

Introduction to Hardening Operating Systems

Fortificación de S.O.

Master en Seguridad Informática. 2022/2023

Universidade da Coruña

Universidade de Vigo

Antonio Yáñez Izquierdo

José Rodríguez Pereira

Contents I

1 Hardening Operating Systems

- Operating systems security
- Principles of security
- Stages of hardening O.S.

2 Hardening Linux Operating Systems

- Linux Operating System
- Linux Operating System: Distributions
- Hardening Linux Operating System

Hardening Operating Systems

Hardening Operating Systems

→ Operating systems security

Operating Systems security

- A newly installed operating system is inherently unsecure
- it usually has a certain number of vulnerabilities arising from
 - age of the operating system
 - services it provides
 - inclusion of applications not already patched
 - default policies where security is not the primary goal

What is hardening an O.S.?

- By *hardening* an O.S. we refer to the process of configuring an O.S. with the aim of making it as secure as possible
- it usually involves
 - applying patches
 - uninstalling applications
 - disabling services
 - restricting user and application privileges
 - changing default O.S. policies

Principles of O.S. security

- We want to make the system as secure as possible
- We have to minimize the risk of it (and the information it contains) being compromised
- To achieve this we have to
 - identify possible threats and vulnerabilities
 - have several lines of defence
 - always apply the principle of least privilege

Possible threats and vulnerabilities

- For an information system, threats can come from
 - Malfunctioning applications
 - Hostile (or dumb) *outside* individuals
 - Hostile (or dumb) *legitimate* users in the system
- The outcome of an exploited vulnerability of our system can be
 - our system ceases to perform as intended
 - information in our system is destroyed or leaked

Secure applications

- all software running in a secure system must be both **reliable** and **secure**
- **reliable** software is the one that **DOES CORRECTLY** the task it is intended for
- **secure** software is the one that **ONLY DOES** the task it is intended for

Hardening Operating Systems

→ Principles of security

Principles of security

- As we seen before, there are two principles that must be taken into account when hardening an O.S.
 - Principle of several lines of defence
 - Principle of least privilege

Several lines of defence

- We have to assume that any security measure that we implement will eventually fail
 - We add another security measure in place for when the previous one fails, and so on . . .
 - We also have to know which security measures have failed
- As seen in the following example

Several lines of defence. Example

- Let's think of the physical security of our system
 - *line 1* We keep our system in a locked room with cameras
 - *line 2* We disable booting from external devices and changing of the booting configuration in the system's firmware, so, should anyone bypass the locked room he/she won't be able boot from an external device to gain access to our system
 - *line 3* We protect the boot loader so anyone that has access to the machine can not tamper with the boot options to gain access to our system
 - *line 4* We lock the machine to the room so its not easily taken away or opened to get the disk extracted
 - *line 5* We crypt the sensitive information so in case the disks are removed, information cannot be accessed
 - ...

Principle of least privilege

- “Every user or application in the system must have only the privileges necessary to perform its task”
 - As always we have to assume that every security measure will ultimately fail
 - When an application (or a user account for that matter) becomes compromised the amount of damage it can cause is limited by the privileges it has

Principle of least privilege. Examples

- Applications can be jailed in *chroot* cages, or even virtualized
- The idea behind it being: should the application (or the user account for that matter) become compromised the amount of damage it can cause is limited
- Users never should work with administrator privileges
- Never use the administrator account unless necessary. In fact users who need only certain administrator privileges should never become the administrator but be allowed to do **ONLY** the tasks they are supposed perform via the *sudo* command or using groups
- Non administrator accounts can have their privileges further restricted with restricted shells

Hardening Operating Systems

→ Stages of hardening O.S.

Stages of hardening an O.S.

- To harden an O.S. we must consider three different stages
 - 1 Hardening during the installation
 - 2 Post-installation hardening
 - 3 Maintenance

Hardening during installation

- This usually affects the choice of what to install and what security policies can be chosen at installation time
- The ideal thing would be to install the system *isolated* and then do some post-configuration and patch-applying before connecting it to the network
- As many of today's O.S. installations are done via network we trust (we shouldn't really!) that the installation scripts apply the adequate patches fast enough

Post instalation hardening

- This is where most of the O.S. hardening is done
- Here we change the configuration of the system to make the system more secure
- It usually involves
 - disabling system services
 - removing applications o restricting access to some of them
 - changing user accounts
 - changing default system policies

Maintenance

- Once the system is up and running (hopefully) securely we have to keep it that way
- To achieve that we must
 - Apply application patches and system updates regularly
 - Using both the logs (which must be secured) and certain tools (Lynix, openvax . . .) we can monitor the system behaviour searching for vulnerabilities and unauthorized accesses

Hardening Linux Operating Systems

Hardening Linux Operating Systems

→ Linux Operating System

Linux Operating System

- Linux is a UNIXlike Operating System implementing mostly the POSIX standard
- It is free software, conforming to the GNU public license
- Consists of a kernel and a userland set of applications
- The kernel conforms to the GNU public licence (some firmware drivers do not, so some distributions refuse to include them by default)
- Most of the userland applications are distributed under the GNU public license
- Some applications are *non-free* although they are supplied with the source code
- There are still some applications distributed only in binary form

Hardening Linux Operating Systems

→ Linux Operating System: Distributions

Linux Operating System: Distributions

- Linux is available for many different hardware platforms, *Sparc, UltraSparc, ARM, intel x86, intel ia64, amd64, alpha* ...
- Although the ARM platform is gaining importance, still most of linux systems are (as is the case with *windows* systems) *amd64* platform (32 bit *intel x86* platform is becoming obsolete).
- Being free software means that, as the source code is available, it can be audited by anyone and checked for vulnerabilities
- This also means that we do not depend of some firm's policy to decide whether a certain vulnerability is worth to be patched
- These are good characteristics towards getting a secure system

Linux Operating System: Distributions

- Being free software also means that anyone can get whatever version of the kernel and whatever set of userland applications and build their own system
- This is what we call a distribution
- Different distributions are targeted to different type of users, there are desktop-targeted, server-targeted, intrusion-targeted . . . distributions.

Linux Operating System: Distributions

- Different distributions usually have different package management system (in fact, this criteria is sometimes used to classify distributions), different desktop environment, different set of applications installed.
- There are even distributions designed to be executed without installing (live distributions)
- The most widespread linux distributions are Debian (and some of its derivatives: mint, ubuntu, devuan . . .), Fedora, OpenSuse, CentOs, gentoo, kali . . .

Linux Operating System: Distributions

- Most of the concepts and solutions we will be presenting are of application to most linux distributions.
- However, we have to take in consideration that some software packages (Pluggable Authentication Modules-PAM, SELinux, Apparmor . . .)
 - Are installed by default in some distributions
 - Are available as an option in some other distributions
 - There might be unavailable for other distributions
- Also, we have to be aware that, as package versions vary from one distribution to the other, configuration files might also be different.

Hardening Linux Operating Systems

→ Hardening Linux Operating System

Hardening Linux operating systems

- In the present course, we'll use the Debian distro (or its derivative devuan) as the target platform in our lab assignments
- we'll deal with the several parts of the operating system, and for each of these parts we'll try to
 - Give a brief introduction to its fundamentals and its working principles
 - Try to identify its possible points of vulnerability
 - Give solutions or hints on how minimize the risk of this vulnerabilities being exploited

Hardening Linux operating systems. Topics

- 1) Hardening the boot procedure
 - Boot procedure details. Hardening firmware. Grub boot loader vulnerabilities. Hardening the grub boot loader. Other boot loaders.
- 2) Hardening user accounts
 - Introduction to users and groups. Pam modules. Hardening authentication. Limiting privileges. Restricted shells. Becoming *root*. *sudo* and *sudoers*

Hardening Linux operating systems. Topics

3) Hardening File Systems

- File system concepts: partitions, logical and physical volumes, filesystems. Formatting and mounting filesystems. Permissions. Quotas. Crypting Filesystems

4) Hardening applications

- Unused applications/packages. setcpulimt. chroot. cgroups. LXC. SELinux. Apparmor

Hardening Linux operating systems. Topics

5) Hardening the network

- Eliminating and disabling services. Limiting access to services.
Packet filtering: iptables, nftables

6) Maintenance

- System logs. Log configuration. Securing logs. Patches.
Hardening tools