Lab A6 Overview: SELinux Fundamentals and Access Control Enforcement

Ayoub Zouargui
Master 1 – Networks and Embedded Systems
University of Algiers 1 (2025–2026)
https://github.com/AyoubSecurity

Introduction

Security-Enhanced Linux (SELinux) is a mandatory access control (MAC) system integrated into the Linux kernel. It enforces security policies that restrict how processes interact with files, directories, and system resources, even when traditional UNIX permissions would allow access. SELinux enhances system security by limiting the impact of misconfigurations or compromised services.

Lab Objective

The objective of this lab is to observe SELinux in enforcing mode, intentionally trigger an access denial, analyze the resulting security alerts, and correctly resolve the issue by applying appropriate SELinux file contexts without disabling security mechanisms.

Lab Environment

Operating System: Red Hat Enterprise Linux

SELinux Policy: Targeted

SELinux Mode: Enforcing

Web Service: Apache HTTP Server (`httpd`)

Execution Steps

Step 1: Verify SELinux Status

```
getenforce
sestatus
```

These commands confirm that SELinux is enabled and running in enforcing mode.

Step 2: Install and Enable Apache HTTP Server

```
sudo dnf install -y httpd
sudo systemctl enable --now httpd
systemctl status httpd
```

Apache is installed, started, and configured to run automatically at system boot.

Step 3: Create a Web Directory with an Incorrect SELinux Context

```
sudo mkdir -p /srv/selinux-test
echo "SELinux-test-page" | sudo tee /srv/selinux-test/index.html
sudo ln -s /srv/selinux-test /var/www/html/selinux-test
```

This directory is intentionally created outside the default web context to trigger an SELinux denial.

Step 4: Test Web Access and Observe Denial

```
curl http://localhost/selinux-test/
```

The request returns a `403 Forbidden` response due to SELinux access restrictions.

Step 5: Inspect SELinux Denial Logs

```
sudo ausearch -m avc -ts recent
sudo journalctl -t setroubleshoot --since "10 minutes ago"
```

These logs reveal access vector cache (AVC) denials generated by SELinux.

Step 6: Identify Current File Context

```
ls -Z /srv/selinux-test/index.html
```

The file is labeled with a generic context that is not permitted for the Apache process.

Step 7: Apply Correct SELinux File Context

```
sudo semanage fcontext -a -t httpd_sys_content_t "/srv/selinux-test(/.)?"
sudo restorecon -Rv /srv/selinux-test
```

This assigns the appropriate context required for Apache to read the content.

Step 8: Verify Updated Context and Retest Access

```
ls -Z /srv/selinux-test/index.html
curl http://localhost/selinux-test/
```

The web content is successfully displayed, confirming that SELinux policies are satisfied.

Step 9: Confirm SELinux Remains Enforcing

```
getenforce
```

SELinux remains in enforcing mode, ensuring system security is preserved.

Outcome and Learning

This lab demonstrated how SELinux enforces mandatory access control independently of traditional file permissions. By resolving access issues through proper context labeling rather than disabling SELinux, the lab reinforces best practices for maintaining secure and resilient Linux systems.