el7.noarch
                              (trimmed output)
```

What do we get? The same bad advice: to make a policy module change that allows a bunch of junk permissions that shouldn't be there- AND it's 100% confident this is the right answer, after all! This shows the danger this can cause.
Think hard before trusting what the advice sealrt gives. Usually seaudit will give a few options, one *might* be acceptable

SELinux files and locations

 - /etc/SELinux/targeted
This directory contains config files specific to the targeted SELinux policy. (policy modules, contexts, and configs)
 - /etc/SELinux/targeted/policy contains compiled binaries of policies
 - /etc/SELinux/targeted/contexts contains exactly that, i.e. the file default_type cats out system_r:unconfined_t

The /etc/SELinux/targeted/contexts/file_contexts files are what holds the default maps of directories and files to labels.
Syntax for file_contexts content= regexp [-type] (context | <<none>>)
Examples ("--" means a file instead of a directory (-d), and -c for block or character special files) :

```
/home/[^/]+          -d          system_u:object_r:user_home_dir_t
/home/[^/]+/.+                   system_u:object_r:user_home_t
/mnt/[^/]*/.*                    <<none>>
```

The /selinux pseudofilesystem (mounted similarly as /proc) exposes runtime SELinux data, like current security context of processes and files, via virtual files. Is used by the OS and SELinux-aware commands to interact with, obtain status and config info

The /etc/SELinux/targeted/src/ directory is created when you install the selinux-policy-targeted-sources package.
 - /etc/SELinux/targeted/src/policy --- source tree --- contains .fc (file context) and .te (type enforcement) files
 - /etc/SELinux/targeted/src/policy/file_contexts --- has source info for building the file_contexts file (for files, resources
 - /etc/SELinux/targeted/src/policy/file_contexts/program/ contains fc files for specific programs, commands (processes)
 - /etc/SELinux/targeted/src/policy/domains --- individual domains/ contexts, the rules for specific programs or services
 - /etc/SELinux/targeted/src/policy/modules --- for modules for policy rules for specific functionalities or components

Creating Policies for Unsupported Software/ Items

Custom policies are needed if you have a program not represented by default, or to change the defaults (i.e., httpd).
You need to operate on the source of the targeted policy to make customizations.
Running "rpm -qa | grep SELinux" Brings us SELinux-policy-targeted-x.xx.xx These are just the binaries- we need to run rpm -Uvh SELinux-policy-targeted-source-x.xx.xx.rpm" to get the source files.

The installed /etc/SELinux/targeted/src/ directory is where you can start working.
Files named *.fc contain file context definitions, those named *.te contain SELinux policy, Type Enforcement (TE) rules

Here are the fundamental steps for compiling and installing a custom policy:
1. Edit then compile the .te files into a binary policy module (.mod file) and do error checking
```
checkmodule -M -m -o <module_name>.mod <module_name>.te
```
2. Package the *.mod file into a policy module package (.pp file) The -f option specifies a *.fc file to use
```
semodule_package -o <module_name>.pp -m <module_name>.mod -f <module_name>.fc
```
3. Install the *.pp file into the system's running policy directory's module directory:
```
sudo semodule -i <module_name>.pp
```

Once the .pp file is compiled and installed, the file context info in the *.pp file is accessed to label the filesystem when it is relabeled and that's it.

The *.fc file entries map file paths to SELinux security labels. Here are some examples for httpd:
```
/usr/bin/httpd httpd_exec_t  # assigns httpd_exec_t label to the httpd binary
/var/www/html httpd_sys_content_t  # assigns httpd_sys_content_t to the web root
/var/log/httpd httpd_log_t # assigns httpd_log_t to the log files
```
Other options: read access to configuration files (httpd_config_t), to content files (httpd_content_t); write access to log files (httpd_log_t), network access to specific ports (tcp_port_t)

The *te file entries define how processes with a specific label (e.g., httpd_t) can interact with labeled objects (files, network sockets)
Syntax is allow | neverallow subject object:object_class {permissions}
```
allow httpd_t httpd_sys_content_t { read write append };  # Allow httpd to modify web server content
allow httpd_t tcp_socket connectto port 80, 443;  # Allow httpd to connect to web ports
allow httpd_t var_log_t { write append };  # Allow httpd to write to log files
allow httpd_suexec_t self:capability { setuid setgid };  #Allow to gain elevated privileges for CGI scripts
bool httpd_enable_ftp_server false;  # Boolean for if httpd can run an FTP server
if (httpd_enable_ftp_server) {
 allow httpd_t ftp_port_t:tcp_socket name_bind;   #Allow httpd to bind to the port 21
}
```

/etc/SELinux/: Primary configuration directory.
/etc/sysconfig/SELinux/: A symlink to /etc/SELinux/config which dictates default mode and policy
/etc/sysconfig/SELinux/restorecond.conf: Used for restoring contexts on objects.
/etc/sysconfig/SELinux/semanage.conf: Config file for the semanage utility.
/etc/SELinux/targeted/modules/active/booleans.local: Location for local boolean settings.
/etc/SELinux/targeted/booleans: Directory for SELinux boolean settings.
libsemanage - Library provides an API for the manipulation of SELinux binary policies.

SELINUX COMMANDS AND PACKAGES

Pkg	Command	Description	Example
☾	system-config-selinux	GUI for configuring policies and settings	system-config-selinux
☆	sesearch	Searches policies for rules matching specified criteria	Find allow rules for Apache- sesearch -A -s httpd_t -p all
◇	sealert	View SELinux-related alerts and recommendations	sealert
○	audit2why	Explain AVC denial messages.	audit2why < AVC_denial_message
△	setfiles	Set default contexts based on file context info stored in the SELinux policy	setfiles -v /path/to/directory
△	restorecon	Relabels files to their default values (or changed by semanage, etc)	restorecon file.txt
△	restorecon_xattr	Restores SELinux extended attributes of files and directories	restorecon_xattr file.txt
□	avcstat	Displays average AVC statistics	avcstat
✶	sedta	Performs domain transition analyses on a policy file	sedta -f policy_file
✶	seinfoflow	Performs detailed information flow analysis	seinfoflow -d /usr/sbin/httpd
POLICY CREATION AND MANAGEMENT			
☾	selinux-polgengui	GUI for generating SELinux policies	(see graphic interface it gives you)
☆	apol	GUI to browse policy (types, classes, roles, users), rules (TE, RBAC, MLS)	(see graphic interface it gives you)
△	semodule	Manage policy modules (install, upgrade, listing, removing)	To install my_module.pp: semodule -i my_module.pp
☆	sechecker	Check SELinux policy for errors and common mistakes	sechecker /path/to/policy
☆	sediff	Compare two policies, reports differences	sediff policy1 policy2
☆	seinfo	Show info about policies, types, and attributes	seinfo /path/to/policy
△	load_policy	Load new SELinux policy into the kernel	load_policy /etc/selinux/targeted/policy/policy
△	sepolicy (semodule is better suited for this)	Manage policies, including loading, querying, and modifying policy rules	sepolicy <subcmd> <policy_rule> (there are query generate compile load and list options)
△	sepolgen	Generate policy interfaces based on input files	sepolgen input_file > output_file
△	sepolgen-ifgen	Generates interfaces in a similar format to sepolgen	sepolgen-ifgen existing_policy.pp > interfaces.cfg
○	audit2allow	Converts SELinux AVC denial messages into policy allow rules	Make rules from AVC denials in denials.log - audit2allow -i denials.log
△	semodule_link	Link a policy module into the current policy	semodule_link -i my_module
△	secon	Convert binary policy files to text	secon -t <policy.bin >policy.txt
△	semodule_expand	Expand modularized policy (pp) into one flat policy file (te)	semodule_expand -o my_policy.te my_module.pp
△	semodule_package/ semodule_unpackage	Create policy module package from current policy source file (or unpackage)	semodule_package -o my_policy.pp / semodule_unpackage my_module.pp
✧	macro-expander	Expands and shows macros used in policy files	macro-expander /path/to/policy
✧	checkmodule	Check module source file for errors, generate binary module	checkmodule -M -m -o /path/to/module.mod /path/to/module.te
✧	checkpolicy	Check policy source file for errors, generate a binary policy file	checkpolicy -M -c /path/to/policy.conf
✧	sedismod	Disassemble a binary policy module	sedismod /path/to/module.mod
✧	sedispol	Disassembles a binary SELinux policy	sedispol /path/to/policy.conf
CONTEXT AND CONTEXT CONFIG TOOLS			
□	selinuxconlist	List SELinux contexts.	selinuxconlist -l
□	selinuxdefcon	Displays the default SELinux context	selinuxdefcon
□	selinuxexeccon	Displays security context of a program (requires full path to program)	selinuxexeccon /bin/netstat
□	getpidprevcon	Get previous security context used by specified process	getpidprevcon 1234
□	matchpathcon	Checks if a file or directory has the correct SELinux context	matchpathcon /path/to/file
□	selabel_lookup	Get security context associated with a specified path	selabel_lookup /path/to/file
□	selabel_lookup_best_match	Find the best-matching security context for a specified path	selabel_lookup_best_match /path/to/file
□	selabel_partial_match	Checks if a path partially matches any SELinux file context	selabel_partial_match /path/to/file
□	selinux_check_access	Checks access against the loaded SELinux policy	selinux_check_access -a target_type -t source_type -p permission
□	validatetrans	Checks if any file with a source type is allowed to transition to the target type	validatetrans -t target_type -s source_type
△	genhomedircon	Generate SELinux file context config for home directories	genhomedircon -r /etc/selinux/targeted/contexts/files/file_contexts
□	sefcontext_compile	Compile file context config files into binary	sefcontext_compile
□	selabel_digest	Compute SHA-256 hash of a specified file's security contexts	selabel_digest /path/to/file
□	selabel_get_digests_all_partial_matches	Get hashes of contexts that partially match a path	selabel_get_digests_all_partial_matches /path/to/partial

△ policycoreutils + -devel, □ libselinux-utils, ○ policycoreutils-python-utils, ◇ setroubleshoot-server, ☾ policycoreutils-gui, ☆ setools-console, ▽ checkpolicy, ✧ selinux-policy-devel, ✶ setools-console-analyses