Experiment No 4

Aim: Study hardening of Linux OS.

Theory: Most systems have confidential data that needs to be protected. To safeguard this data, we need to secure our Linux system. But how to properly harden a Linux system? To do this, we start by with physical security measures to prevent unauthorized people from access the system in the first place. Next is doing the installation the right way, so we have a solid foundation. Finally, we will apply a set of common security measures. After we are finished, our server or desktop system should be better protected.

It is believed, Linux is already secure by default**.**

One of the myths about Linux is that it is secure, as it is not susceptible to viruses or other forms of malware. This is partially true, as Linux uses the foundations of the original UNIX operating system. Processes are separated and a normal user is restricted in what he or she can do on the system. Still, Linux is not perfectly secure by default. One of the reasons is the Linux distributions that package the GNU/Linux kernel and the related software. They have to choose between usability, performance, and security.

With the difficult choices that Linux distributions have to make, we can be sure of compromises. These compromises typically result in a lowered level of security. What about malware for Linux? That is a definitely a myth. The Linux platform also has its fair share of backdoors, rootkits, works, and even ransomware. That is one of the reasons why it is important to do system hardening, security auditing, and checking for compliance with technical guidelines.

Areas	Core	Resources	Services	Environment
System Hardening	Boot Process Containers Frameworks **Kernel** Service Manager Virtualization	Accounting **Authentication** Cryptography Logging Network Software Storage Time	Database Mail Middleware Monitoring Printing **Shell** **Web**	Forensics Incident Response Malware Risks Security Monitoring System Integrity
Security Auditing				
Compliance				

What is system hardening?

To improve the security level of a system, we take different types of measures. This could be the removal of an existing system service or uninstall some software components.

System hardening is the process of doing the ‘right’ things. The goal is to enhance the security level of the system. There are many aspects to securing a system properly. Yet, the basics are similar for most operating systems. So the system hardening process for Linux desktop and servers is that that special.

Core principles of system hardening

If we would put a microscope on system hardening, we could split the process into a few core principles. These include the **principle of least privilege**, **segmentation**, and **reduction**.

Principe of least privilege

The principle of least privileges means that we give users and processes the bare minimum of permission to do their job. It is similar to granting a visitor access to a building. We could give full access to the building, including all sensitive areas. The other option is to only allow our guest to access a single floor where they need to be. The choice is easy, right?

Examples:

- When read-only access is enough, don't give write permissions
- Don't allow executable code in memory areas that are flagged as data segments
- Don't run applications as the root user, instead use a non-privileged user account

Segmentation

The next principle is that we split bigger areas into smaller ones. If we look at that building again, we have split it into multiple floors. Each floor can be further divided into different zones. Maybe our visitor is only allowed on floor 4, in the blue zone. If we translate this to Linux security, this principle would apply to memory usage. Each process can only access their own memory segments.

Reduction

This principle aims to remove something that is not strictly needed for the system to work. It looks like the principle of least privilege, yet focuses on preventing something in the first place. A process that does not have to run, should be stopped. Similar for unneeded user accounts or sensitive data that is no longer being used.

System hardening steps

Overview of hardening steps

1. Install security updates and patches
2. Use strong passwords
3. Bind processes to localhost
4. Implement a firewall
5. Keep things clean
6. Security configurations
7. Limit access
8. Monitor our systems
9. Create backups (and test!)
10. Perform system auditing

1. Install security updates and patches

Most weaknesses in systems are caused by flaws in software. These flaws we call vulnerabilities. Proper care for software patch management help with reducing a lot of the related risks. The activity of installing updates often has a low risk, especially when starting with the security patches first. Most Linux distributions have the option to limit what packages we want to upgrade (all, security only, per package). Make sure that our security

updates are installed as soon as they come available. It goes without saying, before we implementing something, test it first on a (virtual) test system.

Depending on our Linux distribution there might be a way to implement security patches automatically, like unattended upgrades on Debian and Ubuntu. This makes software patch management a lot easier!

2. Use strong passwords

The main gateway to a system is by logging in as a valid user with the related password of that account. Strong passwords make it more difficult for tools to guess the password and let malicious people walk in via the front door. A strong password consists of a variety of characters (alphanumeric, numbers, special like percent, space, or even Unicode characters).

3. Bind processes to localhost

Not all services have to be available via the network. For example, when running a local instance of MySQL on our web server, let it only listen on a local socket or bind to localhost (127.0.0.1). Then configure our application to connect via this local address, which is typically already the default.

4. Implement a firewall

Only allowed traffic should in an ideal situation reach our system. To achieve this, implement a firewall solution like iptables, or the newer nftables.

When creating a policy for our firewall, consider using a “deny all, allow some” policy. So we deny all traffic by default, then define what kind of traffic we want to allow. This is especially useful for incoming traffic, to prevent sharing services we didn’t intend to share.

5. Keep things clean

Everything installed on a system which doesn’t belong there can only negatively impact our machine. It will also increase our backups (and restore times). Or they might contain vulnerabilities. A clean system is often a more healthy and secure system. Therefore minimalization is a great method in the process of Linux hardening.

Actionable tasks include:

- Delete unused package
- Clean up old home directories and remove the users

6. Secure configurations

Most applications have one or more security measures available to protect against some forms of threats to the software or system. Look at the man page for any options and test these options carefully.

7. Limit access

Only allow access to the machine for authorized users. Does someone really need access or are alternative methods possible to give the user what he or she wants?

8. Monitor our systems

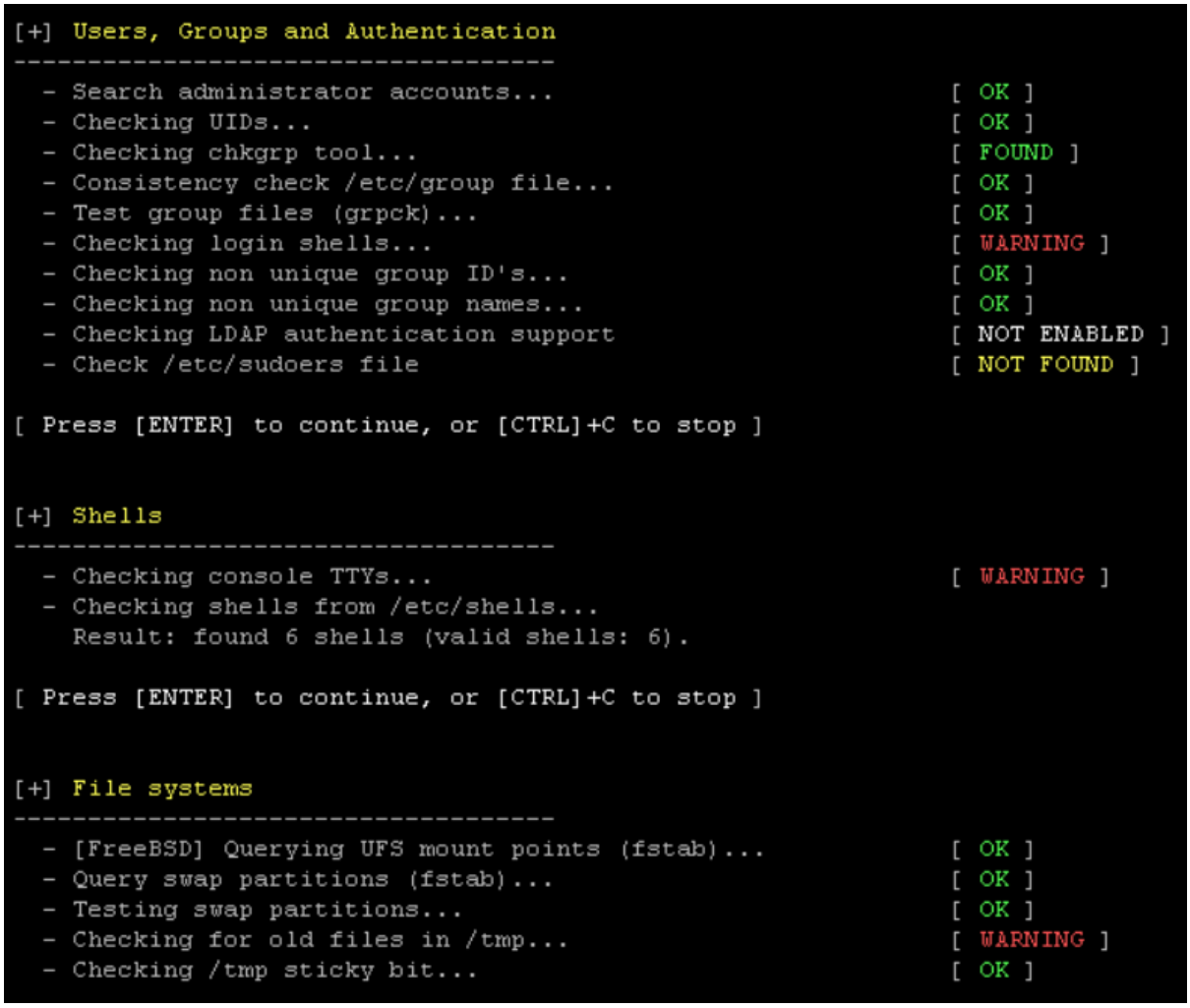
Most intrusions are undetected, due to lack of monitoring. Implement normal system monitoring and implement monitoring on security events. For example, the use of the Linux audit framework increased detection rates of suspected events.

9. Create backups (and test!)

Regularly make a backup of system data. This can prevent data loss. Even more important, test our backups. Having a backup is nice, but it is the restore that really counts!

Backups can be done with existing system tools like tar and scp. Another option to spare bandwidth is synchronizing data with tools like rsync. If we rather want to use a backup program, we can consider Amanda or Bacula.

10. Perform system auditing



Screenshot of a Linux server security audit performed with Lynis.

We can't properly protect a system if We don't measure it.

Use a security tool like Lynis to perform a regular audit of our system. Any findings are showed on the screen and also stored in a data file for further analysis. With an extensive log file, it allows to use all available data and plan next actions for further system hardening.

Lynis runs on almost all Linux systems or Unix flavors. It only requires a normal shell. Root permissions are preferred, yet not needed. The security tool is free to use and open source software (FOSS).

Conclusion: Linux hardening is the process of doing the ‘right’ things for linux OS. The goal is to enhance the security level of the system. There are many aspects to securing a system properly. principle of least privilege, segmentation, and reduction are the core principles of hardening.