Khaula Molapo

Computer Security Week 12

Hands-On Hardening Lab

For this lab, a virtual machine with Ubuntu Linux was used to perform basic system hardening. System hardening means securing the system by reducing vulnerabilities and following best security practices.

1. Updating the System

The first step was to update all software packages to make sure the latest security patches were installed. This was done using the following commands:

- sudo apt update && sudo apt upgrade -y

Regular updates ensure that known vulnerabilities are fixed and the system stays secure.

2. Disabling Non-Essential Services

Next, two unnecessary services were identified and disabled. These were the Bluetooth and Avahi services, which are not needed on a server system.

- sudo systemctl disable bluetooth.service
- sudo systemctl disable avahi-daemon.service

Disabling unused services reduces the attack surface and prevents potential exploitation.

3. Creating a Standard User Account

A new non-administrator (standard) user account was created to avoid using the root account for daily tasks.

- sudo adduser standarduser

After creating the account, administrative tasks can be performed using 'sudo' instead of logging in as root. This limits damage if the user account is compromised.

4. Configuring the Firewall

Ubuntu's built-in firewall (UFW) was used to restrict incoming network traffic. All incoming connections were blocked except for SSH, allowing only secure remote access.

- sudo ufw default deny incoming
- sudo ufw default allow outgoing
- sudo ufw allow ssh
- sudo ufw enable

This configuration allows administrators to manage the server remotely while protecting it from unwanted connections.

Summary

After these steps, the system was more secure. The attack surface was reduced, unnecessary services were disabled, and the firewall restricted network access effectively.

Comparative Analysis: SELinux vs. AppArmor

Linux provides two major security frameworks for Mandatory Access Control (MAC): SELinux and AppArmor. Both improve security by limiting what processes and users can do, even if they are compromised.

1. Design Philosophy

SELinux (Security-Enhanced Linux) uses a model called Type Enforcement. It assigns labels to files, processes, and other objects, and enforces policies based on those labels. This makes SELinux very powerful and precise, but also more complex to configure.

AppArmor, on the other hand, is path-based. It controls access based on file paths rather than labels. This approach is easier to understand and manage but slightly less flexible than SELinux in complex systems.

2. Management Complexity

SELinux is known to be harder to manage because it requires defining detailed policies and understanding the labeling system. When something goes wrong, troubleshooting SELinux permissions can take more time.

AppArmor is simpler to set up and use. Profiles are stored as text files, making it easy to see which paths a process can access. It is better for users who prefer ease of use and straightforward configuration.

3. Use Cases

SELinux is commonly used in enterprise environments like Red Hat and CentOS systems. It is ideal for environments that demand strict security controls, such as financial or government systems.

AppArmor is used by Ubuntu and SUSE Linux distributions. It is a good choice for web servers or general-purpose systems where ease of management is important.

In summary, SELinux offers stronger and more detailed security controls but requires more knowledge to manage. AppArmor is simpler and still provides good protection with less complexity. For a web server, AppArmor is recommended because it is easier to configure and maintain. For a database or high-security system, SELinux would be the better choice due to its strict control model.