Securitatea sistemelor de operare Windows și Linux

Bucharest, Romania

November 2021

Alin Puncioiu

Concepts

- *Protection:*
  - Mechanisms and policy to keep programs and users from accessing or changing stuff they should not do
  - *Internal* to OS
- *Security:*
  - Issues *external* to OS
  - Authentication of user, validation of messages, malicious or accidental introduction of flaws, etc.

Understanding OS Security: Threats and Security Controls

- The term operating system (OS) security refers to practices and measures that can ensure the **confidentiality, integrity, and availability (CIA) of operating systems**.
- The goal of OS security is to **protect the OS from various threats**, including malicious software such as worms, trojans and other viruses, misconfigurations, and remote intrusions.
- OS security typically involves the **implementation of control techniques** that can protect your assets from unauthorized modification and deletion or theft.
- The most common techniques used to protect operating systems include the use of antivirus software and other endpoint protection measures, regular OS patch updates, a firewall for monitoring network traffic, and enforcement of secure access through least privileges and user controls.

Understanding OS Security: Threats and Security Controls

Smss.exe – Session Manager, responsible for creating new sessions.

-- Suspicious behavior – sessions 0 and 1 are normal. Additional sessions may be created by RDP sessions and Fast User switching on shared computers.

Crss.exe – Client/Server Run Subsystem Process, responsible for managing processes and threads, mapping drive letters, creating temp files and handling the shutdown process.

-- Suspicious behavior – threat actors can masquerade malware to appear as this process by naming it very similarly: cssrs.exe, cssrss.exe, csrsss.exe.

Lsass.exe – Local Security Authority Subsystem, responsible for user authentication and generating access tokens.

-- Suspicious behavior –different variations of spelling for this process (lass.exe, lsasss.exe) and multiple instances of the process.

Other core processes – svchost.exe, taskhost.exe, explorer.exe, lsm.exe, winlogon.exe, wininit.exe

Understanding OS Security: Threats and Security Controls

DLL Injection is the most common technique used to inject malware into another process, or processes.
DLL Injection is accomplished using following steps:

1. **Locate process.** – malware needs to find a target process to inject the malicious DLL into.
    - Windows API: CreateToolhel32Snapshot(), Process32First(), Process32Next().
2. **Open Process** – once the malware finds the process, it opens the process.
    - Windows API: GetModuleHandle(), GetProcAddress(), OpenProcess().
3. **Allocate Memory** – The malware then needs to find a location in order to write the path to the malicious DLL.
    - Windows API: VirtualAllocEx().
4. **Copy** – The malware will write the path to the malicious DLL into the allocated memory location.
    - Windows API: WriteProcessMemory().
5. **Execute** – The malware will execute the malicious DLL in another process by starting a new thread.
    - Windows API: CreateRemoteThread(), LoadLibrary().

Attackers can use undocumented functions to execute malware: NTCreateThreadEx(), RtlCreateUserThread() –used by Mimikatz and Metasploit.
For this technique to work, the path of the malicious DLL needs to reside on disk.

CPU operation modes

- In Kernel mode, the executing code has complete and unrestricted access to the underlying hardware. It can execute any CPU instruction and reference any memory address. Kernel mode is generally reserved for the lowest-level, most trusted functions of the operating system. Crashes in kernel mode are catastrophic; they will halt the entire PC.

- In User mode, the executing code has no ability to *directly* access hardware or reference memory. Code running in user mode must delegate to system APIs to access hardware or memory. Due to the protection afforded by this sort of isolation, crashes in user mode are always recoverable. Most of the code running on your computer will execute in user mode.

https://blog.codinghorror.com/understanding-user-and-kernel-mode/
https://en.wikipedia.org/wiki/Protection_ring

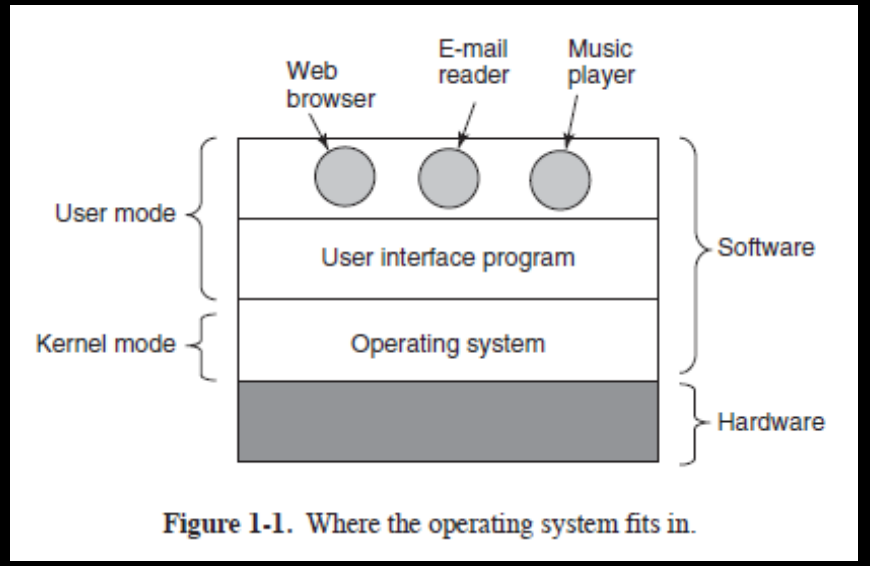


Figure 1-1. Where the operating system fits in.

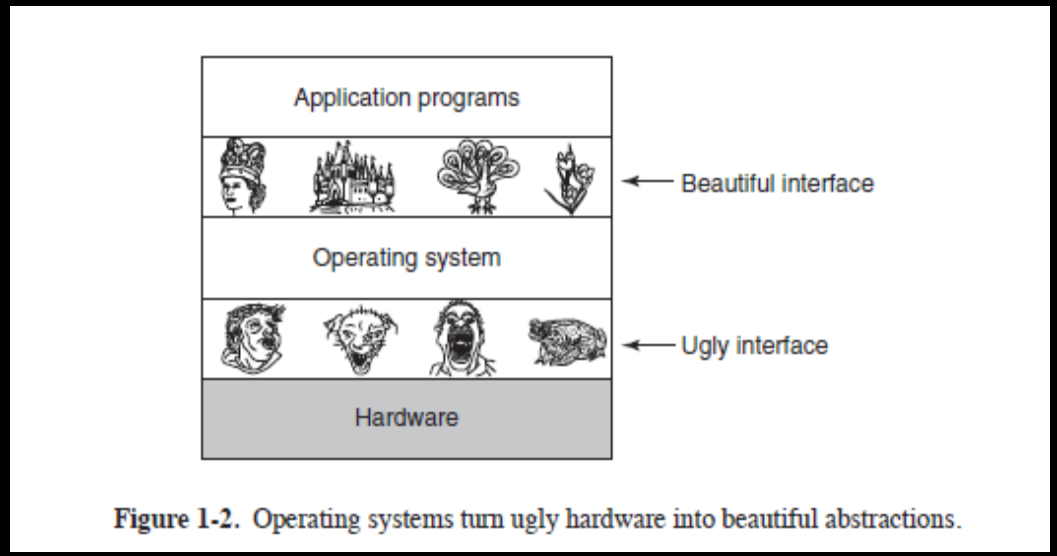
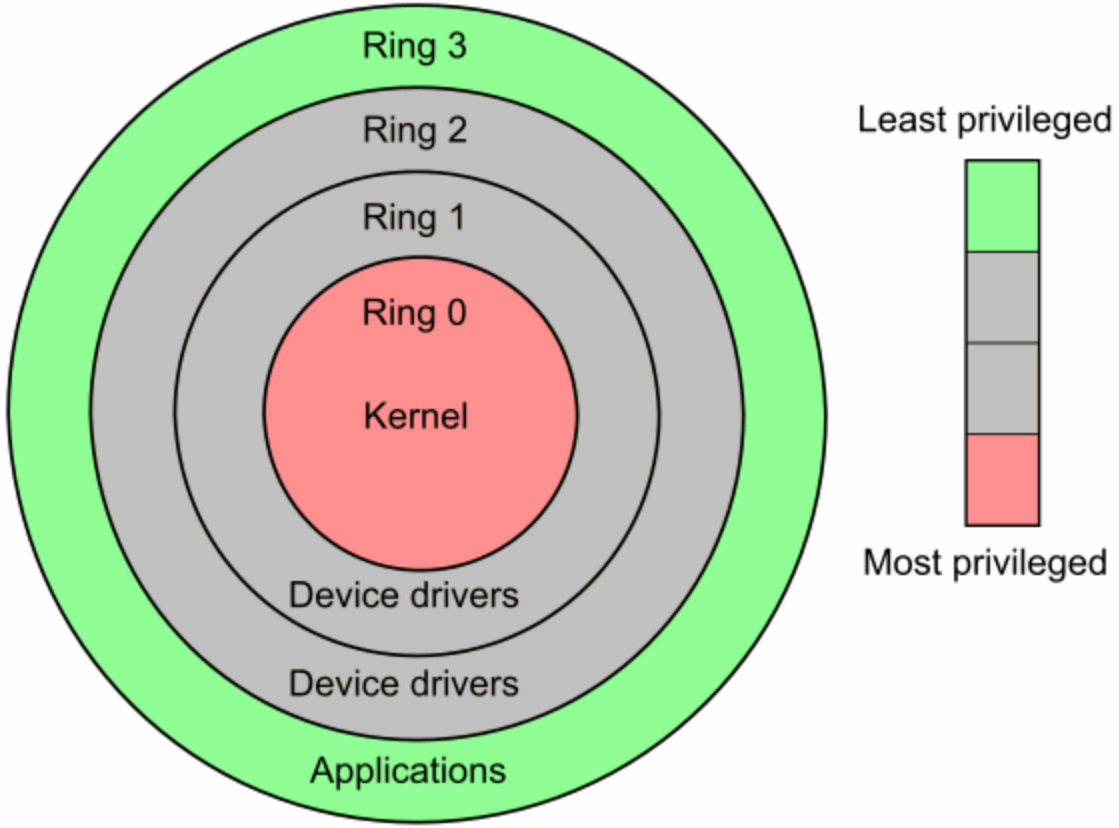


Figure 1-2. Operating systems turn ugly hardware into beautiful abstractions.

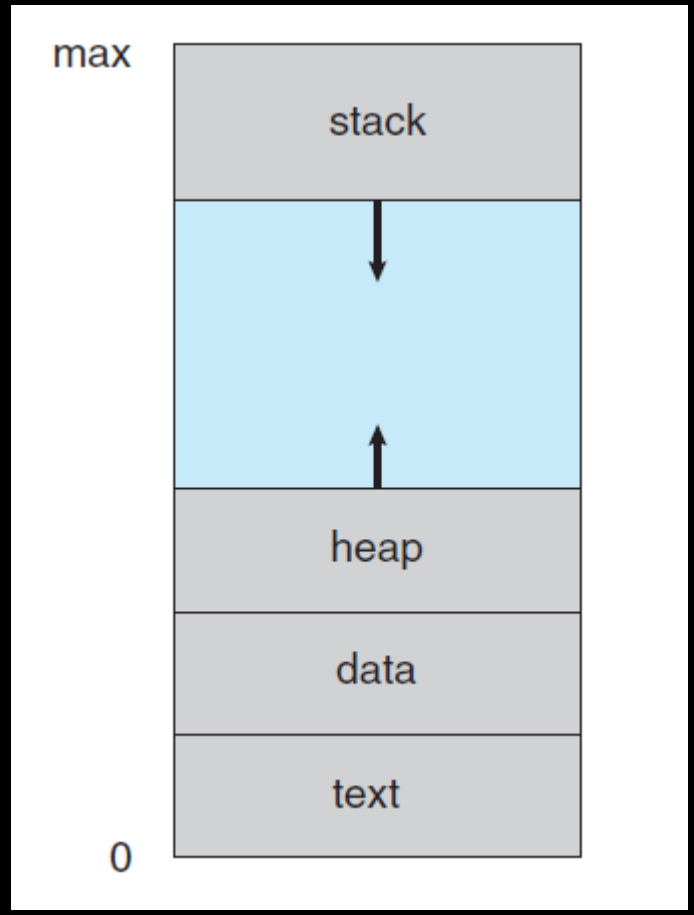
x86 CPU hardware actually provides four protection rings: 0, 1, 2, and 3. Only rings 0 (Kernel) and 3 (User) are typically used.



Process Management

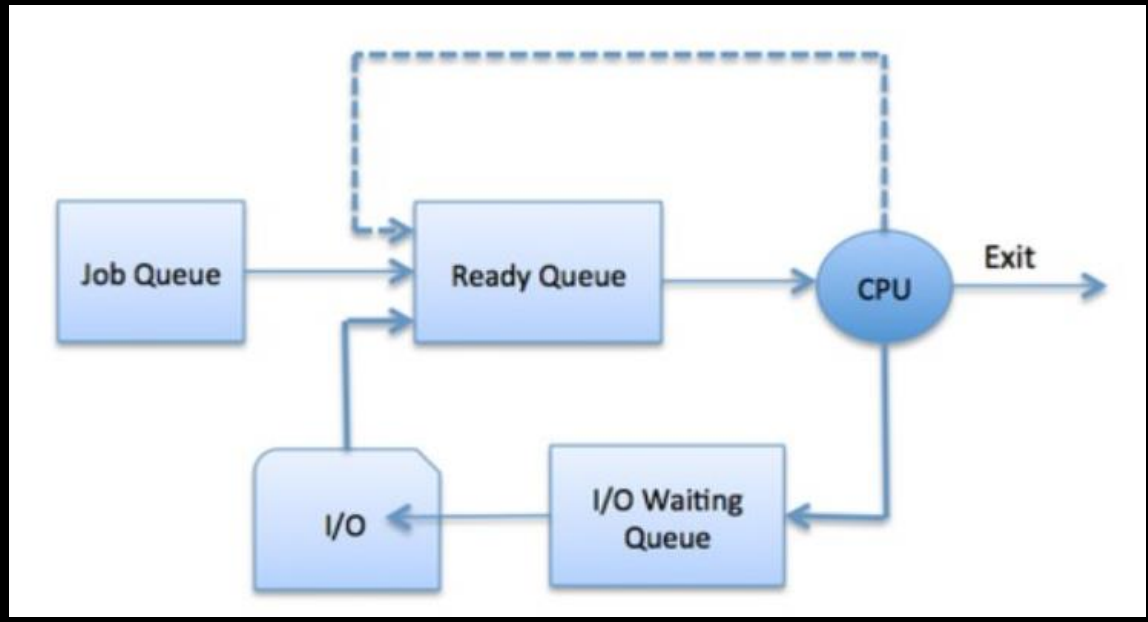
Processes

- **A program** is a passive entity, such as a file **containing a list of instructions stored on disk** (often called an executable file)
- A **program becomes a process** when an executable file is loaded into memory and activated by the operating system, so its instructions are running.
- A process can be in a **running state** (CPU is executing its instructions and data), **ready state** (waiting to send instructions to the CPU), or **blocked state** (waiting for input data, such as keystrokes, from a user).
- Each process has its own stack, which contains temporary data (such as function parameters, return addresses, and local variables) and a data section, which contains global variables.



Process scheduling

- Scheduling and synchronizing various processes and their activities is a responsibility of the operating system.
- After a process uses resources, it needs to release them back to the resource pool, otherwise the system may run out of critical resources.
- In some systems, if a requested resource is unavailable for a certain period, the operating system kills the process that is “holding on” to that resource.



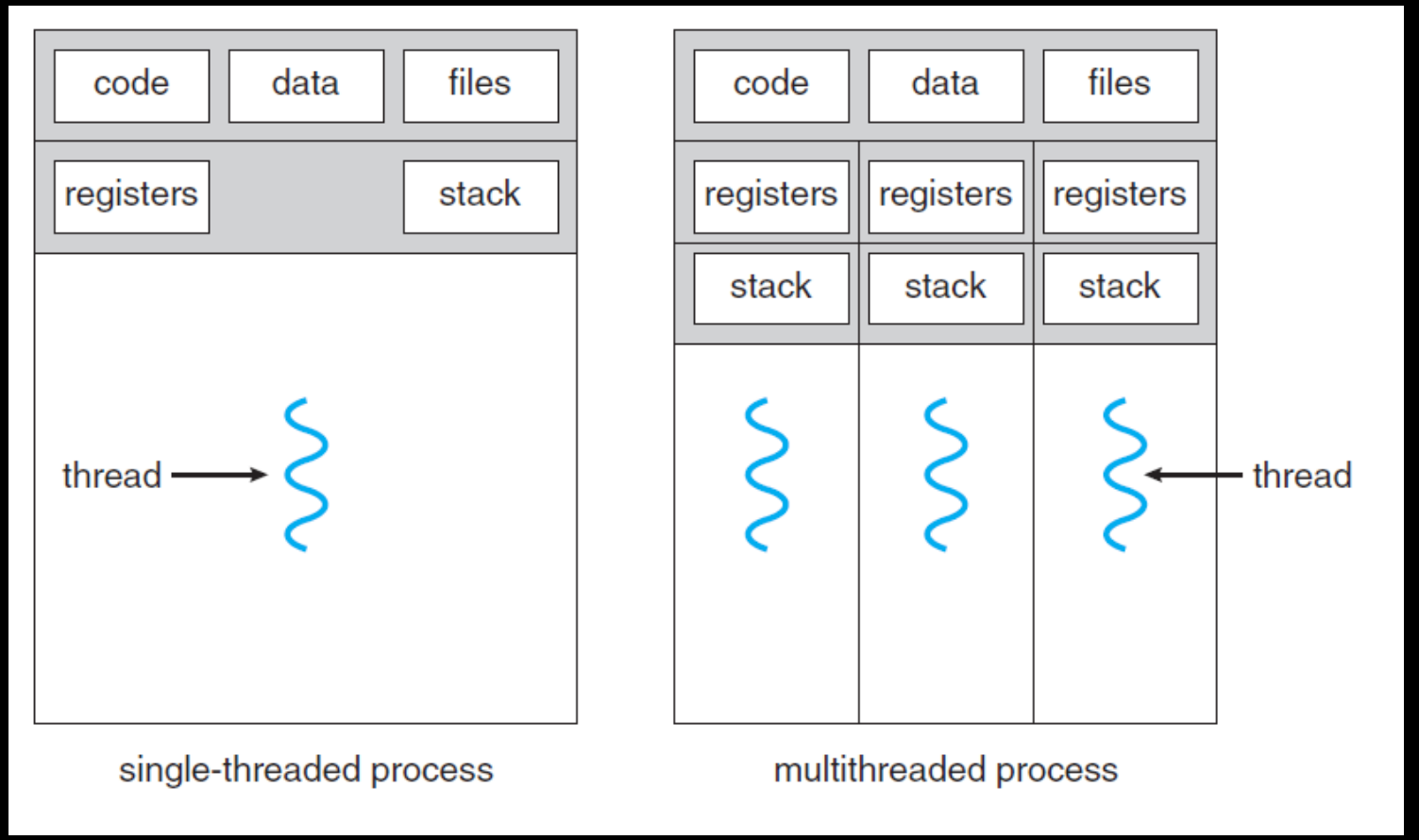
Process isolation

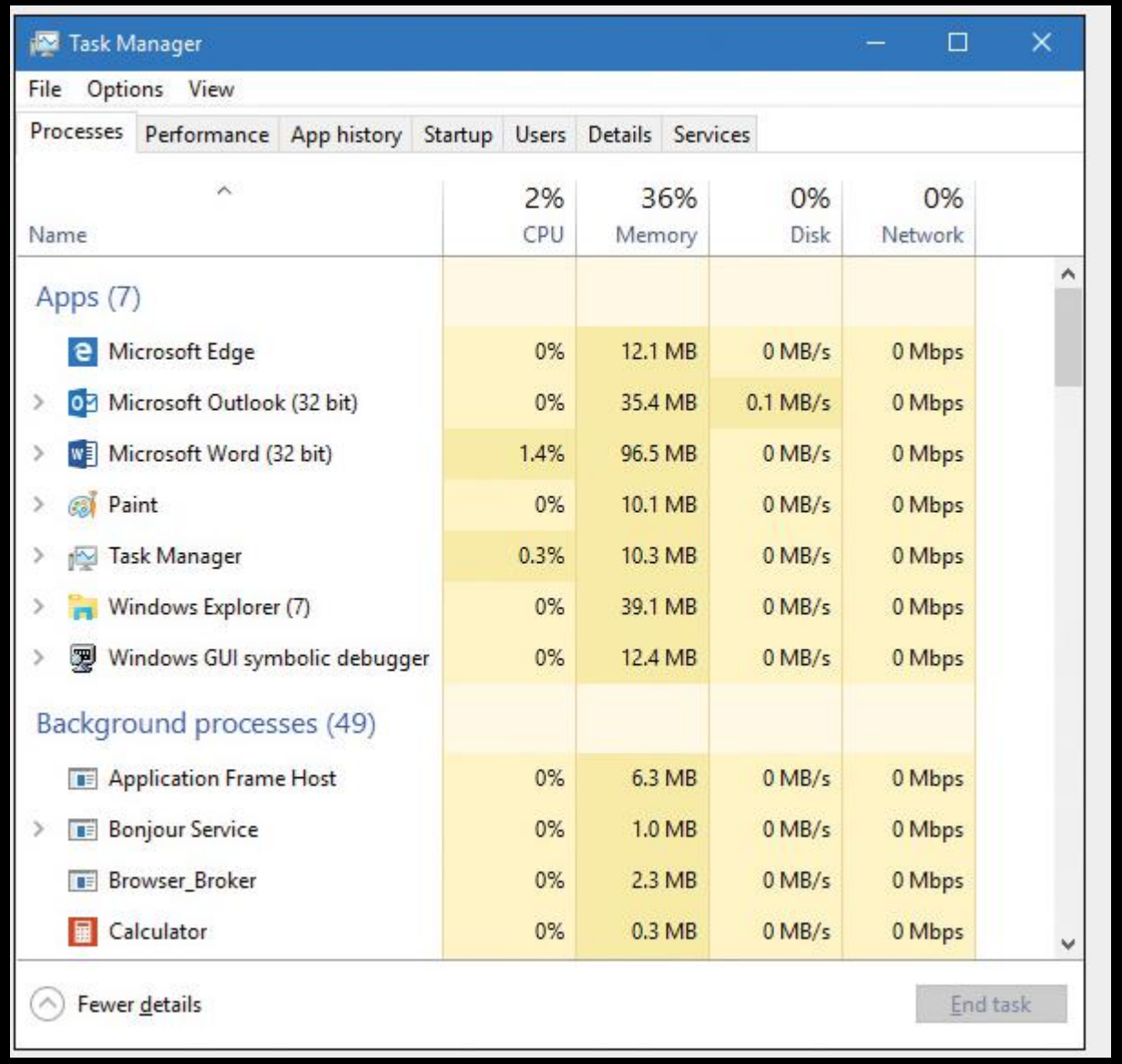
- Necessary to ensure that processes do not communicate in an insecure manner, or negatively affect each other’s productivity.
- Enforcement through encapsulation of objects, time multiplexing of shared resources, naming distinctions or virtual memory mapping.

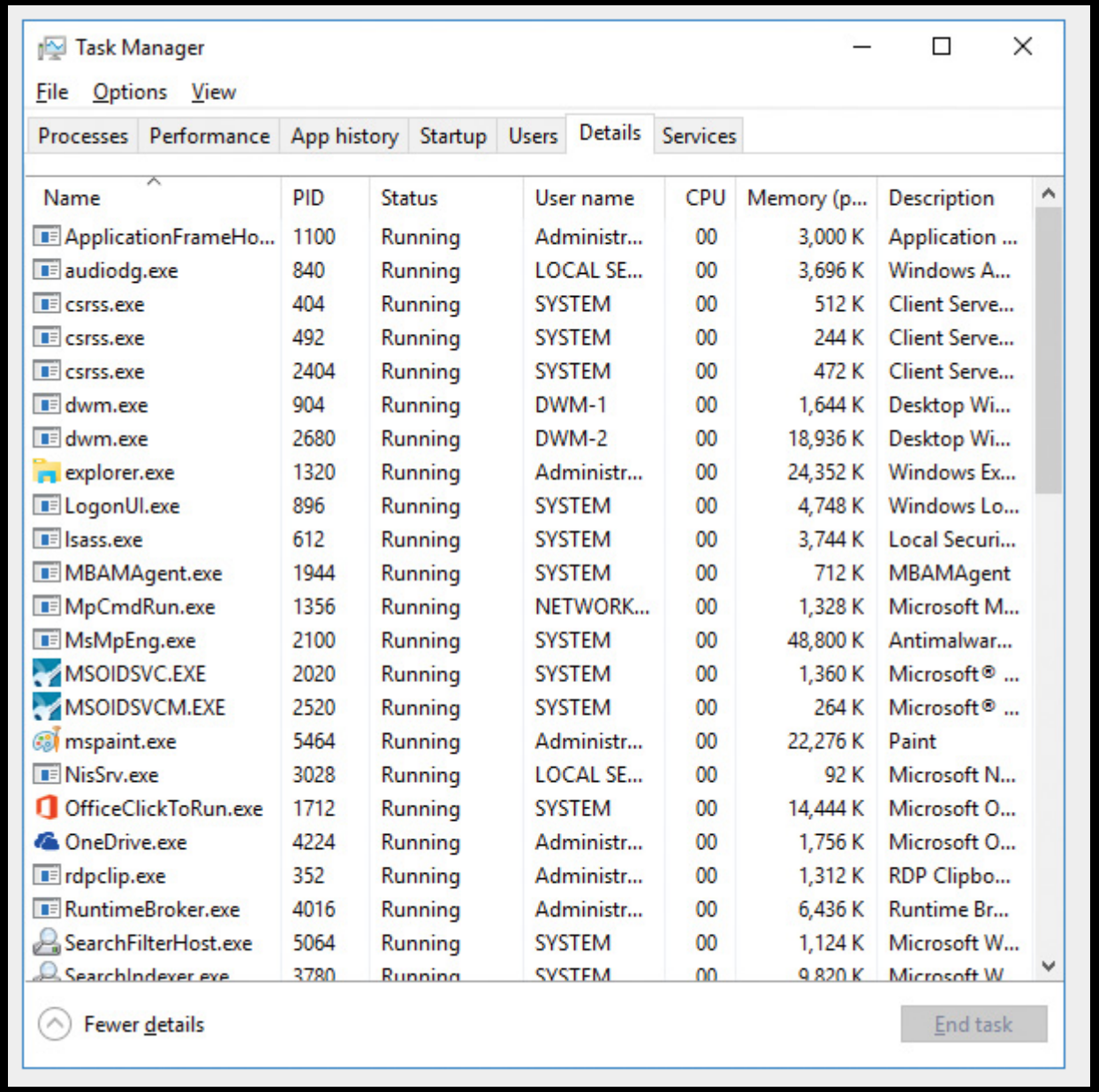
Threads

Most applications have several different functions, each requires a thread (instruction set) to be dynamically generated.

A thread is a basic unit of CPU utilization; it comprises a thread ID, a program counter, a register set, and a stack. It shares with other threads belonging to the same process its code section, data section, and other operating-system resources, such as open files and signals. A traditional (or heavyweight) process has a single thread of control. If a process has multiple threads of control, it can perform more than one task at a time.




Task Manager
File Options View
Processes Performance App history Startup Users Details Services
Name
2% CPU
36% Memory
0% Disk
0% Network
Apps (7)
Microsoft Edge 0% 12.1 MB 0 MB/s 0 Mbps
Microsoft Outlook (32 bit) 0% 35.4 MB 0.1 MB/s 0 Mbps
Microsoft Word (32 bit) 1.4% 96.5 MB 0 MB/s 0 Mbps
Paint 0% 10.1 MB 0 MB/s 0 Mbps
Task Manager 0.3% 10.3 MB 0 MB/s 0 Mbps
Windows Explorer (7) 0% 39.1 MB 0 MB/s 0 Mbps
Windows GUI symbolic debugger 0% 12.4 MB 0 MB/s 0 Mbps
Background processes (49)
Application Frame Host 0% 6.3 MB 0 MB/s 0 Mbps
Bonjour Service 0% 1.0 MB 0 MB/s 0 Mbps
Browser_Broker 0% 2.3 MB 0 MB/s 0 Mbps
Calculator 0% 0.3 MB 0 MB/s 0 Mbps
Fewer details
End task

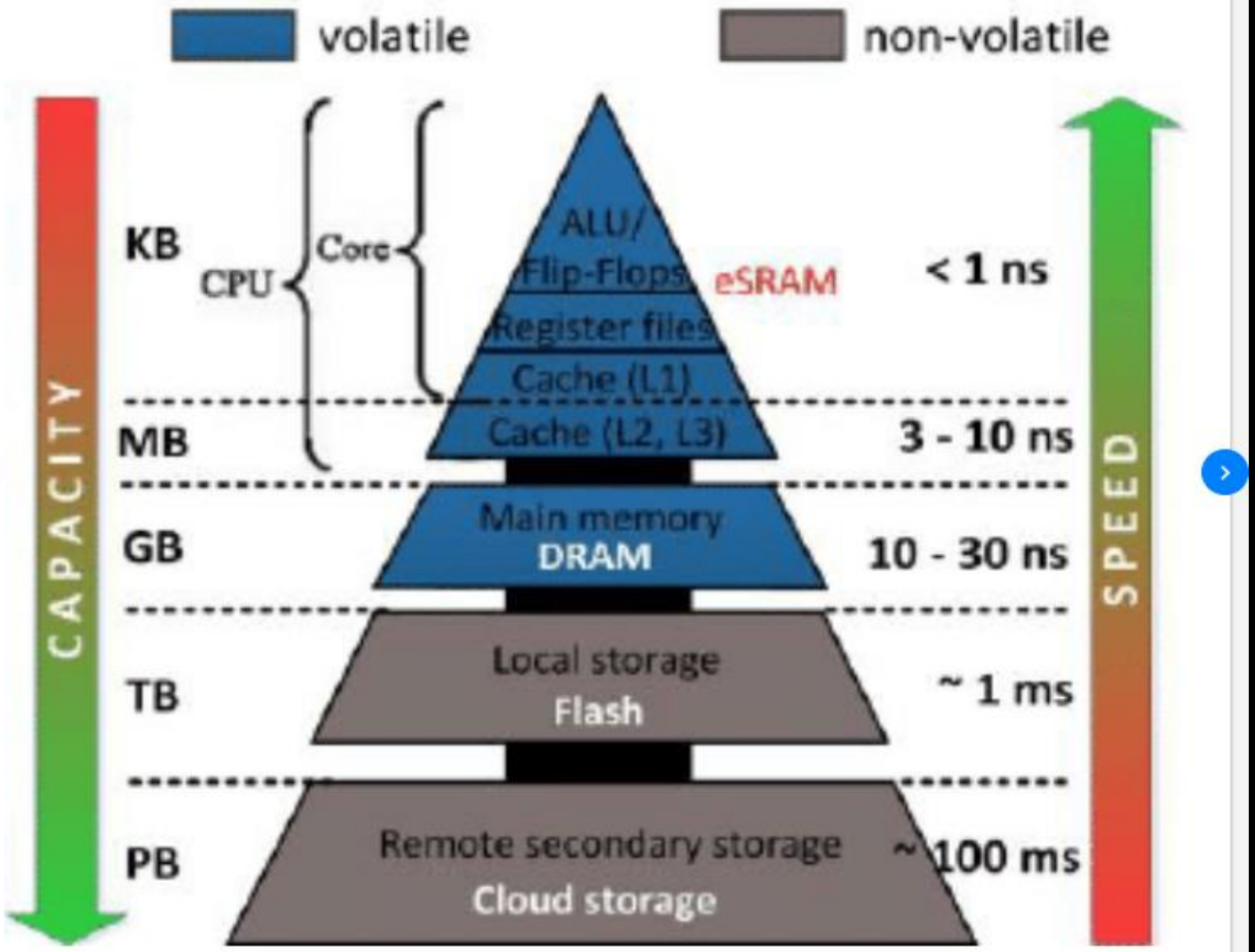

Task Manager
File Options View
Processes Performance App history Startup Users Details Services
Name PID Status User name CPU Memory (p... Description
ApplicationFrameHo... 1100 Running Administr... 00 3,000 K Application ...
audiodg.exe 840 Running LOCAL SE... 00 3,696 K Windows A...
csrss.exe 404 Running SYSTEM 00 512 K Client Serve...
csrss.exe 492 Running SYSTEM 00 244 K Client Serve...
csrss.exe 2404 Running SYSTEM 00 472 K Client Serve...
dwm.exe 904 Running DWM-1 00 1,644 K Desktop Wi...
dwm.exe 2680 Running DWM-2 00 18,936 K Desktop Wi...
explorer.exe 1320 Running Administr... 00 24,352 K Windows Ex...
LogonUI.exe 896 Running SYSTEM 00 4,748 K Windows Lo...
lsass.exe 612 Running SYSTEM 00 3,744 K Local Securi...
MBAMAgent.exe 1944 Running SYSTEM 00 712 K MBAMAgent
MpCmdRun.exe 1356 Running NETWORK... 00 1,328 K Microsoft M...
MsMpEng.exe 2100 Running SYSTEM 00 48,800 K Antimalwar...
MSOIDSVC.EXE 2020 Running SYSTEM 00 1,360 K Microsoft® ...
MSOIDSVCM.EXE 2520 Running SYSTEM 00 264 K Microsoft® ...
mspaint.exe 5464 Running Administr... 00 22,276 K Paint
NisSrv.exe 3028 Running LOCAL SE... 00 92 K Microsoft N...
OfficeClickToRun.exe 1712 Running SYSTEM 00 14,444 K Microsoft O...
OneDrive.exe 4224 Running Administr... 00 1,756 K Microsoft O...
rdpclip.exe 352 Running Administr... 00 1,312 K RDP Clipbo...
RuntimeBroker.exe 4016 Running Administr... 00 6,436 K Runtime Br...
SearchFilterHost.exe 5064 Running SYSTEM 00 1,124 K Microsoft W...
Fewer details
End task

```
Tasks: 238 total,   1 running, 184 sleeping,   0 stopped,   0 zombie
%Cpu(s):  7.0 us,  1.3 sy,  0.0 ni, 91.8 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :  5939268 total,  1367448 free,  1171108 used,  3400712 buff/cache
KiB Swap:  6801404 total,  6288476 free,   512928 used.  4051952 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
30510 paras     20   0 1238776 201476  78084 S  15.4  3.4   0:26.53 chrome
30591 paras     20   0   41944   3692   3004 R   7.7  0.1   0:00.14 top
 1071 root      20   0  469284 110808  90276 S   2.6  1.9  34:35.39 Xorg
 1324 rabbitmq  20   0 2190040  14520   3164 S   2.6  0.2   7:36.91 beam.smp
 2036 paras     20   0  351068  11348   3800 S   2.6  0.2   0:56.86 ibus-daemon
 2256 paras     20   0 1606948  94192  45184 S   2.6  1.6  36:58.63 compiz
29789 paras     20   0  666292  36848  28652 S   2.6  0.6   0:03.85 gnome-terminal-
    1 root      20   0  185800   4556   2936 S   0.0  0.1   0:03.14 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.03 kthreadd
    4 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kworker/0:0H
    6 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 mm_percpu_wq
    7 root      20   0       0      0      0 S   0.0  0.0   0:01.55 ksoftirqd/0
    8 root      20   0       0      0      0 I   0.0  0.0   0:52.59 rcu_sched
    9 root      20   0       0      0      0 I   0.0  0.0   0:00.00 rcu_bh
```

```
top - 20:17:44 up 1 day, 20:59,  1 user,  load average: 0.92, 1.09, 1.31
Tasks: 237 total,   1 running, 183 sleeping,   0 stopped,   0 zombie
%Cpu(s):   0.8/0.2     1[|                                                    ]
KiB Mem :  5939268 total,  1463788 free,  1093036 used,  3382444 buff/cache
KiB Swap:  6801404 total,  6288476 free,   512928 used.  4129196 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
30956 paras     20   0   41980   3704   3016 R   0.3  0.1   0:01.13 top -u paras
 1856 paras     20   0   45360   2564   2128 S   0.0  0.0   0:00.07 /lib/systemd/systemd --user
 1857 paras     20   0   63880    416      0 S   0.0  0.0   0:00.00 (sd-pam)
 1865 paras     20   0  205300   5340   4728 S   0.0  0.1   0:00.99 /usr/bin/gnome-keyring-daemon --daemonize --login
 1896 paras     20   0   46444   3440   2492 S   0.0  0.1   0:00.65 /sbin/upstart --user
 1988 paras     20   0   32860   1696   1568 S   0.0  0.0   0:00.15 upstart-udev-bridge --daemon --user
 1999 paras     20   0   43968   3944   2612 S   0.0  0.1   0:18.17 dbus-daemon --fork --session --address=unix:abstract=/tmp/dbus-WLnJWhB0Kz
 2011 paras     20   0   86344   3852   3592 S   0.0  0.1   0:00.77 /usr/lib/x86_64-linux-gnu/hud/window-stack-bridge
 2043 paras     20   0  274532   3068   2656 S   0.0  0.1   0:00.14 /usr/lib/gvfs/gvfsd
 2048 paras     20   0  406864   2536   2536 S   0.0  0.0   0:00.00 /usr/lib/gvfs/gvfsd-fuse /run/user/1000/gvfs -f -o big_writes
 2057 paras     20   0  264272   3552   3232 S   0.0  0.1   0:00.02 /usr/lib/ibus/ibus-dconf
 2058 paras     20   0  481844  14316   9072 S   0.0  0.2   0:26.53 /usr/lib/ibus/ibus-ui-gtk3
 2060 paras     20   0  427648   9228   8048 S   0.0  0.2   0:08.17 /usr/lib/ibus/ibus-x11 --kill-daemon
 2080 paras     20   0   32868   1168    968 S   0.0  0.0   0:03.73 upstart-dbus-bridge --daemon --session --user --bus-name session
 2081 paras     20   0   32792    124      0 S   0.0  0.0   0:02.10 upstart-dbus-bridge --daemon --system --user --bus-name system
 2091 paras     20   0  188388   2676   2584 S   0.0  0.0   0:14.66 /usr/lib/ibus/ibus-engine-simple
 2114 paras     20   0   41416   1844   1652 S   0.0  0.0   0:00.05 upstart-file-bridge --daemon --user
 2121 paras     20   0  524848  14400   9612 S   0.0  0.2   0:19.63 /usr/lib/x86_64-linux-gnu/bamf/bamfdaemon
 2122 paras     20   0  166536   2188   2000 S   0.0  0.0   0:00.26 gpg-agent --homedir /home/paras/.gnupg --use-standard-socket --daemon
```

https://www.geeksforgeeks.org/top-command-in-linux-with-examples/

Memory Management

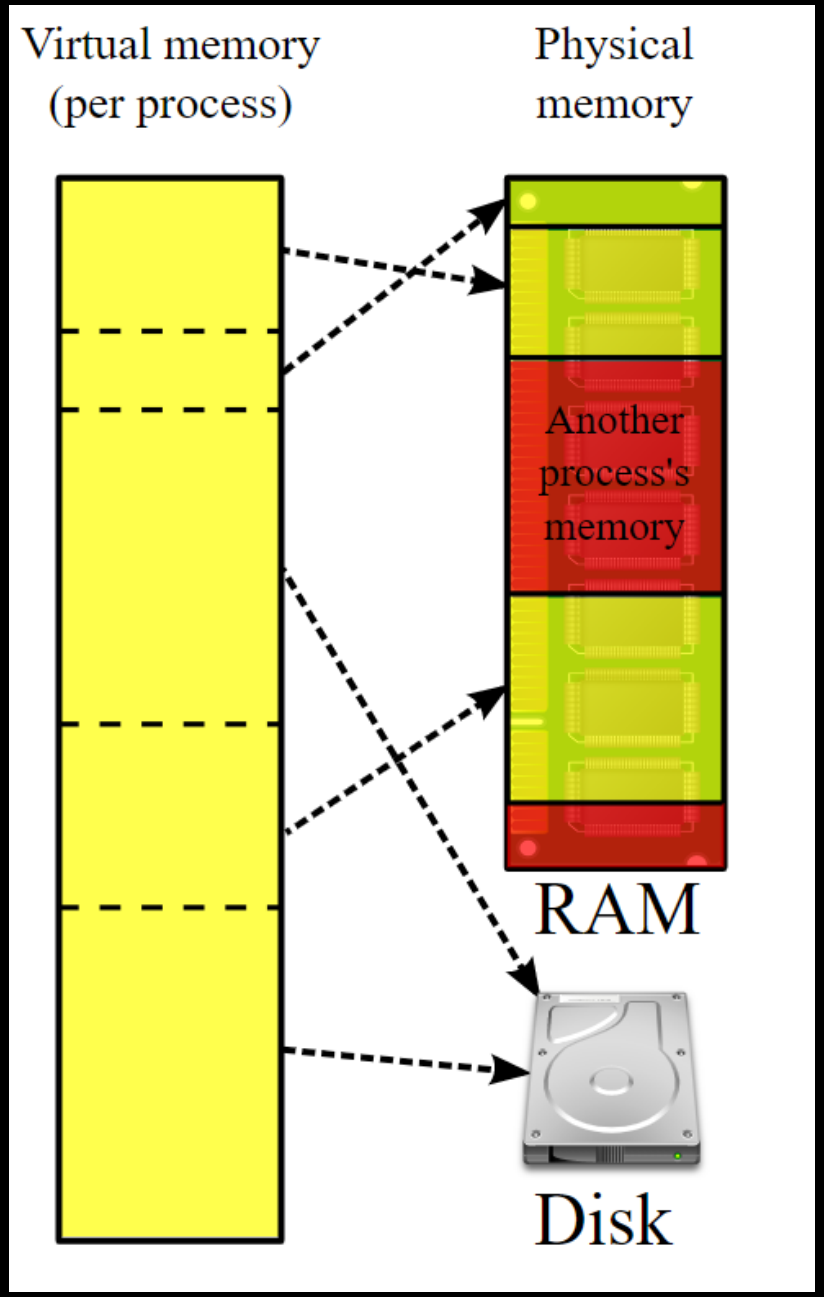


Typical structure of a computer memory hierarchy.

https://www.researchgate.net/figure/Typical-structure-of-a-computer-memory-hierarchy_fig1_281805561

Main Memory

- No memory abstraction – Basic Hardware
- Memory Abstraction: Address spaces
- Virtual Memory

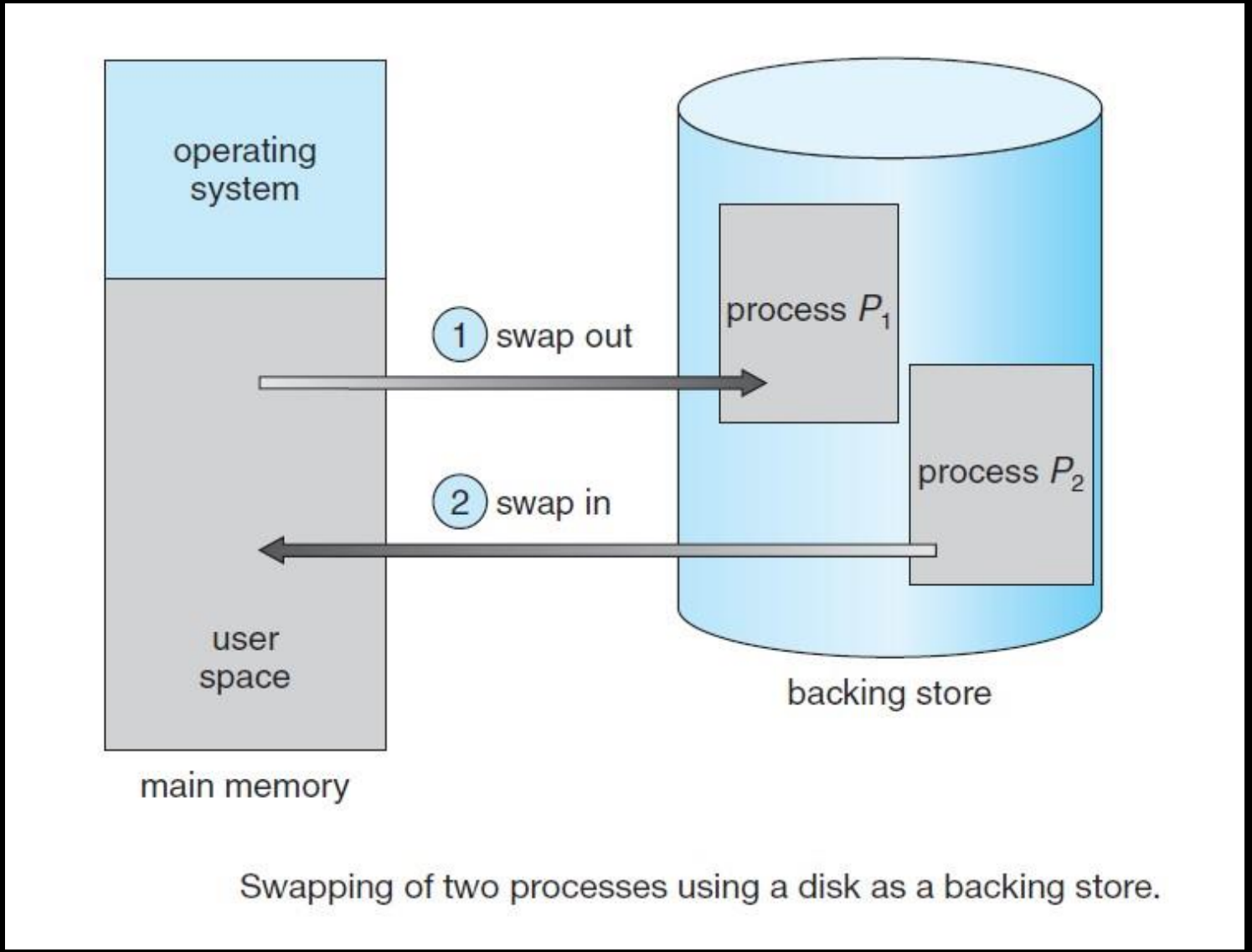


- Main memory and the registers built into the processor itself are the only general-purpose storage that the CPU can access directly.

- There are machine instructions that take memory addresses as arguments, but none that take disk addresses. Therefore, any instructions in execution, and any data being used by the instructions, must be in one of these direct-access storage devices.

- If they are not in memory, they must be moved there before the CPU can operate on them.

Memory Swapping



Swapping of two processes using a disk as a backing store.

Memory allocation

- First fit
  allocate the first hole that is big enough. Searching can start either at the beginning of the set of holes or at the location where the previous first-fit search ended. We can stop searching as soon as we find a free hole that is large enough.

- **Best fit**. Allocate the smallest hole that is big enough. We must search the entire list, unless the list is ordered by size. This strategy produces the smallest leftover hole.

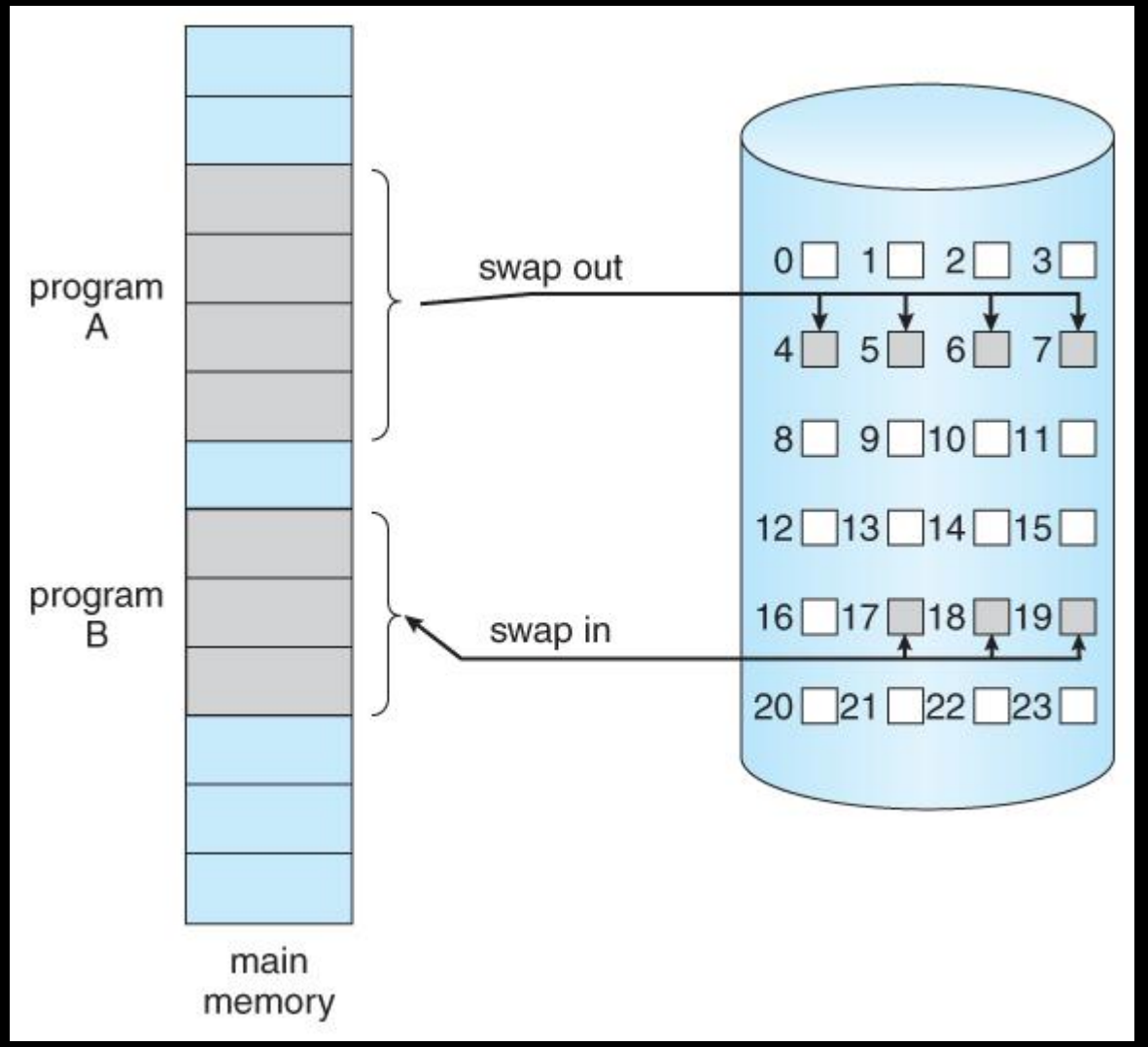
- **Worst fit**. Allocate the largest hole. Again, we must search the entire list, unless it is sorted by size. This strategy produces the largest leftover hole, which may be more useful than the smaller leftover hole from a best-fit approach.

Memory Paging

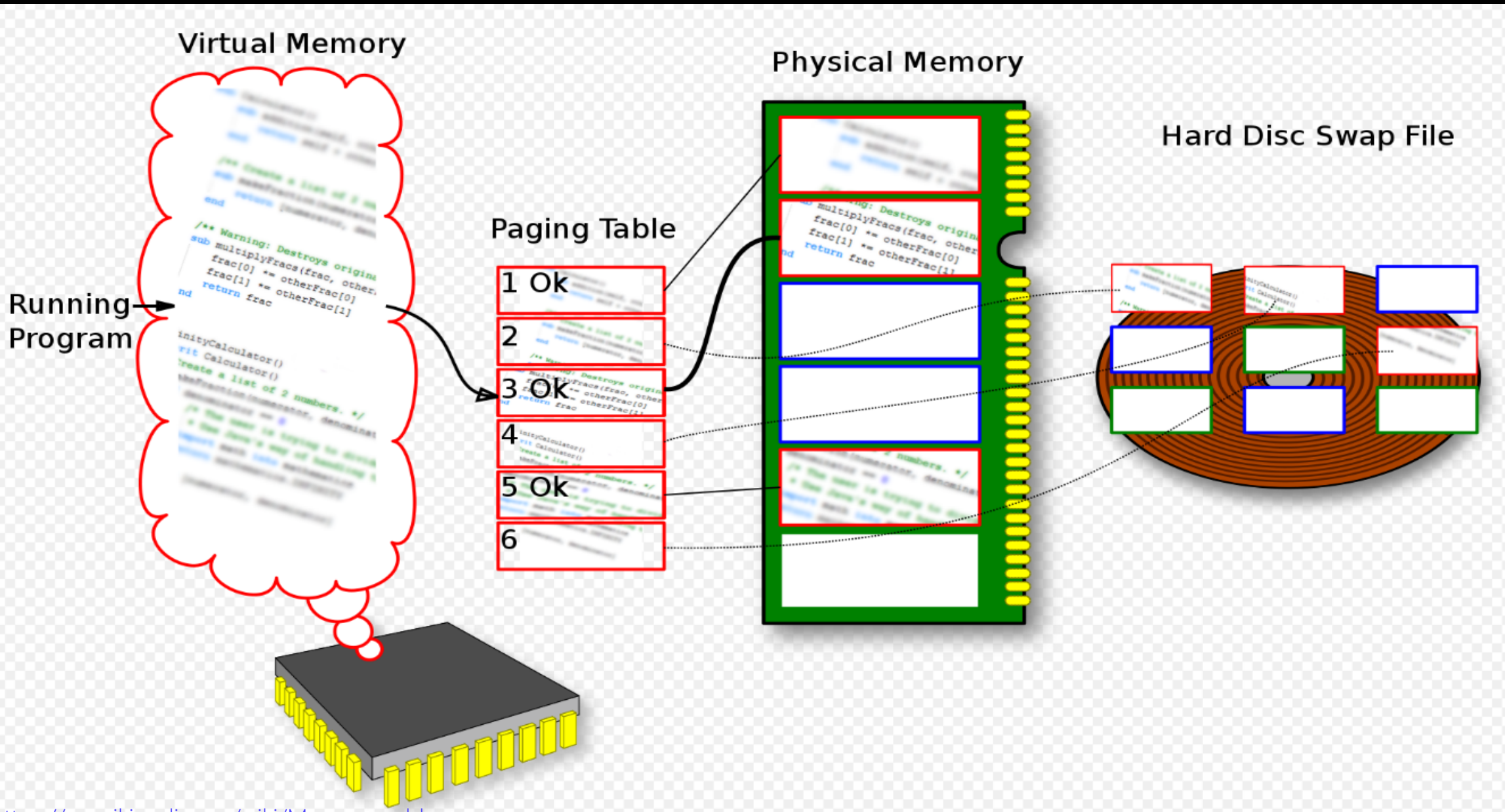
- Paging is a computer memory management function that presents storage locations to the computer's CPU as additional memory, called virtual memory. Each piece of data needs a storage address.

- A page is a fixed-length (4KB) contiguous block of virtual memory residing on disk.
- Pages are the logical blocks, while frames are the physical ones.

- Swapping occurs when whole process is transferred to disk. Paging occurs when some part of a process is transferred to disk.



https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/9_VirtualMemory.html



https://en.wikipedia.org/wiki/Memory_address

Part II

- Controlling access to resources
- Authentication
- Operating systems security

Access Control Context

- **Authentication**: Verification that the credentials of a user or other system entity are valid
- **Authorization**: The granting of a right or permission to a system entity to access a system resource. This function determines who is trusted for a given purpose
- **Audit**: An independent review and examination of system records and activities in order to test for adequacy of system controls, to ensure compliance with established policy and operational procedures, to detect breaches in security, and to recommend any indicated changes in control, policy and procedures

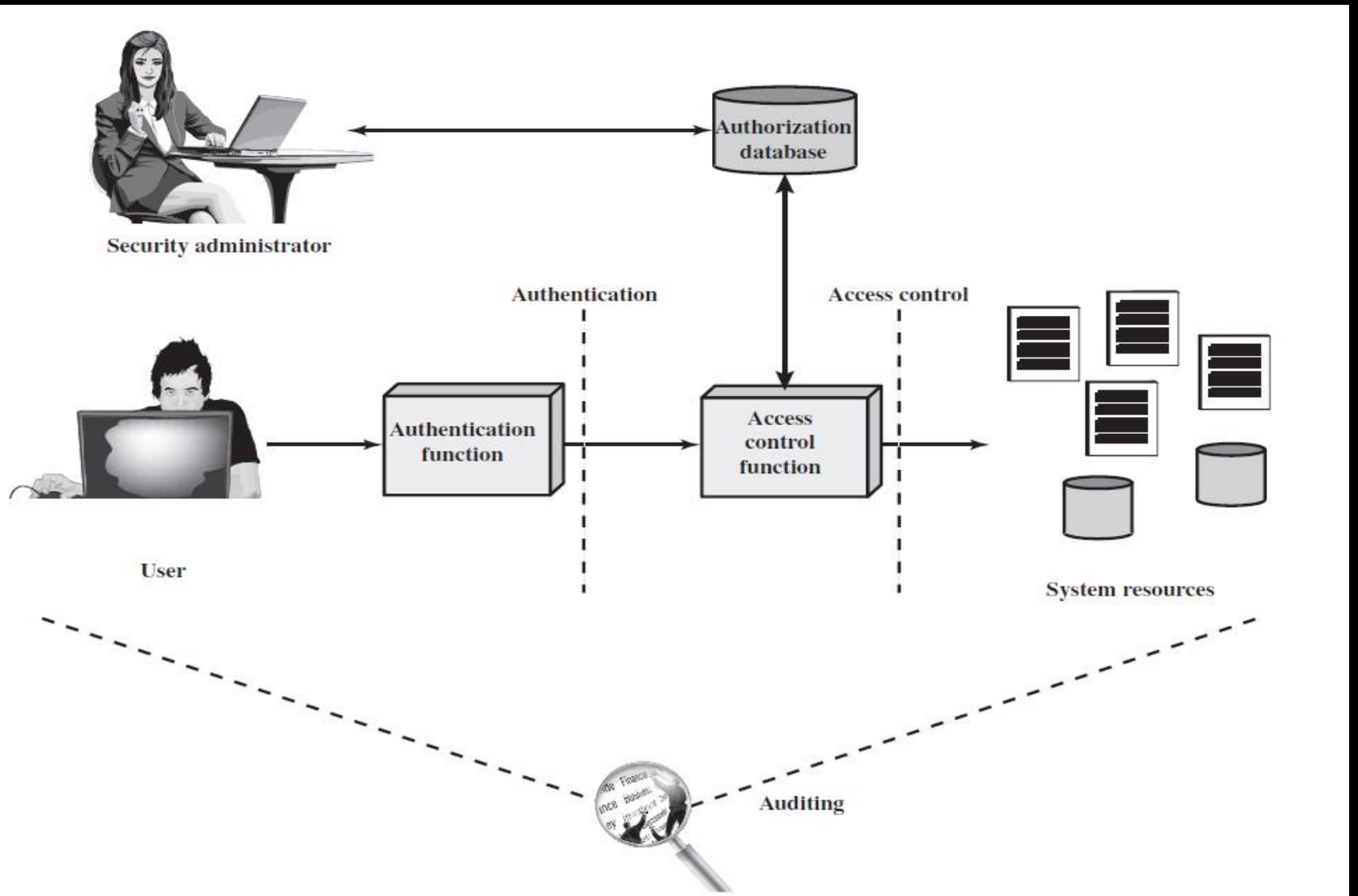


Figure **Relationship Among Access Control and Other Security Functions**
Source: Based on [SAND94].

SAND94 Sandhu, R., and Samarati, P. "Access Control: Principles and Practice." IEEE Communications Magazine, February 1994.

Access Control Policies

- **Discretionary access control (DAC):** Controls access based on the identity of the requestor and on access rules (authorizations) stating what requestors are (or are not) allowed to do. This policy is termed discretionary because an entity might have access rights that permit the entity, by its own volition, to enable another entity to access some resource.

- **Mandatory access control (MAC):** Controls access based on comparing **security labels** (which indicate how sensitive or critical system resources are) with security clearances (which indicate system entities that are eligible to access certain resources). This policy is termed mandatory because an entity that has clearance to access a resource may not, just by its own volition, enable another entity to access that resource.

- **Role-based access control (RBAC):** Controls access based on the roles that users have within the system and on rules stating what accesses are granted to users in given roles. Access is controlled at the system level, outside of user control.

- **Attribute-based access control (ABAC):** Controls access based on attributes of the user, the resource to be accessed and current environmental conditions.

Subjects and Objects

A **subject** is an entity capable of accessing objects. Generally, the concept of subject equates with that of process. Any user or application gains access to an object by means of a process that represents that user or application. The process takes on the attributes of the user, such as access rights.

• **Owner**: This may be the creator of a resource, such as a file. For system resources, ownership may belong to a system administrator. For project resources, a project administrator or leader may be assigned ownership.

• **Group**: In addition to the privileges assigned to an owner, a named group of users may also be granted access rights, such that membership in the group is sufficient to exercise these access rights. In most schemes, a user may belong to multiple groups.

• **World**: The least amount of access is granted to users that can access the system but are not included in the categories owner or group for this resource.

An **object** is a resource to which access is controlled. In general, an object is an entity used to contain and/or receive information.

- Examples include records, blocks, pages, segments, files, portions of files, directories, directory trees, mailboxes, messages, and programs.

- Some access control systems also encompass, bits, bytes, words, processors, communication ports, clocks, and network nodes.

- An **access right** describes the way in which a **subject** may access an **object**

  - **Read**: User may view information in a system resource (e.g., a file, selected records in a file, selected fields within a record, or some combination). Read access includes the ability to copy or print.

  - **Write**: User may add, modify, or delete data in system resource (e.g., files, records, programs). Write access includes read access.

  - **Execute**: User may execute specified programs

  - **Delete**: User may delete certain system resources, such as files or records

  - **Create**: User may create new files, records, or fields

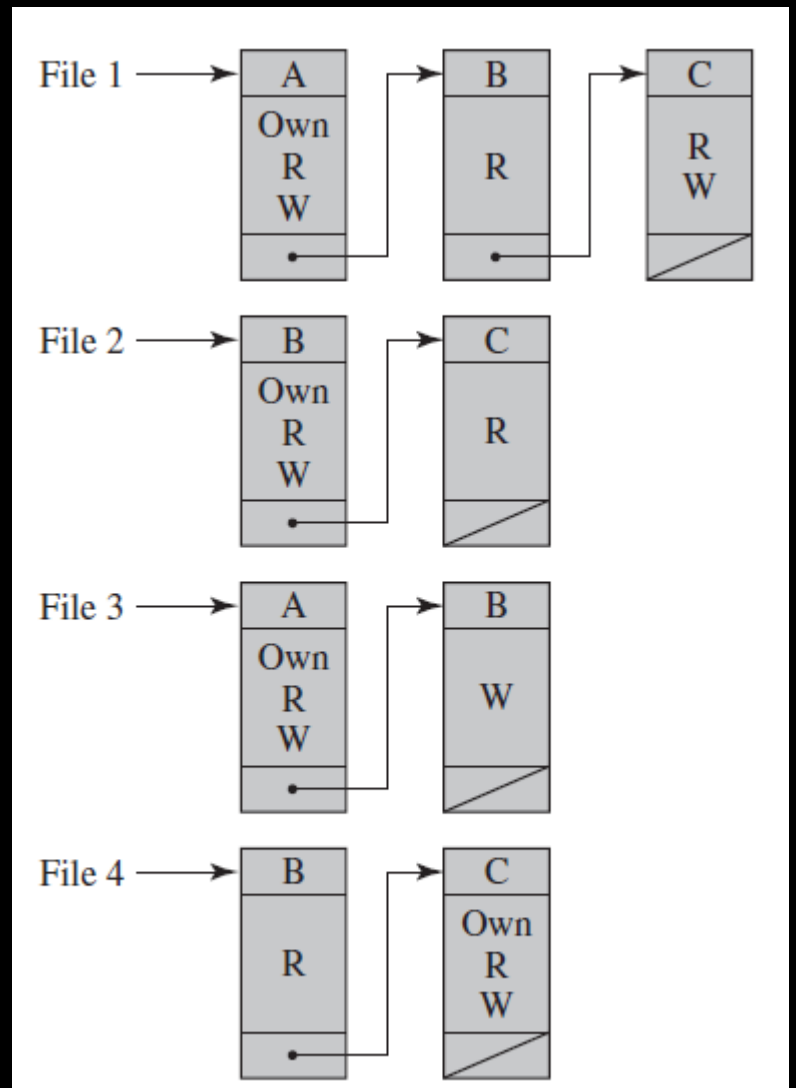
  - **Search**: User may list the files in a directory or otherwise search the directory

DAC – Discretionary Access Control

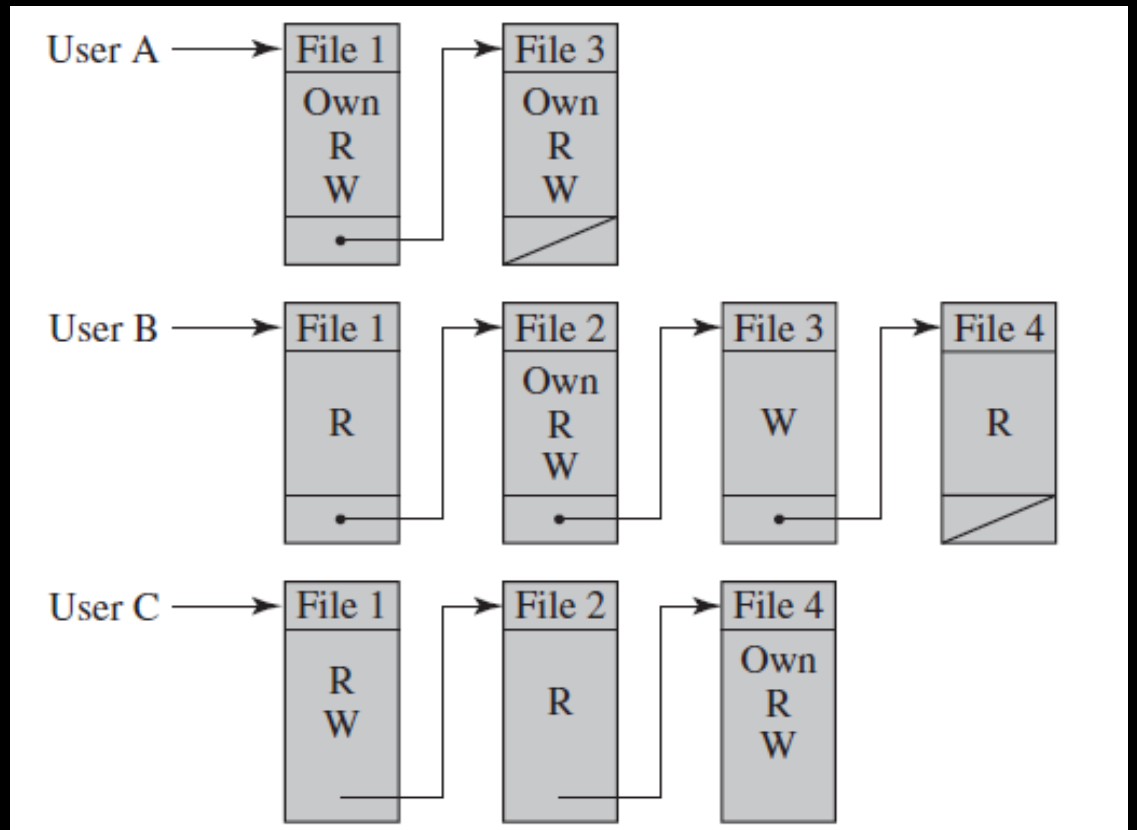
- Access control matrix
- Access Control Lists
- Capabilities lists

		OBJECTS			
		File 1	File 2	File 3	File 4
SUBJECTS	User A	Own Read Write		Own Read Write	
	User B	Read	Own Read Write	Write	Read
	User C	Read Write	Read		Own Read Write

(a) Access matrix



(b) Access control lists for files of part (a)



(c) Capability lists for files of part (a)

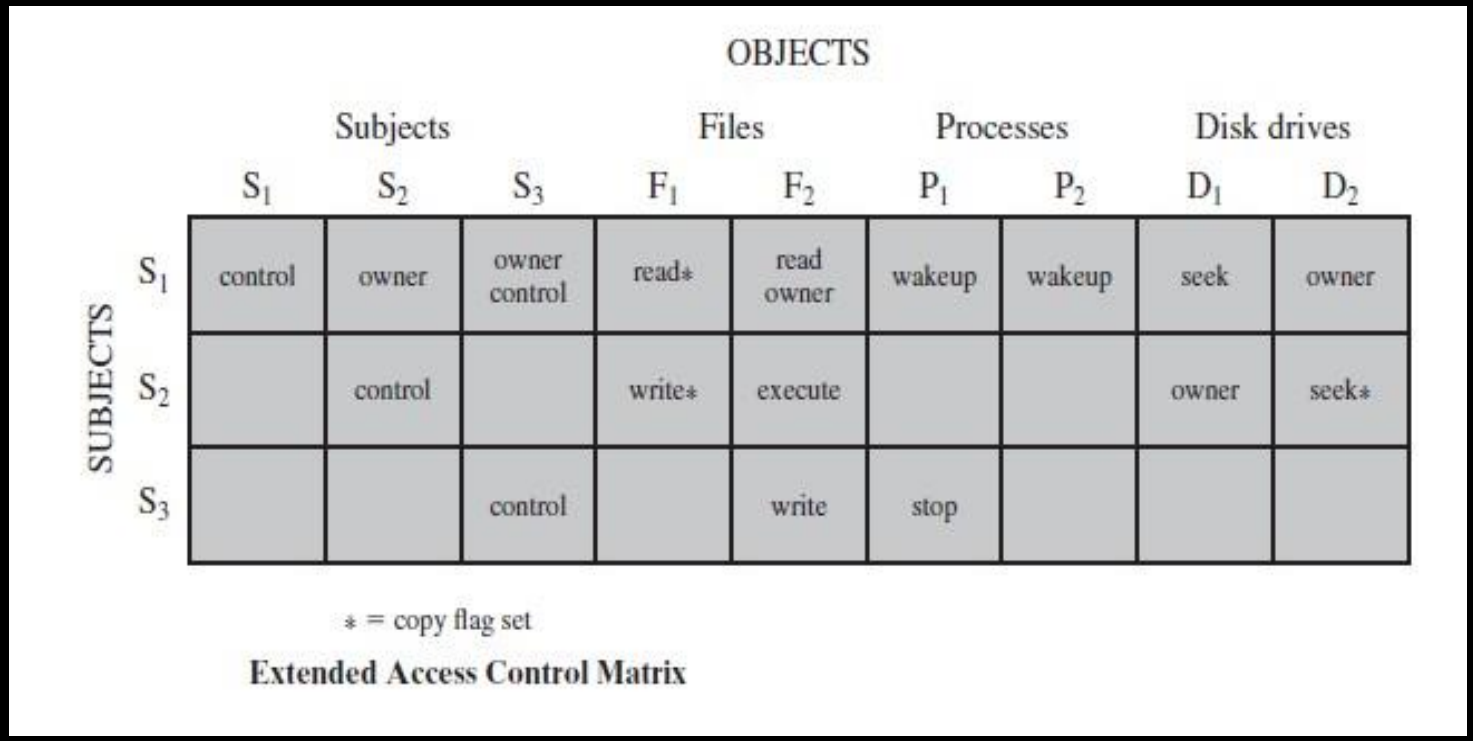
An Access Control Model

- Processes: Access rights include the ability to delete a process, stop (block), and wake up a process

- Devices: Access rights include the ability to read/write the device, to control its operation (e.g., a disk seek), and to block/unblock the device for use

- Memory locations or regions: Access rights include the ability to read/write certain regions of memory that are protected such that the default is to disallow access

- Subjects: Access rights with respect to a subject have to do with the ability to grant or delete access rights of that subject to other objects.



Extended Access Control Matrix

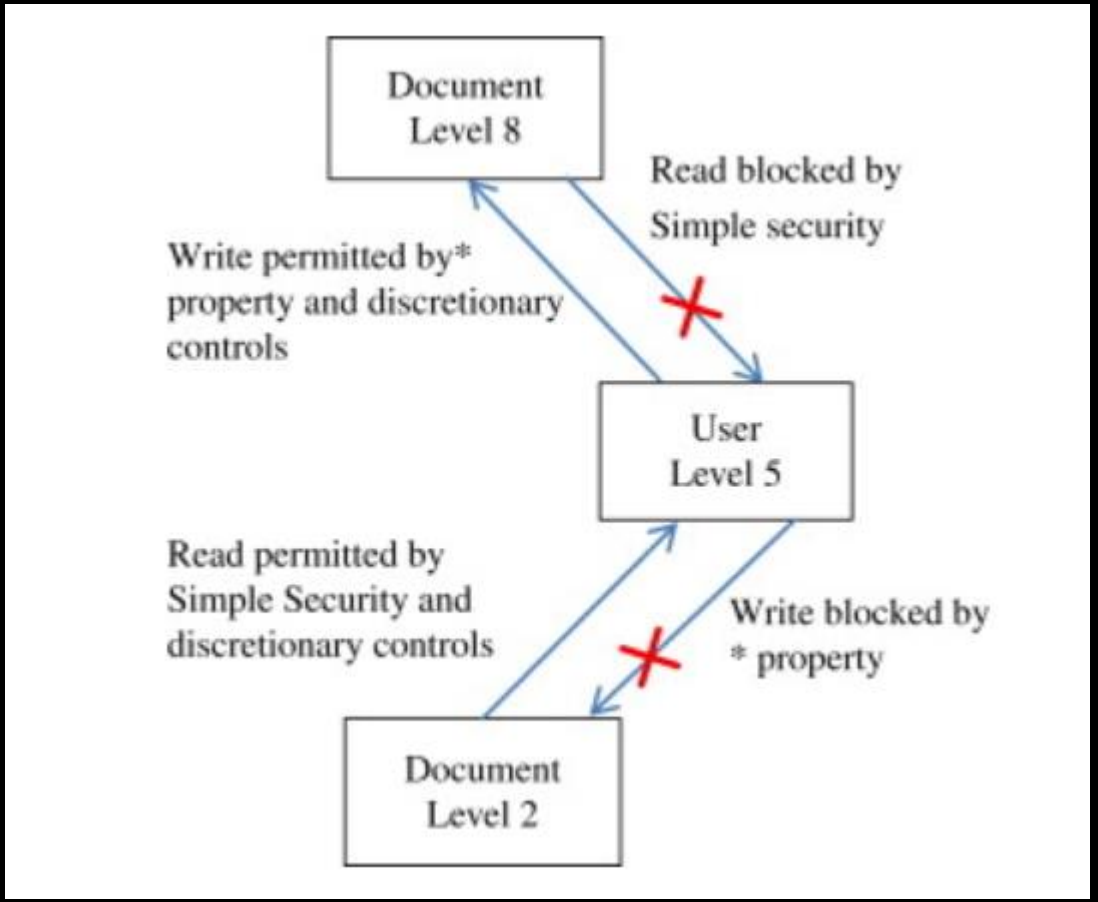
MAC – Mandatory Access Control

The Bell-LaPadula Model

- Designed for military security – unclassified, confidential, secret and top secret. Focused on confidentiality.
- The simple security property: A process running at security level k can read only objects at its level or lower. For example, a general can read a lieutenant's documents but a lieutenant cannot read a general's Documents (no read up).
- The *(star) security property: A process running at security level k can write only objects at its level or higher. For example, a lieutenant can append a message to a general's mailbox telling everything he knows, but a general cannot append a message to a lieutenant's mailbox telling everything he knows because the general may have seen top-secret documents that may not be disclosed to a lieutenant (no write down).

In addition, the BLP model makes a provision for discretionary access control (DAC).

• **Discretionary Security Property**: An individual (or role) may grant to another individual (or role) access to a document based on the owner's discretion, constrained by the MAC rules. Thus, a subject can exercise only accesses for which it has the necessary authorization, and which satisfy the MAC rules.



https://www.sciencedirect.com/topics/computer-science/lapadula-model

Authentication

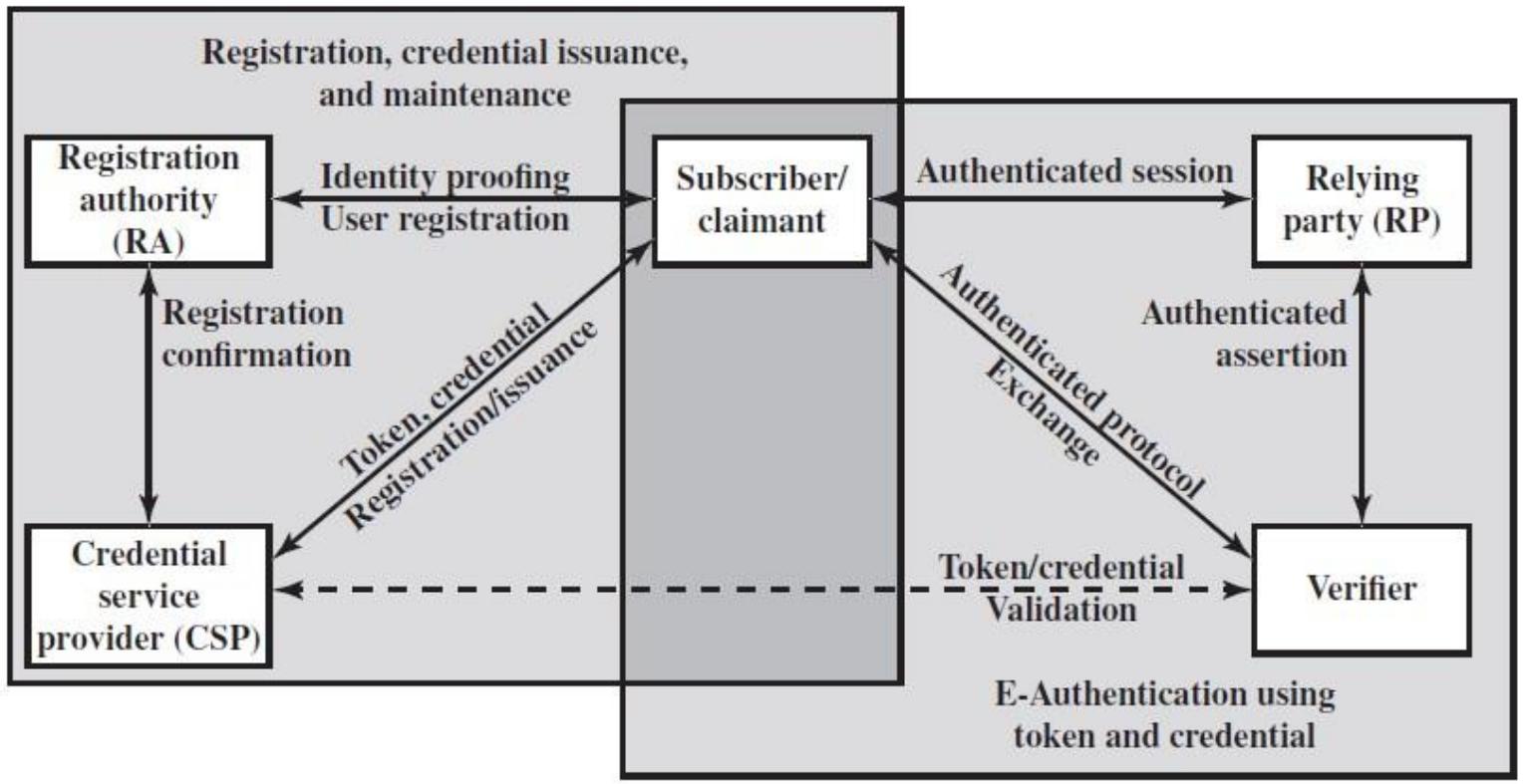
• Something the individual knows: Examples includes a password, a personal identification number (PIN), or answers to a prearranged set of questions.

• Something the individual possesses: Examples include electronic keycards, smart cards, and physical keys. This type of authenticator is referred to as a token.

• Something the individual is (static biometrics): Examples include recognition by fingerprint, retina, and face.

• Something the individual does (dynamic biometrics): Examples include recognition by voice pattern, handwriting characteristics, and typing rhythm.

NIST Special Publication 800-63-3
Digital Identity Guidelines



The NIST SP 800-63-2 E-Authentication Architectural Model

Operating Systems Security

System Security Planning

- The purpose of the system, the type of information stored, the applications and services provided, and their security requirements
- The categories of users of the system, the privileges they have, and the types of information they can access
- How the users are authenticated
- How access to the information stored on the system is managed
- What access the system has to information stored on other hosts, such as file or database servers, and how this is managed.
- Who will administer the system, and how they will manage the system (via local or remote access)
- Any additional security measures required on the system, including the use of host firewalls, anti-virus or other malware protection mechanisms, and logging

Operating Systems Hardening

- Install and patch the operating system
- Harden and configure the operating system to adequately address the identified security needs of the system by:
  - Removing unnecessary services, applications, and protocols
  - Configuring users, groups, and permissions
  - Configuring resource controls
- Install and configure additional security controls, such as anti-virus, host-based firewalls, and intrusion detection systems (IDS), if needed
- Test the security of the basic operating system to ensure that the steps taken adequately address its security needs
- Microsoft Security Baselines - Microsoft-recommended configuration settings, broadly known and well-tested

Windows vs Linux

- Modularity and User Privileges

Windows is not modular; user accounts mostly have administrative privileges (UAC). Linux user accounts typically do not have root access, which is required to make major changes to the system.

- Automated Functions

Windows automates as many functions as possible for the user thus also increasing risk: removable media malware. On Linux is that running an ".exe" is much harder. You'll have to chmod +x a file before you can run it.

- Open-Source and Transparency

Linux is open-source - if a vulnerability is present in a program or distribution, developers tend to find it faster and find solutions to close that vulnerability. While open-source programs exist for Windows, the operating system as a whole is closed.

- Security Through Variety

Not exactly a security feature, but the variety available to Linux users in both operating system and applications makes it much harder for malware to target a majority of users.

Desktop OS market share: Windows has between 85-90% while Linux has 2-3%.

Linux Vs. Windows

	Linux	Windows
Price	Free	$$$
Ease	Not user-friendly	User friendly
Reliability	Very reliable. Often runs for months or years	Often requires reboot
Software	Mostly enterprise level softwares	Much larger selection of softwares e.g. office, games, utlities etc.
Multi-tasking	Best for multi-tasking	Multi-tasking is avaliable but with very high cpu or memory resources
Security	Very secure	Some what secure
Open source	Open to public	No an open source OS

Security and hardening – Documenting your actions

It is best practice to document every hardening activity you perform.

This is useful for audits, investigations and future maintenance on performed on the same host.

The document should contain the benchmark or standard used for reference and the following information about the host:

- **Hostname**
- **IP address**
- **Mac address**
- **Name of the person who is doing the hardening**
- **Date**
- **Asset Number**

Services – Uninstall unused services

Unless a system is specifically set up to run these services, it is recommended that this packages be removed to reduce the potential attack surface. Check the following list:

Internet Daemon (xinetd, inetd, openbsd-inetd), **X Window System** (xserver-xorg*)**, Avahi Server** (avahi-daemon)**, CUPS** (cups)**, DHCP server** (isc-dhcp-server)**, LDAP server** (slapd)**, NFS** (rpcbind)**, DNS Server** (bind9)**, FTP Server** (vsftpd)**, HTTP server** (apache2)**, IMAP and POP3** (dovecot-imapd, dovecot-pop3d)**, Samba** (samba)**, HTTP Proxy server** (squid)**, SNMP Server** (snmpd), **rsync** (rsync), NIS server (nis)

Remediation:

Uninstall the app: **# apt purge <<service name>>**

* for Avahi, run these commands before uninstalling the package:

Stop the service: # **systemctl stop avahi-daaemon.service**

Close the socket: **# systemctl stop avahi-daemon.socket**

Audit:

Check if the app is installed: # **dpkg -s <<service name>>**

This should be relevant for Linux systems configured to run in both server and workstation mode

Services – Uninstall unused service clients

If not used it is recommended to uninstall this packages to reduce the potential attack surface. Check the following list:

NIS server (nis), **RSH client** (rsh-client), **Talk** (talk), **Telnet** (telnet), LDAP client (ldap-utils), Remote Procedure Call (rpcbind),

Remediation:

Uninstall the app: **# apt purge <<service name>>**

Audit:

Check if the app is installed: # **dpkg -s <<service name>>**

This should be relevant for Linux systems configured to run in both server and workstation mode

Initial Setup – Disable unused filesystems

If a filesystem type is not needed it should be disabled in order to reduce the local attack surface of the system.

Native Linux file systems are designed to ensure that built-in security controls function as expected.

Non-native filesystems can lead to unexpected consequences for security and functionality of the system.

These file systems should be considered for evaluation: Cramfs, freevxfs, jffs2, hfs, hfsplus, squashfs, udf

Remediation:

List the supported filesystems: # **cat /proc/filesystems**

Create a file named:# **touch /etc/modprobe.d/<<fs_name>>.conf**

Add this line in the file above: **install <<fsname>> /bin/true**

Unload the module from the kernel: **# rmmod <<fs_name>>**

Audit:

Test what happens when the kernel tries to load one of the modules for the disabled file system: **# modprobe -n -v <<fs_name>>**

install /bin/true

 # lsmod | grep <<fs_name>>

No result

This should be relevant for Linux systems configured to run in both server and workstation mode

Initial Setup – Enable sticky bit on all world-writable dir

Setting the sticky bit on world writable directories prevents users from deleting or renaming files in that directory that are not owned by them.

With the sticky key set only the owner and the root user can rename or delete files in the world-writable directories.

Remediation:

Manual for each directory: **chmod +t <</dir_name>>**

Script: **# df --local -P | awk '{if (NR!=1) print $6}' | xargs -I '{}' find '{}' -xdev -type d \(-perm -0002 -a ! -perm -1000 \) 2>/dev/null | xargs -I '{}' chmod a+t '{}'**

Audit:

Run: **# df --local -P | awk '{if (NR!=1) print $6}' | xargs -I '{}' find '{}' -xdev -type d \(-perm -0002 -a ! -perm -1000 \) 2>/dev/null**

Sticky bit is an attribute relevant only for directories. If the sticky bit is set for a world-writable directory, only the owner of the file, owner of the directory or root user can rename or delete the file. The sticky bit can be set for files, but the Linux kernel will ignore it.

Initial Setup – Disable automount

With automounting enabled anyone with physical access could attach a USB drive or disc and have its contents available in system even if they lacked permissions to mount it themselves.

This control should align with the tolerance of the use of portable drives and optical media in the organization. On a server requiring an admin to manually mount media can be part of defense-in-depth to reduce the risk of unapproved software or information being introduced or proprietary software or information being exfiltrated. If admins commonly use flash drives and Server access has sufficient physical controls, requiring manual mounting may not increase security.

Remediation:

Disable automounting: **# systemctl --now mask autofs**

Or

Uninstall autofs altogether: # **apt purge autofs**

Audit:

Run: **#dpkg –s autofs**

This should be relevant for Linux systems configured to run in server mode

Initial Setup – Disable USB Storage

USB storage provides a means to transfer and store files insuring persistence and availability of the files independent of network connection status. Its popularity and utility has led to USB-based malware being a simple and common means for network infiltration and a first step to establishing a persistent threat within a networked environment.

An alternative solution to disabling the usb-storage module may be found in USBGuard. Use of USBGuard and construction of USB device policies should be done in alignment with company policy.

Remediation:

Disable usb storage: **# install usb-storage /bin/true**

and

Unload module from kernel: # **rmmod usb-storage**

Audit:

Check what happens if the kernel tries to load the module: **# modprobe -n -v usb-storage install /bin/true**

Check if the module is currently loaded: **# lsmod | grep usb-storage**

This should be relevant for Linux systems configured to run in serve mode

Initial Setup – Configure a Filesystem Integrity Checking

AIDE is a file integrity checking tool which can detect unauthorized changes to configuration files by alerting when the files are changed.

The tool takes a snapshot of filesystem state including modification times, permissions, and file hashes which can then be used to compare against the current state of the filesystem to detect modifications to the system.

Configure the tool to perform checks regularly and make sure Prelink app is not enabled as it will interfere with AIDE.

Remediation:

Install the tool: **# apt install aide aide-common**

Initialize the tool: **# aideinit**

and

 # mv /var/lib/aide/aide.db.new /var/lib/aide/aide.db

Configure a cronjob**: # crontab -u root –e**

And

Add the following line to crontab: **0 5 * * * /usr/bin/aide.wrapper --config /etc/aide/aide.conf --check**

Audit:

Check the install status: **# dpkg -s aide | grep 'Status: install ok installed'**

and

 # dpkg -s aide-common | grep 'Status: install ok installed'

Check if a cronjob is configured: **# crontab -u root -l | grep aide**

and

 # find /etc/cron.* /etc/crontab -name 'aide' -type f

This should be relevant for Linux systems configured to run in server mode

Initial Setup – Configure sudo

sudo allows a permitted user to execute a command as the superuser or another user, as specified by the security policy. The invoking user's real (not effective) user ID is used to determine the username with which to query the security policy.

Configure sudo to use PTY - Attackers can run a malicious program using sudo, which would again fork a background process that remains even when the main program has finished executing.

While by default sudo would log events in /var/log/auth.log, you can configure a custom log file which simplifies auditing.

Remediation:

Install sudo: # **apt install sudo**

Edit /etc/sudoers and add: **Defaults use_pty**

Edit /etc/sudoers and add: **Default logfile=<<path to log file>>**

Audit:

Check if sudo is installed: **# dpkg -s sudo**

Check that pty is the default: **# grep 'use_pty' /etc/sudoers /etc/sudoers.d**

This should be relevant for Linux systems configured to run in both server and workstation mode

Initial Setup – Set bootloader password

Setting the boot loader password will force anyone rebooting the system to enter a password before being able to set command line boot parameters. Requiring a boot password upon execution of the boot loader will prevent an unauthorized user from entering boot parameters or changing the boot partition.

To avoid a misconfiguration that might result in the inability to boot the OS, add "--unrestricted" to /etc/grub.d/10_custom. The line should look like this: CLASS="--class gnu-linux --class gnu --class os --unrestricted".

The misconfiguration will result in the inability to access grub config during boot-up, but you are still able to load the OS and recover by reediting the files bellow.

Remediation:
Create an encrypted password: **# grub-mkpasswd-pbkdf2**
Enter password: <password>
Reenter password: <password>
PBKDF2 hash of your password is <encrypted-password>
Add the encrypted password into one of the custom /etc/grub.d configuration files (new or existing):
cat <<EOF
set superusers="<username>"
password_pbkdf2 <username> <encrypted-password>
EOF

Update GRUB: **# update-grub**

Audit:
Check if a super user and password are defined in the grub configuration file:
 # grep "^set superusers" /boot/grub/grub.cfg
and
 # grep "^password" /boot/grub/grub.cfg

This should be relevant for Linux systems configured to run in both server and workstation mode

Initial Setup – Set permissions on the bootloader config

Setting the permissions to read and write for root only prevents non-root users from seeing the boot parameters or changing them. Non-root users who read the boot parameters may be able to identify weaknesses in security upon boot and be able to exploit them

Remediation:

Change the owner of the grub config file: **# chown root:root /boot/grub/grub.cfg**

Configure permissions for the file: **# chmod og-rwx /boot/grub/grub.cfg**

Audit:

Check the permission on the file: **# stat /boot/grub/grub.cfg**

This should be relevant for Linux systems configured to run in both server and workstation mode

Initial Setup – Restrict access to single user mode

Single user mode is used for recovery when the system detects an issue during boot or by manual selection from the bootloader.

Remediation:
Set a password for root: **# passwd root**
Audit:
Check if the user root has a password: # **grep -a root /etc/shadow**

The root password is not set during the installation process.

Initial Setup – Activate ASLR

Address space layout randomization (ASLR) is an exploit mitigation technique which randomly arranges the address space of key data areas of a process.

Randomly placing virtual memory regions will make it difficult to write memory page exploits as the memory placement will be consistently shifting

Remediation:

Add the following line in /etc/sysctl.conf or a /etc/sysctl.d/* file: **kernel.randomize_va_space = 2**

and

Activate the kernel parameter: **# sysctl -w kernel.randomize_va_space=2**

Audit:

Check the output of **# sysctl kernel.randomize_va_space**

Or

 # grep "kernel\.randomize_va_space" /etc/sysctl.conf /etc/sysctl.d/*

By default, ASLR is enabled but it might still be disabled on developer systems for debugging purposes. Keeping this setting enabled does not affect system performance.

Initial Setup – Install and configure AppArmor

AppArmor provides a Mandatory Access Control (MAC) system that greatly augments the default Discretionary Access Control (DAC) model. Under AppArmor MAC rules are applied by file paths instead of by security contexts as in other MAC systems. As such it does not require support in the filesystem and can be applied to network mounted filesystems for example. AppArmor security policies define what system resources applications can access and what privileges they can do so with. This automatically limits the damage that the software can do to files accessible by the calling user.

Remediation:

Install AppArmor: **# apt install apparmor**

Edit /etc/default/grub and add the appermor=1 and security=apparmor parameters to the GRUB_CMDLINE_LINUX= line:

GRUB_CMDLINE_LINUX="apparmor=1 security=apparmor“

Update GRUB: **# update-grub**

Set all profiles to enforce / complain mode: **# aa-enforce /etc/apparmor.d/***

Or

 # aa-complain /etc/apparmor.d/*

Make sure the profiles are enforced: **# aa-enforce /etc/apparmor.d/***

Audit:

Check the install status: # **dpkg -s apparmor**

Check the status: **# apparmor_status**

Selinux is a more complex solution but it is not fully supported by Ubuntu based systems. For this OS AppArmor is the recommended solution

Initial Setup – Ensure core dumps are restricted

A core dump is the memory of an executable program. It is generally used to determine why a program aborted. It can also be used to glean confidential information from a core file. The system provides the ability to set a soft limit for core dumps, but this can be overridden by the user.

Remediation:

Add the following line to /etc/security/limits.conf or a /etc/security/limits.d/* file: *** hard core 0**

Set the following parameter in /etc/sysctl.conf or a /etc/sysctl.d/* file: **fs.suid_dumpable = 0**

Run the following command to set the active kernel parameter: **# sysctl -w fs.suid_dumpable=0**

If systemd-coredump is installed edit /etc/systemd/coredump.conf and add/modify the following lines:

Storage=none

ProcessSizeMax=0

Run the command: **systemctl daemon-reload**

Audit:

The following commands must all return value 0: **# grep -E '*hard.*core.*' /etc/security/limits.conf /etc/security/limits.d/***

and

 # sysctl fs.suid_dumpable

and

 # grep "fs.suid_dumpable" /etc/sysctl.conf /etc/sysctl.d/*

Check if system-coredump is installed: **# systemctl is-enabled coredump.service**

This should be relevant for Linux systems configured to run in server mode

Initial Setup – Manage the graphical display manager

If a graphical login is not required, it should be removed to reduce the attack surface of the system. If a graphical login is required, last logged in user display should be disabled, and a warning banner should be configured. Displaying the last logged in user eliminates half of the Userid/Password equation that an attacker would need to log on.

Remediation:

To stop displaying the users on the login screen:

Locate the light display manager (lightdm): **# grep –i greeter /etc/**

Add following line to the conf file: **greeter-hide-users=true**

Audit:

Locate and inspect the light display manager config file (lightdm): **# grep –i greeter /etc/**

The configured users are not displayed on the login screen

This should be relevant for Linux systems configured to run in server mode

Initial Setup – Harden /tmp

The /tmp is a world-writable directory used for temporary storage by all users and some applications.

Making /tmp its own file system or mounting it in a separate partition allows an administrator to set the noexec, nosuid, nodev options on the mount.

This way an attacker can't execute code, create hardlinks to setuid programs or create block or character special devices in tmp

Remediation 1 (if /tmp has a separate partition):

Edit /etc/fstab: # **sudo nano /etc/fstab**

Add mount options for /tmp: # <<deviceid>> **/tmp ext4 defaults,rw,noexec,nodev,nosuid,relatime 0 2**

NOTE: "relatime" reduces the number of write actions to the disk to update the timestep when a file was last accessed. The other two options which can be set are atime (update for every read action) and noatime (last access timestamp is not updated)

Remediation 2 (if /tmp doesn't have a separate partition):

Edit /etc/fstab: # **sudo nano /etc/fstab**

Add mount options for /tmp: **tmpfs /tmp tmpfs defaults,rw,noexec,nodev,nosuid,relatime,size=2G 0 2**

NOTE: If size is not specified for tmpfs size=2G, it will be set automatically to half the available RAM

Audit:

Check mount options for /tmp: # **mount| grep /tmp**

This should be relevant for Linux systems configured to run in both server and workstation mode

Initial Setup – Harden /dev/shm

/dev/shm is a traditional shared memory concept. Similar to /tmp, /dev/shm is a world-writable directory.

An administrator should set the noexec, nosuid, nodev options on the mount.

This way an attacker can't execute code, create hardlinks to setuid programs or create block or character special devices in tmp

Remediation 1 (if /dev/shm doesn't have a separate partition):
Edit /etc/fstab: # **sudo nano /etc/fstab**
Add mount options for /dev/shm: **tmpfs /dev/shm tmpfs defaults,noexec,nodev,nosuid 0 0**

Audit:
Check mount options for /dev/shm: # **mount| grep /dev/shm**
Or
#sudo cat /etc/mtab
Or
#sudo cat /proc/mounts

This should be relevant for Linux systems configured to run in both server and workstation mode

Initial Setup – Separate partition for /var

The /var directory is used by daemons and other system services to temporarily store dynamic data. Some directories created by these processes may be world-writable.

Since the /var directory may contain world-writable files and directories, there is a risk of resource exhaustion if it is not bound to a separate partition.

Remediation:

For new installations, during installation create a custom partition setup and specify a separate partition for /var .

For systems that were previously installed, create a new partition and configure /etc/fstab as appropriate.

Note: When modifying /var it is advisable to bring the system to emergency mode (so auditd is not running), rename the existing directory, mount the new file system, and migrate the data over before returning to multiuser mode.

Audit:

Search for /var in the mounted partitons: # **sudo cat /proc/mounts | grep /var**

Or

#mount| grep /var

This should be relevant for Linux systems configured to run in server mode

Initial Setup – Harden /var/tmp

The /var/tmp is a world-writable directory used for temporary storage by all users and applications.

Mounting/var/tmp in a separate partition protects the system from resource exhaustion and allows an administrator to set the noexec, nosuid, nodev options on the mount.

This way an attacker can't execute code, create hardlinks to setuid programs or create block or character special devices in /var/tmp

Remediation:

Create a separate ext4 partition and mount it in /var/tmp

Edit /etc/fstab: # **sudo nano /etc/fstab**

Add mount options for /var/tmp: <<device uid>> **/var/tmp ext4 rw,noexec,nodev,nosuid,relatime 0 2**

Audit:

Check mount options for /var/tmp: # **mount| grep /var/tmp**

This should be relevant for Linux systems configured to run in both server and workstation mode

Initial Setup – Separate partition for /var/log/audit

The auditing daemon, auditd, stores log data in the /var/log/audit directory.

Since the /var directory may contain world-writable files and directories, there is a risk of resource exhaustion.

There are two important reasons to ensure that data gathered by auditd is stored on a separate partition: protection against resource exhaustion (since the audit.log file can grow quite large) and protection of audit data. The audit daemon calculates how much free space is left and performs actions based on the results. If other processes (such as syslog) consume space in the same partition as auditd , it may not perform as desired.

Remediation:

For new installations, during installation create a custom partition setup and specify a separate partition for /var/log/audit .

For systems that were previously installed, create a new partition and configure /etc/fstab as appropriate.

Note: When modifying /var/log/audit it is advisable to bring the system to emergency mode (so auditd is not running), rename the existing directory, mount the new file system, and migrate the data over before returning to multiuser mode.

Audit:

Search for /var in the mounted partitions: # **sudo cat /proc/mounts | grep /var/log/audit**

Or

 # mount| grep /var/log/audit

This should be relevant for Linux systems configured to run in server mode

Initial Setup – Harden /home

The /home directory is used to support disk storage needs of local users.

Mounting/home in a separate partition protects the system from resource exhaustion and allows an administrator to set the noexec, nosuid, nodev options on the mount.

Since the user partitions are not intended to support devices, set this option to ensure that users cannot attempt to create block or character special devices.

Remediation:

For new installations, create a custom partition setup and specify a separate partition for /home .

For systems that were previously installed, create a new partition and configure /etc/fstab as appropriate and set the nodev mount option

Edit /etc/fstab: # **sudo nano /etc/fstab**

Add mount options for /home: <<device uid>> **/home ext4 defaults, nodev 0 2**

Audit:

Check mount options for /var/tmp: # **mount| grep /home**

This should be performed on Linux systems for which security is paramount.

Initial Setup – Harden removable media partitions

Administrators should set the nodev, noexec and nosuid options on the mount.

Nodev - removable media containing character and block special devices could be used to circumvent security controls by allowing non-root users to access sensitive device files such as /dev/kmem or the raw disk partitions.

Noexec – Setting this option on a file system prevents users from executing programs from the removable media. This deters users from being able to introduce potentially malicious software on the system.

Nosuid - Setting this option on a file system prevents users from introducing privileged programs onto the system and allowing non-root users to execute them.

Remediation:

Edit the /etc/fstab file and add nodev, nosuid and noexec to the fourth field (mounting options) of all removable media partitions. Look for entries that have mount points that contain words such as floppy or cdrom. See the fstab manual page for more information.

Audit:

Check mount options for removable media partitions # **mount**

This should be relevant for Linux systems for which security is paramount

Services – Mask or remove all nonessential services

Services listening on the system pose a potential risk as an attack vector. These services should be reviewed, and if not required, the service should be stopped, and the package containing the service should be removed. If required packages have a dependency, the service should be stopped and masked to reduce the attack surface of the system.

Remediation:

Run the following command to remove the package containing the service: **# apt purge <package_name>**

Or

Run the following command to stop and mask the service: **# systemctl --now mask <service_name>**

Audit:

Check all listening ports: **# lsof -i -P -n | grep -v "(ESTABLISHED)“**

This should be relevant for Linux systems configured to run in both server and workstation mode

Network configuration – Disable IPv6

If IPv6 or dual stack is not to be used, it is recommended that IPv6 be disabled to reduce the attack surface of the system

Remediation:

Edit /etc/default/grub and add ipv6.disable=1 to the GRUB_CMDLINE_LINUX parameters: **GRUB_CMDLINE_LINUX="ipv6.disable=1"**

Run the following command to update the grub2 configuration: **# update-grub**

Audit:

Check if IPv6 is already disabled: **# grep "^\s*linux" /boot/grub/grub.cfg | grep -v "ipv6.disable=1“**

This should be relevant for Linux systems for which security is paramount

Network configuration – Disable wireless

If wireless is not to be used, wireless devices can be disabled to reduce the potential attack surface

Remediation script:

Disable any wireless interfaces: # **ip link set <<interface>> down**

or

completely disable wireless by blocking modules from loading in the kernel (check the script)

Audit:

Check if what wireless are available: # **iwconfig**

Check what wireless interfaces are active: # **ip link show up**

```
#!/bin/bash
if command -v nmcli >/dev/null 2>&1 ; then
 nmcli radio all off
else
 if [ -n "$(find /sys/class/net/*/ -type d -name wireless)" ]; then
 drivers=$(for driverdir in $(find /sys/class/net/*/ -type d -name
wireless | xargs -0 dirname); do basename "$(readlink -f
"$driverdir"/device/driver)";done | sort -u)
 for dm in $drivers; do
 echo "install $dm /bin/true" >> /etc/modprobe.d/disable_wireless.conf
 done
 fi
fi
```


This should be relevant for Linux systems configured to run in server mode.

Network configuration – Disable packet redirect

ICMP Redirects are used to send routing information to other hosts. As a host itself does not act as a router (in a host only configuration), there is no need to send redirects.

An attacker could use a compromised host to send invalid ICMP redirects to other router devices in an attempt to corrupt routing and have users access a system set up by the attacker as opposed to a valid system.

Remediation script:

Set the following parameters in /etc/sysctl.conf or a /etc/sysctl.d/* file:

net.ipv4.conf.all.send_redirects = 0

net.ipv4.conf.default.send_redirects = 0

Run the following commands to set the active kernel parameters:

 # sysctl -w net.ipv4.conf.all.send_redirects=0

 # sysctl -w net.ipv4.conf.default.send_redirects=0

 # sysctl -w net.ipv4.route.flush=1

Audit:

 # sysctl net.ipv4.conf.all.send_redirects

 # sysctl net.ipv4.conf.default.send_redirects

 # grep "net\.ipv4\.conf\.all\.send_redirects" /etc/sysctl.conf /etc/sysctl.d/*

 # grep "net\.ipv4\.conf\.default\.send_redirects" /etc/sysctl.conf /etc/sysctl.d/*

This should be relevant for Linux systems configured to run in server mode

Network configuration – Configure loopback traffic

Loopback traffic is generated between processes on machine and is typically critical to operation of the system. The loopback interface is the only place that loopback network (127.0.0.0/8 for IPv4 and ::1/128 for IPv6) traffic should be seen, all other interfaces should ignore traffic on this network as an anti-spoofing measure.

Remediation script:

Check the Uncomplicated Firewall status: # **ufw status verbose**

Audit:

 # ufw allow in on lo

 # ufw allow out from lo

 # ufw deny in from 127.0.0.0/8

 # ufw deny in from ::1

This should be relevant for Linux systems configured to run in both server and workstation mode

Network configuration – Disable uncommon net protocols

The Linux kernel modules support several network protocols that are not commonly used. If these protocols are not needed, it is recommended that they be disabled in the kernel.

DCCP, SCTP, RDS, TIPC.

Remediation script:

Block the loading of the modules in the kernel: **# install <<name of the ptotocal>> /bin/true**

Audit:

Check if the modules are installed: **# modprobe -n -v tipc | grep -E '(tipc|install)'**

This should be relevant for Linux systems configured to run in both server and workstation mode

Network configuration – Configure outbound connections

If rules are not in place for new outbound connections all packets will be dropped by the default policy preventing network usage.

Remediation script:
Enable outbound connections: **# ufw allow out on all**
Audit:
Check Firewall status:**# ufw status numbered**

This should be relevant for Linux systems configured to run in both server and workstation mode

Network configuration – Configure FW for open ports

With a default accept policy the firewall will accept any packet that is not configured to be denied. It is easier to white list acceptable usage than to black list unacceptable usage.

The following rules should be considered before applying the default deny:

ufw allow git

ufw allow in http

ufw allow in https

ufw allow out 53

ufw logging on

Remediation:

Deny all traffic not covert by FW policies:

 # ufw default deny incoming

 # ufw default deny outgoing

 # ufw default deny routed

Audit:

Check FW rules: **# ufw status verbose**

This should be relevant for Linux systems configured to run in both server and workstation mode

Logging and Auditing – Configure System accounting

auditd is the userspace component to the Linux Auditing System. It's responsible for writing audit records to the disk.

Remediation:

Install auditd package: **# apt install auditd audispd-plugins**

Enable auditd: **# systemctl --now enable auditd**

To enable auditing of services which started prior to auditd, edit /etc/default/grub and add audit=1 to GRUB_CMDLINE_LINUX:

GRUB_CMDLINE_LINUX="audit=1“

If audit=1 and more than the 64 records are generated during boot, the backlog size needs to be increased by editing /etc/default/grub and adding audit_backlog_limit= to GRUB_CMDLINE_LINUX: with an appropriate value (>=8192 is recommended). E.g.
GRUB_CMDLINE_LINUX="audit_backlog_limit=8192"

Update the grub configuration: **# update-grub**

Audit:

Check FW rules: **# dpkg -s auditd audispd-plugins**

Check if auditd is enabled: **# systemctl is-enabled auditd**

Check the value of audit= and audit_backlog_limit= in the grub config file**:# cat /boot/grub/grub.cfg**

This should be relevant for Linux systems configured to run in server mode

Logging and Auditing – Configure data retention

When auditing, it is important to carefully configure the storage requirements for audit logs. By default, auditd will max out the log files at 5MB and retain only 4 copies of them. Older versions will be deleted. It is possible on a system that the 20 MBs of audit logs may fill up the system causing loss of audit data.

Remediation:

Set the following parameter in /etc/audit/auditd.conf in accordance needs or company policy: **max_log_file = <MB>**

Set the following parameter in /etc/audit/auditd.conf: **max_log_file_action = keep_logs**

Make the audit configuration immutable by addind the following line in /etc/audit/rules.d/99-finalize.rules: **-2 e**

Audit:

Check log file size: **# grep max_log_file /etc/audit/auditd.conf**

Check the log retention settings: **# grep max_log_file_action /etc/audit/auditd.conf**

Check if the audit conf is immutable: **# grep "^\s*[^#]" /etc/audit/rules.d/*.rules | tail -1**

This should be relevant for Linux systems configured to run in server mode

Logging and Auditing – Configure rsyslog

The rsyslog software is recommended as a replacement for the syslogd daemon and provides improvements, such as connection-oriented (i.e. TCP) transmission of logs, the option to log to database formats, and the encryption of log data en route to a central logging server.

Remediation:

Install the app: **# apt install rsyslog**

Enable rsyslog: **# systemctl --now enable rsyslog**

Update the configuration files /etc/rsyslog.conf and /etc/rsyslog.d/*.conf

Edit the /etc/rsyslog.conf and /etc/rsyslog.d/*.conf files and set every instance of $FileCreateMode to 0640 or more restrictive

Audit:

Check if the file is installed: **# dpkg -s rsyslog**

Check if rsyslog is enabled: **# systemctl --now enable rsyslog**

Review a log file to check if events are captured: **# ls -l /var/log/**

Check the log file permissions: **# grep ^\s*\$FileCreateMode /etc/rsyslog.conf /etc/rsyslog.d/*.conf**

This should be relevant for Linux systems configured to run in server mode

Logging and Auditing – Configure journald

systemd-journald is a system service that collects and stores logging data. It creates and maintains structured, indexed journals based on logging information that is received from a variety of sources: Kernel log messages, via kmsg.

Any changes made to the systemd-journald configuration will require a re-start of systemdjournald

Remediation:

Edit the /etc/systemd/journald.conf file and add the following line:

ForwardToSyslog=yes

Compress=yes

Storage=persistent

Audit:

Review the configuration file: **# grep -e Compress /etc/systemd/journald.conf**

This should be relevant for Linux systems configured to run in server mode

Logging and Auditing – Check permissions on all log files

Log files stored in /var/log/ contain logged information from many services on the system, or on log hosts others as well.
It is important to ensure that log files have the correct permissions to ensure that sensitive data is archived and protected.

Remediation:
Sanitize permissions: # **find /var/log -type f -exec chmod g-wx,o-rwx "{}" + -o -type d -exec chmod gw,o-rwx "{}" +**
Audit:
List permissions on all log files: **# find /var/log -type f –ls**

This should be relevant for Linux systems configured to run in server mode

Logging and Auditing – Configured logrotate

The system includes the capability of rotating log files regularly to avoid filling up the system with logs or making the logs unmanageably large. The file /etc/logrotate.d/syslog is the configuration file used to rotate log files created by syslog or rsyslog.

By keeping the log files smaller and more manageable, a system administrator can easily archive these files to another system and spend less time looking through inordinately large log files

If no maxage setting is set for logrotate, it can fail to delete rotated logfiles. It is recommended to set this to a value greater than the longest any log file should exist on your system to ensure that any such logfile is removed but standard rotation settings are not overridden.

Remediation:

Edit /etc/logrotate.conf and /etc/logrotate.d/* to ensure logs are rotated according to needs or company policy.

Audit:

Review /etc/logrotate.conf and /etc/logrotate.d/* and verify logs are rotated according to site policy.

This should be relevant for Linux systems configured to run in server mode

Access, Authentication and Authorization – Configure PAM

PAM (Pluggable Authentication Modules) is a service that implements modular authentication modules on UNIX systems. PAM is implemented as a set of shared objects that are loaded and executed when a program needs to authenticate a user. Files for PAM are typically located in the /etc/pam.d directory. PAM must be carefully configured to secure system authentication.

The pam_pwquality.so module checks the strength of passwords. It performs checks such as making sure a password is not a dictionary word, it is a certain length, contains a mix of characters (e.g. alphabet, numeric, other) and more. The following are definitions of the pam_pwquality.so options.

Remediation:

Install the password quality module: **# apt install libpam-pwquality**

The following options are set in the /etc/security/pwquality.conf file:

minlen = x - password must be x characters or more

minclass = 4 - The minimum number of required classes of characters for

the new password (digits, uppercase, lowercase, others)

OR

dcredit = - 1 - provide at least one digit

ucredit = -1 - provide at least one uppercase character

ocredit = -1 - provide at least one special character

lcredit = -1 - provide at least one lowercase character

Audit:

Check the permission of the files: **/etc/security/pwquality.conf**

This should be relevant for Linux systems configured to run in server mode

AAA – Configure lockout for failed password attempts

Locking out user IDs after n unsuccessful consecutive login attempts mitigates brute force password attacks against your systems.

Remediation:

Edit the /etc/pam.d/common-auth file and add the auth line below: **auth required pam_tally2.so onerr=fail audit silent deny=5 unlock_time=900**

Edit the /etc/pam.d/common-account file and add the account lines bellow:

account requisite pam_deny.so

account required pam_tally2.so

Audit:

Check the current configuration: **# grep "pam_tally2" /etc/pam.d/common-auth**

Check the current configuration: **# grep -E "pam_(tally2|deny)\.so" /etc/pam.d/common-account**

This should be relevant for Linux systems configured to run in server mode

AAA – Configure password reuse

The /etc/security/opasswd file stores the users' old passwords and can be checked to ensure that users are not recycling recent passwords.

Forcing users not to reuse their past 5 passwords make it less likely that an attacker will be able to guess the password.

Remediation:

Edit the /etc/pam.d/common-password file to include the remember option and conform to policy:

password required pam_pwhistory.so remember=X

Audit:

Check the current configuration: **# grep pam_pwhistory /etc/pam.d/commonpassword**

This should be relevant for Linux systems configured to run in server mode

AAA – Configure the password hashing algorithm

The commands below change password encryption from md5 to sha512.

The SHA-512 algorithm provides much stronger hashing than MD5, thus providing additional protection to the system by increasing the level of effort for an attacker to successfully determine passwords.

All existing accounts will need to perform a password change to upgrade the stored hashes to the new algorithm.

Remediation:

Edit the /etc/pam.d/common-password file to include the sha512 option for pam_unix.so as shown:

password [success=1 default=ignore] pam_unix.so sha512

Audit:

Check the current configuration: **# grep –a password /etc/pam.d/common-password**

This should be relevant for Linux systems configured to run in both server and workstation mode

AAA – Configure Shadow Password Suite

While a majority of the password control parameters have been moved to PAM, some parameters are still available through the shadow password suite.

Configure /etc/login.defs with the password expiration age, minimum days between password changes, expiration warning period

Usually a value of -1 will disable these parameters.

Remediation:

Set parameters to /etc/login.defs:

PASS_MAX_DAYS 365

PASS_MIN_DAYS 1

PASS_WARN_AGE 7

Audit:

Check the current configuration: **# cat /etc/login.defs**

This should be relevant for Linux systems configured to run in server mode

AAA – Configure inactive password lock

While a majority of the password control parameters have been moved to PAM, some parameters are still available through the shadow password suite.

User accounts that have been inactive for over a given period of time can be automatically disabled. It is recommended that accounts that are inactive for 30 days after password expiration be disabled.

Remediation:

Set the default password inactivity period: **# useradd -D -f 30**

Audit:

Check the current configuration: **# useradd -D | grep INACTIVE**

This should be relevant for Linux systems configured to run in server mode

AAA – Restrict access to the su command

The su command allows a user to run a command or shell as another user. The program has been superseded by sudo , which allows for more granular control over privileged access. Normally, the su command can be executed by any user. By uncommenting the pam_wheel.so statement in /etc/pam.d/su , the su command will only allow users in a specific groups to execute su. This group should be empty to reinforce the use of sudo for privileged access.

Remediation:

Create an empty group that will be specified for use of the su command (Example): **# groupadd sugroup**

Add the following line to the /etc/pam.d/su file, specifying the empty group: **auth required pam_wheel.so use_uid group=sugroup**

Audit:

Check the current configuration**: # grep pam_wheel.so /etc/pam.d/su**

List the users in the grup allowed to run su: **# grep /etc/group**

This should be relevant for Linux systems configured to run in server mode

AAA – Securing system accounts

There are several accounts provided with most distributions that are used to manage applications and are not intended to provide an interactive shell.

It is important to make sure that accounts that are not being used by regular users are prevented from being used to provide an interactive shell. By default, most distributions set the password field for these accounts to an invalid string, but it is also recommended that the shell field in the password file be set to the nologin shell. This prevents the account from potentially being used to run any commands.

Remediation:

Set nologin shell for specific accounts: **# usermod -s $(which nologin) <<user_name>>**

Audit:

Run these commands (no results should be returned):

awk -F: '($1!="root" && $1!="sync" && $1!="shutdown" && $1!="halt" &&
$1!~/^\+/ && $3<'"$(awk '/^\s*UID_MIN/{print $2}' /etc/login.defs)"' &&
$7!="'"$(which nologin)"'" && $7!="/bin/false") {print}' /etc/passwd

and

awk -F: '($1!="root" && $1!~/^\+/ && $3<'"$(awk '/^\s*UID_MIN/{print $2}'
/etc/login.defs)"') {print $1}' /etc/passwd | xargs -I '{}' passwd -S '{}' |
awk '($2!="L" && $2!="LK") {print $1}'

This should be relevant for Linux systems configured to run in both server and workstation mode

AAA – Lock accounts with an empty password

An account with an empty password field means that anybody may log in as that user without providing a password.
All accounts must have passwords or be locked to prevent the account from being used by an unauthorized user.

Remediation:
Set a password for root
Or
Lock the accounts without a password: **# passwd -l <<user_name>>**
Audit:
Run these commands:
 # awk -F: '($2 == "") { print $1 " does not have a password "}' /etc/shadow

This should be relevant for Linux systems configured to run in server mode

Related attacks

- Buffer Overflow Attacks
- Pass-the-Hash
- Mitre ATT&CK