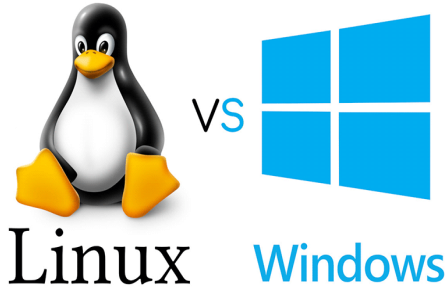
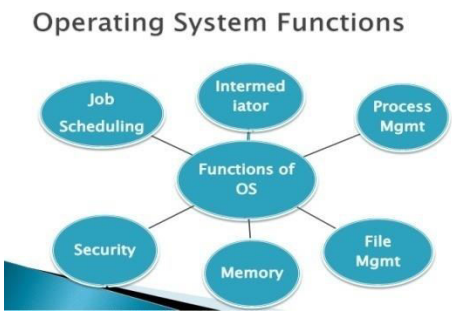
CIT210 OP. SYS. MANAG.

CSUN
COLLEGE OF ENGINEERING AND COMPUTER SCIENCE

Week12Day2 : OS Security - I: Linux Security, Boot and Kernel Security Issues, File System Encryption, iptables, TCP wrappers, SELinux, App Armor





The Basic Linux Kernel Philosophy

- **Includes**
  - **monolithic core**
    - The **monolithic part** of the kernel contains drivers and options essential to the kernel boot process.
  - **modular components**
    - The **modular part** of the kernel adds everything else needed for smooth operation, including many drivers that enable communication with installed hardware.
- ***If that kernel module fails, it does not affect other parts of the Linux operating system***

TABLE 2-1 Basic configuration categories for the Linux kernel.

CATEGORY	DESCRIPTION
General setup	Includes a variety of basic kernel settings
Loadable module support	Sets conditions for how modules are loaded for a modular kernel
Block layer	Defines how block devices can be loaded
Processor type and features	Configures support for a variety of processor types and memory levels
Power management options	Defines available power-management features
Bus options	Specifies support for hardware cards
Executable file formats/ emulations	Associated with executable and linkable format binaries
Networking	Includes network drivers, protocols, and more
Device drivers	Support for a wide variety of hardware
Firmware drivers	Supports nonvolatile systems such as the Basic Input/Output System (BIOS) and the Unified Extensible Firmware Interface (UEFI)
Filesystems	Includes various filesystem formats
Instrumentation support	Adds options for performance modules
Kernel hacking	Designed for kernel debugging
Security options	Includes options such as Security Enhanced Linux (SELinux) and Application Armor (AppArmor)
Cryptographic API	Defines supported encryption algorithms for application programming interfaces (APIs)
Virtualization	Adds code for hardware-based virtual machines
Library routines	Includes modules for cyclic redundancy checks

Security and Kernel Update Issues

- Kernel security can be divided into two interrelated areas.
  - First, there are security issues that arise related to the stock Linux kernel.
  - Second, there are security issues that arise related to the Linux kernel as built by the developers of a distribution.

Security and Kernel Update Issues

- When updating a kernel, it's best to make sure that any existing working kernel is retained on your systems. If there is a problem with the updated kernel, those systems will still be bootable and available with the older working kernel.
- Custom kernels work better for many systems.
- You can install them directly on the distribution.
- Custom kernels can be optimized to load more quickly.
- custom kernels can exclude options such as modules that may be loaded by black-hat hackers to present security risks.

Distribution-Specific Kernel Security Issues

- To update packages on Ubuntu systems command is
  - apt-get update
  - it excludes later versions of the Linux kernel from an update.
- To update packages on Red Hat systems command is
  - yum update
  - It does not exclude later versions of the Linux kernel unless the --exclude=kernel-* switch is included in the command:

 # yum update --exclude=kernel-*

Installing an Updated Kernel

- Binary kernel packages built by the developers of Linux distributions should automatically update applicable bootloader files such as

/etc/lilo.conf or /boot/grub/menu.lst.

- These files should include stanzas for both the current and the new kernel.
- If you've set up custom features for an existing kernel in the bootloader, you may need to modify the stanza for the new kernel to include those same messages.

Before Customizing a Kernel

- The commands clean out any remaining object files and remove any existing kernel configuration in the local directory.
- If you've just installed the kernel source code, these commands are not necessary.
  - # make clean
  - # make mrproper
- The next step is to select a baseline.
- Accept the baseline inherent in the source code.
- Select one of the configuration options available in the configs/ or arch/x86/configs/ subdirectory.
- If the system architecture is not a standard 32- or 64-bit Intel/AMD CPU, substitute for x86/ accordingly.

Kernel Configuration Options

- General Setup
  - General setup options run the from kernel compression modes to the use of swap space to initial RAM disk support.
  - In most cases, you should not change any options in this section

Kernel Configuration Options

- Enable Loadable Module Support
  - The default options automatically load appropriate module devices for existing and newly installed hardware.
  - As Linux distributions rely on a modular kernel, you shouldn't generally change that basic functionality.
  - One feature in the stock kernel is related to module verification, which uses GNU Privacy Guard (GPG) keys to verify modules.
  - It's a common option for kernels built for Red Hat systems.

Kernel Configuration Options

- Enable the Block Layer
  - Block-layer kernel settings are associated with larger files in the terabyte range.
  - Without such settings, fairly common Linux filesystems such as the third and fourth extended filesystems (ext3 and ext4) won't work.
- Processor Type and Features
  - With the development of multi-core CPU processors with different features, this menu includes a number of relatively new options.

Kernel Configuration Options

- **Filesystems**
  - This section includes integrated and modular support for a wide variety of local and network-based filesystems.
  - One way to disable access to certain filesystems such as the Network File System (NFS) and Samba is to disable it in the kernel as configured in this section.
  - Be aware that this section includes support for a number of filesystems at an experimental level.
  - Such options should not be enabled on production systems.

Securing Linux File Systems

- Linux filesystem hierarchy standard (FHS)
- Filesystem mounting options
- Remote filesystems
- Filesystem encryption
- Filesystem quotas

Types of File Systems

- Disk file systems – FAT (File Allocation Table), NTFS, HFS (Hierarchical File System), ext2, ext3, ISO9660 and UDF
- FAT(FAT12, FAT16, FAT32), and especially NTFS are primarily used on Windows operating systems. FAT is also the standard file system for floppy drives and is still used today
- HFS is used by Mac OS, and ext2, ext3 are used on various linux operating systems
- ISO9660 and UDF are used on optical media

Linux File System

- **File system**
  - Enables directories or folders organization
  - Establishes a file-naming convention
  - Includes utilities to compress or encrypt files
  - Provides for both file and data integrity
  - Enables error recovery
  - Stores information about files and folders
- File systems store information about files in information nodes (inodes)

Linux File System (continued)

- Information stored in an inode
  - An inode number
  - Owner of the file
  - Group the file belongs to
  - Size of the file
  - Date the file was created
  - Date the file was last modified or read
- File systems use a fixed number of inodes
  - mounts a file system as a subfile system of the root file system

Linux filesystem (continued)

- In Linux, all data is stored as files.
- A filesystem specifies how those files are stored, marked, and retrieved.
- In Linux, filesystems are associated with a particular device filename.
- Files on a device are accessible only when they're mounted on a Linux directory.
- The way filesystems are mounted on different Linux directories is documented in the /etc/fstab configuration file.

How does the file system handle security?

- The file system is crucial to data integrity.
- Main method of protection is through access control
- Accessing file system operations (ex. modifying or deleting a file) are controlled through access control lists or capabilities
- Capabilities are more secure so they tend to be used by operating systems on file systems like NTFS or ext3.
- Secondary method of protection is through the use of backup and recovery systems

Linux filesystem

- The word “filesystem” means
  - A construct of space on a hard drive such as a partition, RAID array, or logical volume
  - The device files associated with a partition, RAID array, or logical volume, such as /dev/sda1, /dev/md0, or /dev/VolGroup00/LogVol00
  - The format method associated with a partition, RAID array, or logical volume, such as ext3 or vfat
  - A mounted directory, such as the /boot/ directory filesystem

TABLE 5-1 Important filesystem hierarchy standard directories.

DIRECTORY	DESCRIPTION
/	Top-level root directory, always mounted separately
/bin/	Basic command-line utilities, should always be a part of /
/boot/	Linux kernel, initial RAM disk, bootloader files; frequently mounted separately
/dev/	Device files for hardware and software, should always be part of /
/etc/	Configuration files, should always be part of /
/home/	Home directories for every regular user; frequently mounted separately
/lib/	Program libraries; should always be part of / (/lib64/ may exist on 64-bit systems)
/media/	Standard mount point for removable media such as CD/DVD drives and universal serial bus (USB) keys; may also be used for other volumes such as a directory formatted to a Microsoft file system
/mnt/	Common legacy mount point for removable media and temporary filesystems
/opt/	Common location for some third-party applications; may be empty and can be mounted separately
/root/	The home directory for the root administrative user; should always be part of /
/sbin/	Primarily for administrative commands; should always be part of /
/srv/	Directory commonly used for network services such as those that share using FTP and HTTP; may be helpful to mount separately
/tmp/	Common location for temporary files; if the /tmp/ filesystem is full, users would not be able to log into the GUI
/usr/	Small programs generally accessible to all users; could be mounted separately and read-only
/var/	Log files, print spools; some distributions use it for network service files; may be helpful to mount separately

Linux File System (continued)

- *mount* command is used to mount file systems

```
[root@server root]# mount
/dev/hda2 on / type ext3 (rw)
none on /proc type proc (rw)
none on /dev/pts type devpts (rw,gid=5,mode=620)
usbdevfs on /proc/bus/usb type usbdevfs (rw)
/dev/hda1 on /boot type ext3 (rw)
none on /dev/shm type tmpfs (rw)
[root@server root]#
```

Linux File System (continued)

- *df* command displays the currently mounted file systems

```
[root@server /]# df
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/hda2             37073224   6876240  28313768  20% /
/dev/hda1               101089     23754     72116  25% /boot
none                    319860         0    319860   0% /dev/shm
[root@server /]#
```

Good Volume Organization Can Help Secure a System

- if you have more space available on a local system, configure additional directories on different filesystems.
- If you're configuring a bastion host with a small amount of space, it's often best to configure a single partition for the entire system.
- if the system has a little bit of extra space, it's often best to isolate some filesystems

The /boot Filesystem

- By default, Linux distributions set up the /boot/ directory on a separate filesystem, always on a separate partition.
- A /boot/ filesystem is typically pretty small at 500 MB, is infrequently changed, and is therefore easier to back up.
- The 100 or 500 MB that Red Hat configures for /boot/ is typically plenty of space for four or five versions of the Linux kernel

The /home Filesystem

- Set up the /home directory as a separate filesystem.
- Easier back up user files configured in that way.
- Easier to upgrade a distribution to a later release with much less risk to user files.
- It may be helpful to set up the /home directory on a RAID array.
- If your system includes a growing number of users, it may be helpful to set up the /home directory on a logical volume.

The /opt Filesystem

- Standard location for third-party applications.
- Not important in the actual Linux boot process.
- Examples of Linux applications that use this filesystem include the following:
  - • Adobe Acrobat
  - • OpenOffice.org Suite (when installed directly from
  - • RealPlayer for Linux
- Because the /opt directory is not often checked, it may be helpful to set it up as a separate filesystem.
- In fact, it's an excellent candidate for a logical volume.
- If the number of third-party applications on the local system grows, a configuration as a logical volume makes it easier to expand the space allocated to the /opt/ directory.

The /srv Filesystem

- The /srv filesystem is designed to contain files associated with some FTP and HTTP services.
- Set up the /srv directory in a separate filesystem.
- If someone chooses to upload a library of multi-gigabyte movie files to your FTP server, a separate filesystem would act as a barrier, keeping that upload from affecting any other filesystem.

The /tmp Filesystem

- The /tmp directory is a common location for temporary files such as shared downloads.
- It's also a location for files associated with logins to the GUI.
- If the /tmp filesystem is full, users who try to log into the GUI are "stuck" with the command line.
- So if your systems use the /tmp directory for shared downloads,
- it may be helpful to set it up on a separate filesystem for that directory, so users who download too much do not overload the local system.

The /var Filesystem

- Files in the /var directory can easily grow quite large.
- Some distributions use subdirectories of /var/ to contain files associated with some servers.
- Uploads to such servers, if limits can be breached, could quickly overload the space available on just about any system.
- In addition, the /var directory contains log files.
- The logs for enterprise-level Web sites can easily grow to several gigabytes every day.
- If such a directory of logs is not maintained, this array of large files could quickly overload just about any system.
- The configuration of the /var directory on a separate filesystem can limit such risks.

Read-Only Filesystems

- Three filesystem candidates for mounting in read-only mode are /boot, /opt, and /usr.
- if you were to update a kernel or install a new package, you'd have to remount the filesystem in readwrite mode so the kernel or package files could be written to the appropriate directories.
- The following is a possible read-only configuration directive for the /boot directory in the /etc/fstab configuration file:

/dev/sda1 /boot ext2 ro,exec,auto,nouser,async 1 2

Journals, Formats, and File Sizes

- The selection of a filesystem can affect how well it resists an attack.
- A journaling filesystem keeps a journal or log of the changes that are being made to the filesystem during disk writing that can be used to rapidly reconstruct corruptions that may occur due to events such a system crash or power outage.
- The level of journaling performed by the file system can be configured to provide a number of levels of logging depending on your needs and performance requirements.

Filesystem Encryption

- Encryption adds another layer of security for data that is considered confidential.
- Documents such as customer personal information, social security numbers, credit card information, and business plans can be encrypted.
- There are many regulations and laws for protecting consumer's personal data.

Using Encryption

- **Encryption Tools**
  - **Linux supports a wide variety of encryption tools.**
  - **Split into two categories:**
  - **kernel-space and**
    - **Kernel-space tools depend primarily on modules** that can be loaded with the Linux kernel.
    - kernel-space encryption is faster,
  - **user-space.**
    - User-space tools can more easily utilize the libraries loaded in the /lib/ directory.
    - user-space encryption is easier to implement.

TABLE 5-2 Linux file-encryption commands.

COMMAND	DESCRIPTION
bcrypt	Uses the blowfish encryption algorithm, based on a passphrase
ccrypt	Uses the U.S. Advanced Encryption Standard (AES), uses passphrases
gpg	Users may select from different encryption algorithms; may use passphrases or public and private key pairs
ncrypt	Users may select from different encryption algorithms; passphrases are hashed
pad	Uses "one time pad" encryption

Encryption with a Public/Private Key Pair

- **Elgamal—**
  - A probabilistic encryption scheme developed by Taher Elgamal. When running the gpg command, this scheme is paired with DSA for digital signatures.
- **DSA—**
  - Digital Signature Algorithm, used by the U.S. government for digital signatures.
- **RSA—**
  - A public-key algorithm, named for its developers, Rivest, Shamir, and Adelman.

Encryption with a Public/Private Key Pair

- After selecting a key, be prepared to select the following:
  - An encryption-key size of between 1,024 and 4,096 bits
  - A key lifetime in days, weeks, months, or years; alternatively, choose the default setting of 0, which means the key does not expire
  - A name, e-mail address, and comment
  - A passphrase

Encrypted Directories

- Ubuntu method is with the enterprise cryptographic filesystem (eCryptfs), other options are available.
- Uses loopback encrypted filesystems based on the pluggable authentication modules (PAM) mount module.
- The eCryptfs system includes both kernel-space and user-space components.
- The loaded modules related to encryption include the following:
- • Secure hash algorithm (SHA) with 256 bits (sha256_generic)
- • Advanced encryption standard (AES) for x86 CPUs (aes_i586)
- • Disk encryption subsystem (dm_crypt)

Pros and Cons of Filesystem Encryption

Pros

- Simple to implement
- Transparent to the user
- Difficult to hack

Cons

- Entire data in a filesystem is encrypted, including the data that does not need to be encrypted.
- Resizing the filesystem later is difficult.

Linux Kernel Security Options

- **CONFIG_FIREWALL for network firewalls**
- **CONFIG_IP_MASQUERADE for IP address masquerading to help protect systems on a LAN**
  - is a form of Network Address Translation (NAT) which allows internally connected computers that do not have one or more registered Internet IP addresses to communicate to the Internet via the Linux server's Internet IP address.
  - Since IPMASQ is a generic technology, you can connect the Linux server's internal and external to other computers through LAN technologies like Ethernet, TokenRing, and FDDI, as well as dialup connections line PPP or SLIP links.
- **IP_FORWARDING to set up a system as a gateway**

Boot Process Security

- Anyone with physical access to your systems may be able to access boot menus, such as those available through a BIOS or a UEFI menu.
- Such menus should be password protected.
- Network access to the UEFI menu should be disabled.
- Boot process security extends to the boot menu for the operating system.

Common Boot Loaders

- Grand Unified Bootloader (GRUB)
- Linux Loader (LILO)
- Loadlin
- Universal Bootloader (U-Boot)

More Boot Process Issues

- The (TCP/IP) suite includes default port numbers for hundreds of services, such as 22 for SSH and 23 for Telnet.
- Use nonstandard port numbers for key services.
- Obscurity can slow the efforts of black-hat hackers who want to break into a system.

More Boot Process Issues

- Some services require network access during the boot process.
- if you keep servers synchronized with the Network Time Protocol (NTP), those servers access external NTP servers during the boot process.
- The standard NTP port is 123.
- *If you want to block that port after the boot process is complete, you need to make sure the associated firewall rule isn't run until after the NTP client is synchronized.*

Security Issues with the GUI

- Applications that require a GUI are common security risks.
- GUI is a networkable client-server system.
- Users can log into a GUI remotely.
- Users can also run GUI clients over a network.
- Administrators can configure a system to display GUI clients on a remote terminal.
- Malware on one Linux GUI application can be spread across a network to other GUI systems.

The User Authentication Databases

- There are four major user authentication databases available for Linux.
- The files of the shadow password suite are
  - /etc/passwd,
  - /etc/group,
  - /etc/shadow,
  - /etc/gshadow.
- These files include encrypted passwords, user and group accounts, home directories, and login shells.

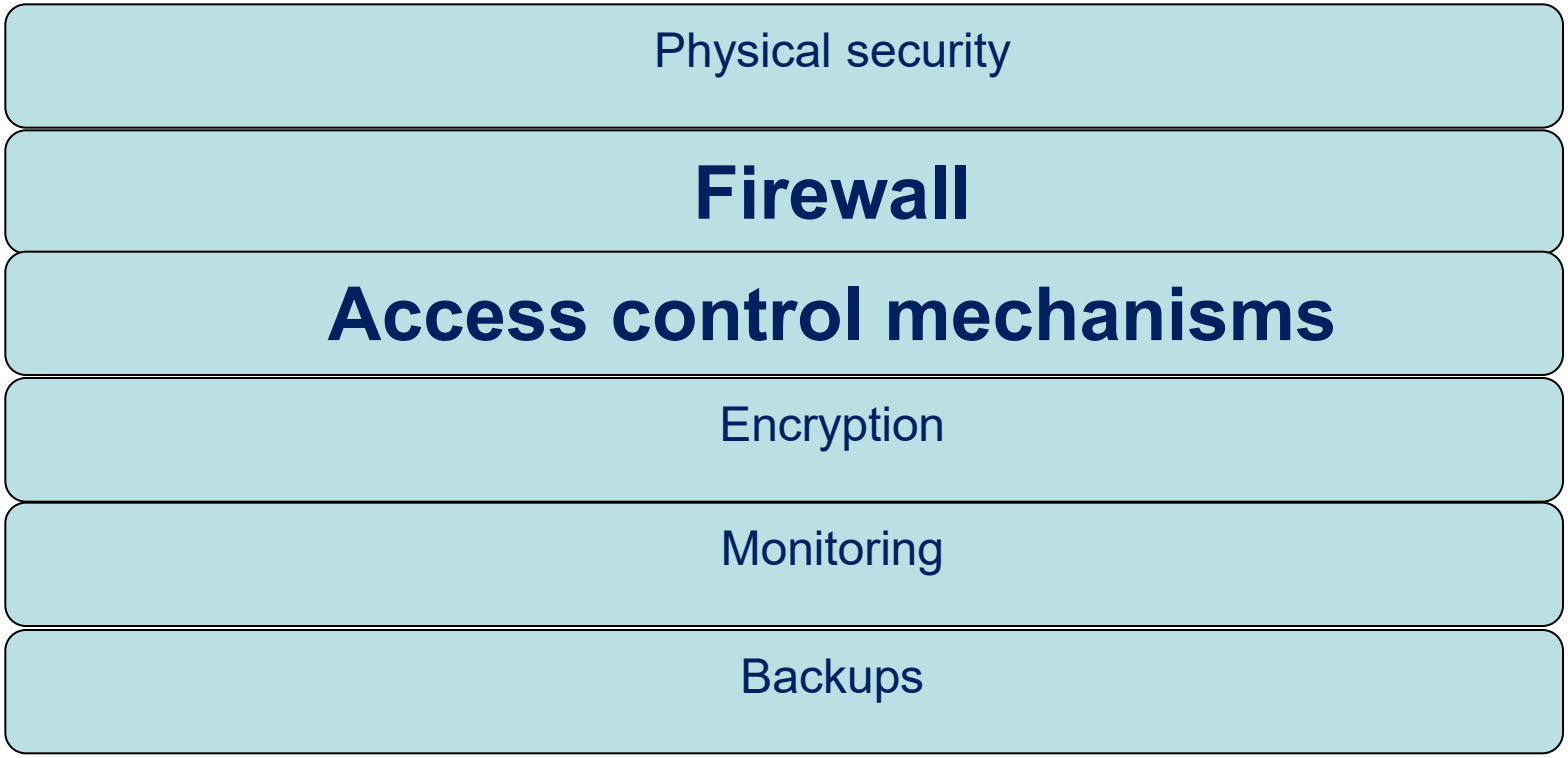
The User Authentication Databases

- Other Two databases are native to Linux:
  - The Network Information Service (NIS) and
  - Lightweight Directory Access Protocol (LDAP).
- To help manage this variety of authentication databases, Linux uses the /etc/nsswitch.conf file, also known as the name service switch file.

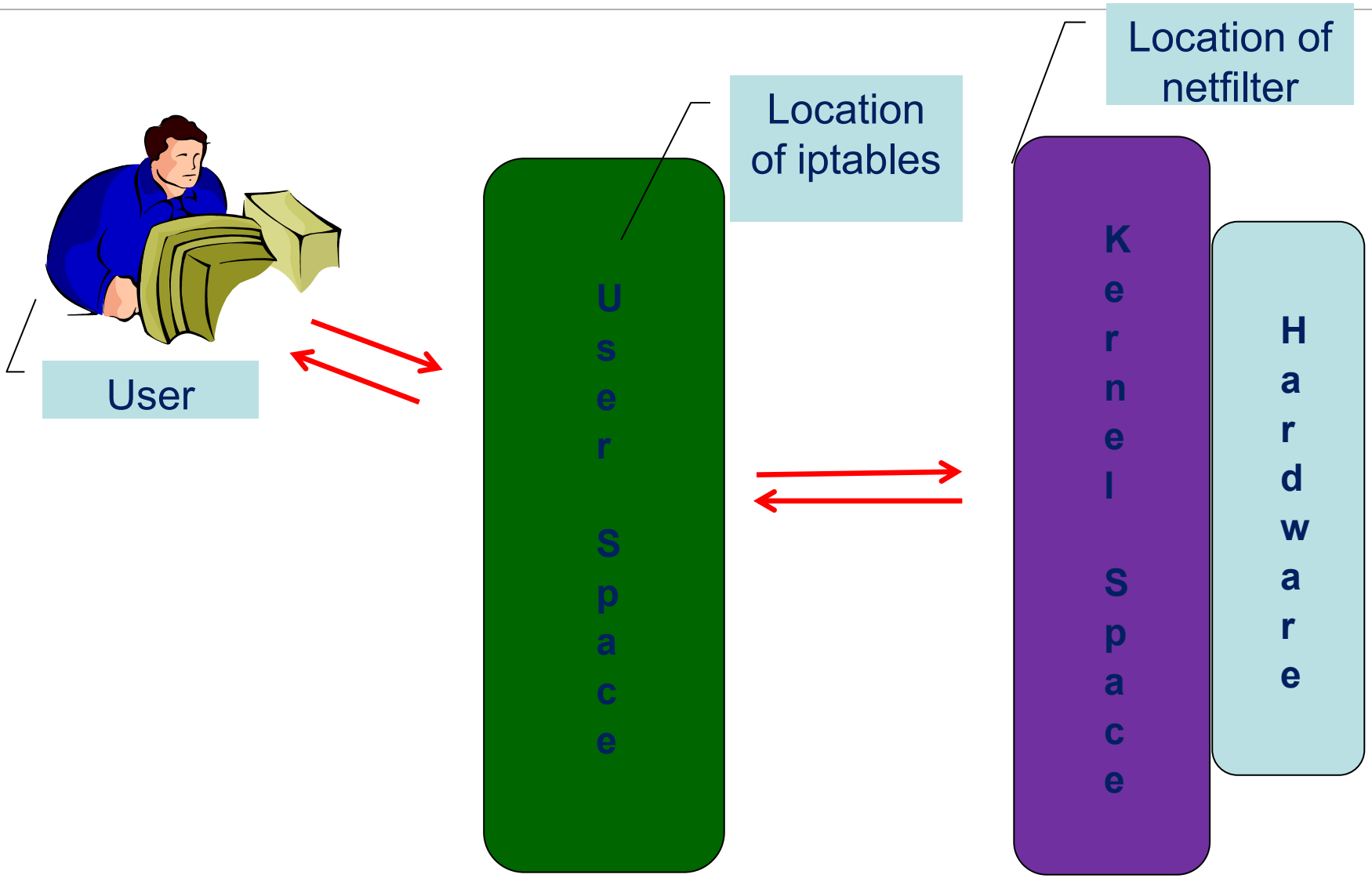
The User Authentication Databases

- Linux supplements authentication databases with a system known as pluggable authentication modules (PAM).
- The rules associated with PAM files in the /etc/pam.d/ directory can further specify local access limits for different administrative tools and commands.

Layered Security



The Linux Firewall



TCP Wrappers

- Controlling access to network services is one of the most important security tasks facing a server administrator.
- iptables-based firewall filters out unwelcome network packets within the kernel's network stack.
- For network services that utilize it, *TCP Wrappers* add an additional layer of protection by defining which hosts are or are not allowed to connect to "*wrapped*" network services.
- It provides for greater and more specific control over local network services and which hosts are allowed to access them

TCP Wrappers

• TCP Wrappers remains useful because it can be configured quickly and easily,

•Adds an additional layer of protection even when used in conjunction with more robust packet filters (like iptables).

- *Transparency to both the client and the wrapped network service —*
  - Both the connecting client and the wrapped network service are unaware that TCP Wrappers are in use.
  - Legitimate users are logged and connected to the requested service while connections from banned clients fail.
- *Centralized management of multiple protocols —*
  - TCP Wrappers operate separately from the network services they protect, allowing many server applications to share a common set of access control configuration files, making for simpler management.

TCP Wrappers

- TCP Wrappers allows or denies access to a given daemon running on a server.
- To allow a host, add its name or IP address to /etc/hosts.allow.
- To deny a host, add its name or IP address to /etc/hosts.deny.

You may want to keep in mind that a rule allowing access to a given service in /etc/hosts.allow takes precedence over a rule in /etc/hosts.deny prohibiting it. Additionally, if two rules apply to the same service, only the first one will be taken into account.

Linux Firewall with IPTABLES

- The Linux kernel features a powerful networking subsystem called *netfilter*.
- The netfilter subsystem provides stateful or stateless packet filtering as well as NAT
- Netfilter also has the ability to *mangle* IP header information for advanced routing and connection state management.
- Netfilter is controlled through the iptables utility.
- The iptables program is a tool to implement the packet handling engine Netfilter.
- At one time, the most popular firewall running on Linux was ipchains, but it had a number of shortcomings.
- To rectify this, the Netfilter organization decided to create a new product called iptables.

Iptables Overview

- The power and flexibility of netfilter is implemented through the IPTables interface.
- This command-line tool is similar in syntax to its predecessor, IPChains;
- PTables uses the netfilter subsystem to enhance network connection, inspection, and processing;
- IPChains used intricate rule sets for filtering source and destination paths, as well as connection ports for both.
- IPTables features advanced logging, pre- and post-routing actions, network address translation, and port forwarding all in one command-line interface.

iptables Improvements

- Better integration with the Linux kernel with the capability of loading iptables-specific kernel modules designed for improved speed and reliability.
- Stateful packet inspection.
  - Firewall keeps track of each connection passing through it and in certain cases will view the contents of data flows in an attempt to anticipate the next action of certain protocols.
  - This is an important feature in the support of active FTP and DNS, as well as many other network services.

iptables Improvements

- Filtering packets based on a MAC address and the values of the flags in the TCP header.
  - This is helpful in preventing attacks using malformed packets and in restricting access from locally attached servers to other networks in spite of their IP addresses.
- System logging that provides the option of adjusting the level of detail of the reporting.
- Better network address translation.
- Support for transparent integration with Web proxy programs as Squid.
- A rate limiting feature that helps iptables block some types of denial of service (DoS) attacks.

Packets and iptables

- All packets inspected by iptables pass through a sequence of built-in tables for processing.
- Each of these queues is dedicated to a particular type of packet activity and is controlled by an associated packet transformation and filtering chain.

Packets and iptables

There are three tables:

- The first is the table which is responsible for the quality of service bits in the TCP header.
- The second table is the filter queue for packet filtering and has three parts:
  - Forward chain: Filters packets to servers protected by the firewall.
  - Input chain: Filters packets destined for the firewall.
  - Output chain: Filters packets originating from the firewall.
- The third table is the NAT queue which is responsible for network address translation and has two parts:
  - Pre-routing chain: NAT packets when the destination address of the packet needs to be changed.
  - Post-routing chain: NAT packets when the source address of the packet needs to be changed

Definition

Input - Traffic coming in

Forward - Traffic moving from one interface to another

Output - Traffic going out

Prerouting - Before a routing decision has been made

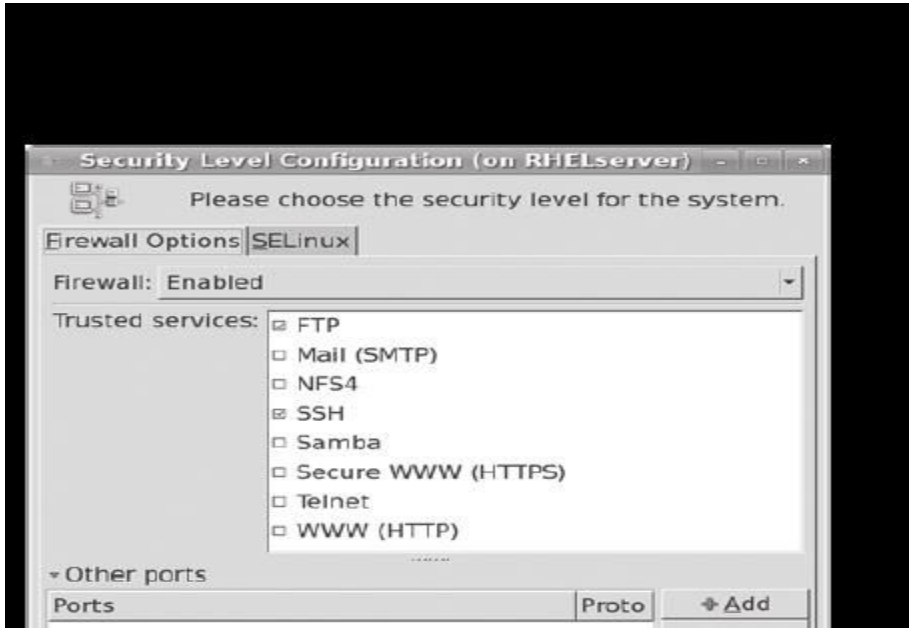
Postrouting - After a routing decision has been made

Firewall and TCP Wrappers

Iptables	▪ Add, remove, and edit rules to a packet filter ruleset ▪ List and flush the rules to a packet filter ruleset ▪ List counters of matched packets to rules
Netfilter	▪ Provides iptables packet filter in the kernel ▪ Performs stateless and stateful packet filtering ▪ Provides network address translation
TCP Wrappers	▪ Allow or deny access to an application based on an Internet Protocol (IP) Address or hostname ▪ Allow or deny access to an application based on time

Firewalls and Mandatory Access Controls

- Firewall Support Options
- Create a firewall at the command-line interface with the iptables
- It can block access by port, by IP address, by protocol type
- Other firewalls are available for services that depend on the Transmission Control Protocol (TCP) Wrappers libraries, such as libwrap.so.0.



Common Linux Access Controls

The firewall allows user access based on the file service port and user's Internet Protocol address.

MAC allows the file sharing service to interact with the shared filesystem.

DAC provides the required permissions to access files.

Access Control Mechanisms

DAC

- Defines the access control for objects in the filesystem

ACLs

- Grants “special” permissions to users or groups for an object in the filesystem that are not specified in the DAC permissions

MAC

- Adds additional categories to objects in the filesystem

Mandatory Access Control Support

- The mandatory access controls associated with SELinux and AppArmor provide another layer of security.
- These controls can check access to services and commands to make sure they're run only by intended users, in properly labeled directories.
- AppArmor includes profiles for specific commands. You can add profiles for additional commands as you learn more about mandatory access control.

Mandatory Access Control Support

- Both AppArmor and SELinux include a trial mode, where violations are logged without preventing access.
- On AppArmor, this is known as complain mode;
- on SELinux, this is known as permissive mode. The resulting log files can help you better customize either of these mandatory access control systems.

Layered Security for FTP Access

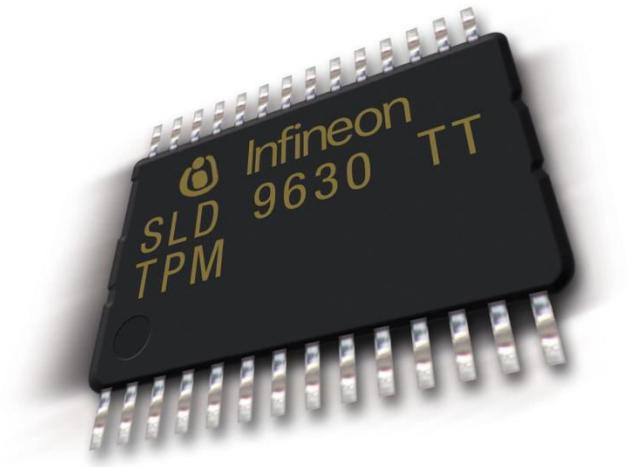
Firewall	TCP Wrapper	SELinux
Protects against unauthorized traffic	Performs specific actions based on a network service running under the xinetd super server	Protects the network service from unauthorized access based on the subject, such as users, applications, or files
Allows access to FTP from local traffic only	Sends an e-mail to the administrator when access is granted during nonbusiness hours	Denies access to home directories to logged-in users

What Is A Trusted Platform Module (TPM)?

Smartcard-like module

on the motherboard that:

- Performs cryptographic functions
  - RSA, SHA-1, RNG
  - Meets encryption export requirements
- Can create, store and manage keys
  - Provides a unique Endorsement Key (EK)
  - Provides a unique Storage Root Key (SRK)
- Performs digital signature operations
- Holds Platform Measurements (hashes)
- Anchors chain of trust for keys and credentials
- Protects itself against attacks



Why Use A TPM?

- Trusted Platforms use Roots-of-Trust
  - A TPM is an implementation of a Root-of-Trust
- A hardware Root-of-Trust has distinct advantages
  - Software can be hacked by Software
    - Difficult to root trust in software that has to validate itself
  - Hardware can be made to be robust against attacks
    - Certified to be tamper resistant
  - Hardware and software combined can protect root secrets better than software alone
- A TPM can ensure that keys and secrets are only available for use when the environment is appropriate
  - Security can be tied to specific hardware and software configurations

Hardware Crypto Capabilities

- RSA Accelerator
  - contains a hardware engine to perform up to 2048 bit RSA encryption/decryption.
  - uses its built-in RSA engine during digital signing and key wrapping operations.
- Engine for SHA-1 hash algorithm
  - uses its built-in hash engine to compute hash values of small pieces of data.
  - Large pieces of data (such as an email message) may be hashed outside of the TPM, for performance reasons.

Security Resources

- cybersecurity,
- tryhackme.com
- hackthebox.com ,
- picoctf.org and
- portswigger.net/web-security are free