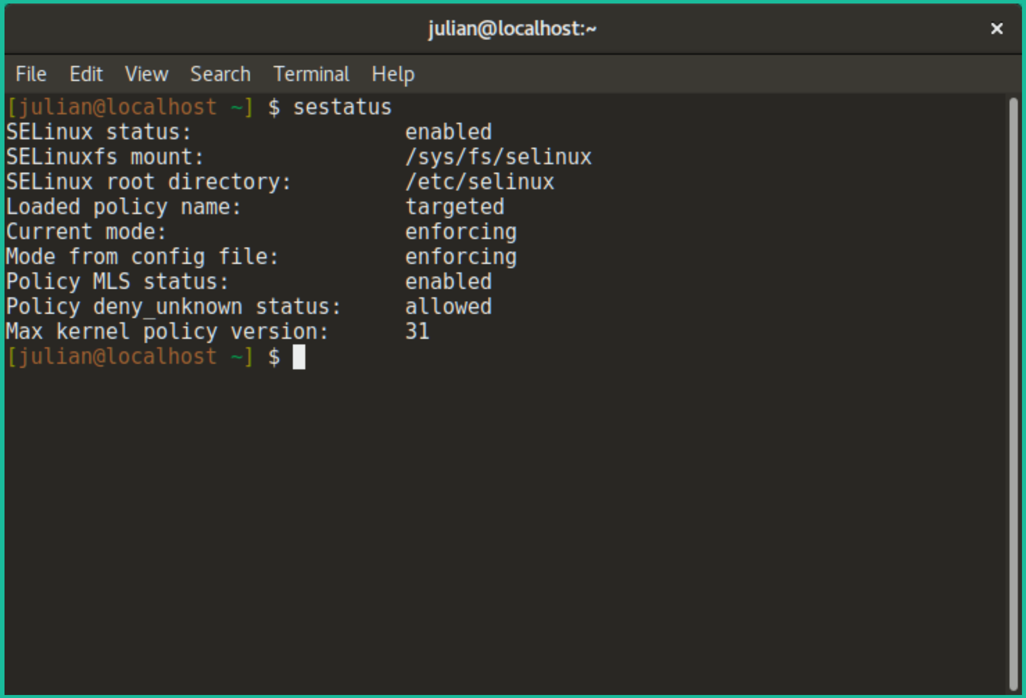


```
[julian@localhost ~] $ sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Max kernel policy version:      31
[julian@localhost ~] $
```

Secure Enhanced Linux

Julian Richen

SELinux?

- **Started as a research project from the National Security Agency (NSA)**
- **A set of patches using the Linux Security Modules (LSM)**
  - Hardening GNU/Linux systems with extra security policies and enforcing Mandatory Access Control (MAC)
  - Similar to modules like AppArmor, Smack, TOMOYO
- **NSA published the code under the GPL in 2000**
- **Upstream Linux kernel adopted patches in 2003**

Who develops it?

- **NSA**
- **Red Hat**
- **MITRE Corporation**
- **Secure Computing Corporation (SSC)**
- **Individual contributors & companies**
  - CUPS Project, SAMBA Project, IBM, Tresys Technology, and more
- **Full list:**
  - https://www.nsa.gov/what-we-do/research/selinux/contributors.shtml

Source?

- **Source:**
  - https://github.com/SELinuxProject/selinux
- **Bugs**
  - NSA: selinux@tycho.nsa.gov
  - Red Hat: https://bugzilla.redhat.com/
- **Policies**
  - https://github.com/TresysTechnology/refpolicy

Who uses it?

- **Linux Distros**
  - RHEL, Fedora, SuSE, CentOS, Debian, Ubuntu
- **United States Government**
  - NSA, DoD, etc...
- **Enterprise**
  - Data sensitive companies, healthcare, or anyone really
- **Android**
  - Google implemented SELinux in Android 4.3 (2015)

What does it solve?

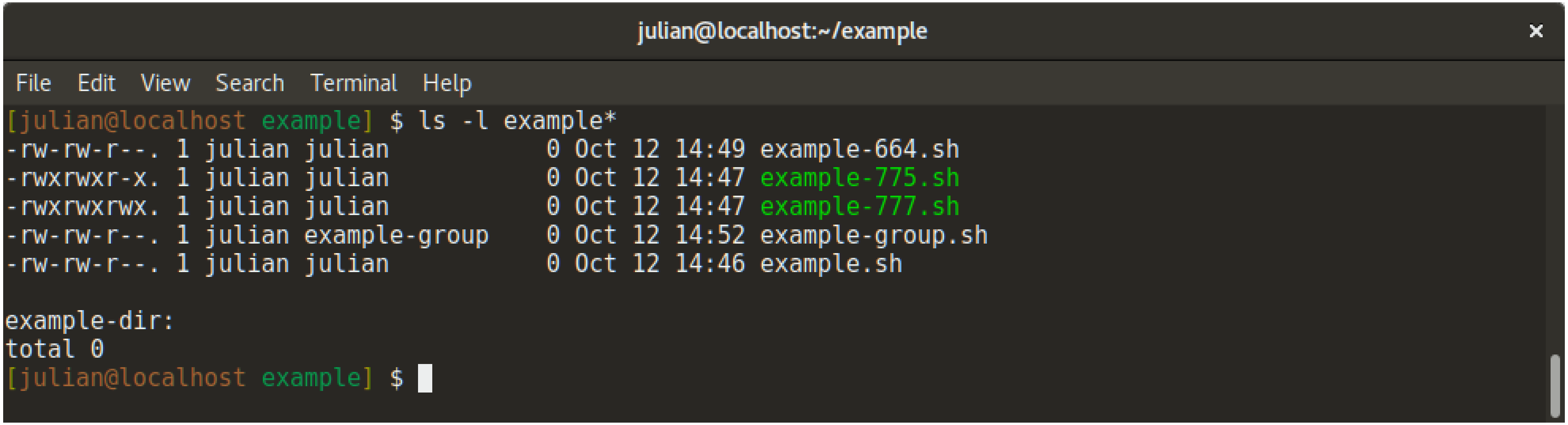
- **Implements Mandatory Access Control (MAC)**
  - Focus on process context instead of role-based security (think DAC)
  - Enhances Discretionary Access Control (DAC); aka Ownership (user, group, other) with read/write/exec permissions
- **MAC policies can be set for:**
  - Users
  - Files
  - Directories
  - Memory
  - Sockets
  - tcp/udp ports
  - And more!

Discretionary Access Controls

- Access to objects is restricted based on the identity of a subject and/or group (ownership + permissions).
- Controls are “discretionary” because subjects have a level of permissions that allow them to reach a subject.

Discretionary Access Controls

User			Group			Other		
r	w	x	r	w	x	r	w	x



```
[julian@localhost example] $ ls -l example*
-rw-rw-r--. 1 julian julian         0 Oct 12 14:49 example-664.sh
-rwxrwxr-x. 1 julian julian         0 Oct 12 14:47 example-775.sh
-rwxrwxrwx. 1 julian julian         0 Oct 12 14:47 example-777.sh
-rw-rw-r--. 1 julian example-group  0 Oct 12 14:52 example-group.sh
-rw-rw-r--. 1 julian julian         0 Oct 12 14:46 example.sh

example-dir:
total 0
[julian@localhost example] $
```

Mandatory Access Control

- Operating Systems constrain the ability of the subject to access or perform operation on an object or target.
- Basically, access to objects is restricted based on the security levels set by the security context.

How does SELinux work?

- **It’s basically Mandatory Access Control**
  - SELinux doesn’t replace DAC, MAC can work alongside DAC
  - SELinux can be enabled/disabled at anytime and system will fallback to DAC
- **SELinux uses “Labels” for MAC**
  - These labels are then followed with “Type Enforcement”
  - SELinux needs extended attributes on file-system to work
    - Labels are added as extended attributes
- **Use or make security policies**
  - Security policies are just pre-made lists of labels for lots of packages on a GNU/Linux system
  - SELinux ships with targeted, minimum and mls as defaults.

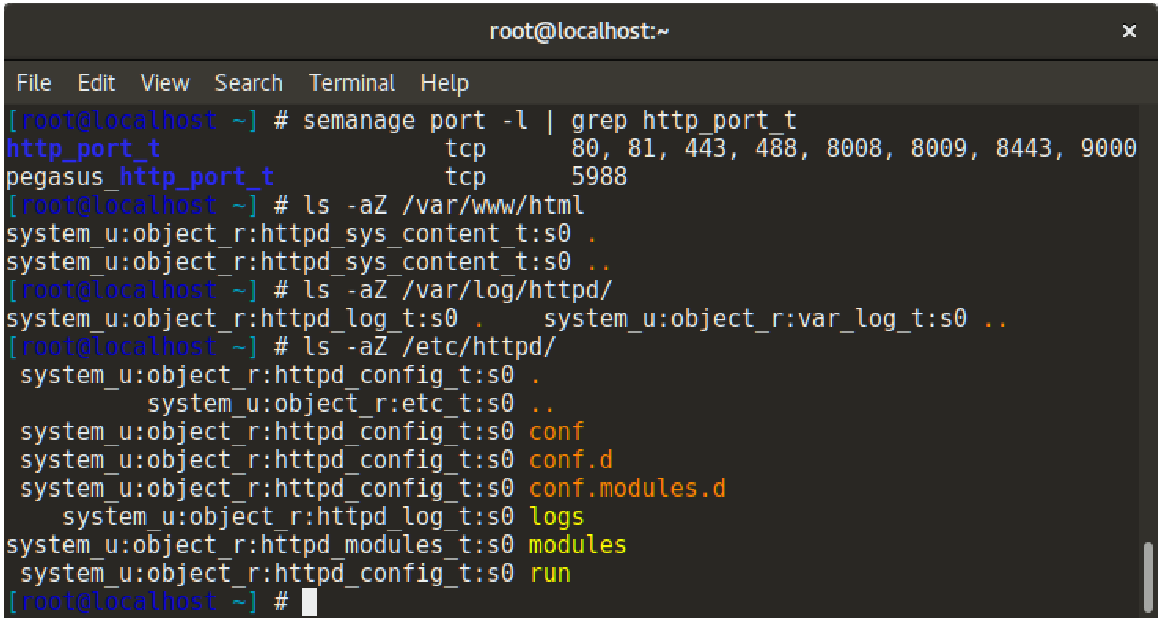
Labeling & Type Enforcement

- **Labeling**
  - Every object (file, process, port, etc..) has a SELinux context/label
    - Label’s job is to create logical groups/levels which the object may interact with
  - Format
    - user:role:type:level(optional)
  - Labels should be logical, e.g a http servers & ports 80/443 should be grouped together because a http will use those ports
- **Type Enforcement**
  - The part of the policy that says a subject with *“abc label”* can interact with an object with *“xyz label”.*

Label & Type Enforcement Example

- **It makes sense that httpd_* labeled objects should interact together.**
- **It doesn’t make sense for httpd labeled content to access sensitive files like /etc/shadow or files in the home directory.**

Object	label
httpd process	httpd
/usr/bin/httpd	httpd_exec_t
/etc/httpd/	httpd_config_t
/var/log/httpd/	httpd_log_t
/var/www/html/	httpd_sys_content_t
Port 80 & 443	httpd_port_t
/etc/shadow	shadow_t
/home/<user>/*	user_home_t

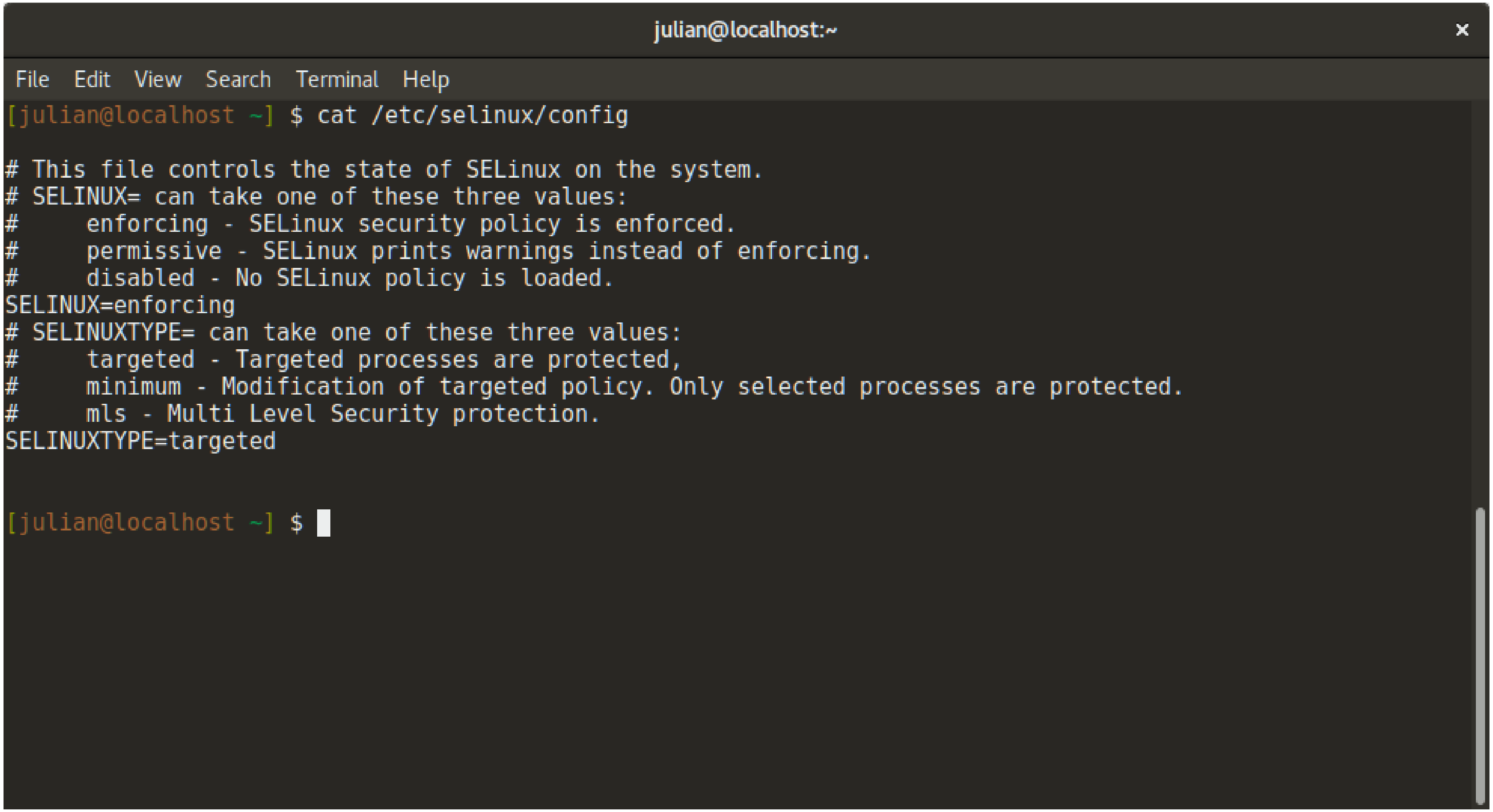


```
[root@localhost ~] # semanage port -l | grep http_port_t
http_port_t                    tcp      80, 81, 443, 488, 8008, 8009, 8443, 9000
pegasus_http_port_t            tcp      5988
[root@localhost ~] # ls -aZ /var/www/html
system_u:object_r:httpd_sys_content_t:s0 .
system_u:object_r:httpd_sys_content_t:s0 ..
[root@localhost ~] # ls -aZ /var/log/httpd/
system_u:object_r:httpd_log_t:s0 .    system_u:object_r:var_log_t:s0 ..
[root@localhost ~] # ls -aZ /etc/httpd/
 system_u:object_r:httpd_config_t:s0 .
          system_u:object_r:etc_t:s0 ..
 system_u:object_r:httpd_config_t:s0 conf
 system_u:object_r:httpd_config_t:s0 conf.d
 system_u:object_r:httpd_config_t:s0 conf.modules.d
    system_u:object_r:httpd_log_t:s0 logs
system_u:object_r:httpd_modules_t:s0 modules
 system_u:object_r:httpd_config_t:s0 run
[root@localhost ~] # 
```

SELinux Policies

- **Policy**
  - `Enforcing`
    - Enforce all policies.
  - `Permissive`
    - Prints warnings instead of enforcing.
  - `Disabled`
    - No policy is loaded.
- **Types**
  - `Targeted`
    - Support a greater number of confined daemons, can confine other users and areas. Good confinement for most use-cases.
  - `Minimum`
    - Support minimal set of confined daemons, rest are set as unconfined. Used for users to test SELinux and devices that only need to confine a few daemons.
  - `MLS`
    - Multi Level Security protection, lots of confined daemons and users. Used in high-security environments (think Government).
  - Write your own
    - You can write policies that fit your machine, business, etc…

cat /etc/selinux/config



```
[julian@localhost ~] $ cat /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.
#     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


[julian@localhost ~] $ 
```

Attributions

- **Docs on SELinux source**
  - https://github.com/SELinuxProject/selinux
- **Red Hat’s Thomas Cameron yearly SELinux presentation:**
  - http://people.redhat.com/tcameron/Summit2017/SElinux/selinux_for_mere_mortals_2017.pdf
- **Fedora docs**
  - https://docs-old.fedoraproject.org/en-US/Fedora/25/html/SELinux_Users_and_Administrators_Guide/index.html
- **SELinux intro by Digital Ocean**
  - https://www.digitalocean.com/community/tutorials/an-introduction-to-selinux-on-centos-7-part-1-basic-concepts