SECURITY IN COMPUTING, FIFTH EDITION

Chapter 5: Operating Systems

Chapter 5 Objectives

- Basic security functions provided by operating systems
- System resources that require operating system protection
- Operating system design principles
- How operating systems control access to resources
- The history of trusted computing
- Characteristics of operating system rootkits

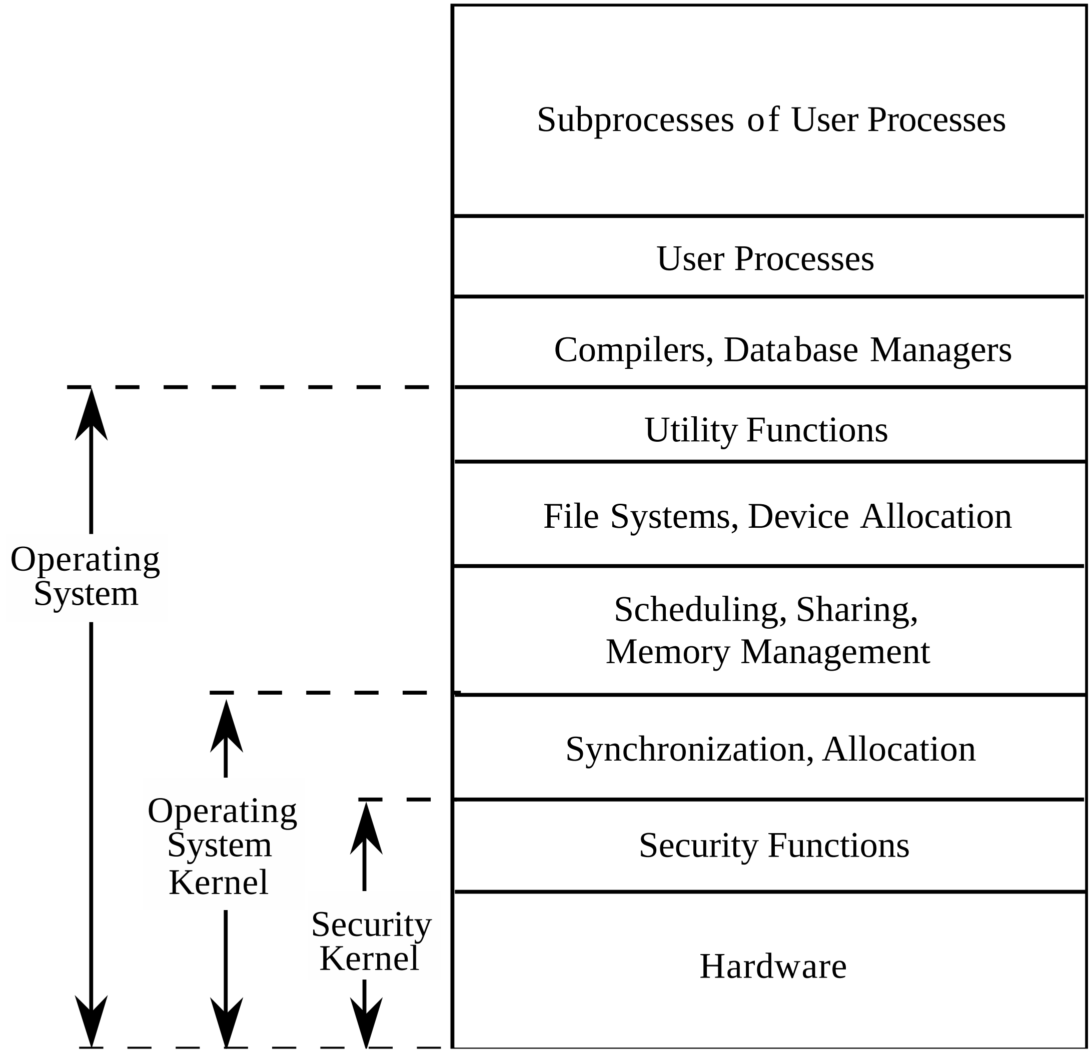
History of Operating Systems

- Single-user systems, no OS
- Multiprogrammed OS, aka monitors
  - Multiple users
  - Multiple programs
  - Scheduling, sharing, concurrent use
- Personal computers

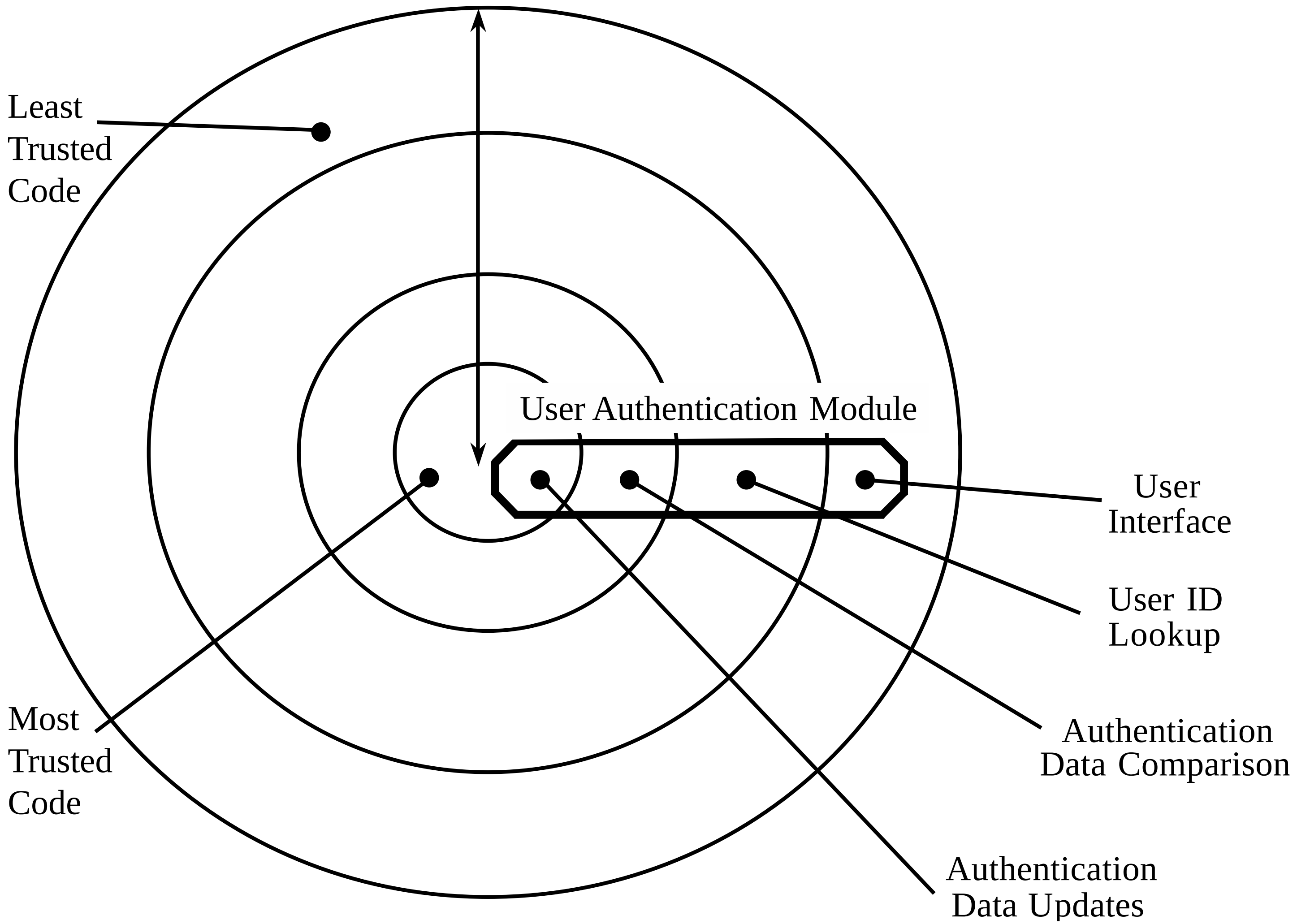
Protected Objects

- Memory
- Sharable I/O devices, such as disks
- Serially reusable I/O devices, such as printers
- Sharable programs and subprocedures
- Networks
- Sharable data

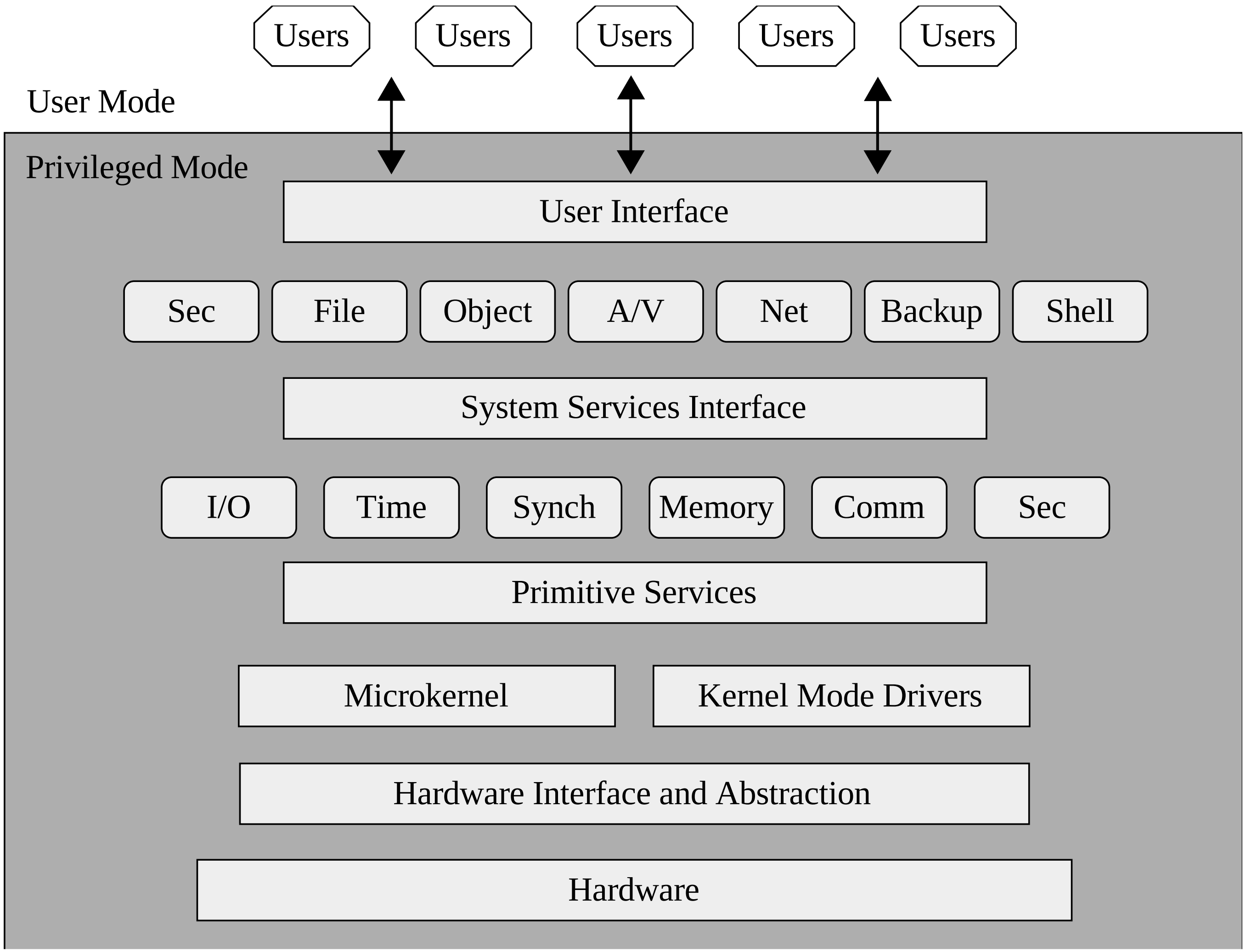
OS Layered Design



Functions Spanning Layers



Modular OS Design



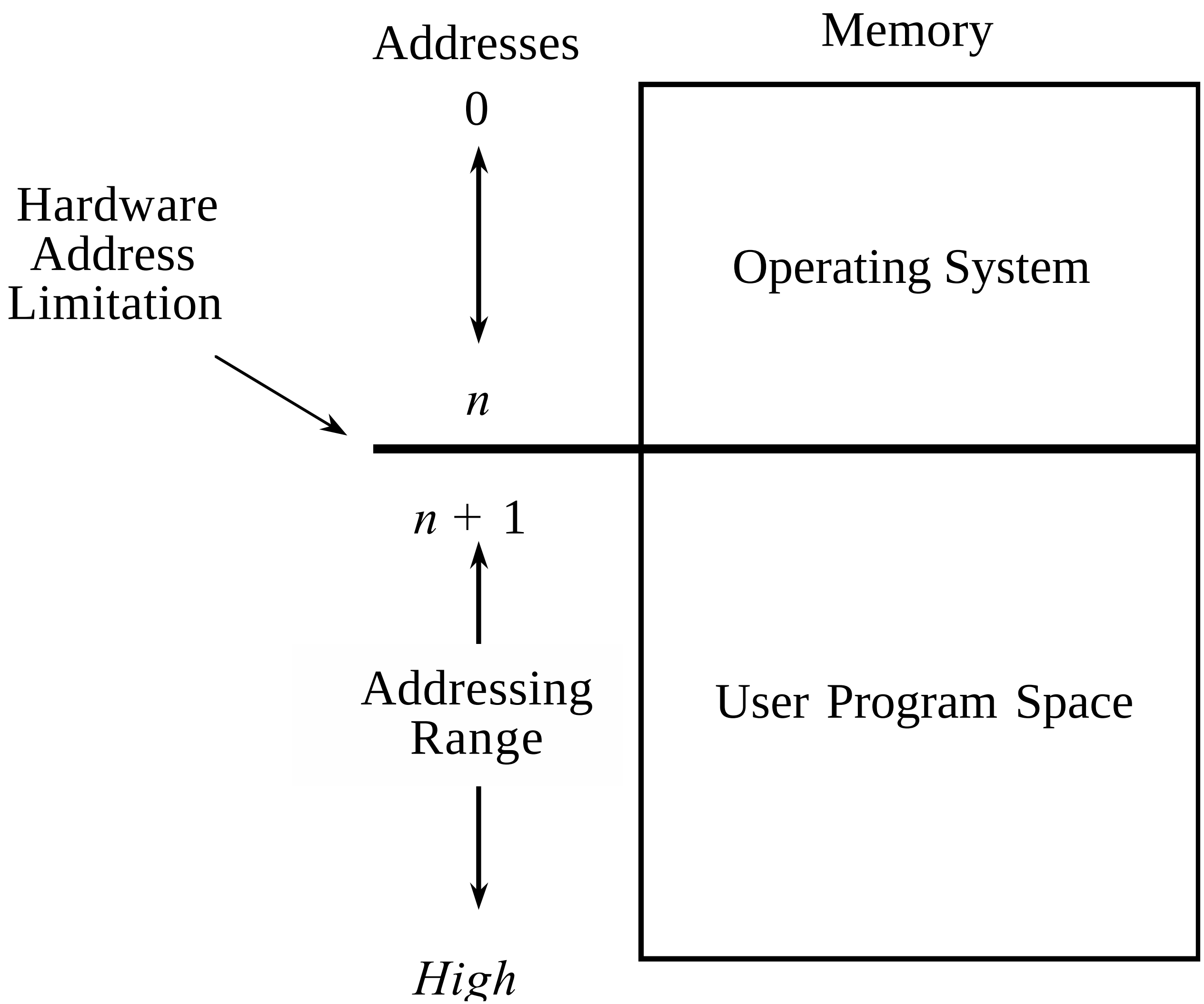
Virtualization

- With virtualization, the OS presents each user with just the resources that user should see
- The user has access to a virtual machine (VM), which contains those resources
- The user cannot access resources that are available to the OS but exist outside the VM
- A hypervisor, or VM monitor, is the software that implements a VM
  - Translates access requests between the VM and the OS
  - Can support multiple OSs in VMs simultaneously
- Honeypot: A VM meant to lure an attacker into an environment that can be both controlled and monitored

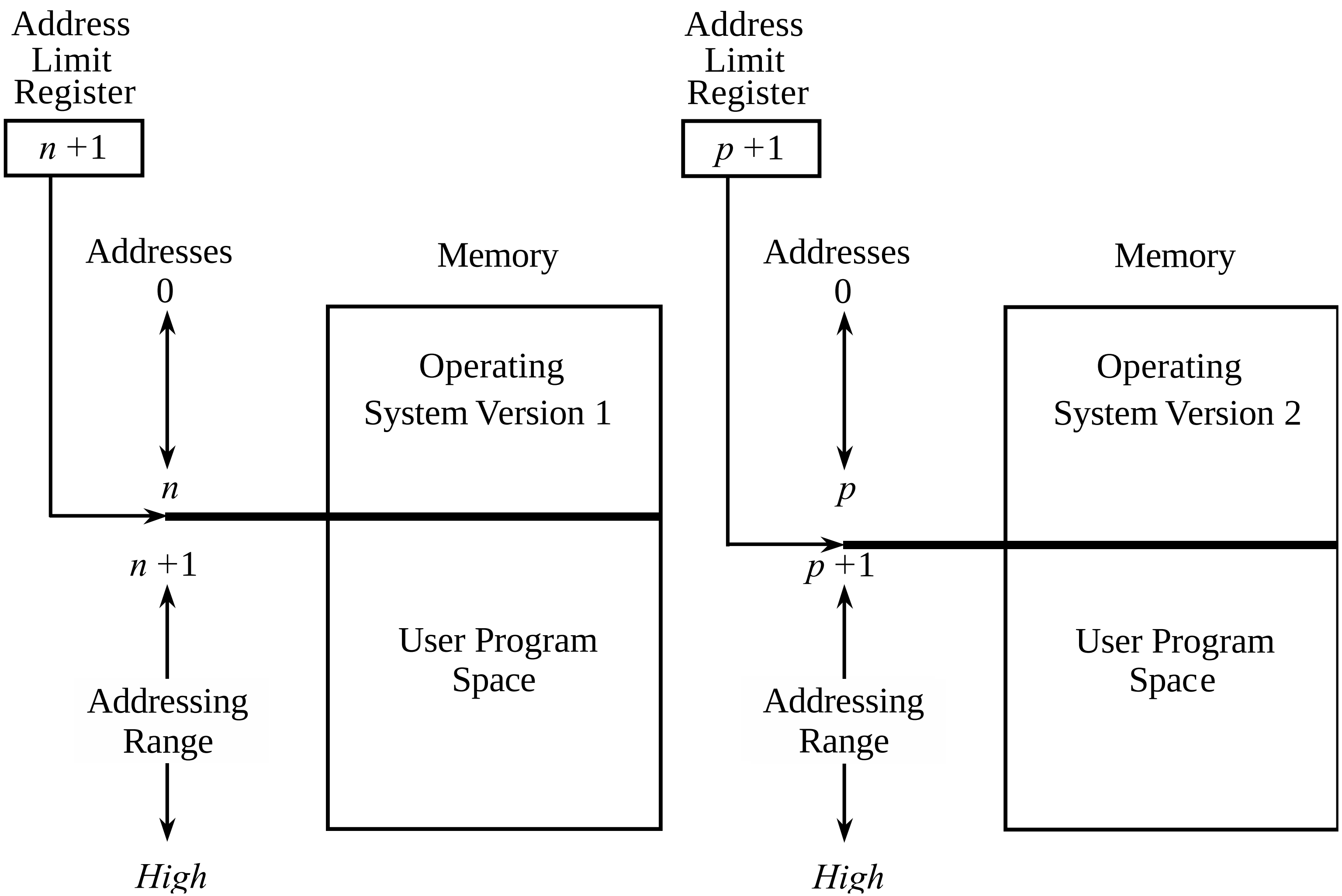
Separation and Sharing

- Methods of separation:
  - Physical
  - Temporal
  - Logical
  - Cryptographic
- Methods of supporting separation/sharing:
  - Do not protect
  - Isolate
  - Share all or share nothing
  - Share but limit access
  - Limit use of an object

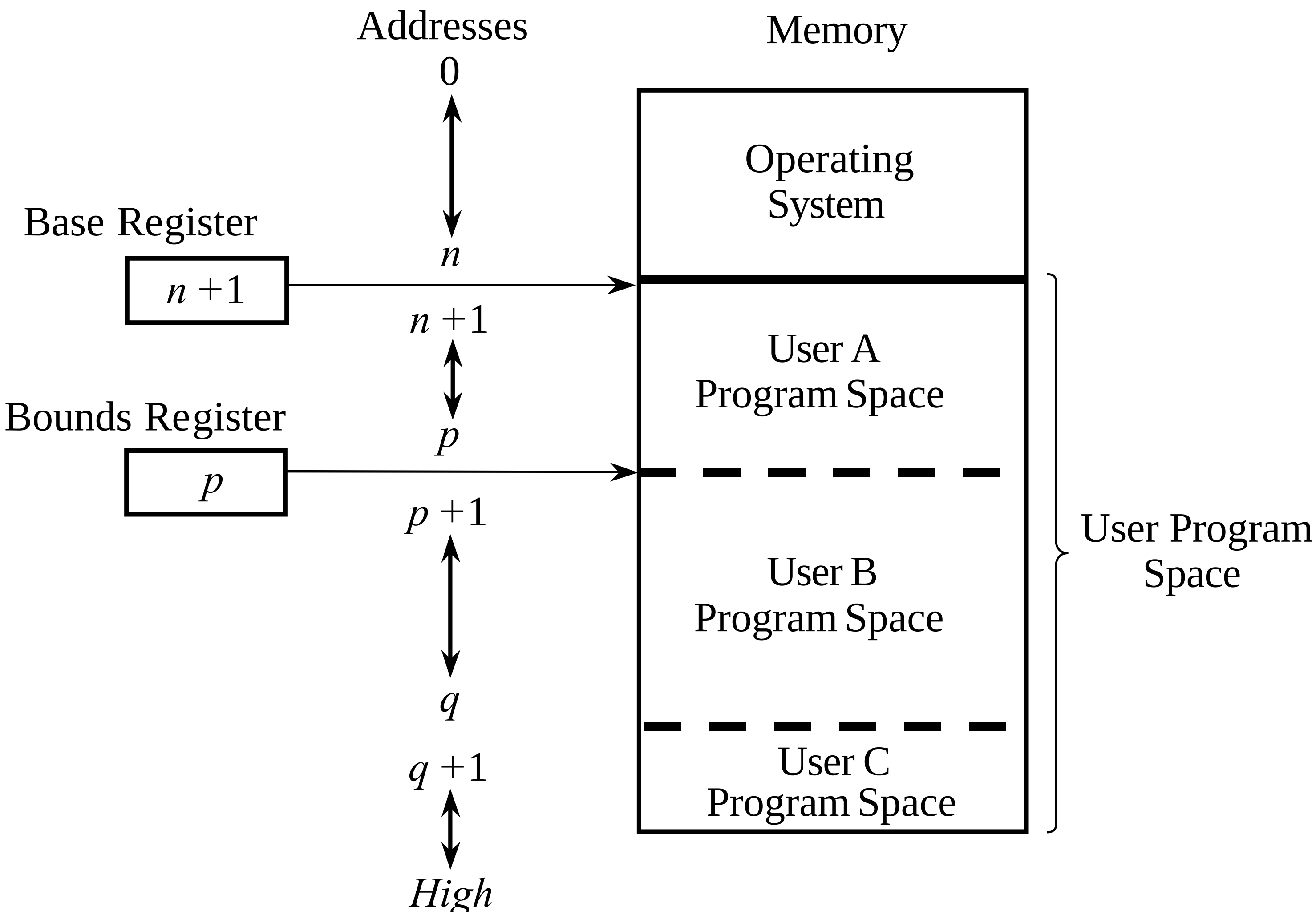
Hardware Protection of Memory



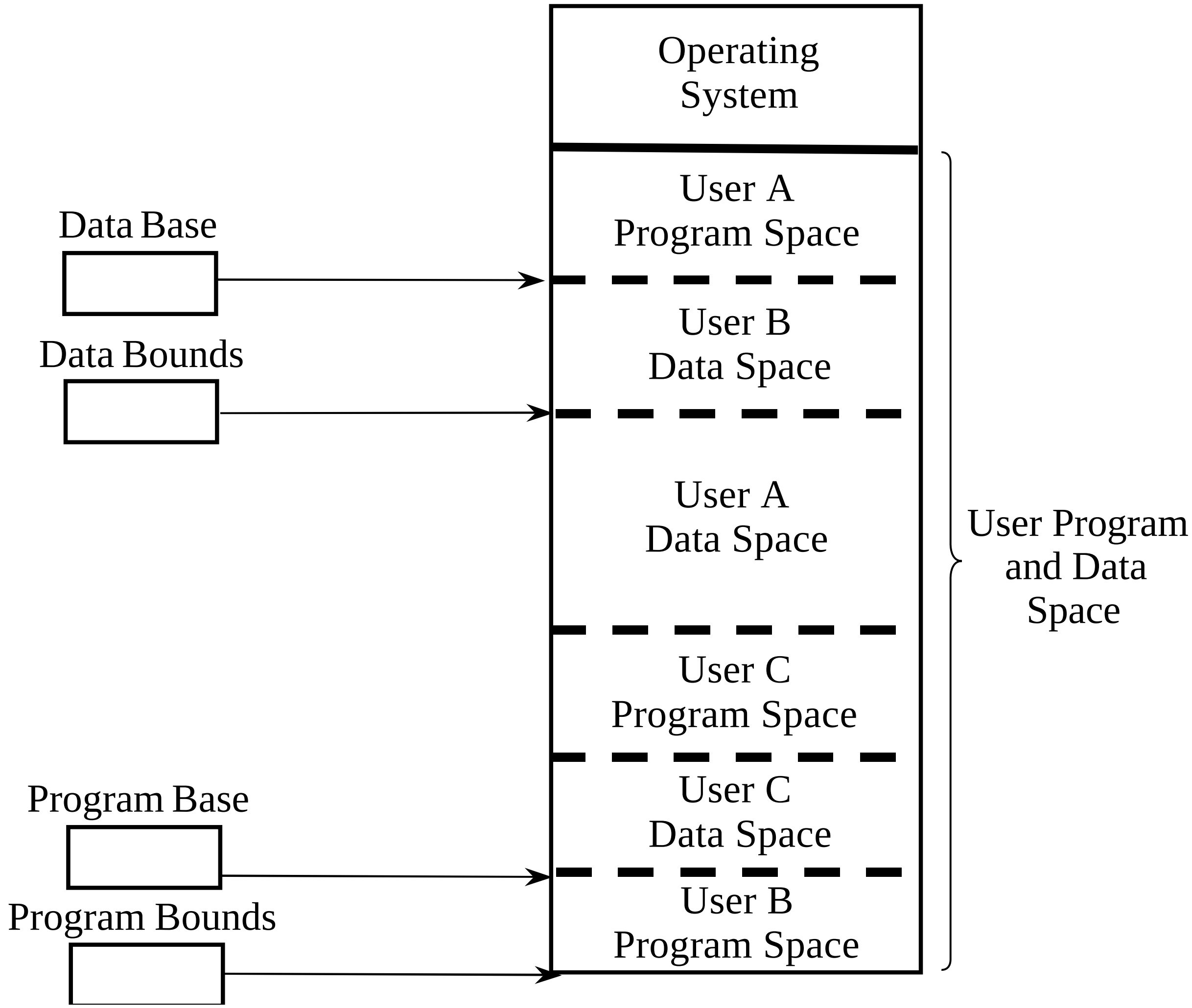
Fence Registers



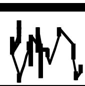
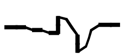
Base/Bounds Registers



Two Pairs of Base/Bounds Registers

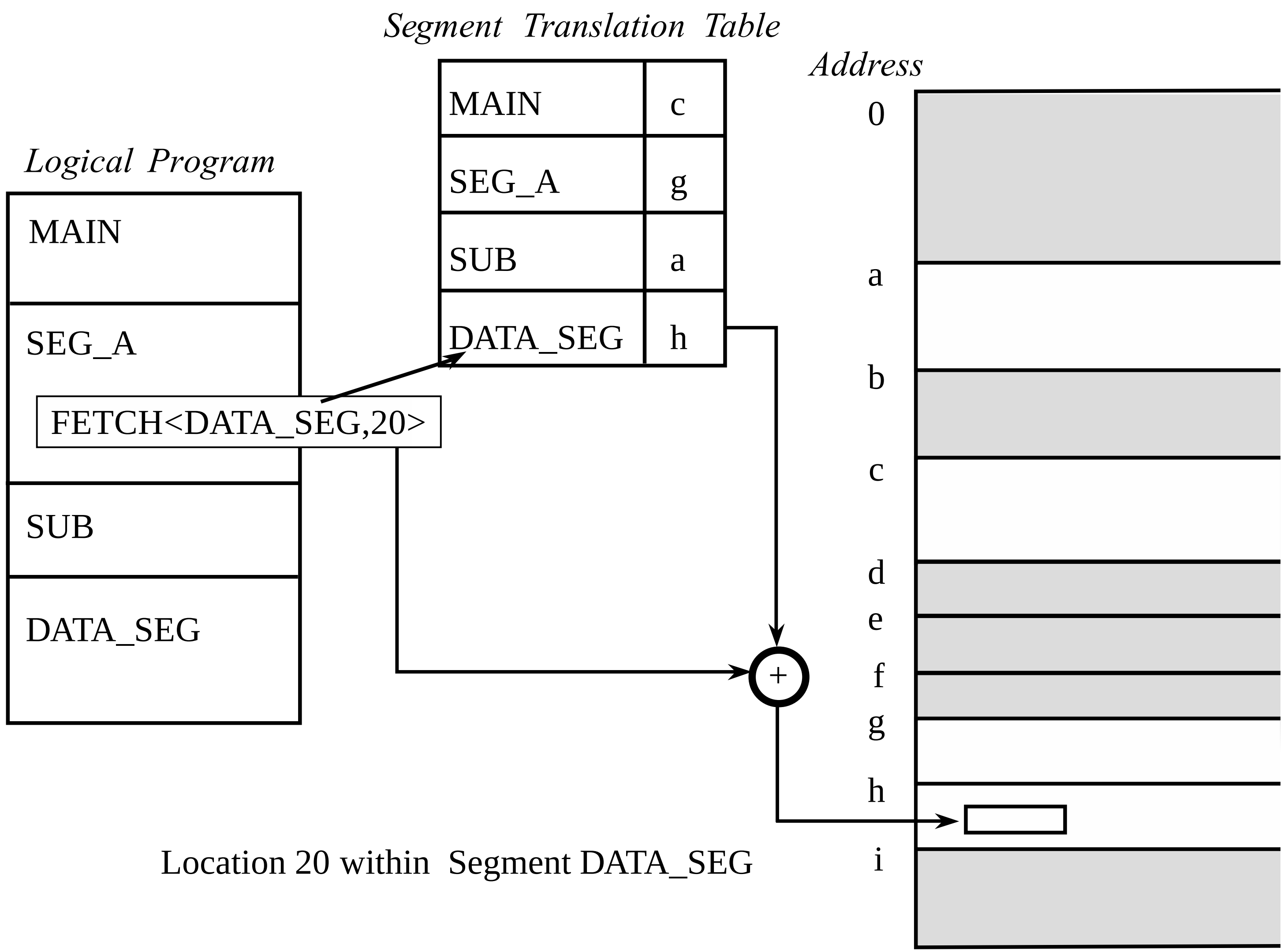


Tagged Architecture

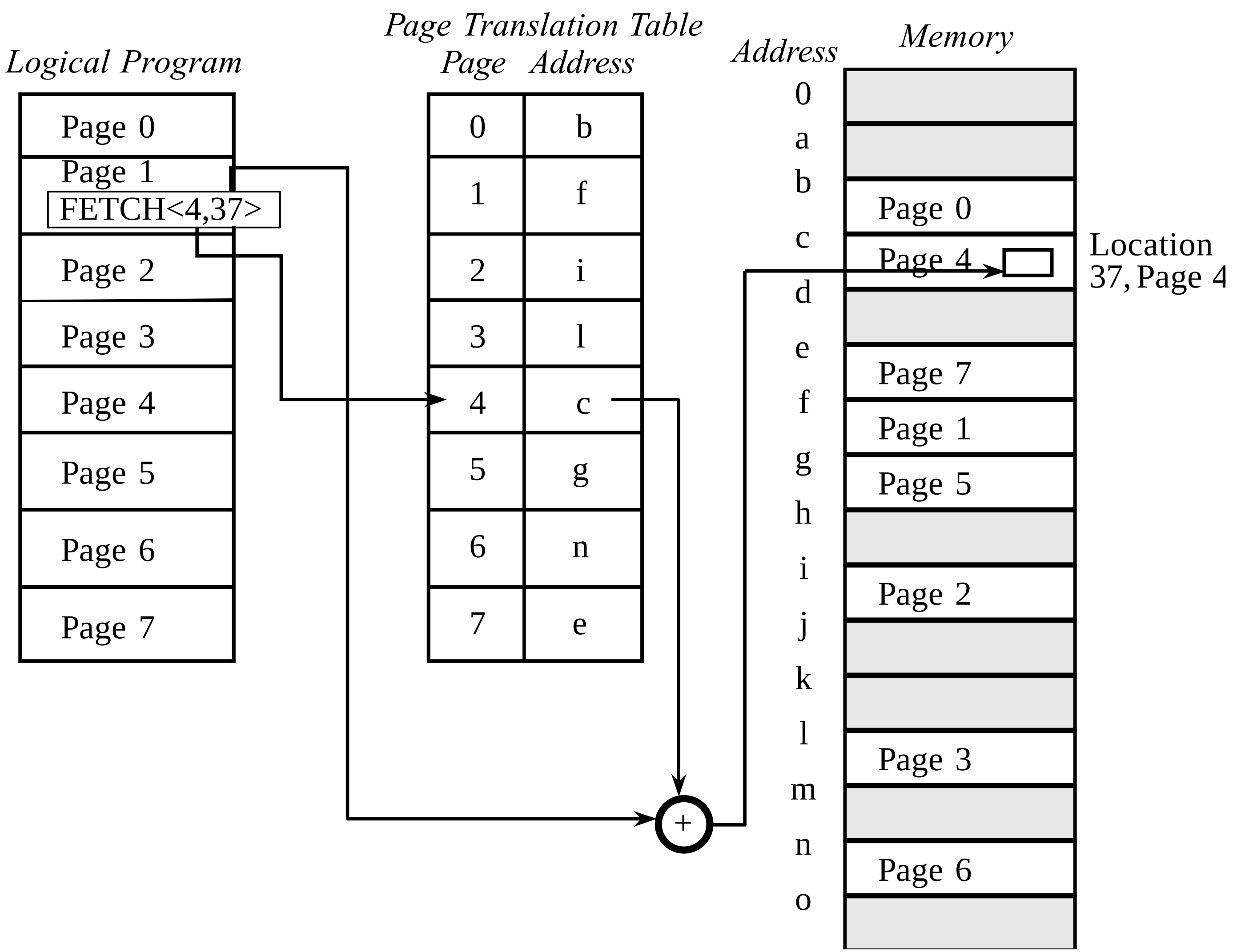
Tag	Memory Word
R	0001
RW	0137
R	0099
X	
X	
X	
X	
X	
X	
R	4091
RW	0002

Code: R = Read-only RW = Read/Write
X = Execute-only

Segment Address Translation

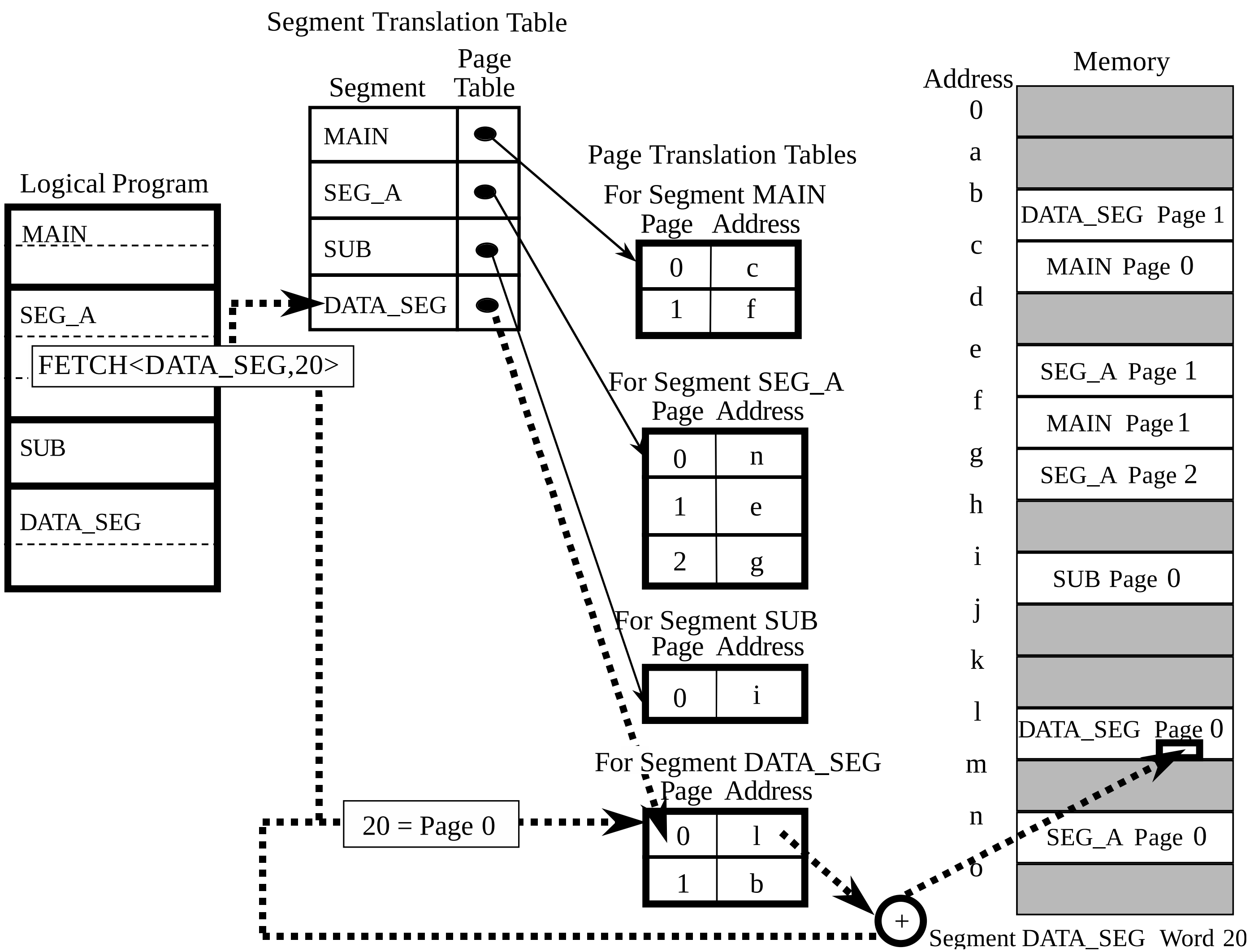


Paging



Logical Program
Page 0
Page 1
FETCH<4,37>
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Page Translation Table
Page Address
0 b
1 f
2 i
3 l
4 c
5 g
6 n
7 e
+
Address
0
a
b
c
d
e
f
g
h
i
j
k
l
m
n
o
Memory
Page 0
Page 4
Page 7
Page 1
Page 5
Page 2
Page 3
Page 6
Location
37, Page 4

Paged Segmentation



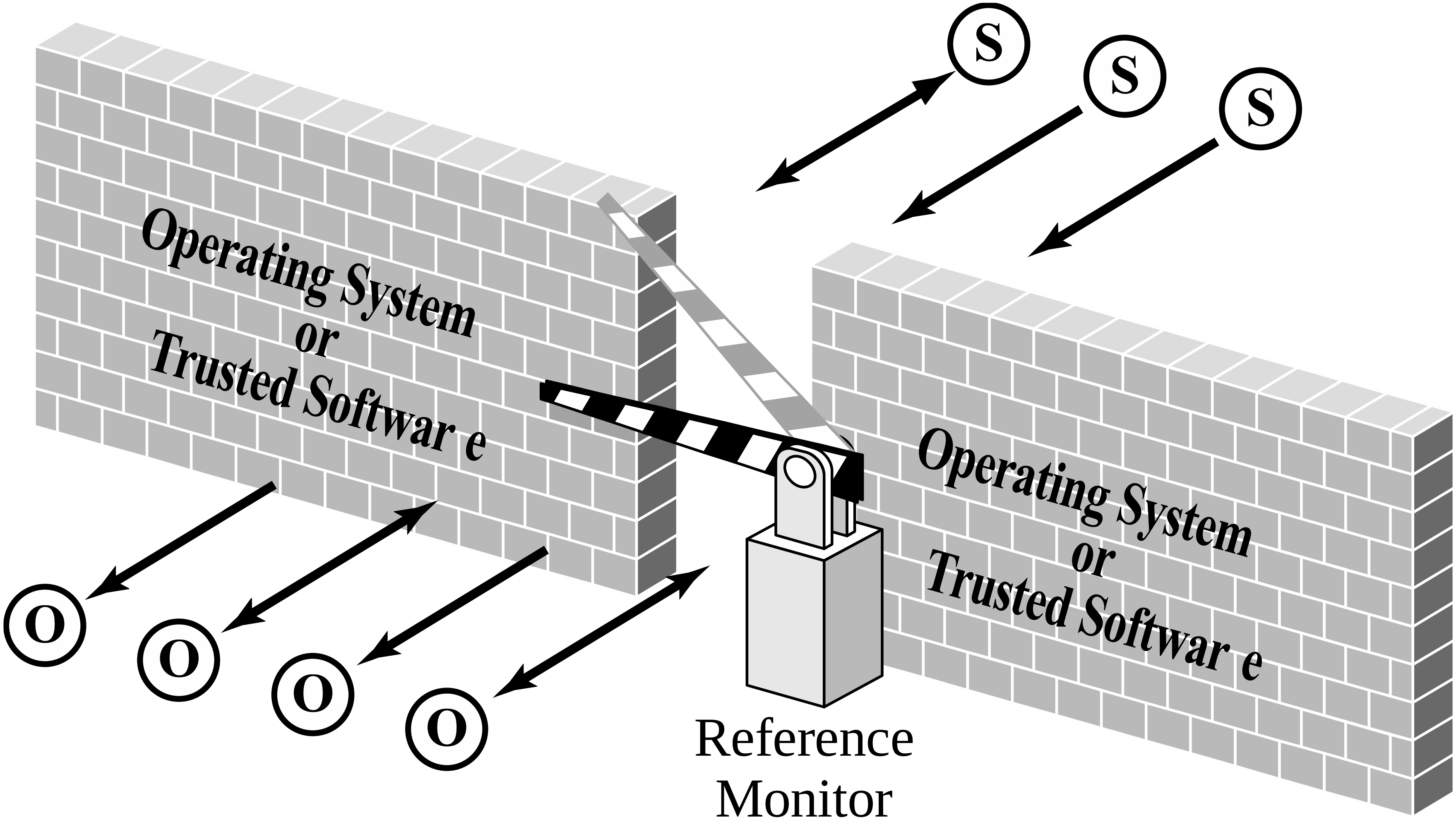
Principles of Secure OS Design

- Simplicity of design
  - OSs are inherently complex, and any unnecessary complexity only makes them harder to understand and secure
- Layered design
  - Enables layered trust
- Layered trust
  - Layering is both a way to keep a design logical and understandable and a way to limit risk
  - Example: very tight access controls on critical OS functions, fewer access controls on important noncritical functions, and few if any access controls on functions that aren't important to the OS

Kernelized Design

- A kernel is the part of the OS that performs the lowest-level functions
  - Synchronization
  - Interprocess communication
  - Message passing
  - Interrupt handling
- A security kernel is responsible for enforcing the security mechanisms of the entire OS
  - Typically contained within the kernel

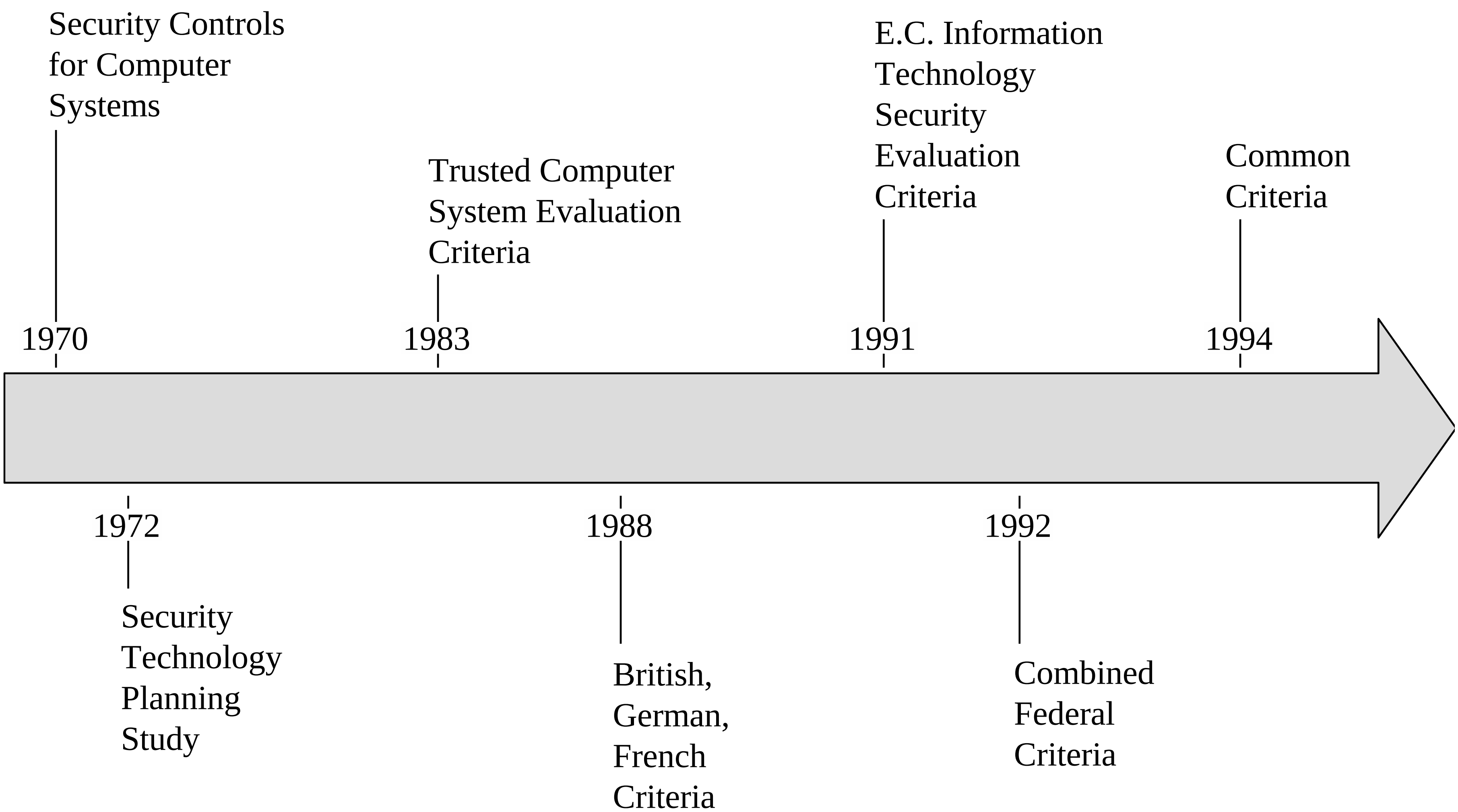
Reference Monitor



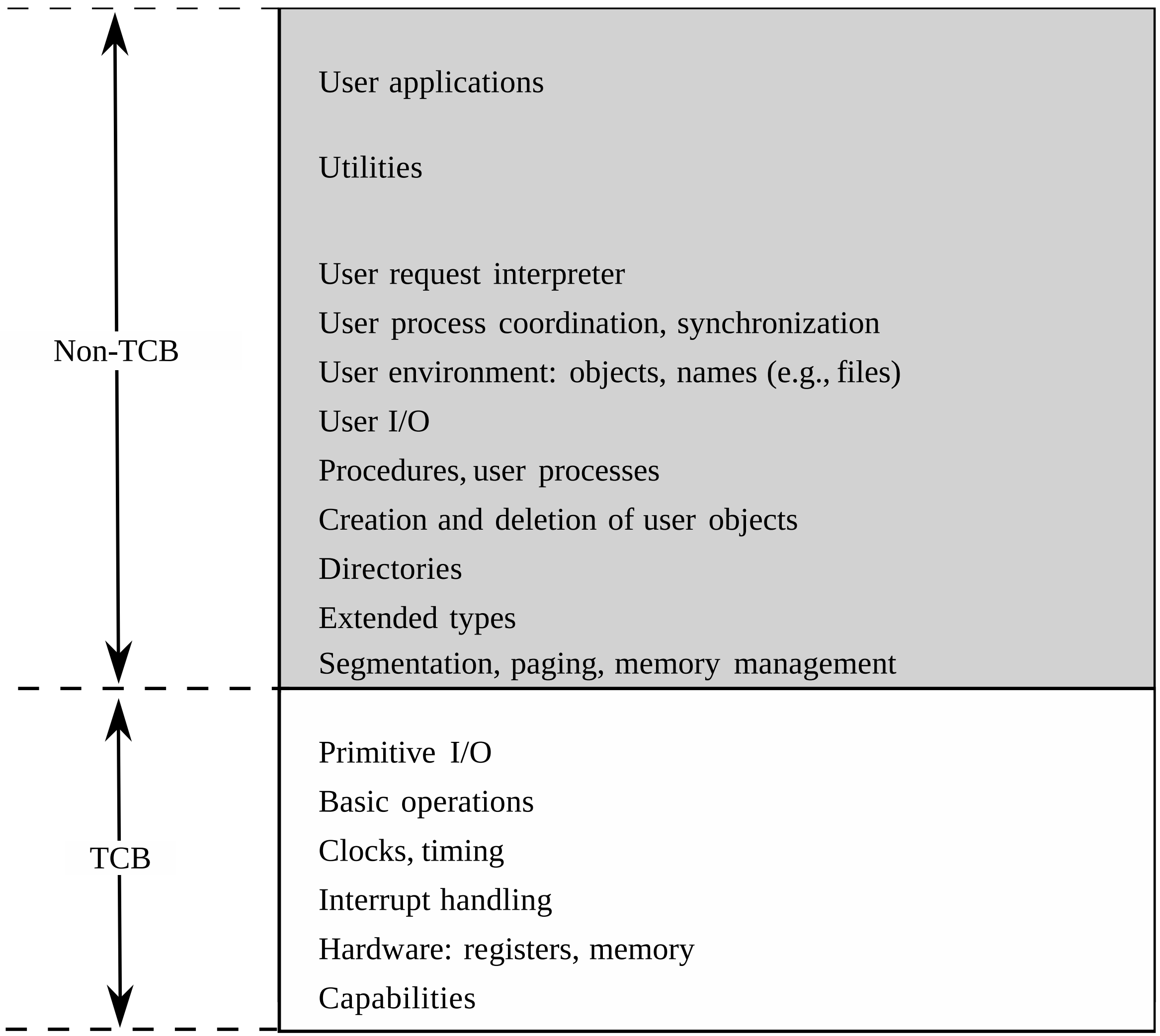
Trusted Systems

- A trusted system is one that has been shown to warrant some degree of trust that it will perform certain activities faithfully
- Characteristics of a trusted system:
  - A defined policy that details what security qualities it enforces
  - Appropriate measures and mechanisms by which it can enforce security adequately
  - Independent scrutiny or evaluation to ensure that the mechanisms have been selected and implemented properly

History of Trusted Systems



Trusted Computing Base (TCB)



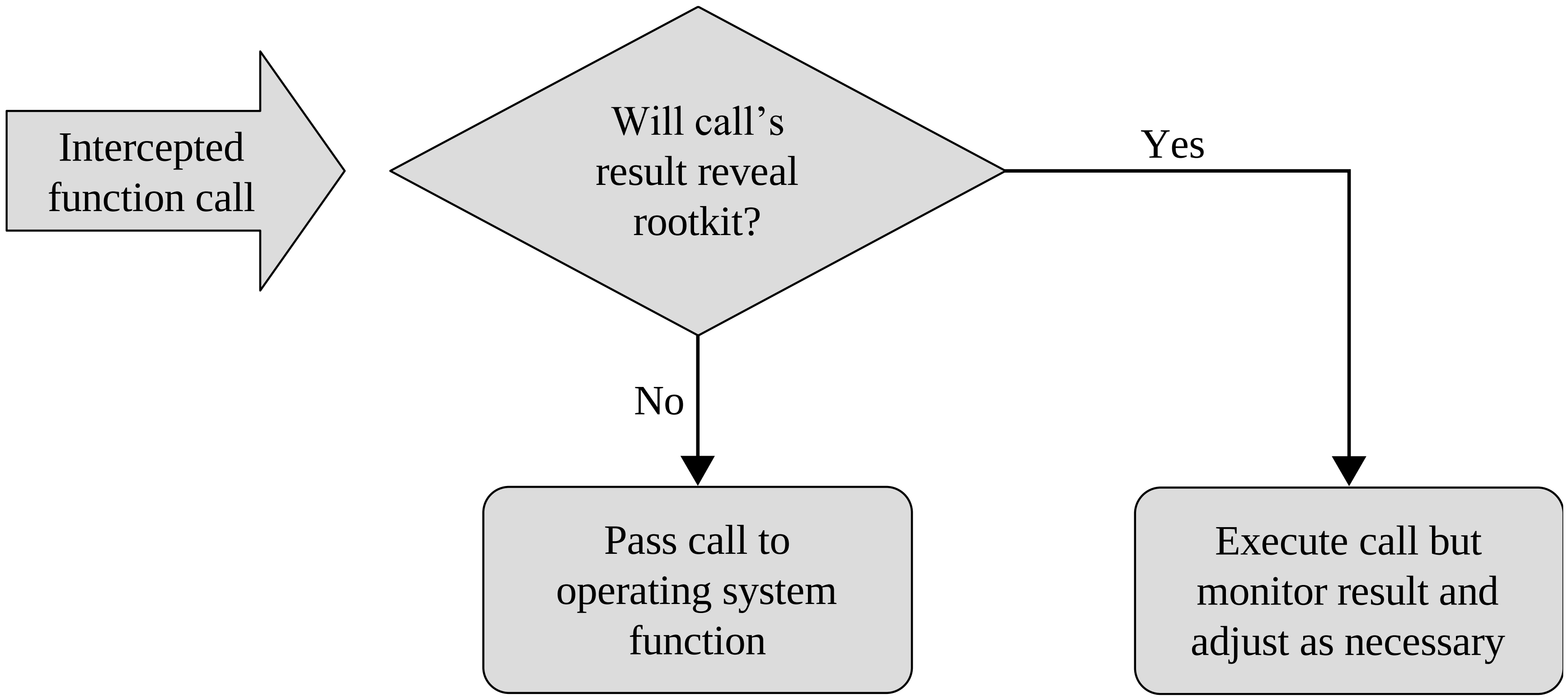
Other Trusted System Characteristics

- Secure startup
  - System startup is a tricky time for security, as most systems load basic I/O functionality before being able to load security functions
- Trusted path
  - An unforgeable connection by which the user can be confident of communicating directly with the OS
- Object reuse control
  - OS clears memory before reassigning it to ensure that leftover data doesn't become compromised
- Audit
  - Trusted systems track security-relevant changes, such as installation of new programs or OS modification
  - Audit logs must be protected against tampering and deletion

Rootkits

- A rootkit is a malicious software package that attains and takes advantage of root status or effectively becomes part of the OS
- Rootkits often go to great length to avoid being discovered or, if discovered and partially removed, to reestablish themselves
  - This can include intercepting or modifying basic OS functions

Rootkit Evading Detection



Summary

- OSs have evolved from supporting single users and single programs to many users and programs at once
- Resources that require OS protection: memory, I/O devices, programs, and networks
- OSs use layered and modular designs for simplification and to separate critical functions from noncritical ones
- Resource access control can be enforced in a number of ways, including virtualization, segmentation, hardware memory protection, and reference monitors
- Rootkits are malicious software packages that attain root status or effectively become part of the OS