2550 Intro to cybersecurity

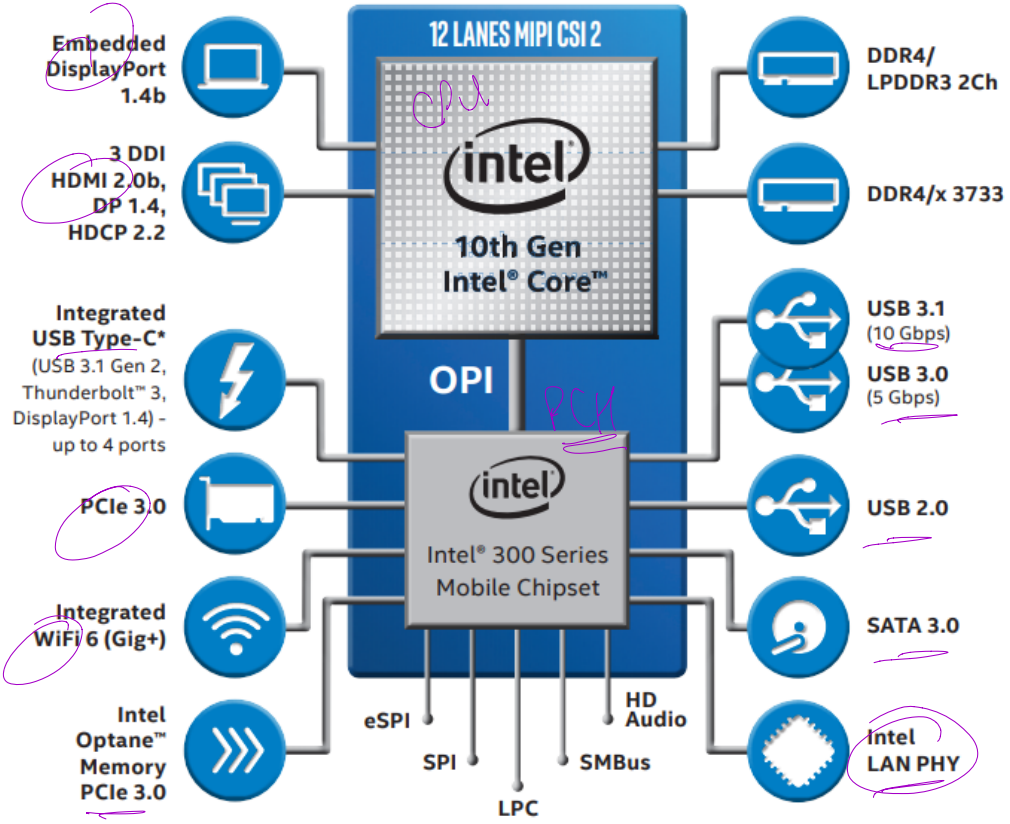
L18: Intro to Systems Security

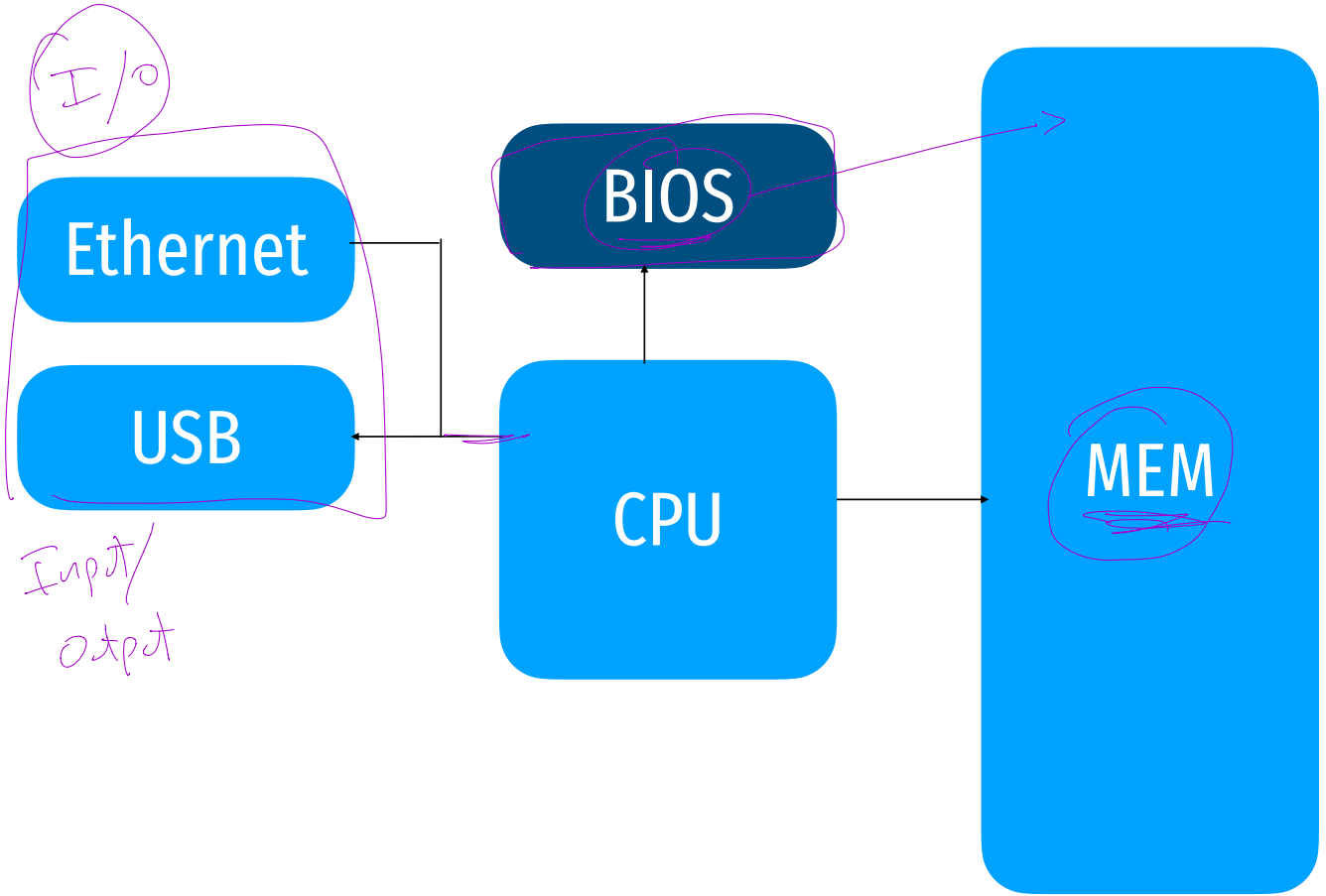
Ran Cohen/abhi shelat

Thanks to Christo for starting point for slides.

Threat Model
Principles
Intro to System Architecture
Hardware Support for Isolation
Examples


SI-System Information, Advanced Edition, (C) Copr 1987, 1988, Peter Norton
Computer Name: IBM-PC
Operating System: DOS 5.00
Built-in BIOS Dated: Wednesday, October 27, 1982
Main Processor: Intel 8088
Serial Ports: 1
Co-Processor: None
Parallel Ports: 2
Video Display Adapter: Monochrome (MDA)
Current Video Mode: Text: 80 x 25 Monochrome
Available Disk Drives: A: - C:
DOS reports 640 K-bytes of memory
128 K-bytes used by DOS and resident programs
512 K-bytes available for application programs
A search for active memory finds:
640 K-bytes main memory (at hex 0000-0280)
32 K-bytes display memory (at hex B000-B020)
ROM-BIOS Extensions are found at hex paragraphs: C000
Computing index (CI), relative to IBM/XT: 1.8
Disk index (DI), relative to IBM/XT: Not computed. No drive Specified
Performance index (PI), relative to IBM/XT: Not computed.
C:\>
IBM
IBM
IBM
IBM
Break


12 LANES MIPI CSI 2
CPU
intel
10th Gen
Intel® Core™
OPI
PCH
intel
Intel® 300 Series
Mobile Chipset
Embedded
DisplayPort
1.4b
3 DDI
HDMI 2.0b,
DP 1.4,
HDCP 2.2
Integrated
USB Type-C*
(USB 3.1 Gen 2,
Thunderbolt™ 3,
DisplayPort 1.4) -
up to 4 ports
PCIe 3.0
Integrated
WiFi 6 (Gig+)
Intel
Optane™
Memory
PCIe 3.0
DDR4/
LPDDR3 2Ch
DDR4/x 3733
USB 3.1
(10 Gbps)
USB 3.0
(5 Gbps)
USB 2.0
SATA 3.0
Intel
LAN PHY
eSPI
SPI
LPC
SMBus
HD
Audio


I/O
Ethernet
USB
Input/
output
BIOS
CPU
MEM

What is Memory?

Memory is essentially a spreadsheet with a single column

- Every row has a number, called an address
- Every cell holds 1 byte of data

Address	Contents
114	
113	0
112	0
111	0
110	8
109	
108	0
107	0
106	0
105	
104	
103	
102	
101	
100	

What is Memory?

Memory is essentially a spreadsheet with a single column

- Every row has a number, called an address
- Every cell holds 1 byte of data

Integers are typically four bytes

Address	Contents
114	
113	0
112	0
111	0
110	8
109	
108	
107	
106	
105	
104	
103	
102	
101	
100	

What is Memory?

Memory is essentially a spreadsheet with a single column

- Every row has a number, called an address
- Every cell holds 1 byte of data

Integers are typically four bytes

Each ASCII character is one byte, Strings are null terminated

Address	Contents
114	
113	0
112	0
111	0
110	8
109	
108	0
107	C
106	B
105	A
104	
103	
102	
101	
100	

What is Memory?

Memory is essentially a spreadsheet with a single column

- Every row has a number, called an address
- Every cell holds 1 byte of data

Integers are typically four bytes

Each ASCII character is one byte, Strings are null terminated

CPUs understand instructions in assembly language

Address	Contents
114	
113	0
112	0
111	0
110	8
109	
108	0
107	C
106	B
105	A
104	
103	0xAF
102	0x3C
101	0x91
100	0xE3

What is Memory?

Memory is essentially a spreadsheet with a single column

- Every row has a number, called an address
- Every cell holds 1 byte of data

All data and running code are held in memory

```
int my_num = 8;
```

Integers are typically four bytes

Each ASCII character is one byte, Strings are null terminated

CPUs understand instructions in assembly language

Address	Contents
114	
113	0
112	0
111	0
110	8
109	
108	0
107	C
106	B
105	A
104	
103	0xAF
102	0x3C
101	0x91
100	0xE3

What is Memory?

Memory is essentially a spreadsheet with a single column

- Every row has a number, called an address
- Every cell holds 1 byte of data

All data and running code are held in memory

```
int my_num = 8;

String my_str = "ABC";
```

Integers are typically four bytes

Each ASCII character is one byte, Strings are null terminated

CPUs understand instructions in assembly language

Address	Contents
114	
113	0
112	0
111	0
110	8
109	
108	0
107	C
106	B
105	A
104	
103	0xAF
102	0x3C
101	0x91
100	0xE3

What is Memory?

Memory is essentially a spreadsheet with a single column

- Every row has a number, called an address
- Every cell holds 1 byte of data

All data and running code are held in memory

```
int my_num = 8;
String my_str = "ABC";
while (my_num > 0) my_num--;
```

Integers are typically four bytes

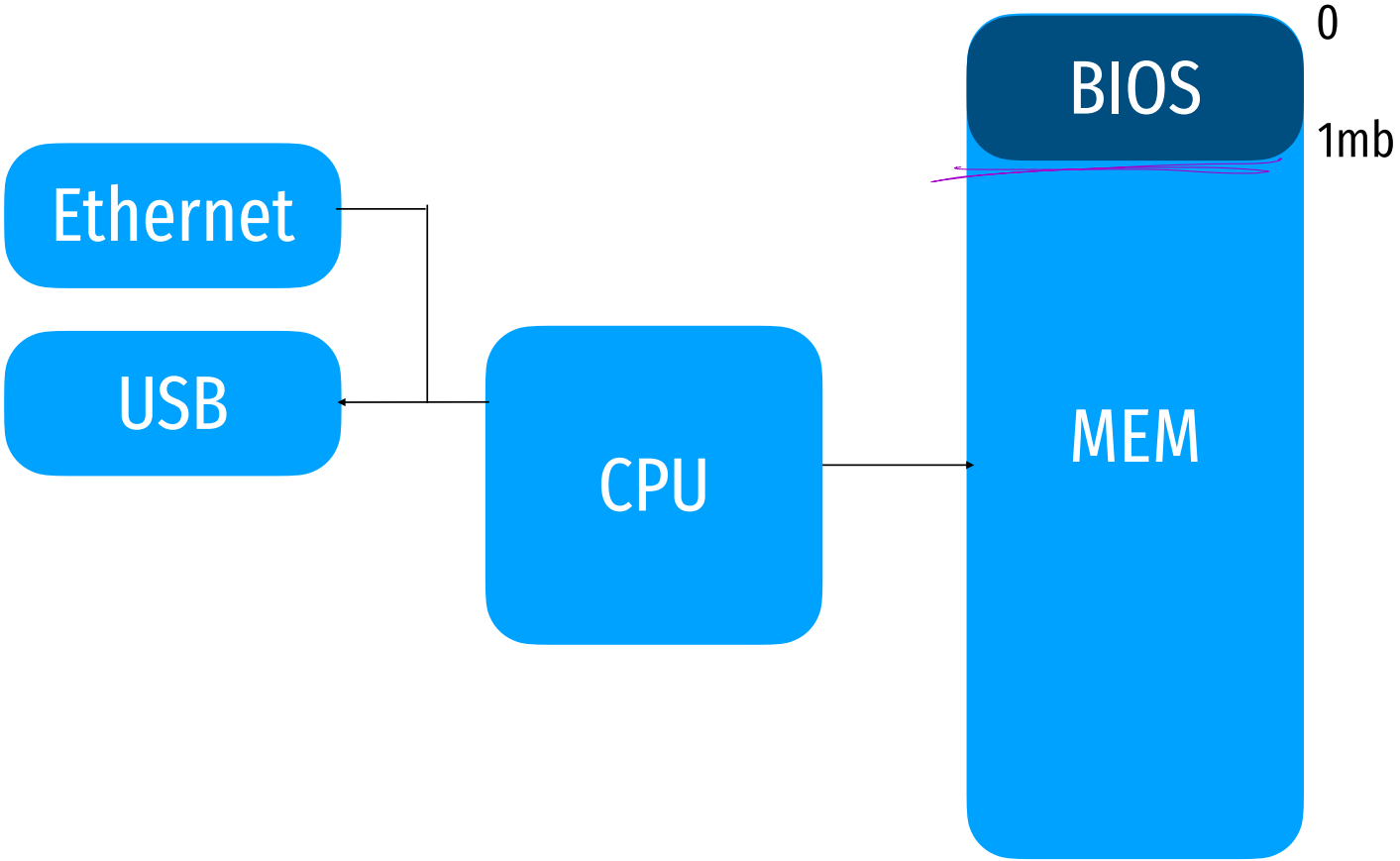
Each ASCII character is one byte, Strings are null terminated

CPUs understand instructions in assembly language

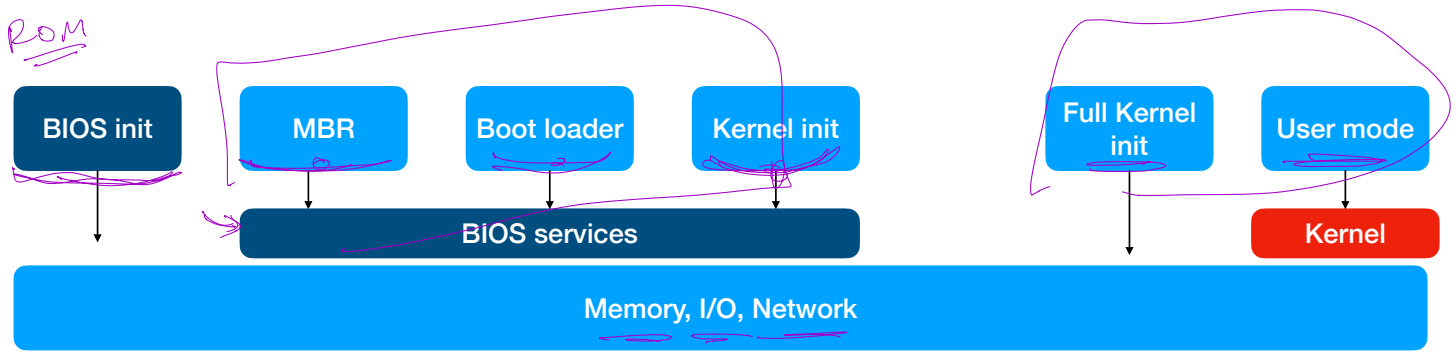
Address	Contents
114	
113	0
112	0
111	0
110	8
109	
108	0
107	C
106	B
105	A
104	
103	0xAF
102	0x3C
101	0x91
100	0xE3

How does a computer boot?

https://youtu.be/MsKb0gR-4AM?t=36

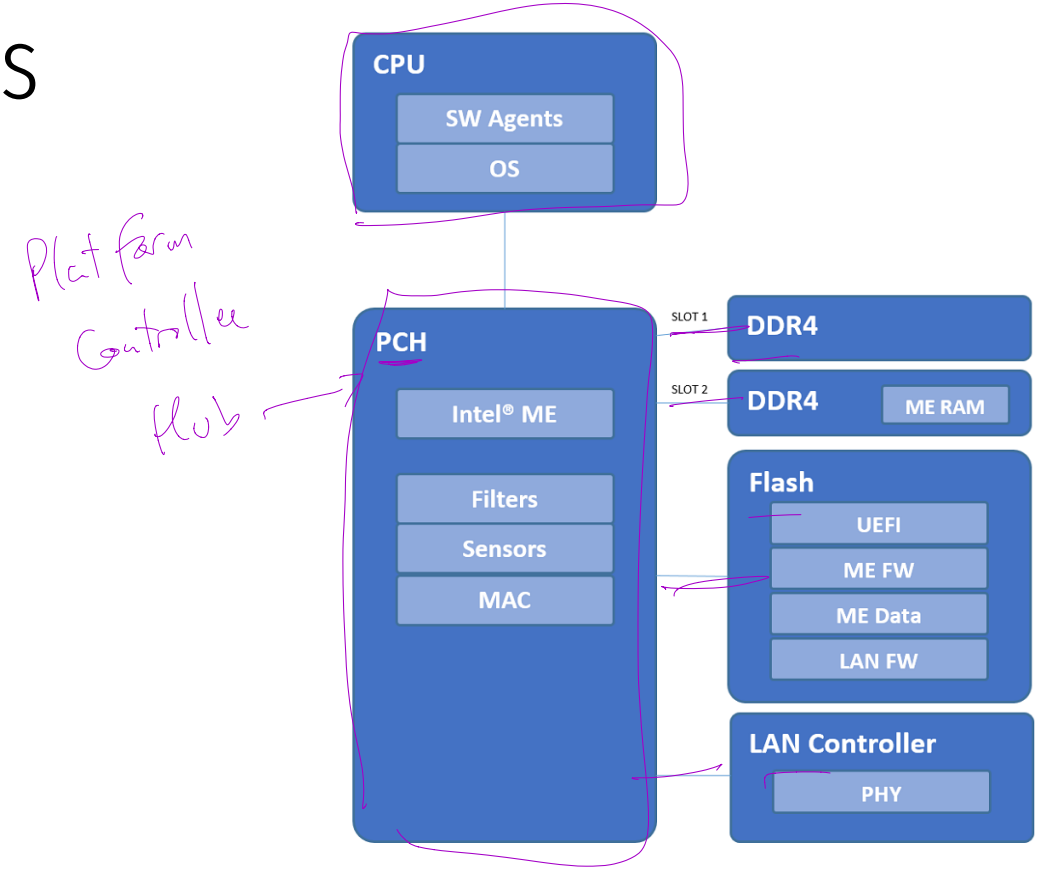

0
BIOS
1mb
Ethernet
USB
CPU
MEM

System Model: how does a computer boot?



https://www.intel.com/content/www/us/en/intelligent-systems/intel-boot-loader-development-kit/minimal-intel-architecture-boot-loader-paper.html

More details



Layout of memory at boot

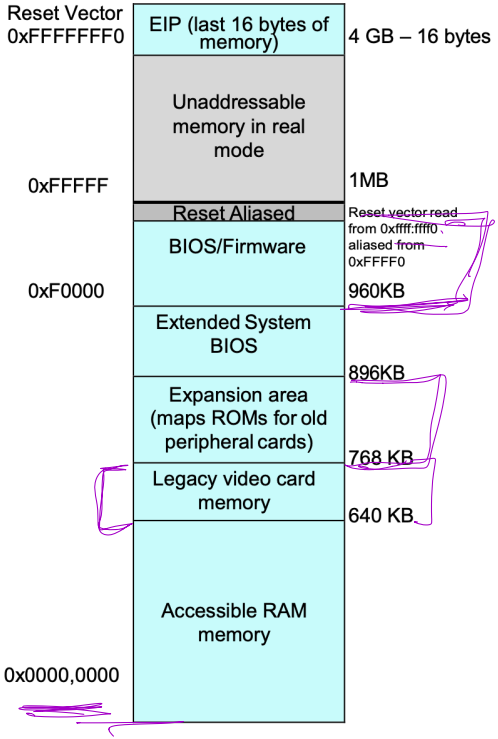


Figure 3 Intel® Architecture Memory Map at Power On

Details

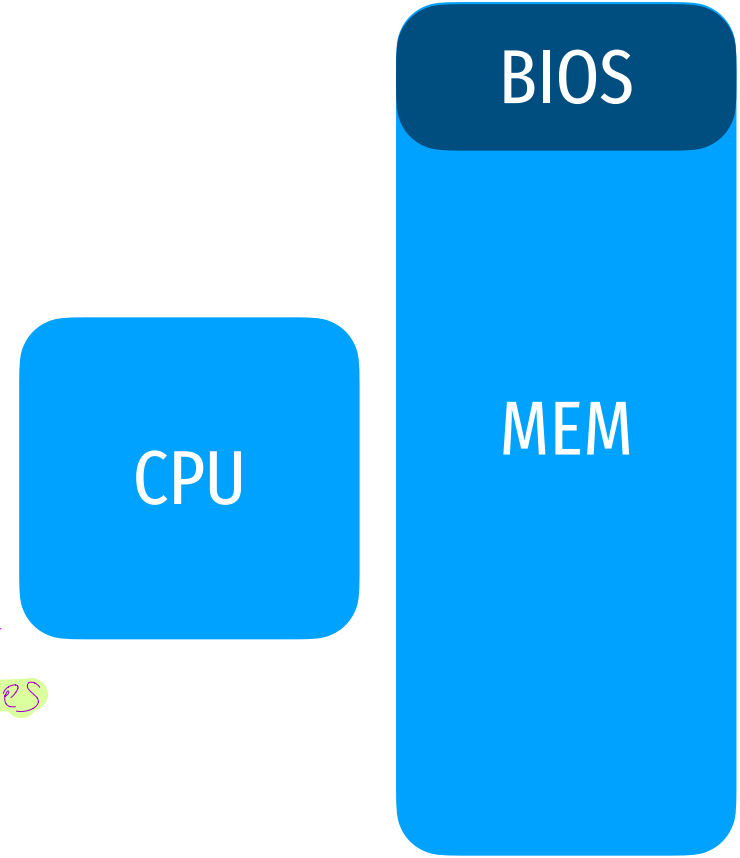
CPU begins executing at f.fff0
BIOS firmware begins init of hw
Applies microcode patches
Execute Firmware Support Pkg (blob)
[Ram is setup]
Copy firmware to RAM
Begin executing in RAM
Setup interrupts, timers, clocks
Bring up other cores
Setup PCI
Setup ACPI tables
Execute OS loader

OS

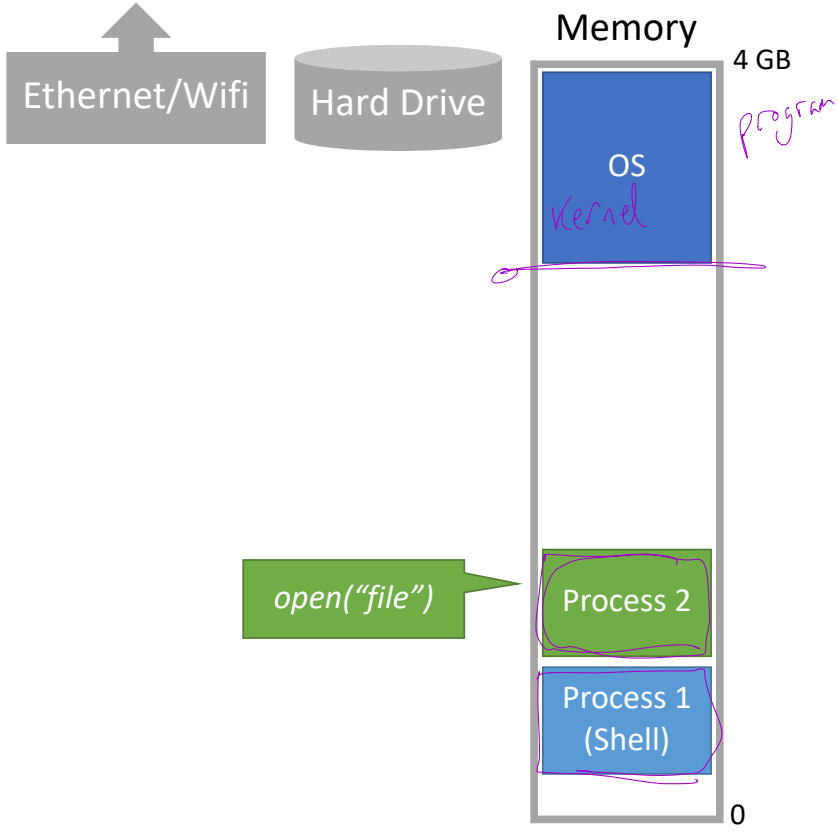
0.0000

in the next slides

power



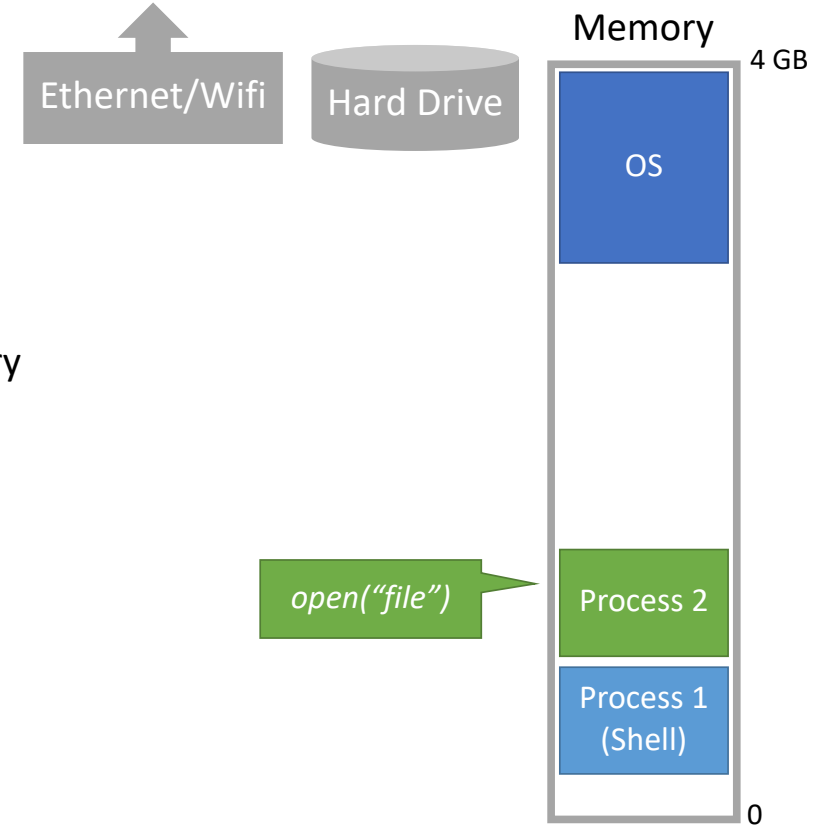
System Model



System Model

On bootup, the Operating System (OS) loads itself into memory

- eg. DOS (before hw isolation)
- Typically places itself in high memory



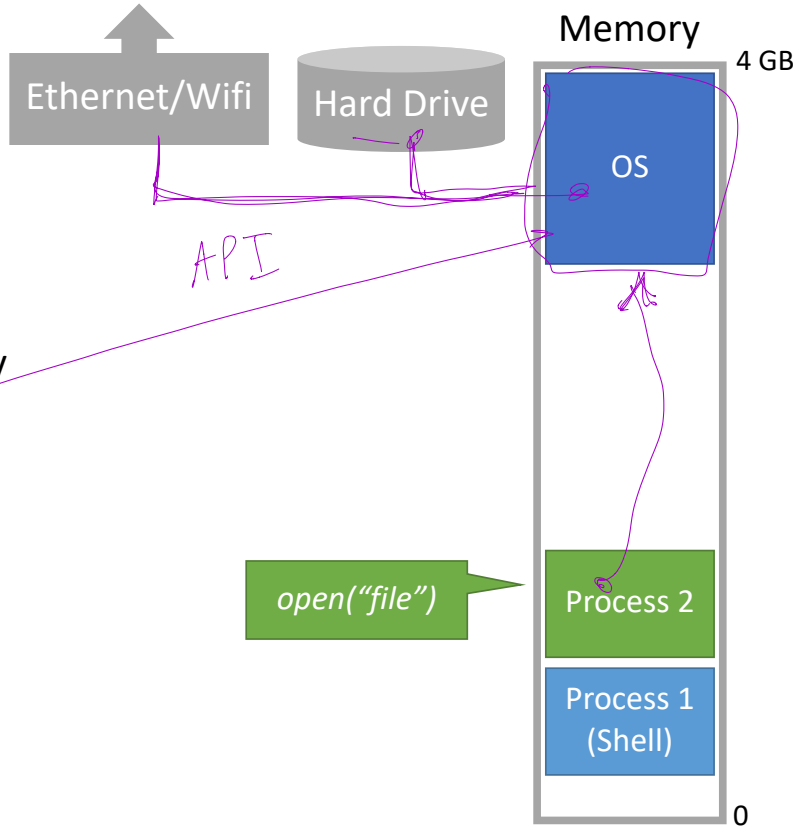
System Model

On bootup, the Operating System (OS) loads itself into memory

- eg. DOS (before hw isolation)
- Typically places itself in high memory

What is the role of the OS?

- Allow the user to run processes
- Often comes with a shell
  - Text shell like bash
  - Graphical shell like the Windows desktop
- Provides APIs to access devices
  - Offered as a convenience to application developers



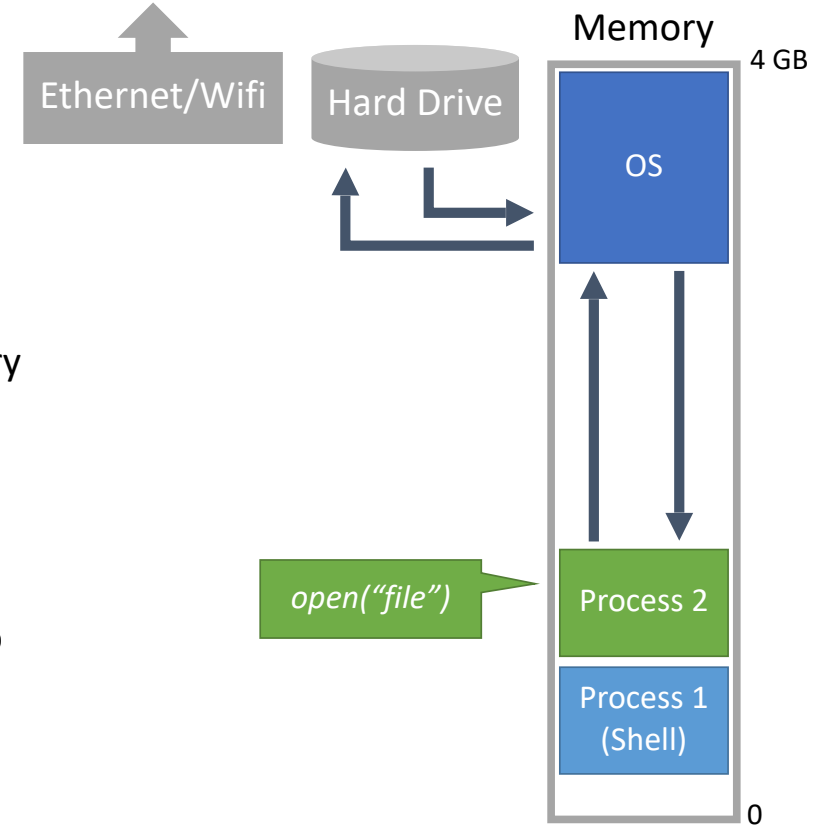
System Model

On bootup, the Operating System (OS) loads itself into memory

- eg. DOS (before hw isolation)
- Typically places itself in high memory

What is the role of the OS?

- Allow the user to run processes
- Often comes with a shell
  - Text shell like bash
  - Graphical shell like the Windows desktop
- Provides APIs to access devices
  - Offered as a convenience to application developers



dmesg

```
[    0.000000] microcode: microcode updated early to revision 0xca, date = 2019-10-03
[    0.000000] Linux version 5.3.0-64-generic (buildd@lcy01-amd64-026) (gcc version 9.2.1 20191008 (Ubuntu 9.2.1-9ubuntu2)) #58-Ubuntu SMP Fri Jul 10 19:33:51 UTC 2020 (Ubuntu 5.3.0-64.58-generic 5.3.18)
[    0.000000] Command line: BOOT_IMAGE=/boot/vmlinuz-5.3.0-64-generic root=/dev/mapper/vgubuntu-root ro quiet splash vt.handoff=7
[    0.000000] KERNEL supported cpus:
[    0.000000]   Intel GenuineIntel
[    0.000000]   AMD AuthenticAMD
[    0.000000]   Hygon HygonGenuine
[    0.000000]   Centaur CentaurHauls
[    0.000000]   zhaoxin   Shanghai
[    0.000000] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x008: 'MPX bounds registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x010: 'MPX CSR'
[    0.000000] x86/fpu: xstate_offset[2]:  576, xstate_sizes[2]:  256
[    0.000000] x86/fpu: xstate_offset[3]:  832, xstate_sizes[3]:   64
[    0.000000] x86/fpu: xstate_offset[4]:  896, xstate_sizes[4]:   64
[    0.000000] x86/fpu: Enabled xstate features 0x1f, context size is 960 bytes, using 'compacted' format.
[    0.000000] BIOS-provided physical RAM map:
[    0.000000] BIOS-e820: [mem 0x0000000000000000-0x000000000009efff] usable
[    0.000000] BIOS-e820: [mem 0x000000000009f000-0x000000000000ffff] reserved
[    0.000000] BIOS-e820: [mem 0x0000000000100000-0x000000006cf53fff] usable
[    0.000000] BIOS-e820: [mem 0x000000006cf54000-0x000000006f057fff] reserved
[    0.000000] BIOS-e820: [mem 0x000000006f058000-0x000000006f069fff] ACPI data
[    0.000000] BIOS-e820: [mem 0x000000006f06a000-0x000000006f06afff] reserved
[    0.000000] BIOS-e820: [mem 0x000000006f06b000-0x000000006f0d7fff] ACPI data
[    0.000000] BIOS-e820: [mem 0x000000006f0d8000-0x000000006f1b1fff] ACPI NVS
[    0.000000] BIOS-e820: [mem 0x000000006f1b2000-0x000000006fc4dfff] reserved
[    0.000000] BIOS-e820: [mem 0x000000006fc4e000-0x000000006fc4efff] usable
[    0.000000] BIOS-e820: [mem 0x000000006fc4f000-0x000000007cffffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000e0000000-0x00000000effffffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fe000000-0x00000000fe010fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fec00000-0x00000000fec00fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fed00000-0x00000000fed03fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fee00000-0x00000000fee00fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000ff000000-0x00000000ffffffff] reserved
[    0.000000] BIOS-e820: [mem 0x0000000100000000-0x0000000880ffffff] usable
[    0.000000] NX (Execute Disable) protection: active
[    0.000000] e820: update [mem 0x657d3018-0x657e3457] usable ==> usable
[    0.000000] e820: update [mem 0x657d3018-0x657e3457] usable ==> usable
[    0.000000] extended physical RAM map:
[    0.000000] reserve setup_data: [mem 0x0000000000000000-0x000000000009efff] usable
[    0.000000] reserve setup_data: [mem 0x000000000009f000-0x000000000000ffff] reserved
[    0.000000] reserve setup_data: [mem 0x0000000000100000-0x000000006657d3017] usable
[    0.000000] reserve setup_data: [mem 0x000000006657d3018-0x000000006657e3457] usable
[    0.000000] reserve setup_data: [mem 0x000000006657e3458-0x000000006cf53fff] usable
[    0.000000] reserve setup_data: [mem 0x000000006cf54000-0x000000006f057fff] reserved
[    0.000000] reserve setup_data: [mem 0x000000006f058000-0x000000006f069fff] ACPI data
[    0.000000] reserve setup_data: [mem 0x000000006f06a000-0x000000006f06afff] reserved
[    0.000000] reserve setup_data: [mem 0x000000006f06b000-0x000000006f0d7fff] ACPI data
[    0.000000] reserve setup_data: [mem 0x000000006f0d8000-0x000000006f1b1fff] ACPI NVS
[    0.000000] reserve setup_data: [mem 0x000000006f1b2000-0x000000006fc4dfff] reserved
[    0.000000] reserve setup_data: [mem 0x000000006fc4e000-0x000000006fc4efff] usable
[    0.000000] reserve setup_data: [mem 0x000000006fc4f000-0x000000007cffffff] reserved
[    0.000000] reserve setup_data: [mem 0x00000000e0000000-0x00000000effffffff] reserved
[    0.000000] reserve setup_data: [mem 0x00000000fe000000-0x00000000fe010fff] reserved
[    0.000000] reserve setup_data: [mem 0x00000000fec00000-0x00000000fec00fff] reserved
[    0.000000] reserve setup_data: [mem 0x00000000fed00000-0x00000000fed03fff] reserved
[    0.000000] reserve setup_data: [mem 0x00000000fee00000-0x00000000fee00fff] reserved
[    0.000000] reserve setup_data: [mem 0x00000000ff000000-0x00000000ffffffff] reserved
[    0.000000] reserve setup_data: [mem 0x0000000100000000-0x0000000880ffffff] usable
[    0.000000] efi: EFI v2.70 by American Megatrends
[    0.000000] efi:  ACPI=0x6f117000  ACPI 2.0=0x6f117014  TPMFinalLog=0x6f11f000  SMBIOS=0x6f9de000  SMBIOS 3.0=0x6f9dd000  MEMATTR=0x66372018  ESRT=0x68146518  TPMEventLog=0x657e4018
[    0.000000] secureboot: Secure boot enabled
[    0.000000] Kernel is locked down from EFI secure boot; see man kernel_lockdown.7
[    0.000000] SMBIOS 3.2.0 present.
[    0.000000] DMI: Intel(R) Client Systems NUC10i7FNH/NUC10i7FNB, BIOS FNCML357.0032.2019.1021.1624 10/21/2019
[    0.000000] tsc: Detected 1600.000 MHz processor
[    0.000376] tsc: Detected 1599.960 MHz TSC
[    0.000376] e820: update [mem 0x00000000-0x00000fff] usable ==> reserved
[    0.000379] e820: remove [mem 0x000a0000-0x000fffff] usable
[    0.000391] last_pfn = 0x881000 max_arch_pfn = 0x400000000
[    0.000398] MTRR default type: write-back
[    0.000399] MTRR fixed ranges enabled:
[    0.000401]   00000-9FFFF write-back
[    0.000403]   A0000-BFFFF uncachable
[    0.000405]   C0000-FFFFF write-protect
```

```
[    0.000406] MTRR variable ranges enabled:
[    0.000409]   0 base 0000000000 mask 7F8000000 uncachable
[    0.000410]   1 base 007C000000 mask 7FFC000000 uncachable
[    0.000412]   2 base 007A000000 mask 7FFE000000 uncachable
[    0.000413]   3 base 0079000000 mask 7FFF000000 uncachable
[    0.000415]   4 base 0078800000 mask 7FFF800000 uncachable
[    0.000417]   5 base 2000000000 mask 6000000000 uncachable
[    0.000418]   6 base 1000000000 mask 7000000000 uncachable
[    0.000420]   7 base 4000000000 mask 4000000000 uncachable
[    0.000421]   8 disabled
[    0.000422]   9 disabled
[    0.001256] x86/PAT: Configuration [0-7]: WB  WC  UC- UC  WB  WP  UC- WT
[    0.001570] last_pfn = 0x6fc4f max_arch_pfn = 0x400000000
[    0.021713] esrt: Reserving ESRT space from 0x0000000068146518 to 0x0000000068146550.
[    0.021729] e820: update [mem 0x68146000-0x68146fff] usable ==> reserved
[    0.021847] check: Scanning 1 areas for low memory corruption
[    0.021853] Using GB pages for direct mapping
[    0.022564] RAMDISK: [mem 0x3ce54000-0x3fffdfff]
[    0.022580] ACPI: Early table checksum verification disabled
[    0.022584] ACPI: RSDP 0x000000006F117014 000024 (v02 INTEL )
[    0.022589] ACPI: XSDT 0x000000006F116728 0000CC (v01 INTEL  NUC9i5FN 00000020 AMI  01000013)
[    0.022598] ACPI: FACP 0x000000006F0D2000 000114 (v06 INTEL  NUC9i5FN 00000020 AMI  00010013)
[    0.022606] ACPI: DSDT 0x000000006F08F000 042561 (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022610] ACPI: FACS 0x000000006F1B1000 000040
[    0.022614] ACPI: MCFG 0x000000006F0D5000 00003C (v01 INTEL  NUC9i5FN 00000020 MSFT 00000097)
[    0.022618] ACPI: SSDT 0x000000006F0D3000 001B4A (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022623] ACPI: FIDT 0x000000006F08E000 00009C (v01 INTEL  NUC9i5FN 00000020 AMI  00010013)
[    0.022627] ACPI: SSDT 0x000000006F08A000 0031C6 (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022631] ACPI: HPET 0x000000006F0D7000 000038 (v01 INTEL  NUC9i5FN 00000020 AMI  01000013)
[    0.022635] ACPI: SSDT 0x000000006F086000 003384 (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022639] ACPI: SSDT 0x000000006F084000 00147B (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022643] ACPI: SSDT 0x000000006F080000 0032BD (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022648] ACPI: NHLT 0x000000006F0D6000 00002D (v00 INTEL  NUC9i5FN 00000020 AMI  01000013)
[    0.022652] ACPI: LPIT 0x000000006F07F000 000094 (v01 INTEL  NUC9i5FN 00000020 AMI  01000013)
[    0.022656] ACPI: SSDT 0x000000006F07B000 002720 (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022660] ACPI: SSDT 0x000000006F07A000 00087C (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022664] ACPI: DBGP 0x000000006F079000 000034 (v01 INTEL  NUC9i5FN 00000020 AMI  01000013)
[    0.022668] ACPI: DBG2 0x000000006F078000 000054 (v00 INTEL  NUC9i5FN 00000020 AMI  01000013)
[    0.022672] ACPI: SSDT 0x000000006F076000 001B66 (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022677] ACPI: TPM2 0x000000006F074000 00004C (v04 INTEL  NUC9i5FN 00000020 AMI  00000000)
[    0.022681] ACPI: DMAR 0x000000006F075000 000A8 (v01 INTEL  NUC9i5FN 00000020          01000013)
[    0.022685] ACPI: WSMT 0x000000006F07E000 000028 (v01 INTEL  NUC9i5FN 00000020 AMI  00010013)
[    0.022689] ACPI: APIC 0x000000006F073000 0000F4 (v04 INTEL  NUC9i5FN 00000020 AMI  00010013)
[    0.022693] ACPI: FPDT 0x000000006F072000 000044 (v01 INTEL  NUC9i5FN 00000020 AMI  01000013)
[    0.022707] ACPI: Local APIC address 0xfee00000
[    0.023236] No NUMA configuration found
[    0.023238] Faking a node at [mem 0x0000000000000000-0x0000000880ffffff]
[    0.023254] NODE_DATA(0) allocated [mem 0x880fd5000-0x880ffffff]
[    0.023698] Zone ranges:
[    0.023700]   DMA      [mem 0x0000000000001000-0x0000000000ffffff]
[    0.023702]   DMA32    [mem 0x0000000001000000-0x00000000ffffffff]
[    0.023703]   Normal   [mem 0x0000000100000000-0x0000000880ffffff]
[    0.023705]   Device   empty
[    0.023706] Movable zone start for each node
[    0.023711] Early memory node ranges
[    0.023713]   node   0: [mem 0x0000000000001000-0x000000000009efff]
[    0.023714]   node   0: [mem 0x0000000000100000-0x000000006cf53fff]
[    0.023716]   node   0: [mem 0x000000006fc4e000-0x000000006fc4efff]
[    0.023717]   node   0: [mem 0x0000000100000000-0x0000000880ffffff]
[    0.024270] Zeroed struct page in unavailable ranges: 41229 pages
[    0.024272] Initmem setup node 0 [mem 0x0000000000001000-0x0000000880ffffff]
[    0.024274] On node 0 totalpages: 8314611
[    0.024276]   DMA zone: 64 pages used for memmap
[    0.024277]   DMA zone: 25 pages reserved
[    0.024278]   DMA zone: 3998 pages, LIFO batch:0
[    0.024379]   DMA32 zone: 6910 pages used for memmap
[    0.024380]   DMA32 zone: 442197 pages, LIFO batch:63
[    0.039990]   Normal zone: 122944 pages used for memmap
[    0.039991]   Normal zone: 7868416 pages, LIFO batch:63
```

```
[    0.207139] Reserving Intel graphics memory at [mem 0x79000000-0x7cffffff]
[    0.207882] ACPI: PM-Timer IO Port: 0x1808
[    0.207884] ACPI: Local APIC address 0xfee00000
[    0.207894] ACPI: LAPIC_NMI (acpi_id[0x01] high edge lint[0x1])
[    0.207895] ACPI: LAPIC_NMI (acpi_id[0x02] high edge lint[0x1])
[    0.207896] ACPI: LAPIC_NMI (acpi_id[0x03] high edge lint[0x1])
[    0.207897] ACPI: LAPIC_NMI (acpi_id[0x04] high edge lint[0x1])
[    0.207897] ACPI: LAPIC_NMI (acpi_id[0x05] high edge lint[0x1])
[    0.207898] ACPI: LAPIC_NMI (acpi_id[0x06] high edge lint[0x1])
[    0.207899] ACPI: LAPIC_NMI (acpi_id[0x07] high edge lint[0x1])
[    0.207900] ACPI: LAPIC_NMI (acpi_id[0x08] high edge lint[0x1])
[    0.207901] ACPI: LAPIC_NMI (acpi_id[0x09] high edge lint[0x1])
[    0.207902] ACPI: LAPIC_NMI (acpi_id[0x0a] high edge lint[0x1])
[    0.207902] ACPI: LAPIC_NMI (acpi_id[0x0b] high edge lint[0x1])
[    0.207903] ACPI: LAPIC_NMI (acpi_id[0x0c] high edge lint[0x1])
[    0.207957] IOAPIC[0]: apic_id 2, version 32, address 0xfec00000, GSI 0-119
[    0.207960] ACPI: INT_SRC_OVR (bus 0 bus_irq 0 global_irq 2 dfl dfl)
[    0.207962] ACPI: INT_SRC_OVR (bus 0 bus_irq 9 global_irq 9 high level)
[    0.207963] ACPI: IRQ0 used by override.
[    0.207965] ACPI: IRQ9 used by override.
[    0.207968] Using ACPI (MADT) for SMP configuration information
[    0.207969] ACPI: HPET id: 0x8086a201 base: 0xfed00000
[    0.207974] smpboot: Allowing 12 CPUs, 0 hotplug CPUs
[    0.208001] PM: Registered nosave memory: [mem 0x00000000-0x00000fff]
[    0.208003] PM: Registered nosave memory: [mem 0x0009f000-0x000fffff]
[    0.208005] PM: Registered nosave memory: [mem 0x657d3000-0x657d3fff]
[    0.208008] PM: Registered nosave memory: [mem 0x657e3000-0x657e3fff]
[    0.208010] PM: Registered nosave memory: [mem 0x68146000-0x68146fff]
[    0.208012] PM: Registered nosave memory: [mem 0x6cf54000-0x6f057fff]
[    0.208013] PM: Registered nosave memory: [mem 0x6f058000-0x6f069fff]
[    0.208014] PM: Registered nosave memory: [mem 0x6f06a000-0x6f06afff]
[    0.208015] PM: Registered nosave memory: [mem 0x6f06b000-0x6f0d7fff]
[    0.208015] PM: Registered nosave memory: [mem 0x6f0d8000-0x6f1b1fff]
[    0.208016] PM: Registered nosave memory: [mem 0x6f1b2000-0x6fc4dfff]
[    0.208018] PM: Registered nosave memory: [mem 0x6fc4f000-0x7cffffff]
[    0.208019] PM: Registered nosave memory: [mem 0x7d000000-0xdfffffff]
[    0.208020] PM: Registered nosave memory: [mem 0xe0000000-0xefffffff]
[    0.208021] PM: Registered nosave memory: [mem 0xf0000000-0xfdffffff]
[    0.208022] PM: Registered nosave memory: [mem 0xfe000000-0xfe010fff]
[    0.208022] PM: Registered nosave memory: [mem 0xfe011000-0xfebffffff]
[    0.208023] PM: Registered nosave memory: [mem 0xfec00000-0xfec00fff]
[    0.208024] PM: Registered nosave memory: [mem 0xfec01000-0xfecffffff]
[    0.208025] PM: Registered nosave memory: [mem 0xfed00000-0xfed03fff]
[    0.208025] PM: Registered nosave memory: [mem 0xfed04000-0xfedffffff]
[    0.208026] PM: Registered nosave memory: [mem 0xfee00000-0xfee00fff]
[    0.208027] PM: Registered nosave memory: [mem 0xfee01000-0xfeffffff]
[    0.208028] PM: Registered nosave memory: [mem 0xff000000-0xffffffff]
[    0.208030] [mem 0x7d000000-0xdfffffff] available for PCI devices
[    0.208033] Booting paravirtualized kernel on bare hardware
[    0.208036] clocksource: refined-jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns: 7645519600211568 ns
[    0.208045] setup_percpu: NR_CPUS:8192 nr_cpumask_bits:12 nr_cpu_ids:12 nr_node_ids:1
[    0.208519] percpu: Embedded 54 pages/cpu s184320 r8192 d28672 u262144
[    0.208530] pcpu-alloc: s184320 r8192 d28672 u262144 alloc=1*2097152
[    0.208531] pcpu-alloc: [0] 00 01 02 03 04 05 06 07 [0] 08 09 10 11 -- -- -- --
[    0.208576] Built 1 zonelists, mobility grouping on.  Total pages: 8184668
[    0.208577] Policy zone: Normal
[    0.208579] Kernel command line: BOOT_IMAGE=/boot/vmlinuz-5.3.0-64-generic root=/dev/mapper/vgubuntu-root ro quiet splash vt.handoff=7
[    0.211269] Dentry cache hash table entries: 4194304 (order: 13, 33554432 bytes, linear)
[    0.212573] Inode-cache hash table entries: 2097152 (order: 12, 16777216 bytes, linear)
[    0.212724] mem auto-init: stack:off, heap alloc:on, heap free:off
[    0.217959] Calgary: detecting Calgary via BIOS EBDA area
[    0.217961] Calgary: Unable to locate Rio Grande table in EBDA - bailing!
[    0.326171] Memory: 32410828K/33258444K available (14339K kernel code, 2387K rwdata, 4732K rodata, 2680K init, 5040K bss, 847616K reserved, 0K cma-reserved)
[    0.326180] random: get_random_u64 called from kmem_cache_open+0x2d/0x410 with crng_init=0
[    0.326431] SLUB: HWalign=64, Order=0-3, MinObjects=0, CPUs=12, Nodes=1
[    0.326453] ftrace: allocating 43632 entries in 171 pages
[    0.355621] rcu: Hierarchical RCU implementation.
[    0.355623] rcu:     RCU restricting CPUs from NR_CPUS=8192 to nr_cpu_ids=12.
[    0.355624]  Tasks RCU enabled.
[    0.355626] rcu: RCU calculated value of scheduler-enlistment delay is 25 jiffies.
[    0.355627] rcu: Adjusting geometry for rcu_fanout_leaf=16, nr_cpu_ids=12
[    0.361328] NR_IRQS: 524544, nr_irqs: 2152, preallocated irqs: 16
[    0.361842] random: crng done (trusting CPU's manufacturer)
[    0.361879] vt handoff: transparent VT on vt#7
[    0.361889] Console: colour dummy device 80x25
[    0.361895] printk: console [tty0] enabled
[    0.361918] ACPI: Core revision 20190703
[    0.362625] clocksource: hpet: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns: 79635855245 ns
[    0.362761] APIC: Switch to symmetric I/O mode setup
[    0.362764] DMAR: Host address width 39
[    0.362766] DMAR: DRHD base: 0x000000fed90000 flags: 0x0
[    0.362774] DMAR: dmar0: reg_base_addr fed90000 ver 1:0 cap 1c0000c40660462 ecap 19e2ff0505e
```

```
[    0.696153] PCI host bridge to bus 0000:00
[    0.696156] pci_bus 0000:00: root bus resource [io  0x0000-0x0cf7 window]
[    0.696158] pci_bus 0000:00: root bus resource [io  0x0d00-0xffff window]
[    0.696159] pci_bus 0000:00: root bus resource [mem 0x000a0000-0x000bffff window]
[    0.696161] pci_bus 0000:00: root bus resource [mem 0x000e0000-0x000e3fff window]
[    0.696162] pci_bus 0000:00: root bus resource [mem 0x000e4000-0x000e7fff window]
[    0.696163] pci_bus 0000:00: root bus resource [mem 0x000e8000-0x000ebfff window]
[    0.696165] pci_bus 0000:00: root bus resource [mem 0x000ec000-0x000effff window]
[    0.696166] pci_bus 0000:00: root bus resource [mem 0x000f0000-0x000fffff window]
[    0.696167] pci_bus 0000:00: root bus resource [mem 0x7d000000-0xdfffffff window]
[    0.696169] pci_bus 0000:00: root bus resource [mem 0x4000000000-0x7fffffffff window]
[    0.696170] pci_bus 0000:00: root bus resource [mem 0xfc800000-0xfe7fffff window]
[    0.696172] pci_bus 0000:00: root bus resource [bus 00-fe]
[    0.696187] pci 0000:00:00.0: [8086:9b51] type 00 class 0x060000
[    0.696942] pci 0000:00:02.0: [8086:9bca] type 00 class 0x030000
[    0.696959] pci 0000:00:02.0: reg 0x10: [mem 0x6022000000-0x6022ffffff 64bit]
[    0.696967] pci 0000:00:02.0: reg 0x18: [mem 0x4000000000-0x400fffffff 64bit pref]
[    0.696973] pci 0000:00:02.0: reg 0x20: [io  0x3000-0x303f]
[    0.697334] pci 0000:00:08.0: [8086:1911] type 00 class 0x088000
[    0.697352] pci 0000:00:08.0: reg 0x10: [mem 0x6023120000-0x6023120fff 64bit]
[    0.697682] pci 0000:00:12.0: [8086:02f9] type 00 class 0x118000
[    0.697708] pci 0000:00:12.0: reg 0x10: [mem 0x602311f000-0x602311ffff 64bit]
[    0.698071] pci 0000:00:14.0: [8086:02ed] type 00 class 0x0c0330
[    0.698096] pci 0000:00:14.0: reg 0x10: [mem 0x6023100000-0x602310ffff 64bit]
[    0.698176] pci 0000:00:14.0: PME# supported from D3hot D3cold
[    0.698639] pci 0000:00:14.2: [8086:02ef] type 00 class 0x050000
[    0.698662] pci 0000:00:14.2: reg 0x10: [mem 0x6023118000-0x6023119fff 64bit]
[    0.698675] pci 0000:00:14.2: reg 0x18: [mem 0x602311e000-0x602311efff 64bit]
[    0.699033] pci 0000:00:14.3: [8086:02f0] type 00 class 0x028000
[    0.699130] pci 0000:00:14.3: reg 0x10: [mem 0x6023114000-0x6023117fff 64bit]
[    0.699383] pci 0000:00:14.3: PME# supported from D0 D3hot D3cold
[    0.699892] pci 0000:00:15.0: [8086:02e8] type 00 class 0x0c8000
[    0.700034] pci 0000:00:15.0: reg 0x10: [mem 0x00000000-0x00000fff 64bit]
[    0.700757] pci 0000:00:15.2: [8086:02ea] type 00 class 0x0c8000
[    0.700899] pci 0000:00:15.2: reg 0x10: [mem 0x00000000-0x00000fff 64bit]
[    0.701605] pci 0000:00:16.0: [8086:02e0] type 00 class 0x078000
[    0.701636] pci 0000:00:16.0: reg 0x10: [mem 0x602311b000-0x602311bfff 64bit]
[    0.701726] pci 0000:00:16.0: PME# supported from D3hot
[    0.702164] pci 0000:00:17.0: [8086:02d3] type 00 class 0x010601
[    0.702186] pci 0000:00:17.0: reg 0x10: [mem 0x96220000-0x96221fff]
[    0.702195] pci 0000:00:17.0: reg 0x14: [mem 0x96223000-0x962230ff]
[    0.702204] pci 0000:00:17.0: reg 0x18: [io  0x3090-0x3097]
[    0.702213] pci 0000:00:17.0: reg 0x1c: [io  0x3080-0x3083]
[    0.702222] pci 0000:00:17.0: reg 0x20: [io  0x3060-0x307f]
[    0.702230] pci 0000:00:17.0: reg 0x24: [mem 0x96222000-0x962227ff]
```

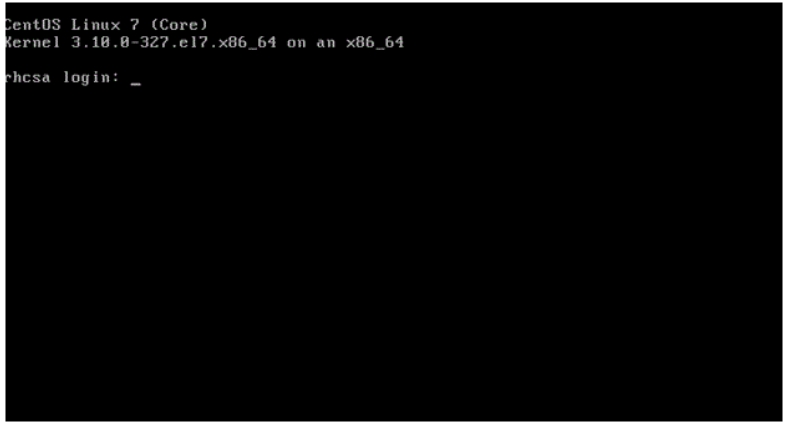
```
[    0.814539] NET: Registered protocol family 2
[    0.814767] tcp_listen_portaddr_hash hash table entries: 16384 (order: 6, 262144 bytes, linear)
[    0.815038] TCP established hash table entries: 262144 (order: 9, 2097152 bytes, linear)
[    0.815477] TCP bind hash table entries: 65536 (order: 8, 1048576 bytes, linear)
[    0.815587] TCP: Hash tables configured (established 262144 bind 65536)
[    0.815676] UDP hash table entries: 16384 (order: 7, 524288 bytes, linear)
[    0.815819] UDP-Lite hash table entries: 16384 (order: 7, 524288 bytes, linear)
[    0.815965] NET: Registered protocol family 1
[    0.815971] NET: Registered protocol family 44
[    0.815984] pci 0000:00:02.0: Video device with shadowed ROM at [mem 0x000c0000-0x000dffff]
[    0.816626] pci 0000:01:00.0: CLS mismatch (64 != 128), using 64 bytes
[    0.816828] pci 0000:01:00.0: enabling device (0002 -> 0003)
[    0.817224] Trying to unpack rootfs image as initramfs...
[    0.986156] Initramfs unpacking failed: Decoding failed
[    0.993397] Freeing initrd memory: 50856K
[    0.993425] DMAR: Intel-IOMMU force enabled due to platform opt in
[    0.993459] DMAR: No ATSR found
[    0.993546] DMAR: dmar0: Using Queued invalidation
```

```
[    4.346737] idma64 idma64.1: Found Intel integrated DMA 64-bit
[    4.351728] mei_me 0000:00:16.0: hbm: dma setup response: failure = 3 REJECTED
[    4.352889] Bluetooth: Core ver 2.22
[    4.352898] NET: Registered protocol family 31
[    4.352898] Bluetooth: HCI device and connection manager initialized
[    4.352901] Bluetooth: HCI socket layer initialized
[    4.352902] Bluetooth: L2CAP socket layer initialized
[    4.352903] Bluetooth: SCO socket layer initialized
[    4.389562] 88x2bu: loading out-of-tree module taints kernel.
[    4.403328] cryptd: max_cpu_qlen set to 1000
[    4.406575] Intel(R) Wireless WiFi driver for Linux
[    4.406576] Copyright(c) 2003- 2015 Intel Corporation
[    4.406641] iwlwifi 0000:00:14.3: enabling device (0000 -> 0002)
[    4.415431] iwlwifi 0000:00:14.3: TLV_FW_FSEQ_VERSION: FSEQ Version: 43.2.23.17
[    4.415434] iwlwifi 0000:00:14.3: Found debug destination: EXTERNAL_DRAM
[    4.415435] iwlwifi 0000:00:14.3: Found debug configuration: 0
[    4.415592] iwlwifi 0000:00:14.3: loaded firmware version 48.4fa0041f.0 op_mode iwlmvm
[    4.424971] AVX2 version of gcm_enc/dec engaged.
[    4.424972] AES CTR mode by8 optimization enabled
[    4.427194] usbcore: registered new interface driver btusb
[    4.428100] Bluetooth: hci0: Firmware revision 0.0 build 62 week 31 2019
```

```
[    1.103257] usb usb1: New USB device found, idVendor=1d6b, idProduct=0002, bcdDevice= 5.03
[    1.103259] usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber=1
[    1.103261] usb usb1: Product: xHCI Host Controller
[    1.103263] usb usb1: Manufacturer: Linux 5.3.0-64-generic xhci-hcd
[    1.103264] usb usb1: SerialNumber: 0000:00:14.0
[    1.103469] hub 1-0:1.0: USB hub found
[    1.103489] hub 1-0:1.0: 12 ports detected
[    1.106061] xhci_hcd 0000:00:14.0: xHCI Host Controller
[    1.106066] xhci_hcd 0000:00:14.0: new USB bus registered, assigned bus number 2
[    1.106071] xhci_hcd 0000:00:14.0: Host supports USB 3.1 Enhanced SuperSpeed
[    1.106121] usb usb2: New USB device found, idVendor=1d6b, idProduct=0003, bcdDevice= 5.03
[    1.106122] usb usb2: New USB device strings: Mfr=3, Product=2, SerialNumber=1
[    1.106124] usb usb2: Product: xHCI Host Controller
[    1.106126] usb usb2: Manufacturer: Linux 5.3.0-64-generic xhci-hcd
[    1.106127] usb usb2: SerialNumber: 0000:00:14.0
[    1.106307] hub 2-0:1.0: USB hub found
[    1.106321] hub 2-0:1.0: 6 ports detected
[    1.107605] usb: port power management may be unreliable
[    1.107867] xhci_hcd 0000:39:00.0: xHCI Host Controller
[    1.107874] xhci_hcd 0000:39:00.0: new USB bus registered, assigned bus number 3
[    1.109069] xhci_hcd 0000:39:00.0: hcc params 0x200077c1 hci version 0x110 quirks 0x0000000200009810
[    1.109318] usb usb3: New USB device found, idVendor=1d6b, idProduct=0002, bcdDevice= 5.03
[    1.109320] usb usb3: New USB device strings: Mfr=3, Product=2, SerialNumber=1
[    1.109322] usb usb3: Product: xHCI Host Controller
[    1.109323] usb usb3: Manufacturer: Linux 5.3.0-64-generic xhci-hcd
[    1.109325] usb usb3: SerialNumber: 0000:39:00.0
[    1.109508] hub 3-0:1.0: USB hub found
[    1.109520] hub 3-0:1.0: 2 ports detected
[    1.109708] xhci_hcd 0000:39:00.0: xHCI Host Controller
```

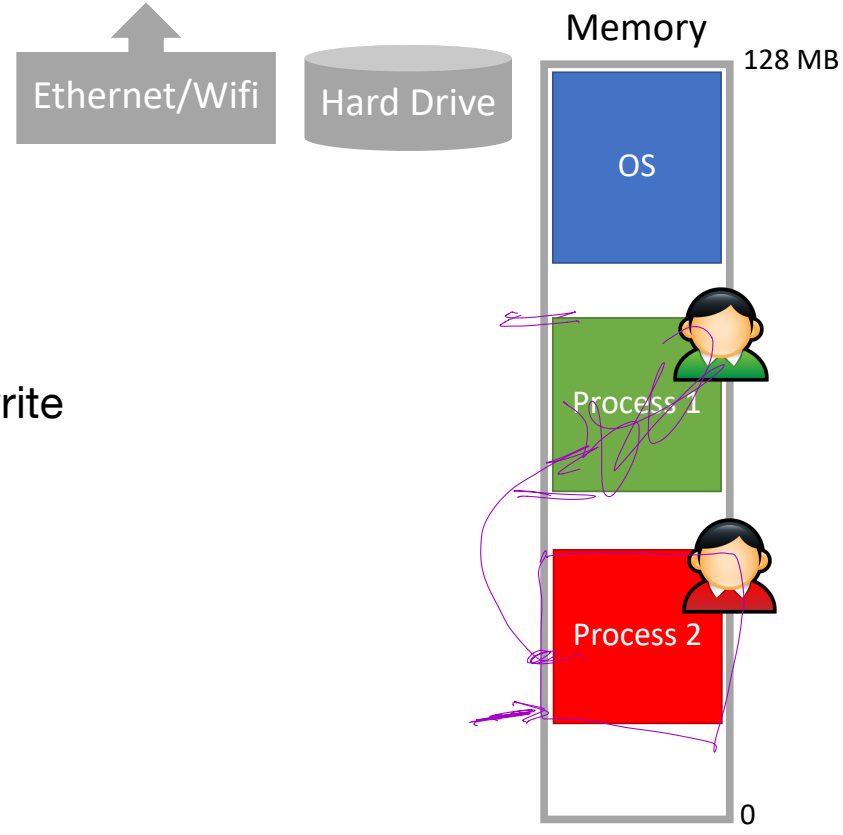
```
[    8.658281] audit: type=1400 audit(1602118928.642:68): apparmor="STATUS" operation="profile_
docker.dockerd" name="docker-default" pid=1506 comm="apparmor_parser"
[    8.810186] Initializing XFRM netlink socket
[   10.867709] e1000e: eno1 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: Rx/Tx
[   10.867863] IPv6: ADDRCONF(NETDEV_CHANGE): eno1: link becomes ready
[   98.291764] rfkill: input handler disabled
[  142.368628] audit: type=1400 audit(1602119062.356:69): apparmor="DENIED" operation="open" p
ompose" name="/proc/2958/mounts" pid=2958 comm="python3" requested_mask="r" denied_mask="r" fsu
```

What happens after boot?



Memory Unsafety

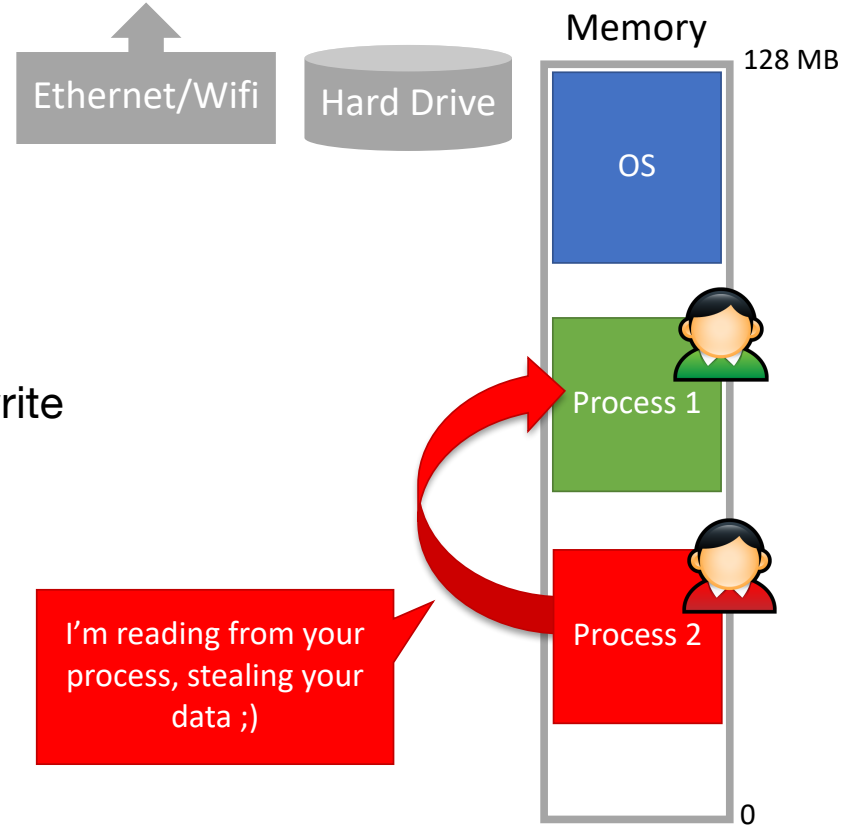
DOS



Problem: any process can read/write any memory

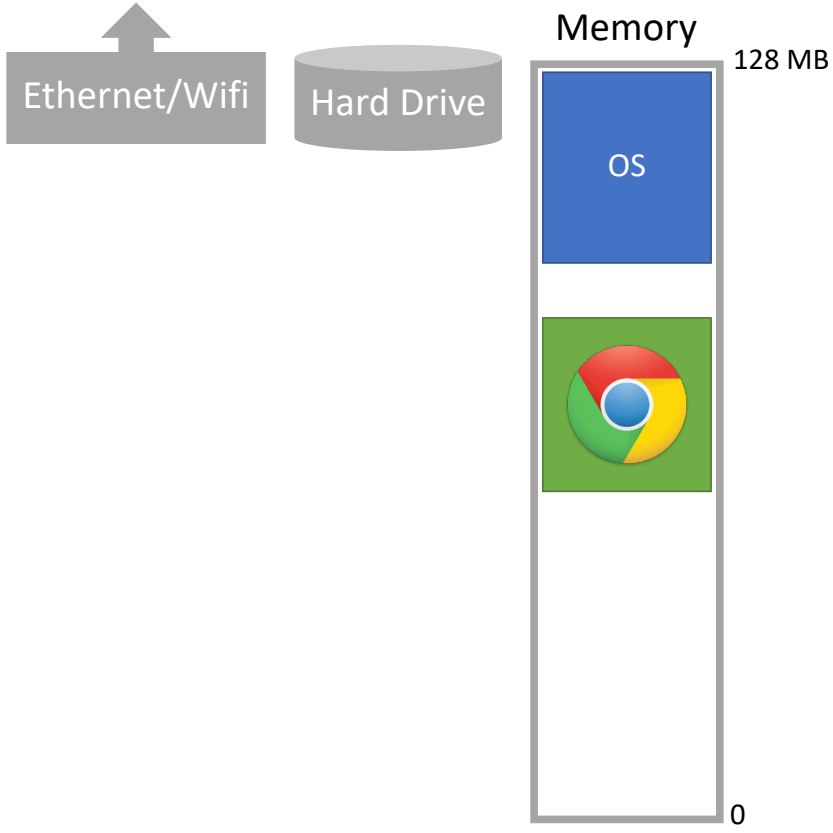
Memory Unsafety

Problem: any process can read/write any memory



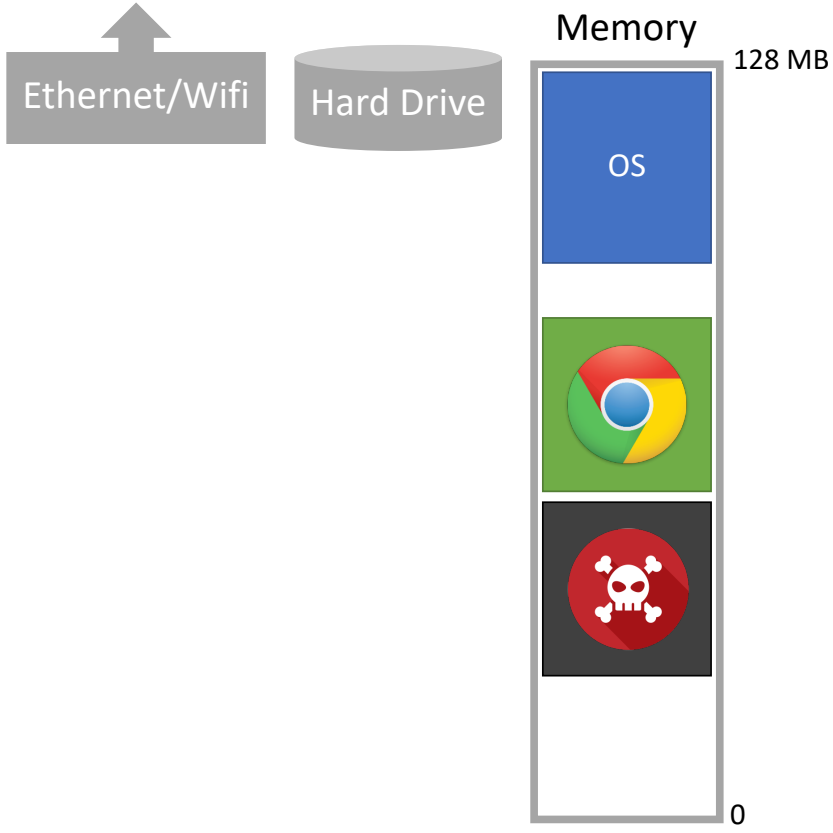
Memory Unsafety

Problem: any process can read/write any memory



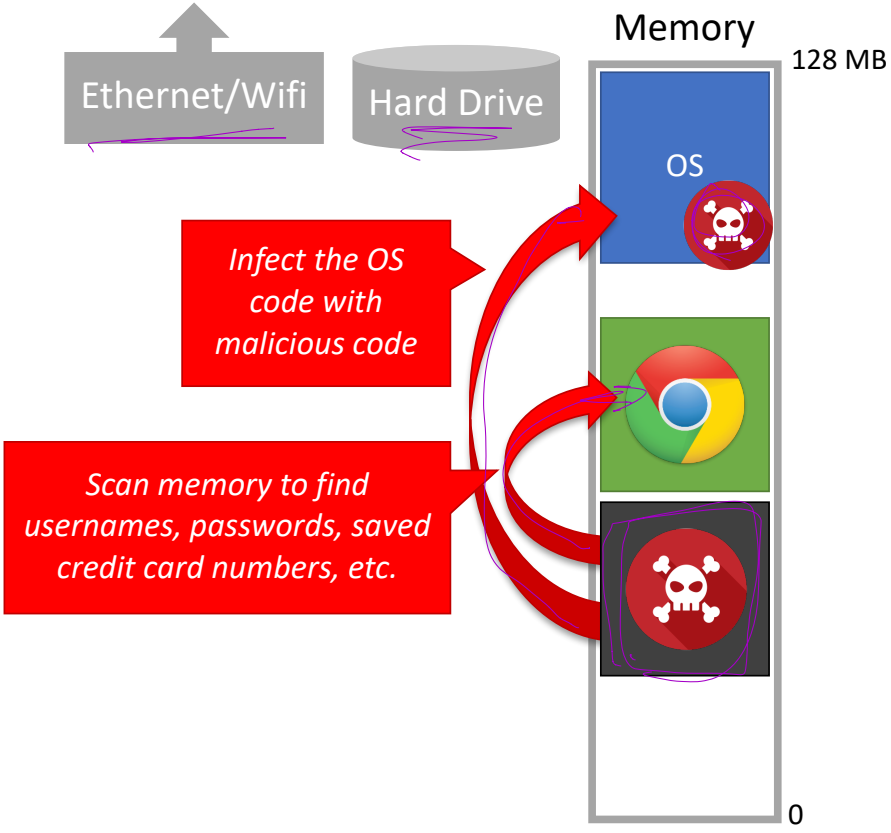
Memory Unsafety

Problem: any process can read/write any memory



Memory Unsafety

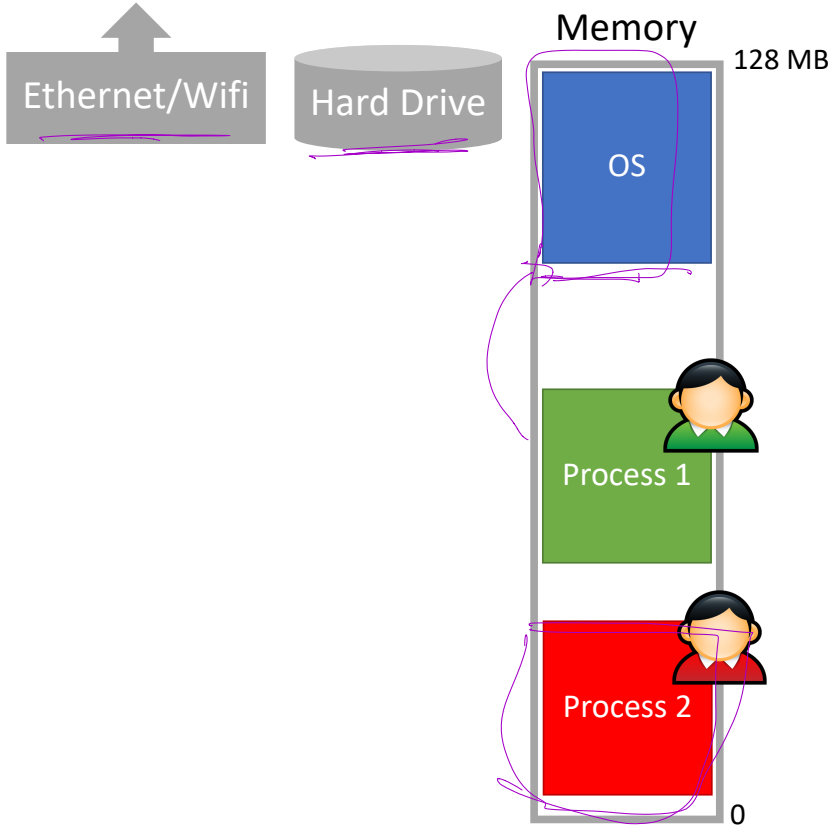
Problem: any process can read/write any memory



Device Unsafety

Problem: any process can access any hardware device directly

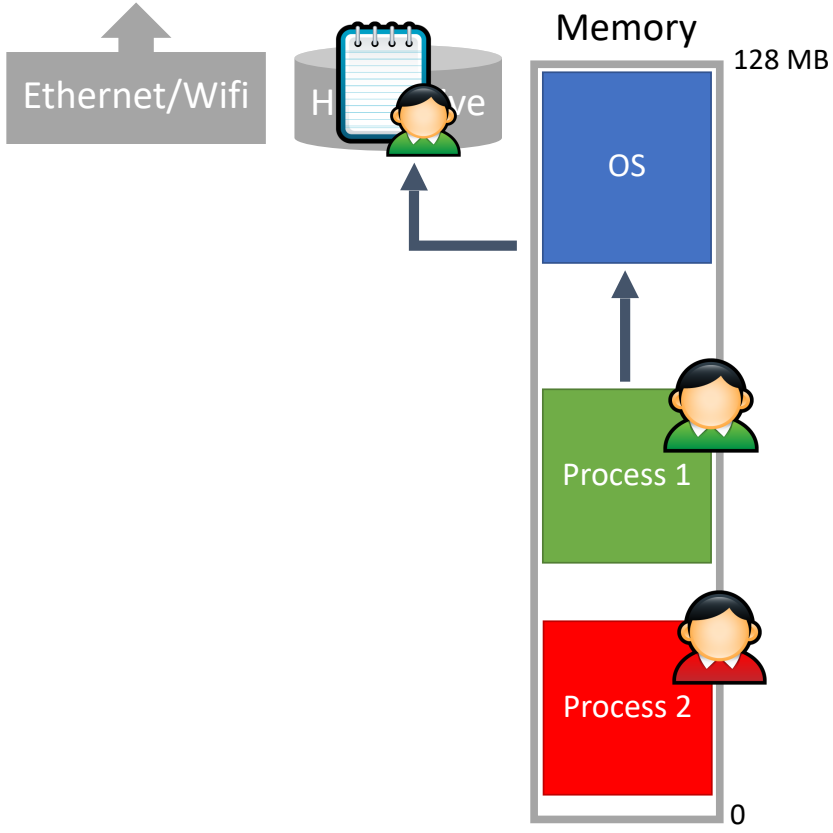
Access control is enforced by the OS, but OS APIs can be bypassed



Device Unsafety

Problem: any process can access any hardware device directly

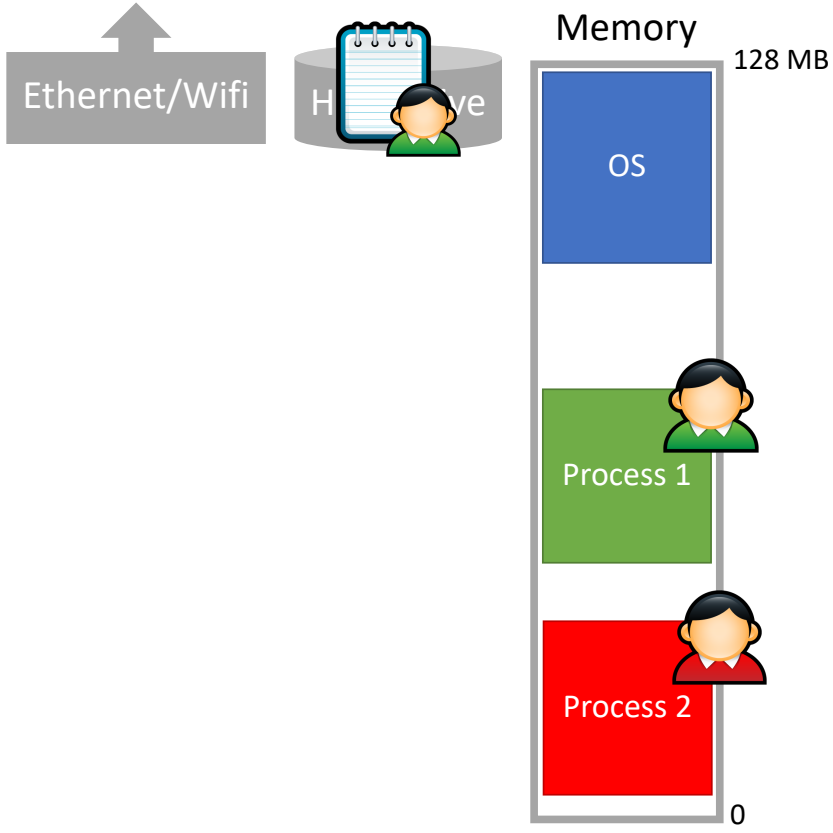
Access control is enforced by the OS, but OS APIs can be bypassed



Device Unsafety

Problem: any process can access any hardware device directly

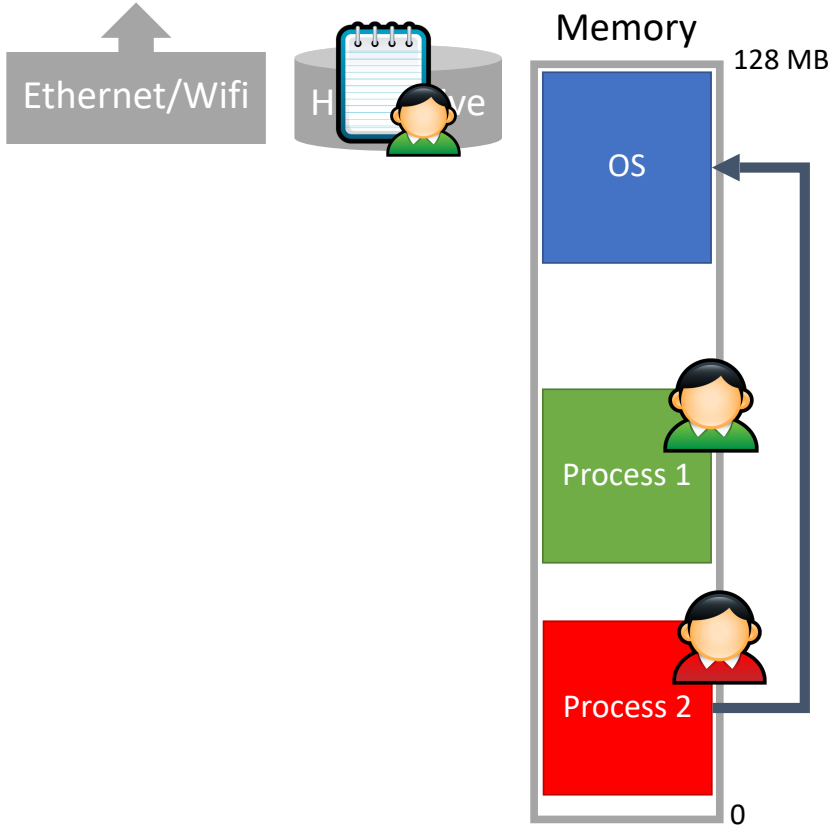
Access control is enforced by the OS, but OS APIs can be bypassed



Device Unsafety

Problem: any process can access any hardware device directly

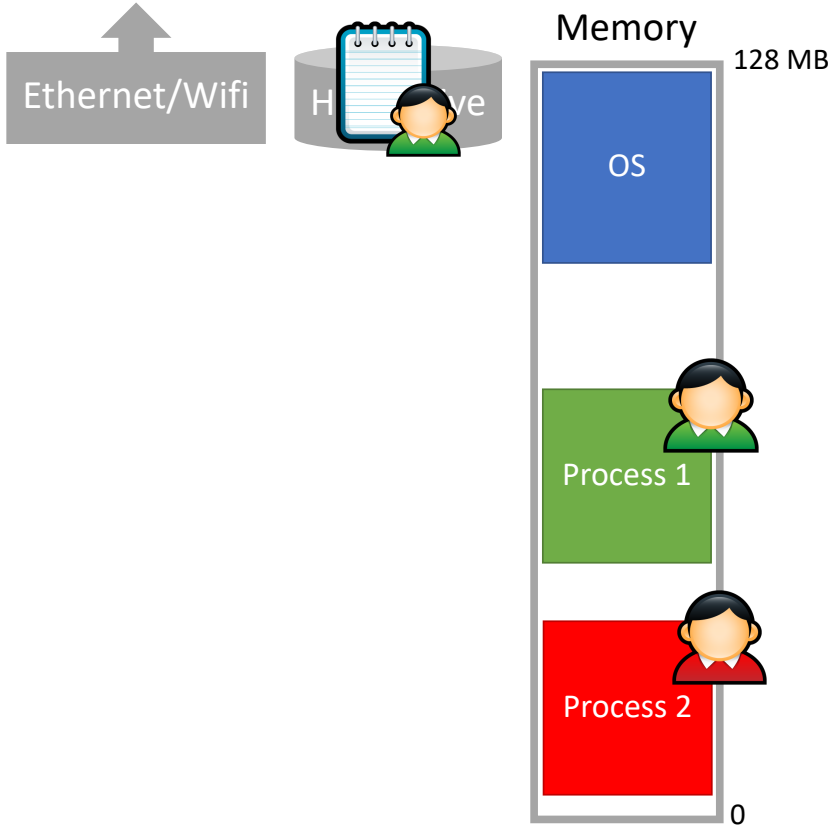
Access control is enforced by the OS, but OS APIs can be bypassed



Device Unsafety

Problem: any process can access any hardware device directly

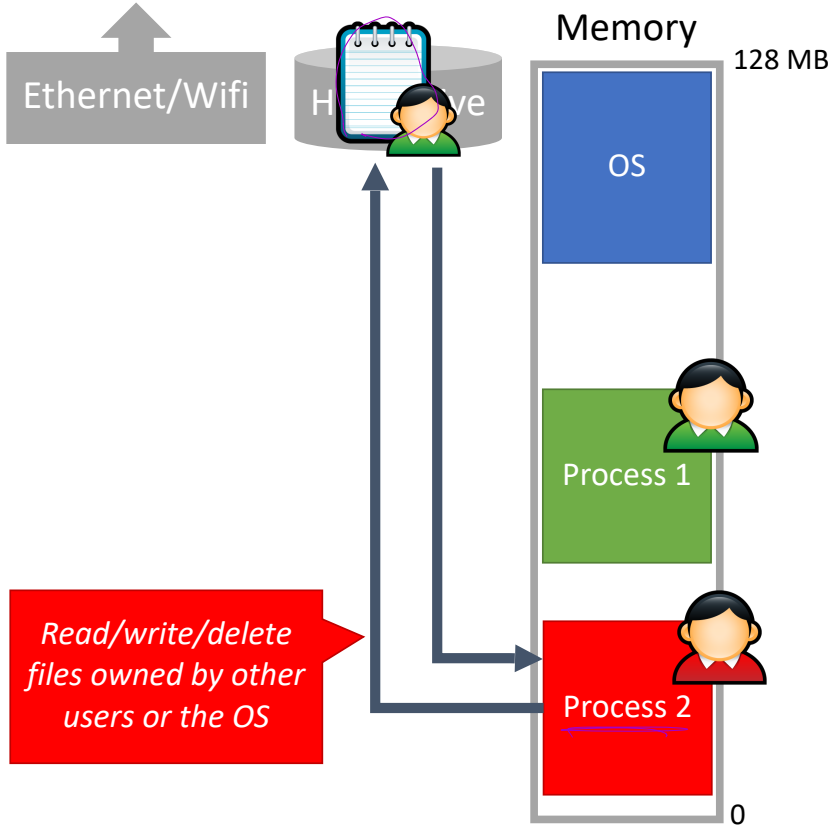
Access control is enforced by the OS, but OS APIs can be bypassed



Device Unsafety

Problem: any process can access any hardware device directly

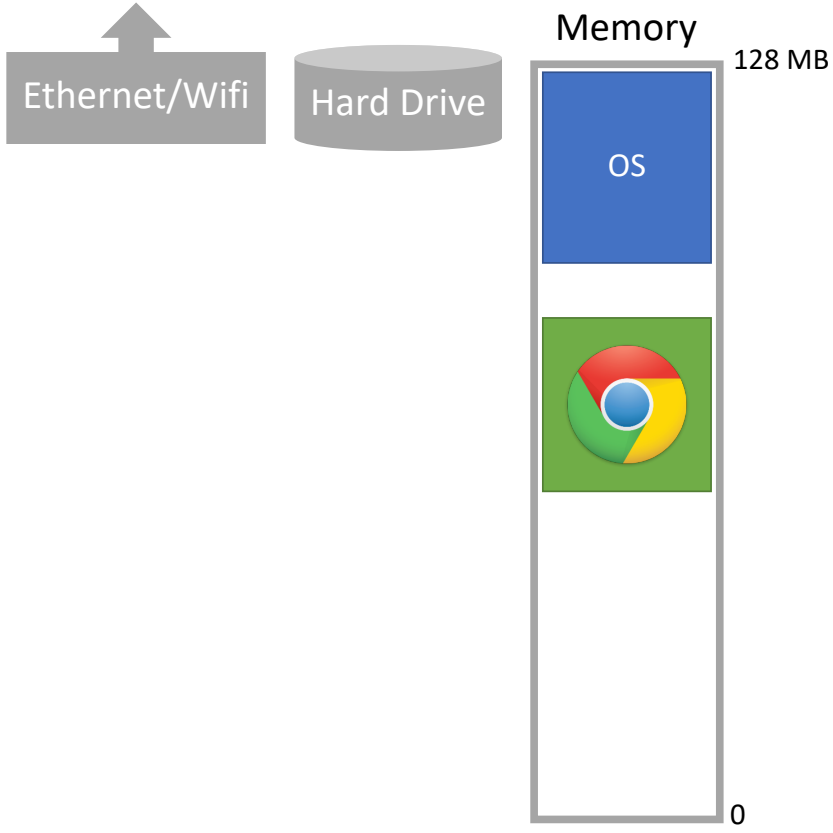
Access control is enforced by the OS, but OS APIs can be bypassed



Device Unsafety

Problem: any process can access any hardware device directly

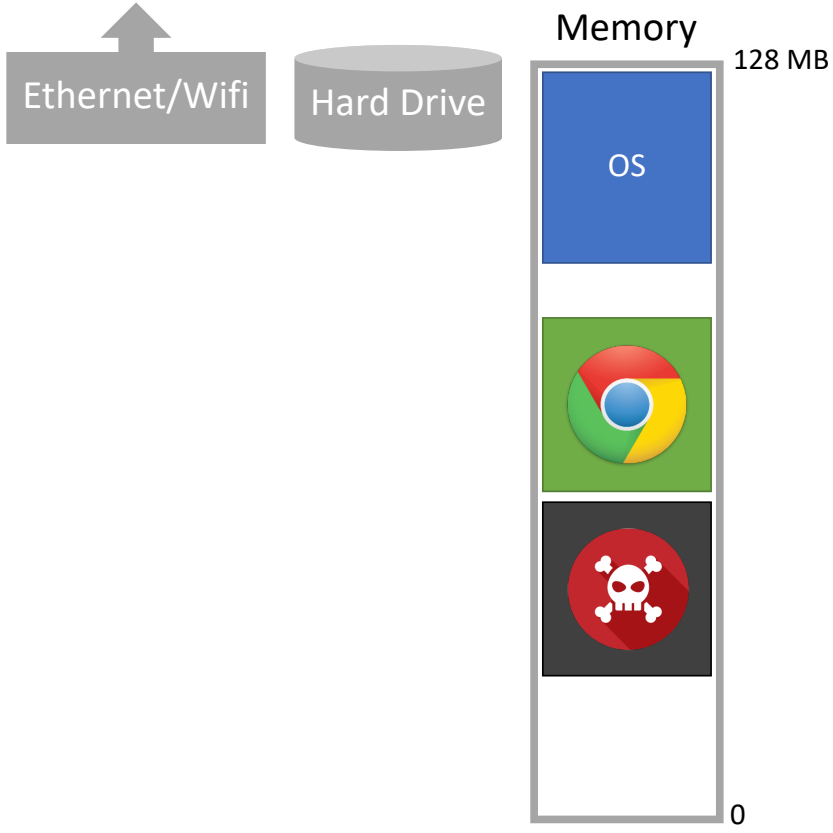
Access control is enforced by the OS, but OS APIs can be bypassed



Device Unsafety

Problem: any process can access any hardware device directly

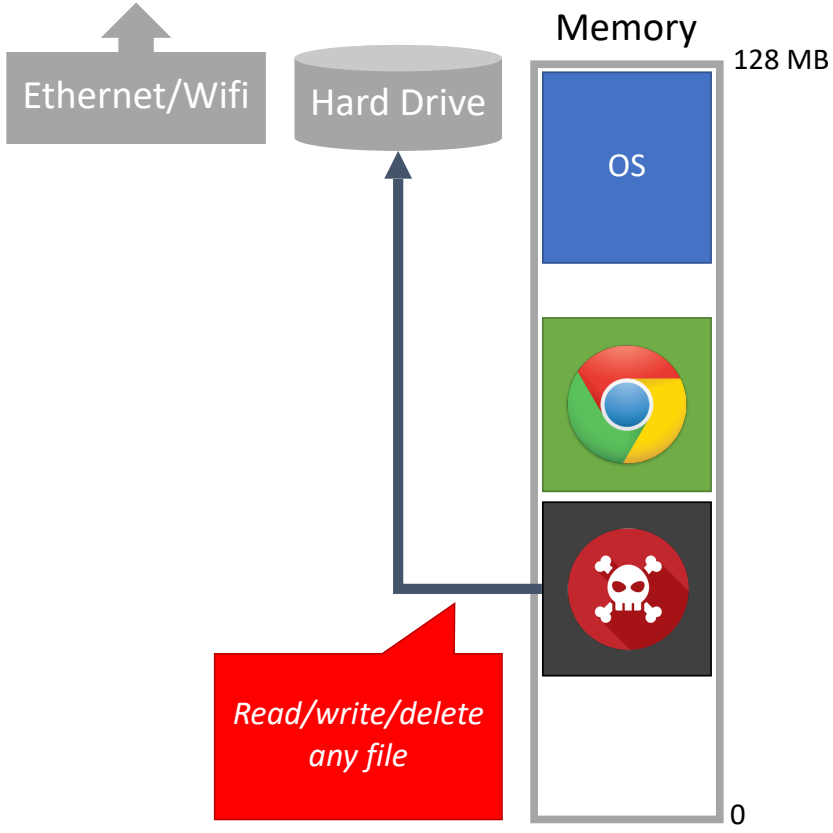
Access control is enforced by the OS, but OS APIs can be bypassed



Device Unsafety

Problem: any process can access any hardware device directly

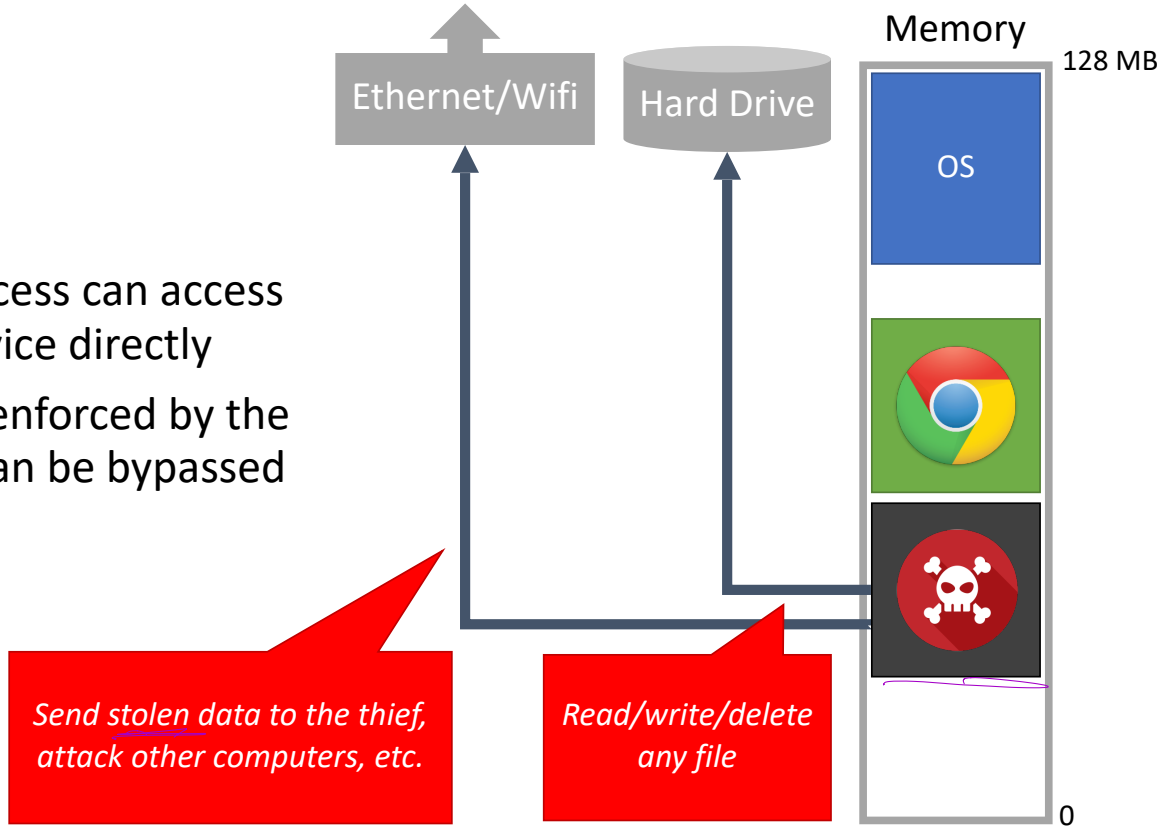
Access control is enforced by the OS, but OS APIs can be bypassed



Device Unsafety

Problem: any process can access any hardware device directly

Access control is enforced by the OS, but OS APIs can be bypassed



Review

Old systems did not protect memory or devices

- Any process could access any memory
- Any process could access any device

Problems

- No way to enforce access controls on users or devices
- Processes can steal from or destroy each other
- Processes can modify or destroy the OS

On old computers, systems security was literally impossible

ISOLATION

Threat Model
Principles
Intro to System Architecture
Hardware Support for Isolation
Examples


ASUS

Towards Modern Architecture

To achieve systems security, we need process isolation

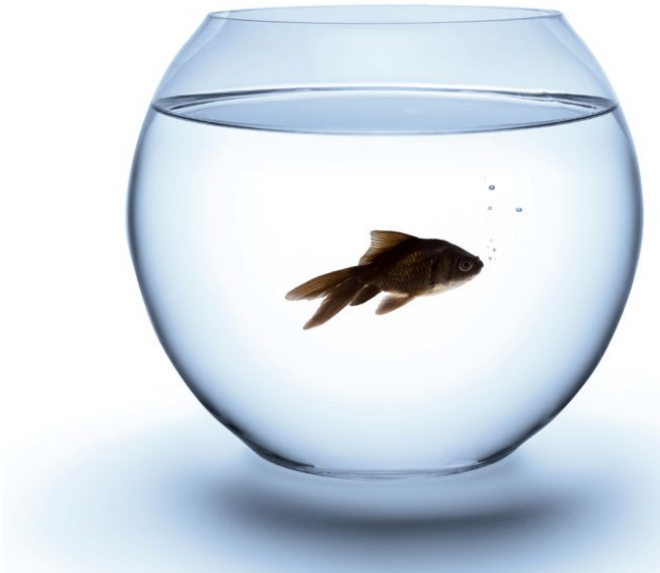
- Processes cannot read/write memory arbitrarily
- Processes cannot access devices directly

How do we achieve this?

Hardware support for isolation

1. Protected mode execution (a.k.a. process rings)
2. Virtual memory

Needed for process isolation.



Protected Mode

Protected Mode

Most modern CPUs support protected mode

x86 CPUs support three rings with different privileges

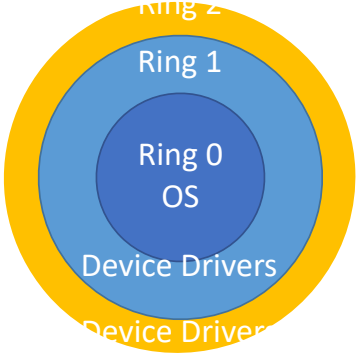
- Ring 0: Operating System
  - Code in this ring may directly access any device

Protected Mode

Most modern CPUs support protected mode

x86 CPUs support three rings with different privileges

- Ring 0: Operating System
  - Code in this ring may directly access any device
- Ring 1, 2: device drivers
  - Code in these rings may directly access some devices
  - May not change the protection level of the CPU

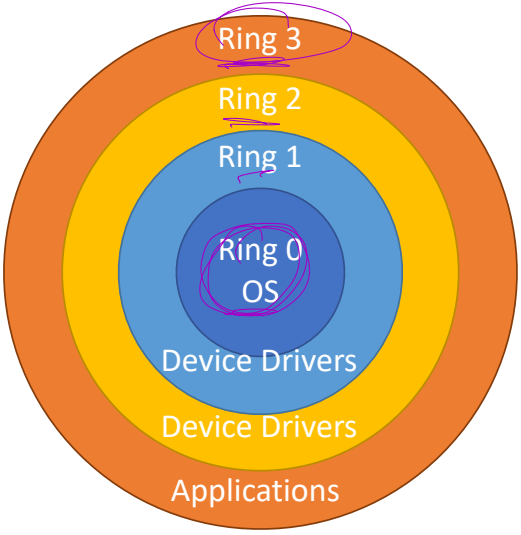


Protected Mode

Most modern CPUs support protected mode

x86 CPUs support three rings with different privileges

- Ring 0: Operating System
  - Code in this ring may directly access any device
- Ring 1, 2: device drivers
  - Code in these rings may directly access some devices
  - May not change the protection level of the CPU
- Ring 3: userland
  - Code in this ring may not directly access devices
  - All device access must be via OS APIs
  - May not change the protection level of the CPU



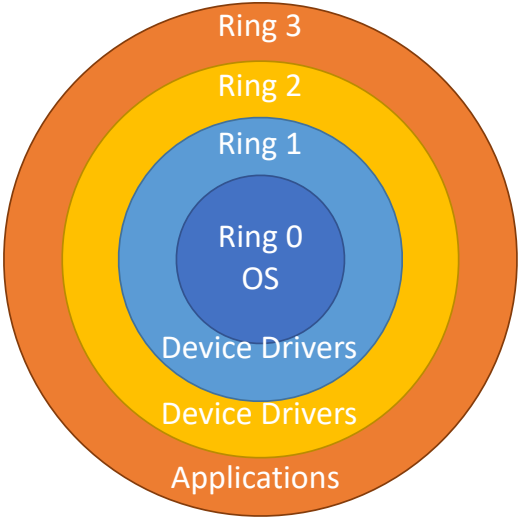
Protected Mode

Most modern CPUs support protected mode

x86 CPUs support three rings with different privileges

- Ring 0: Operating System
  - Code in this ring may directly access any device
- Ring 1, 2: device drivers
  - Code in these rings may directly access some devices
  - May not change the protection level of the CPU
- Ring 3: userland
  - Code in this ring may not directly access devices
  - All device access must be via OS APIs
  - May not change the protection level of the CPU

Most OSes only use rings 0 and 3



Ring -1,-2,-3

“Google cited worries that the Intel ME (actually MINIX) code runs on their CPU's deepest access level — Ring "-3" — and also runs a web server component that allows anyone to remotely connect to remote computers, even when the main OS is turned off.”

System Boot Sequence

1. On startup, the CPU starts in 16-bit real mode
   - Protected mode is disabled
   - Any process can access any device

System Boot Sequence

1. On startup, the CPU starts in 16-bit real mode
   - Protected mode is disabled
   - Any process can access any device
2. BIOS executes, finds and loads the OS

System Boot Sequence

1. On startup, the CPU starts in 16-bit real mode
   - Protected mode is disabled
   - Any process can access any device
2. BIOS executes, finds and loads the OS
3. OS switches CPU to 32-bit protected mode
   - OS code is now running in Ring 0
   - OS decides what Ring to place other processes in

System Boot Sequence

1. On startup, the CPU starts in 16-bit real mode
   - Protected mode is disabled
   - Any process can access any device
2. BIOS executes, finds and loads the OS
3. OS switches CPU to 32-bit protected mode
   - OS code is now running in Ring 0
   - OS decides what Ring to place other processes in
4. Shell gets executed, user may run programs
   - User processes are placed in Ring 3

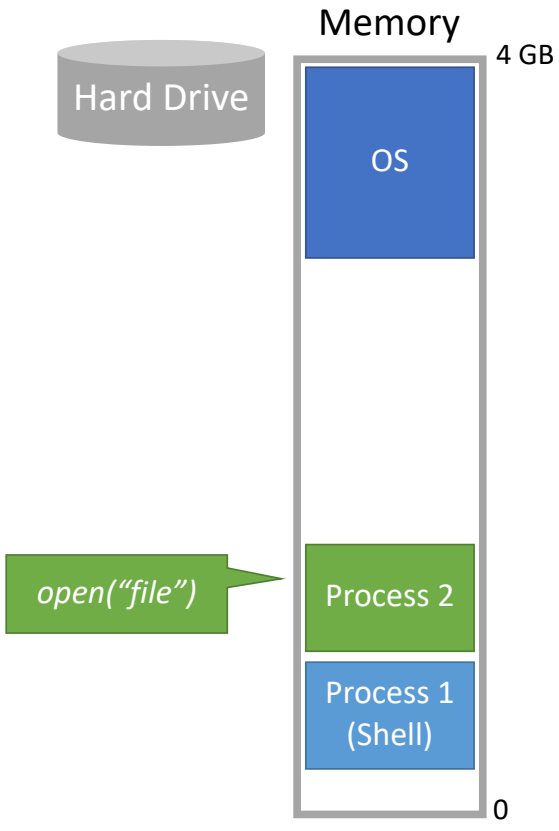
Restriction on Privileged Instructions

What CPU instructions are restricted in protected mode?

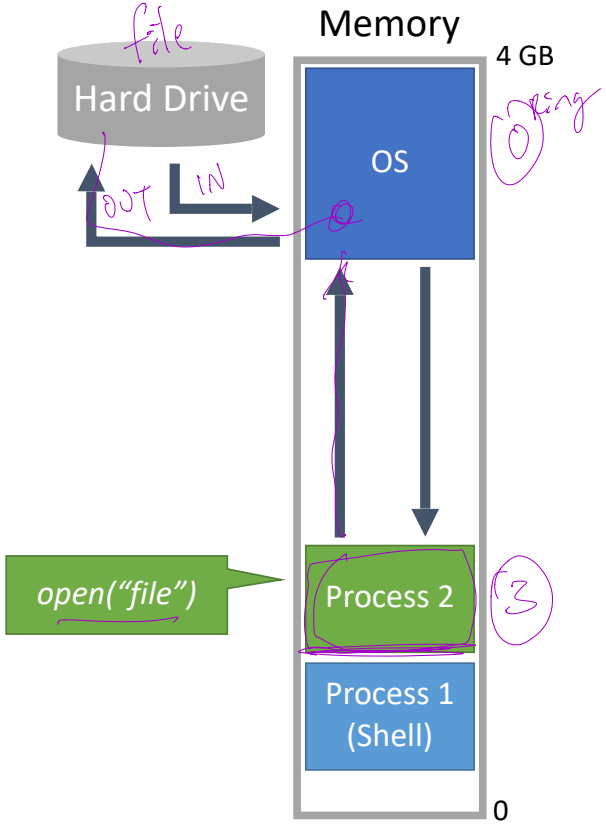
- Any instruction that modifies the CR0 register
  - Controls whether protected mode is enabled
- Any instruction that modifies the CR3 register
  - Controls the virtual memory configuration
  - More on this later…
- hlt – Halts the CPU
- sti/cli – enable and disable interrupts
- in/out – directly access hardware devices

If a Ring 3 process tries any of these things, it immediately crashes

How to change modes



How to change modes



Changing Modes

Applications often need to access the OS APIs

- Writing files
- Displaying things on the screen
- Receiving data from the network
- etc…

But the OS is Ring 0, and processes are Ring 3

How do processes get access to the OS?

Interrupt handlers

Changing Modes

Applications often need to access the OS APIs

- Writing files
- Displaying things on the screen
- Receiving data from the network
- etc…

But the OS is Ring 0, and processes are Ring 3

How do processes get access to the OS?

- Invoke OS APIs with special assembly instructions
  - Interrupt: int 0x80
  - System call: sysenter or syscall
- int/sysenter/syscall cause a mode transfer from Ring 3 to Ring 0

Mode Transfer

Userland

1. Application executes trap (int) instruction
   - EIP, CS, and EFLAGS get pushed onto the stack
   - Mode switches from ring 3 to ring 0

Kernel Mode

Mode Transfer

Userland

1. Application executes trap (int) instruction
   - EIP, CS, and EFLAGS get pushed onto the stack
   - Mode switches from ring 3 to ring 0

Kernel Mode

2. Save the state of the current process
   - Push EAX, EBX, …, etc. onto the stack

Mode Transfer

Userland

1. Application executes trap (int) instruction
   - EIP, CS, and EFLAGS get pushed onto the stack
   - Mode switches from ring 3 to ring 0

Kernel Mode

2. Save the state of the current process
   - Push EAX, EBX, …, etc. onto the stack
3. Locate and execute the correct syscall handler

Mode Transfer

Userland

1. Application executes trap (int) instruction
   - EIP, CS, and EFLAGS get pushed onto the stack
   - Mode switches from ring 3 to ring 0

Kernel Mode

2. Save the state of the current process
   - Push EAX, EBX, …, etc. onto the stack
3. Locate and execute the correct syscall handler
4. Restore the state of process
   - Pop EAX, EBX, … etc.

Mode Transfer

Userland

1. Application executes trap (int) instruction
   - EIP, CS, and EFLAGS get pushed onto the stack
   - Mode switches from ring 3 to ring 0

Kernel Mode

2. Save the state of the current process
   - Push EAX, EBX, …, etc. onto the stack
3. Locate and execute the correct syscall handler
4. Restore the state of process
   - Pop EAX, EBX, … etc.
5. Place the return value in EAX

Mode Transfer

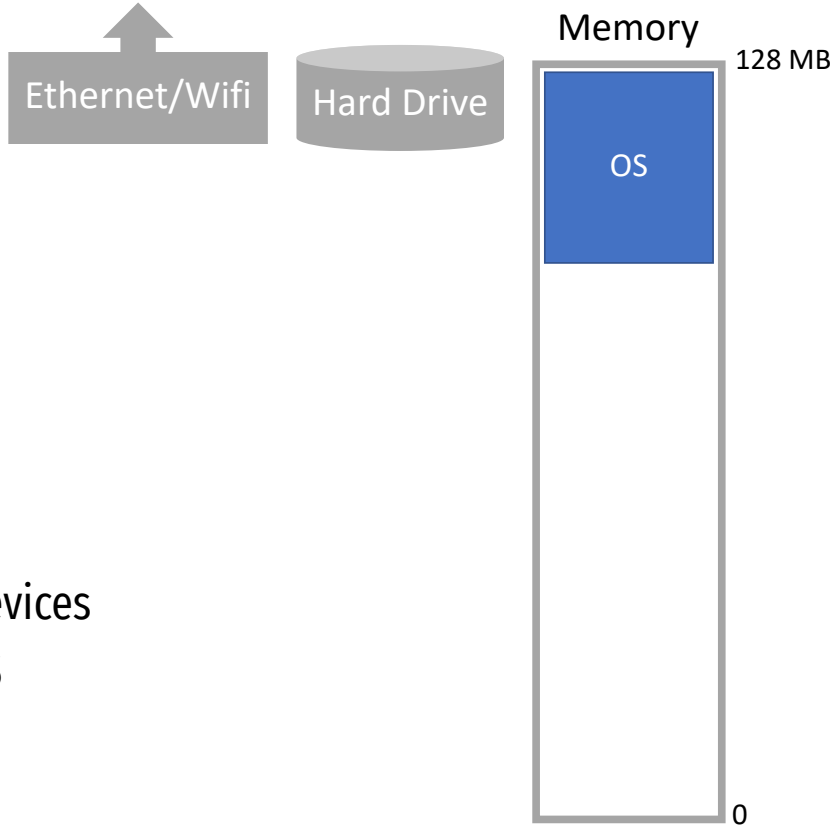
Userland

1. Application executes trap (int) instruction
   - EIP, CS, and EFLAGS get pushed onto the stack
   - Mode switches from ring 3 to ring 0

Kernel Mode

2. Save the state of the current process
   - Push EAX, EBX, …, etc. onto the stack
3. Locate and execute the correct syscall handler
4. Restore the state of process
   - Pop EAX, EBX, … etc.
5. Place the return value in EAX
6. Use iret to return to the process
   - Switches back to the original mode (typically 3)

Protection in Action

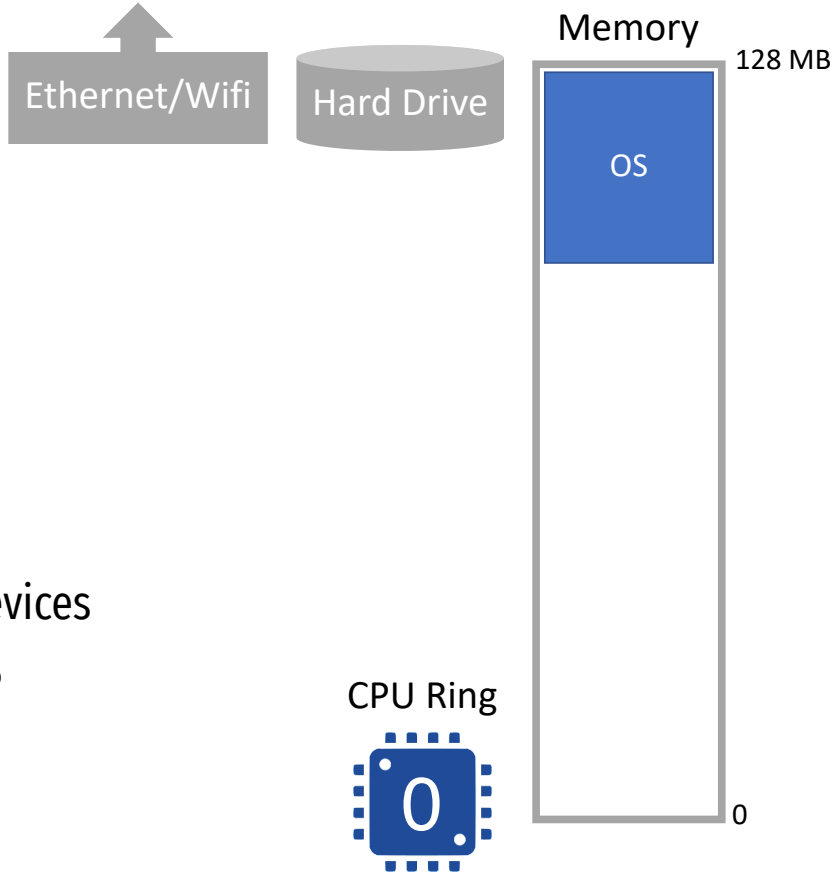


Protected mode stops direct access to devices

All device access must go through the OS

OS will impose access control checks

Protection in Action

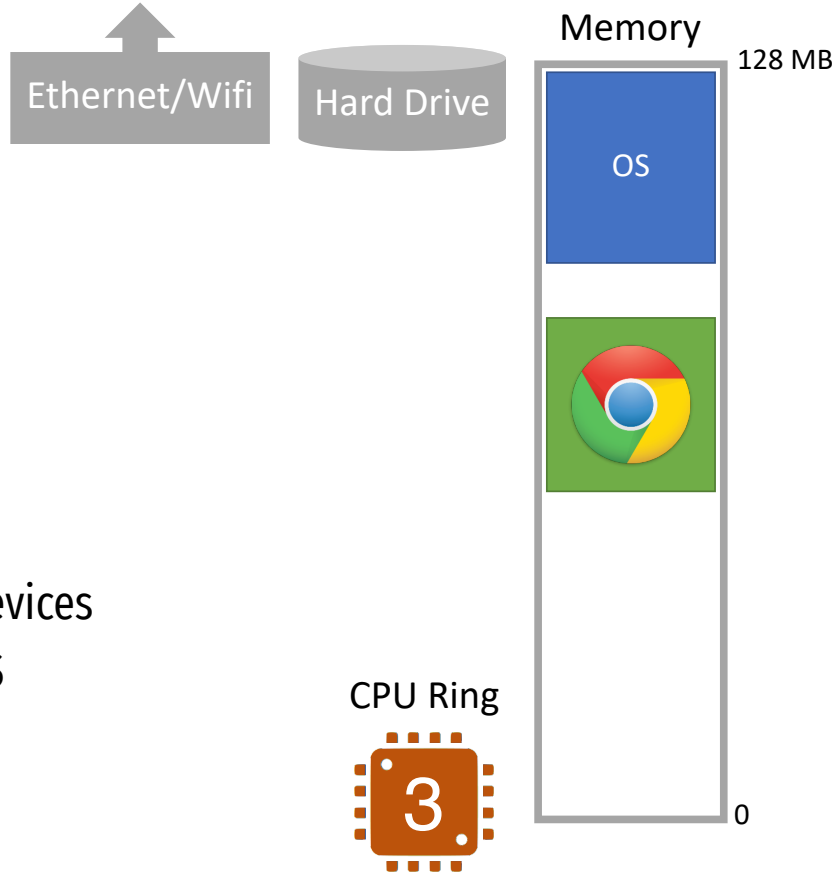


Protected mode stops direct access to devices

All device access must go through the OS

OS will impose access control checks

Protection in Action

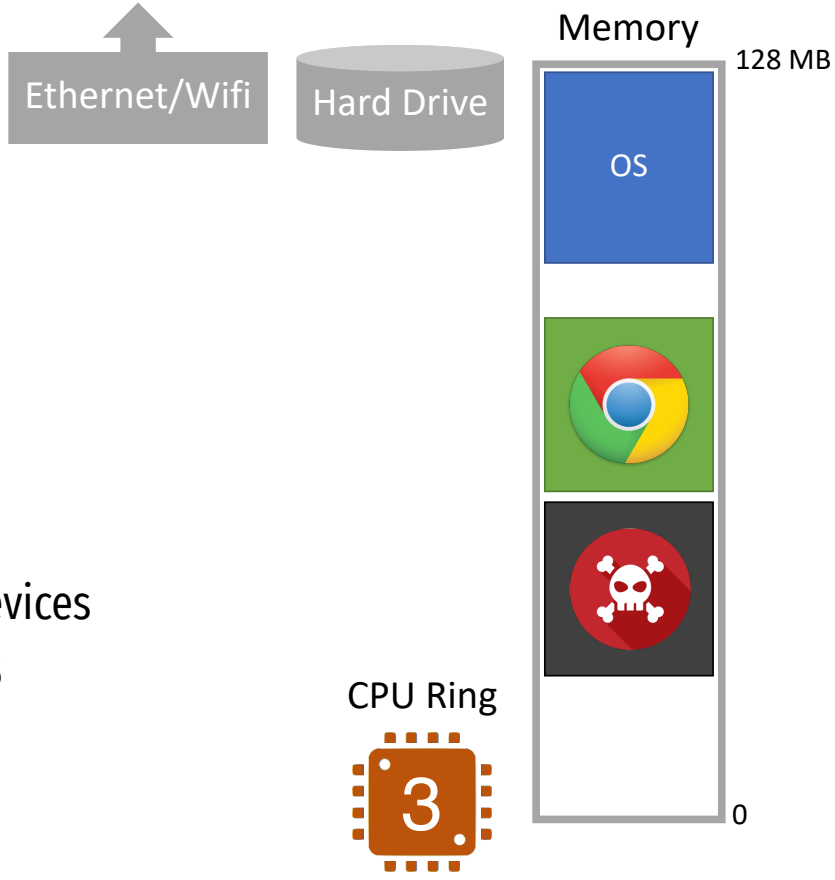


Protected mode stops direct access to devices

All device access must go through the OS

OS will impose access control checks

Protection in Action

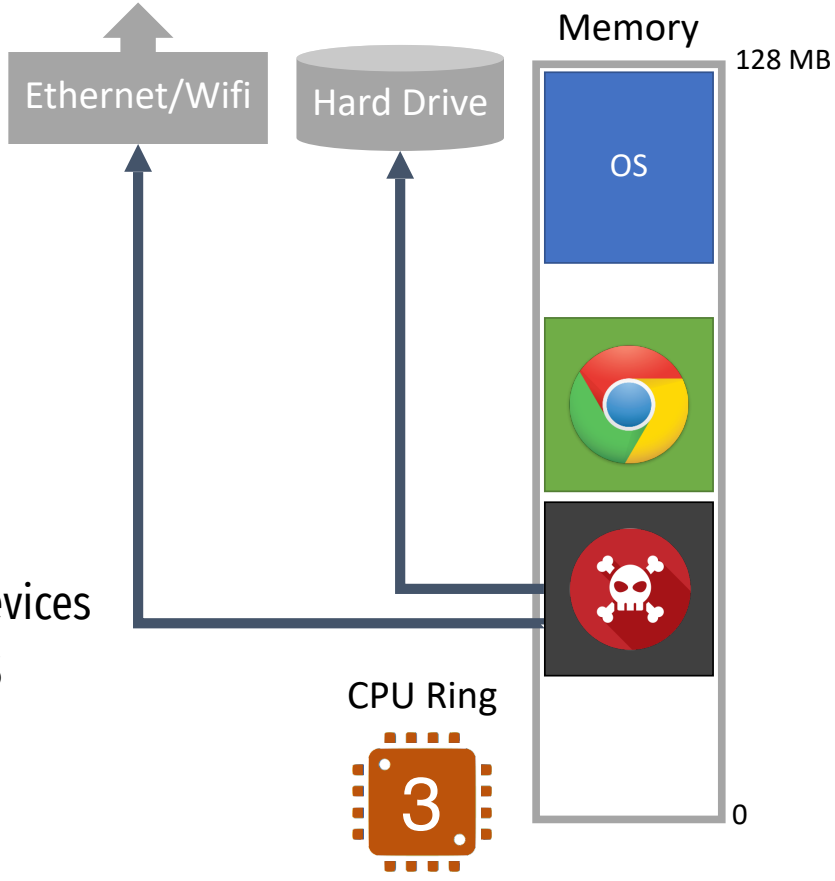


Protected mode stops direct access to devices

All device access must go through the OS

OS will impose access control checks

Protection in Action

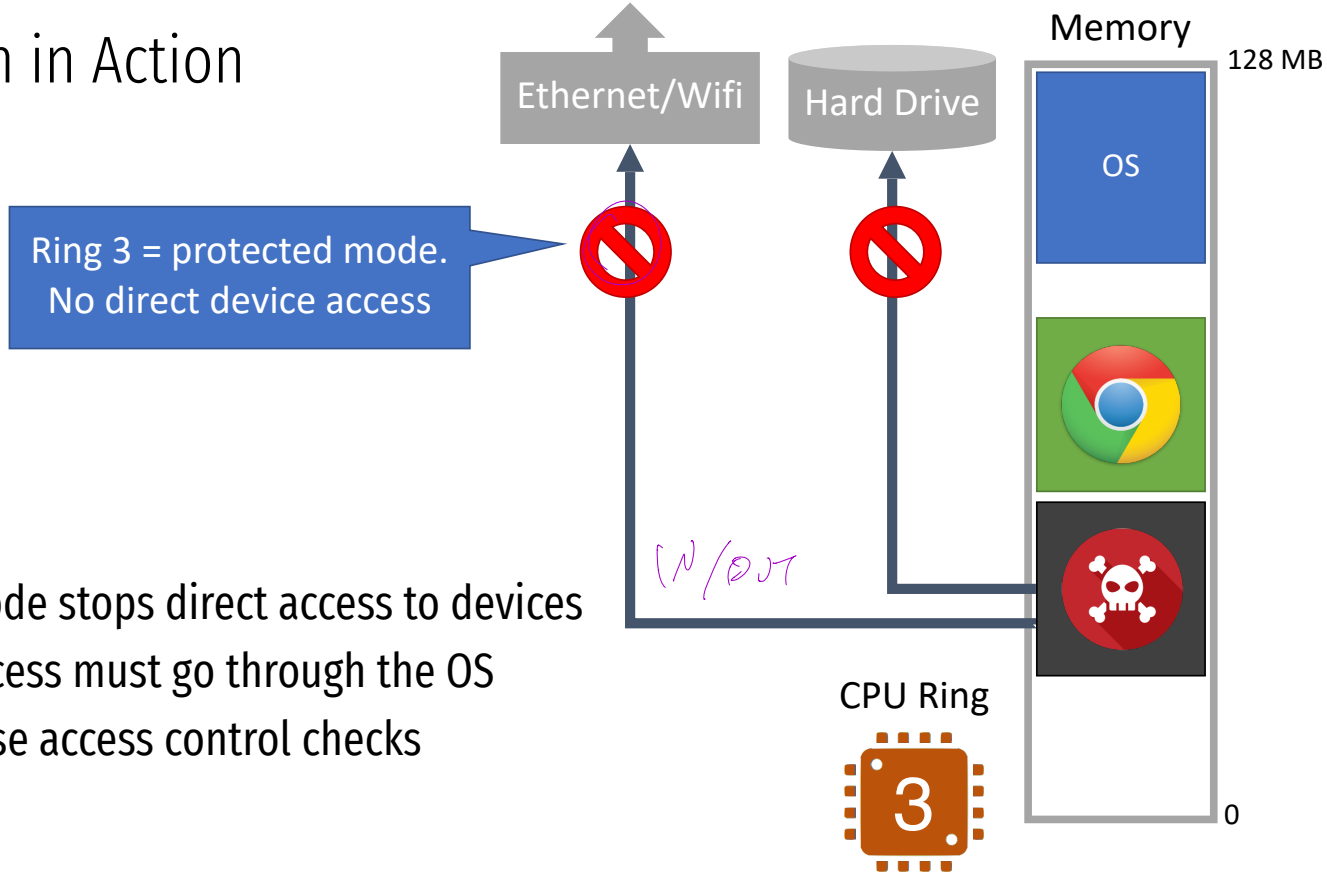


Protected mode stops direct access to devices

All device access must go through the OS

OS will impose access control checks

Protection in Action

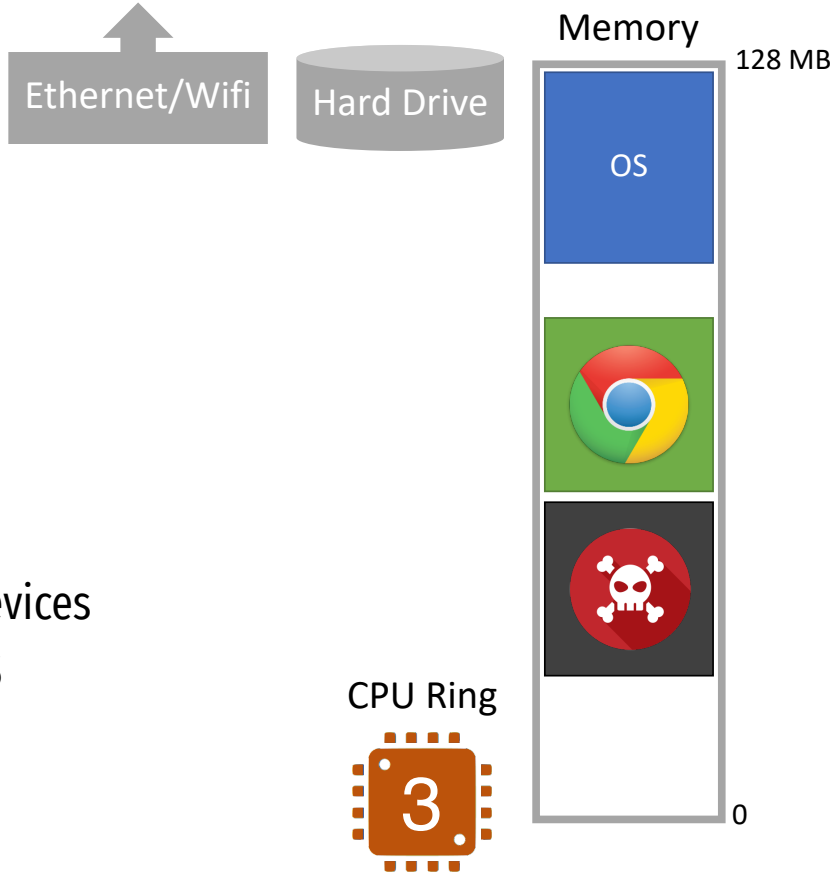


Protected mode stops direct access to devices

All device access must go through the OS

OS will impose access control checks

Protection in Action

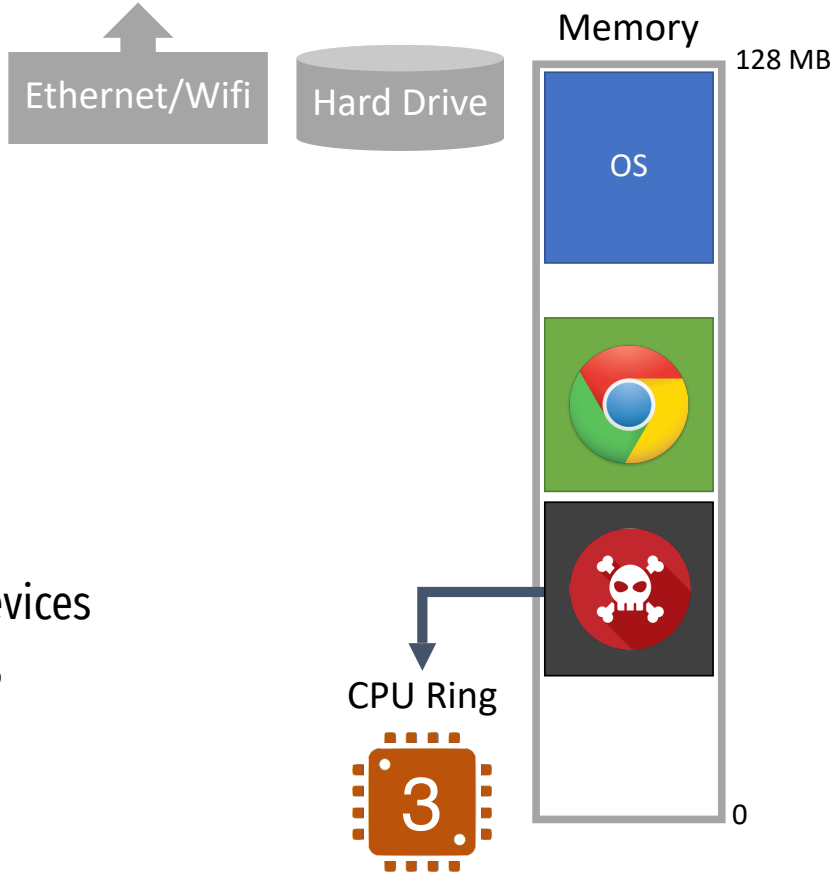


Protected mode stops direct access to devices

All device access must go through the OS

OS will impose access control checks

Protection in Action

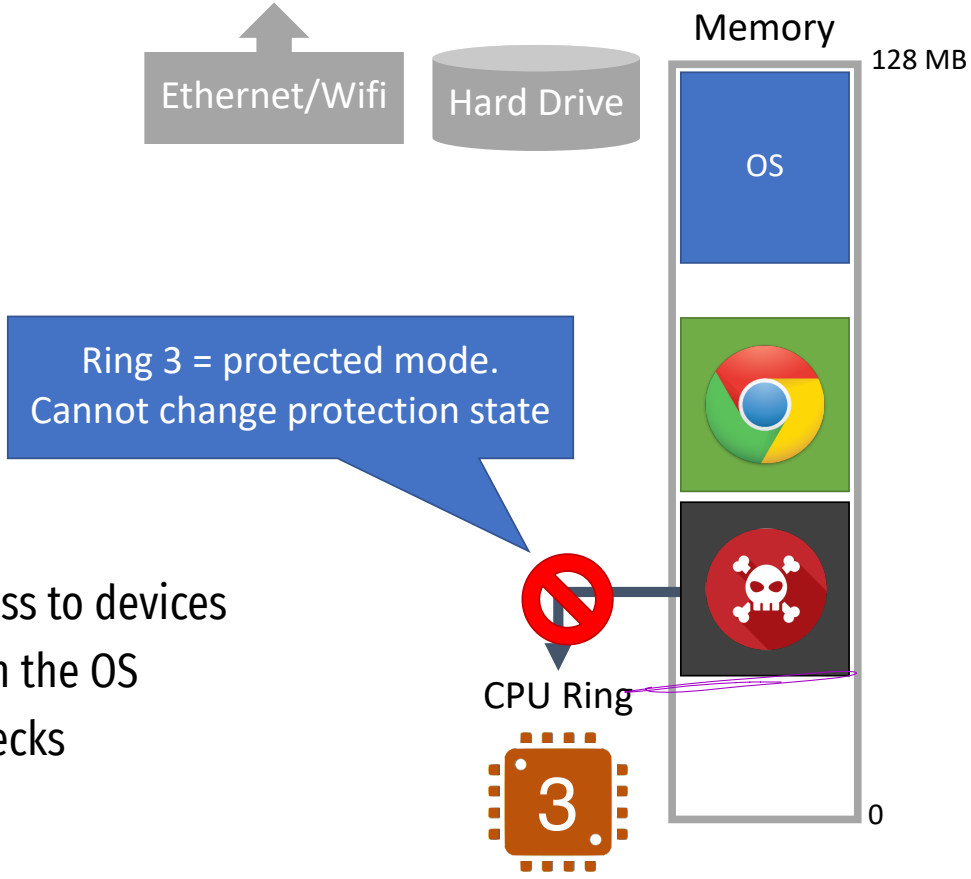


Protected mode stops direct access to devices

All device access must go through the OS

OS will impose access control checks

Protection in Action

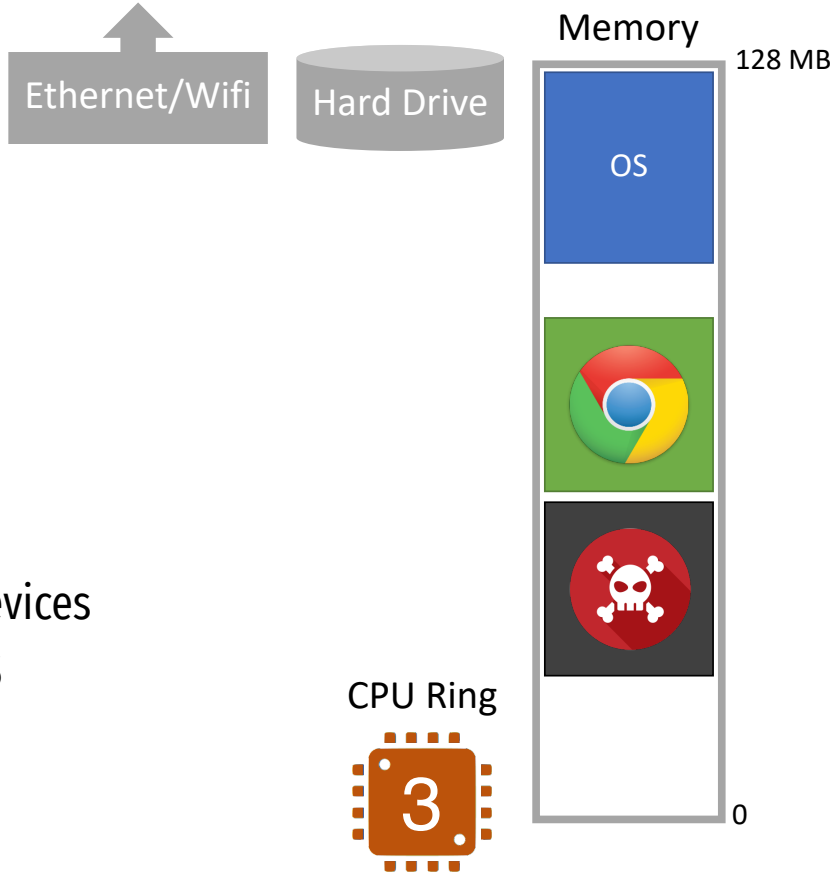


Protected mode stops direct access to devices

All device access must go through the OS

OS will impose access control checks

Protection in Action

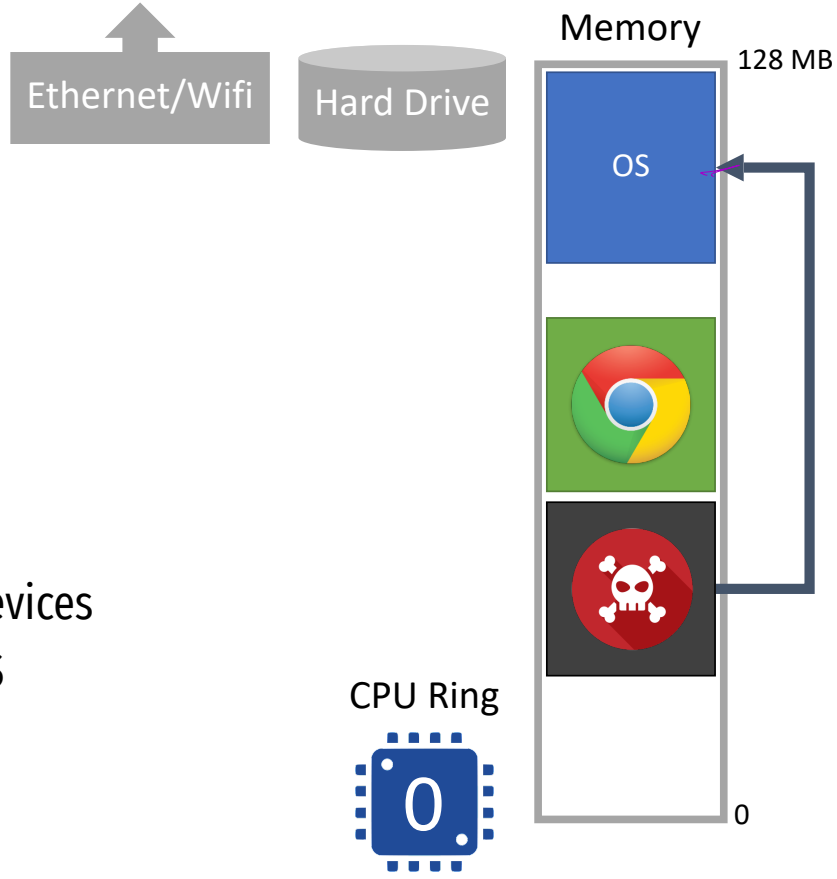


Protected mode stops direct access to devices

All device access must go through the OS

OS will impose access control checks

Protection in Action

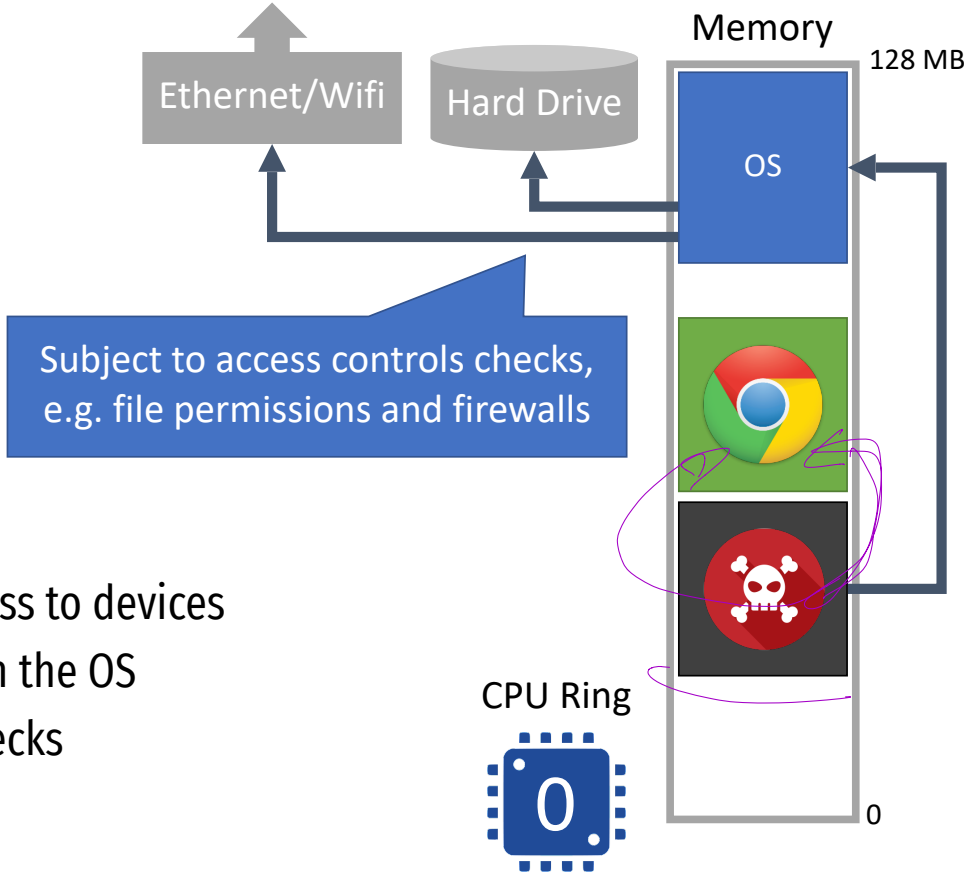


Protected mode stops direct access to devices

All device access must go through the OS

OS will impose access control checks

Protection in Action



Protected mode stops direct access to devices

All device access must go through the OS

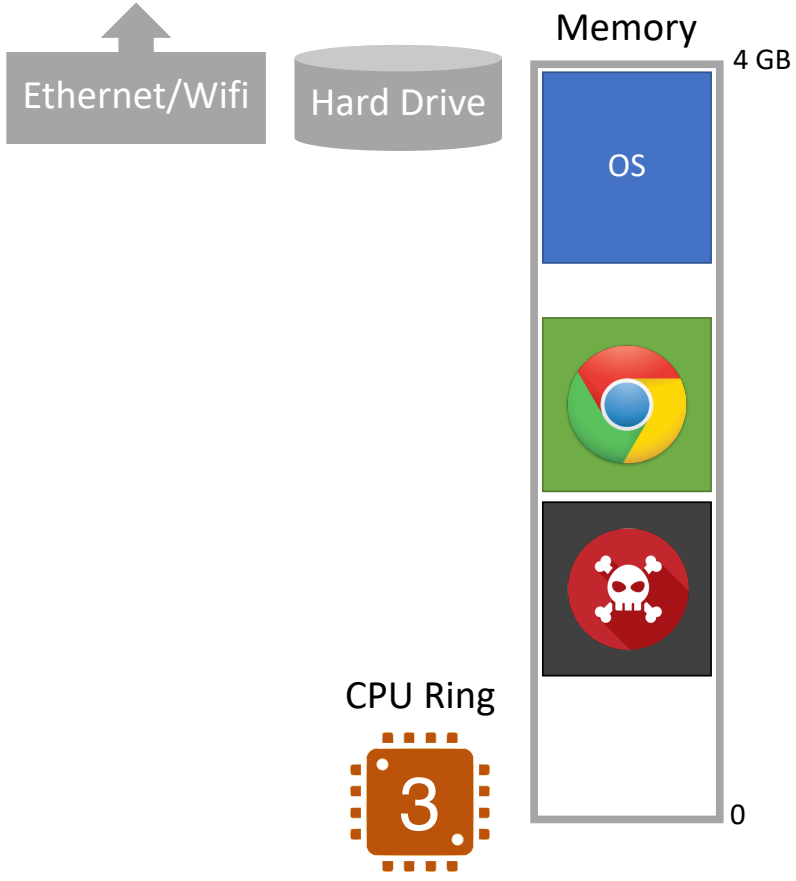
OS will impose access control checks

Virtual Memory

Status Check

At this point we have protected the devices attached to the system…

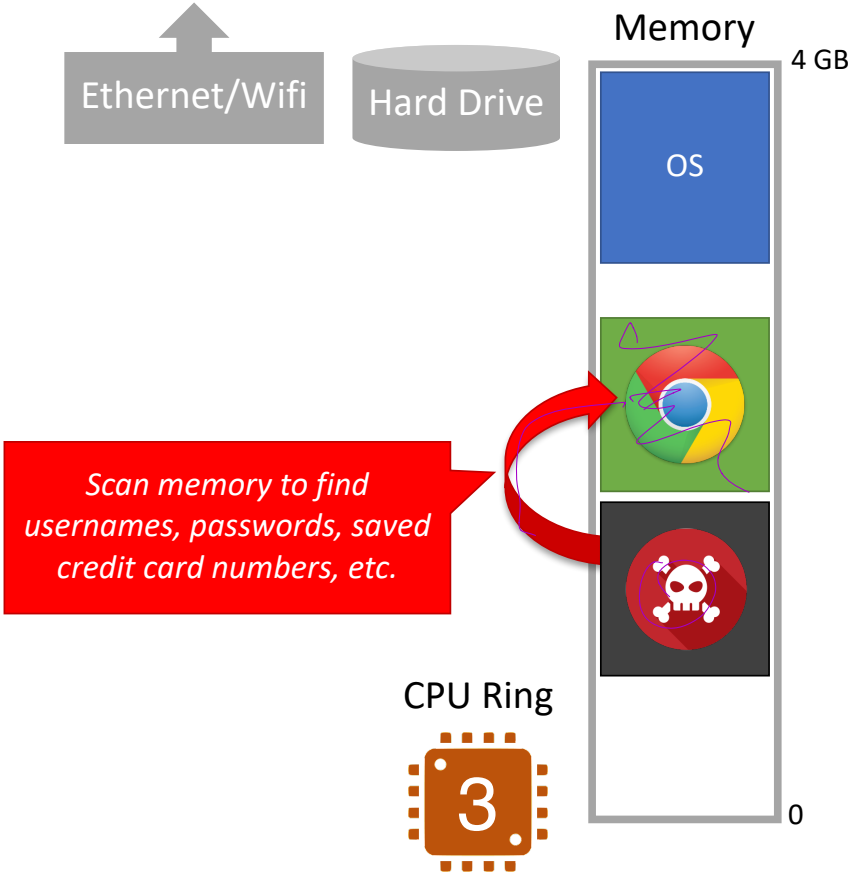
… But we have not protected memory



Status Check

At this point we have protected the devices attached to the system…

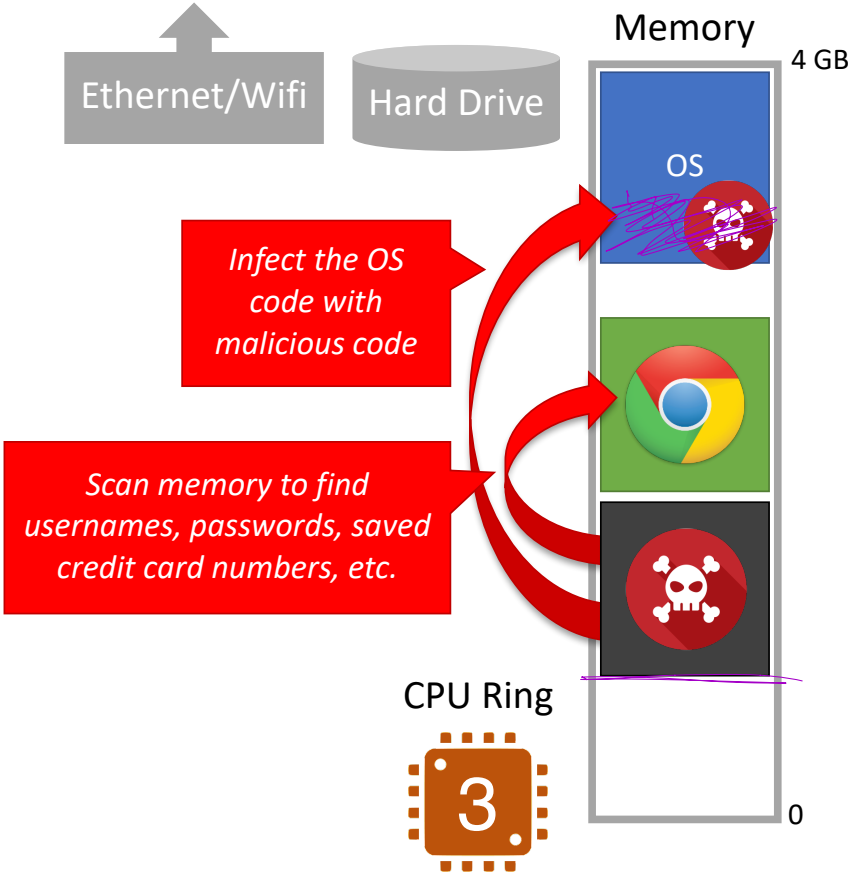
… But we have not protected memory



Status Check

At this point we have protected the devices attached to the system…

… But we have not protected memory



Memory Isolation and Virtual Memory

Modern CPUs support virtual memory

Creates the illusion that each process runs in its own, empty memory space

- Processes can not read/write memory used by other processes
- Processes can not read/write memory used by the OS

Memory Isolation and Virtual Memory

Modern CPUs support virtual memory

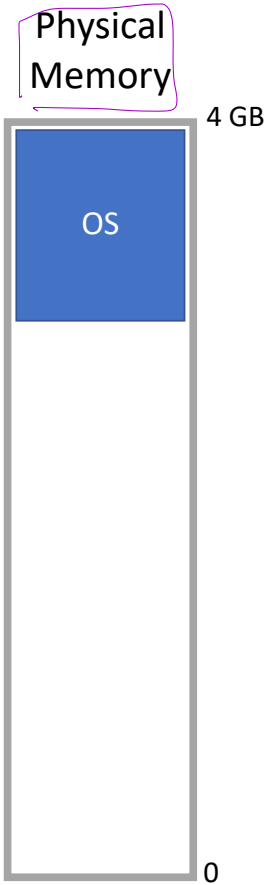
Creates the illusion that each process runs in its own, empty memory space

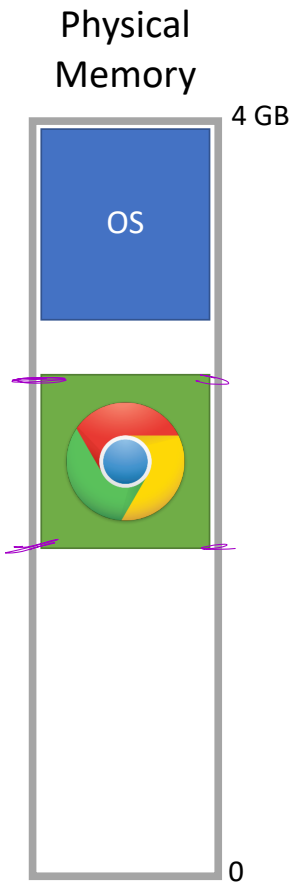
- Processes can not read/write memory used by other processes
- Processes can not read/write memory used by the OS

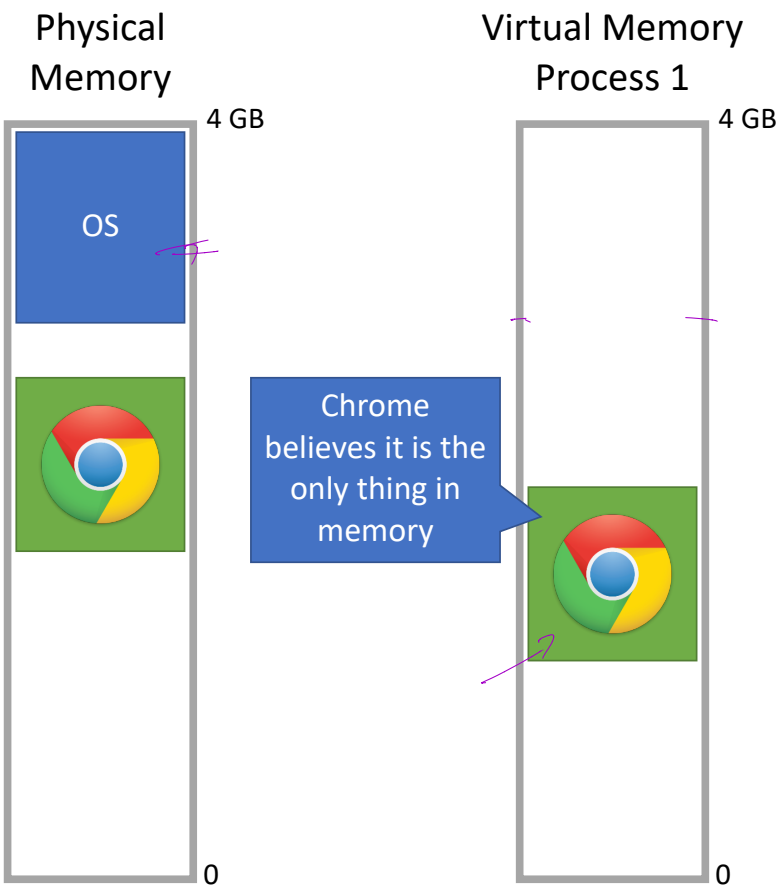
In later courses, you will learn how virtual memory is implemented

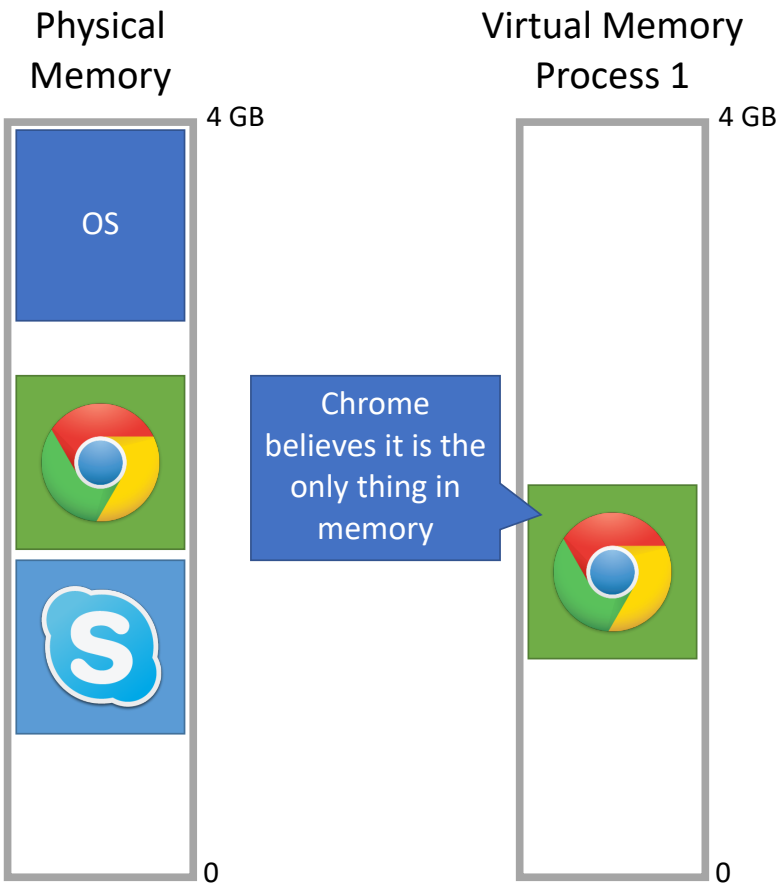
- Base and bound registers
- Segmentation
- Page tables

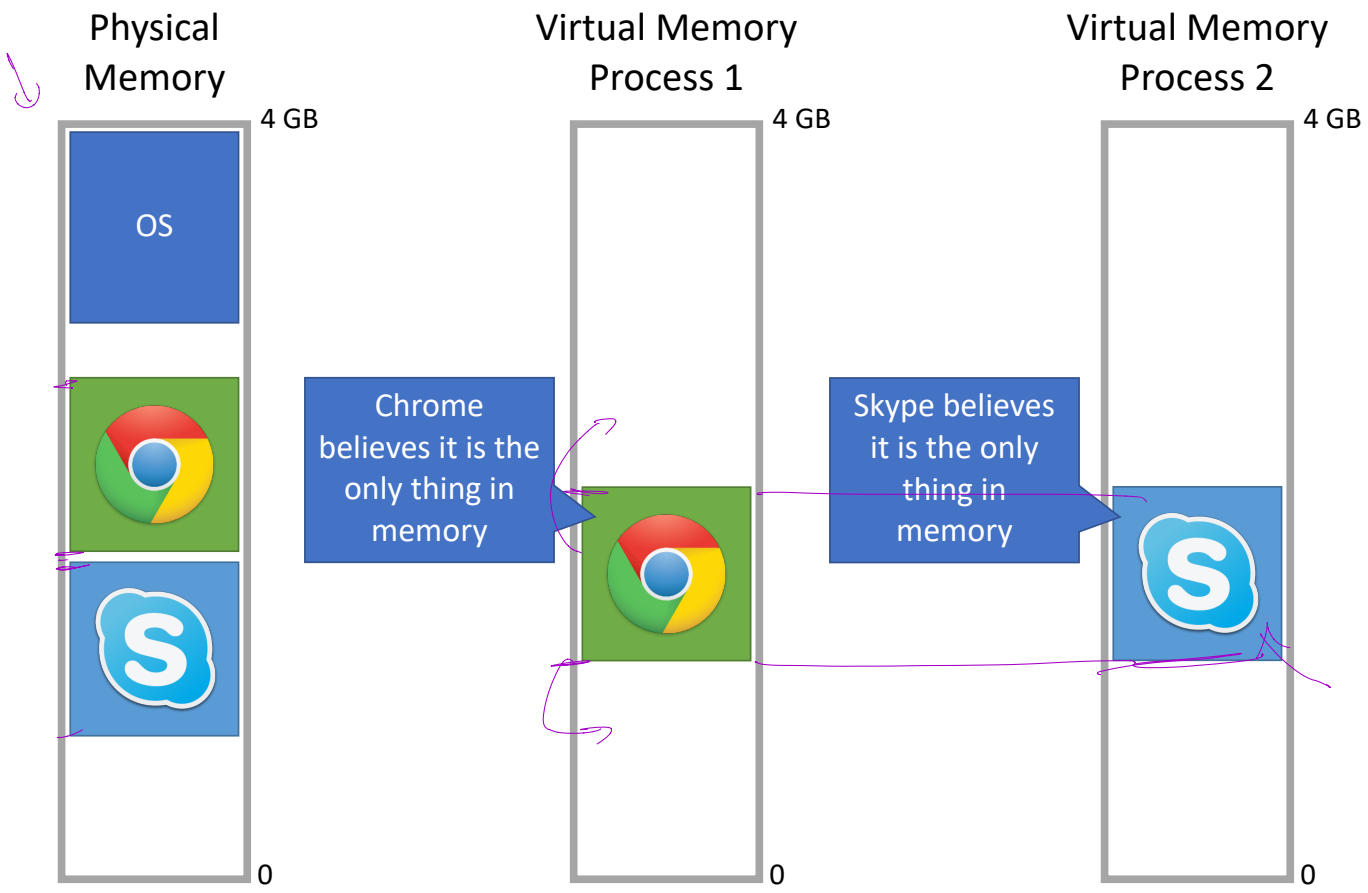
Today, we will do the cliffnotes version…


Physical Memory
4 GB
OS
0

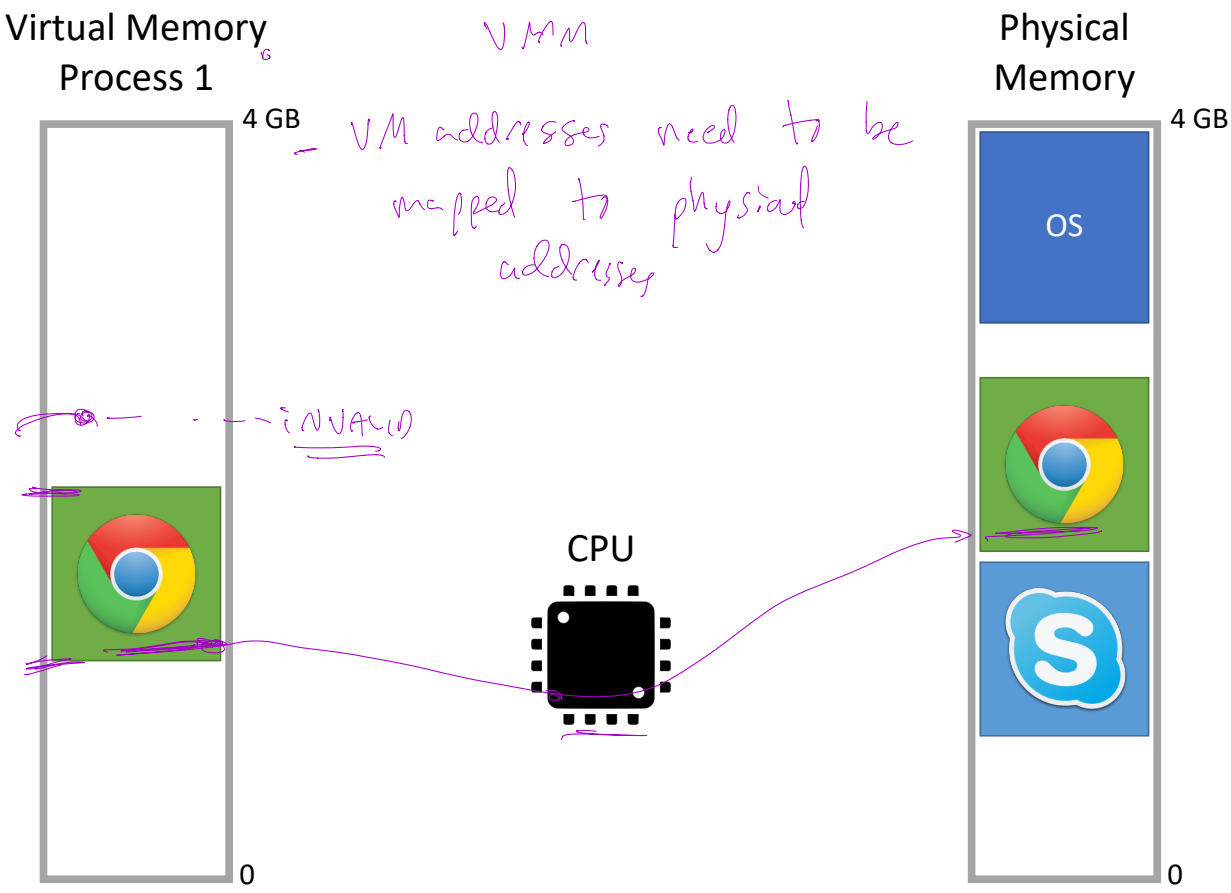

Physical
Memory
4 GB
OS
0


Physical Memory
4 GB
OS
0
Virtual Memory Process 1
4 GB
Chrome believes it is the only thing in memory
0


Physical Memory
4 GB
OS
0
Virtual Memory Process 1
4 GB
Chrome believes it is the only thing in memory
0

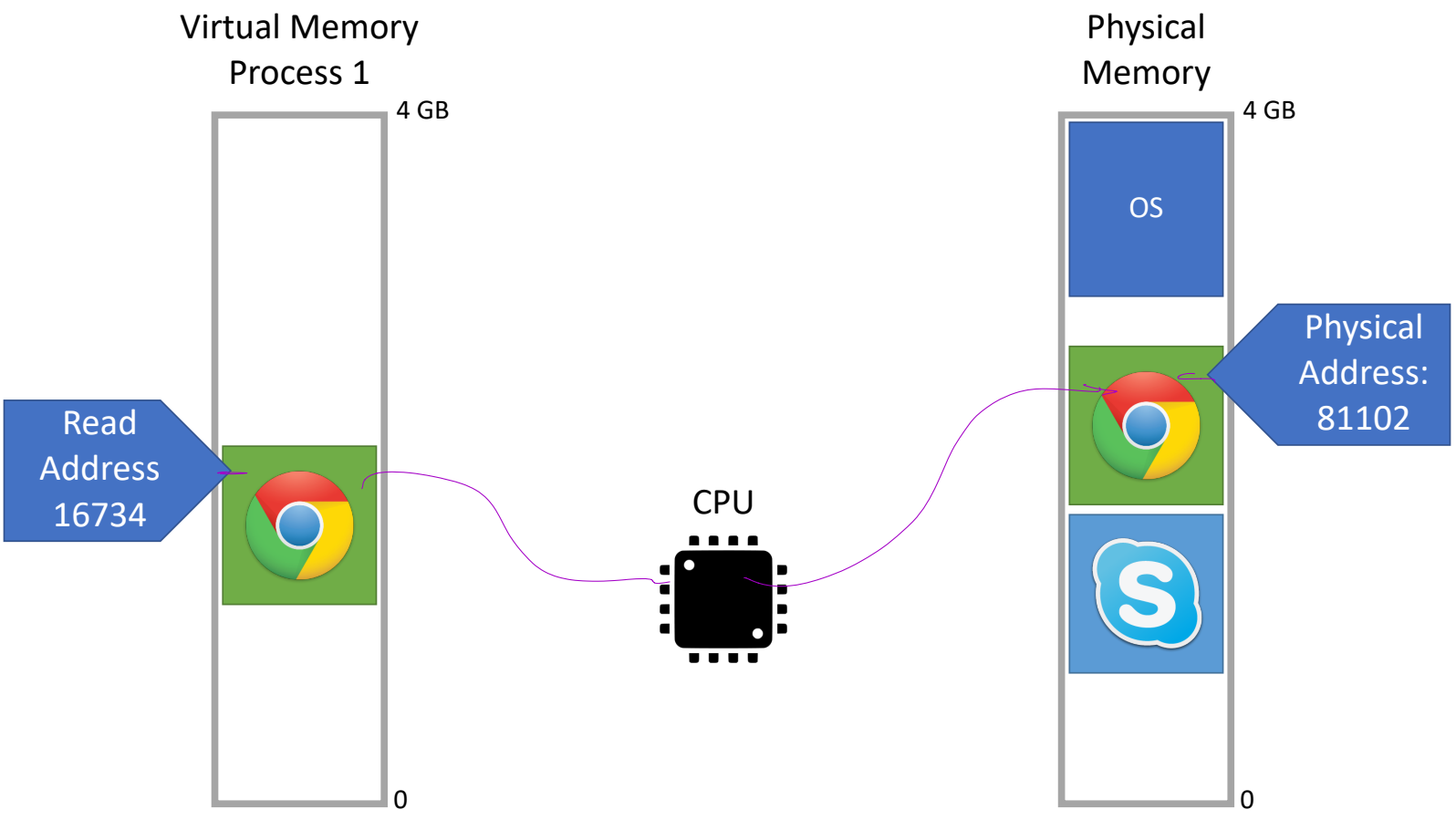

Physical Memory
4 GB
OS
0
Chrome believes it is the only thing in memory
Virtual Memory Process 1
4 GB
0
Skype believes it is the only thing in memory
Virtual Memory Process 2
4 GB
0

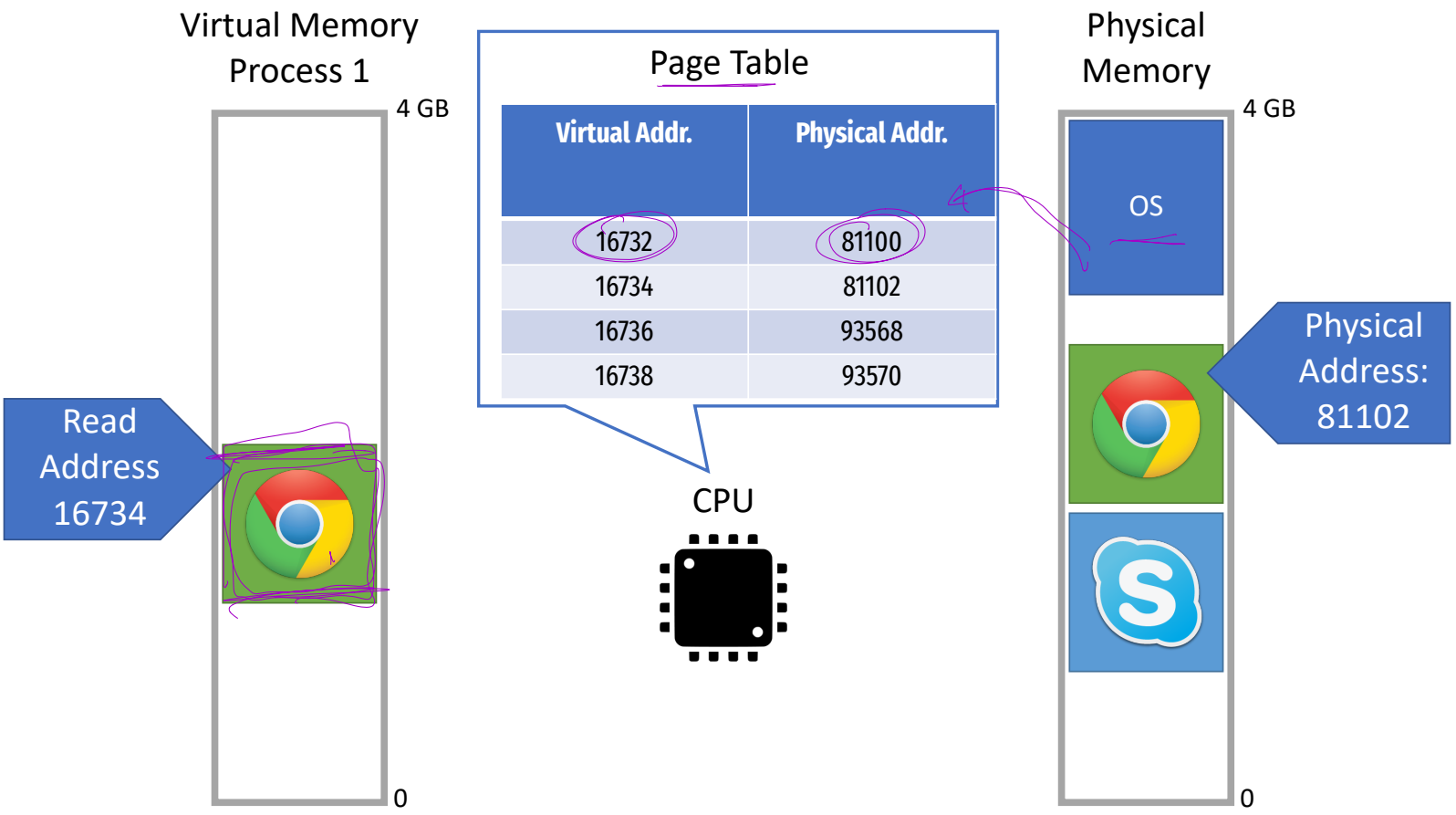
VMM

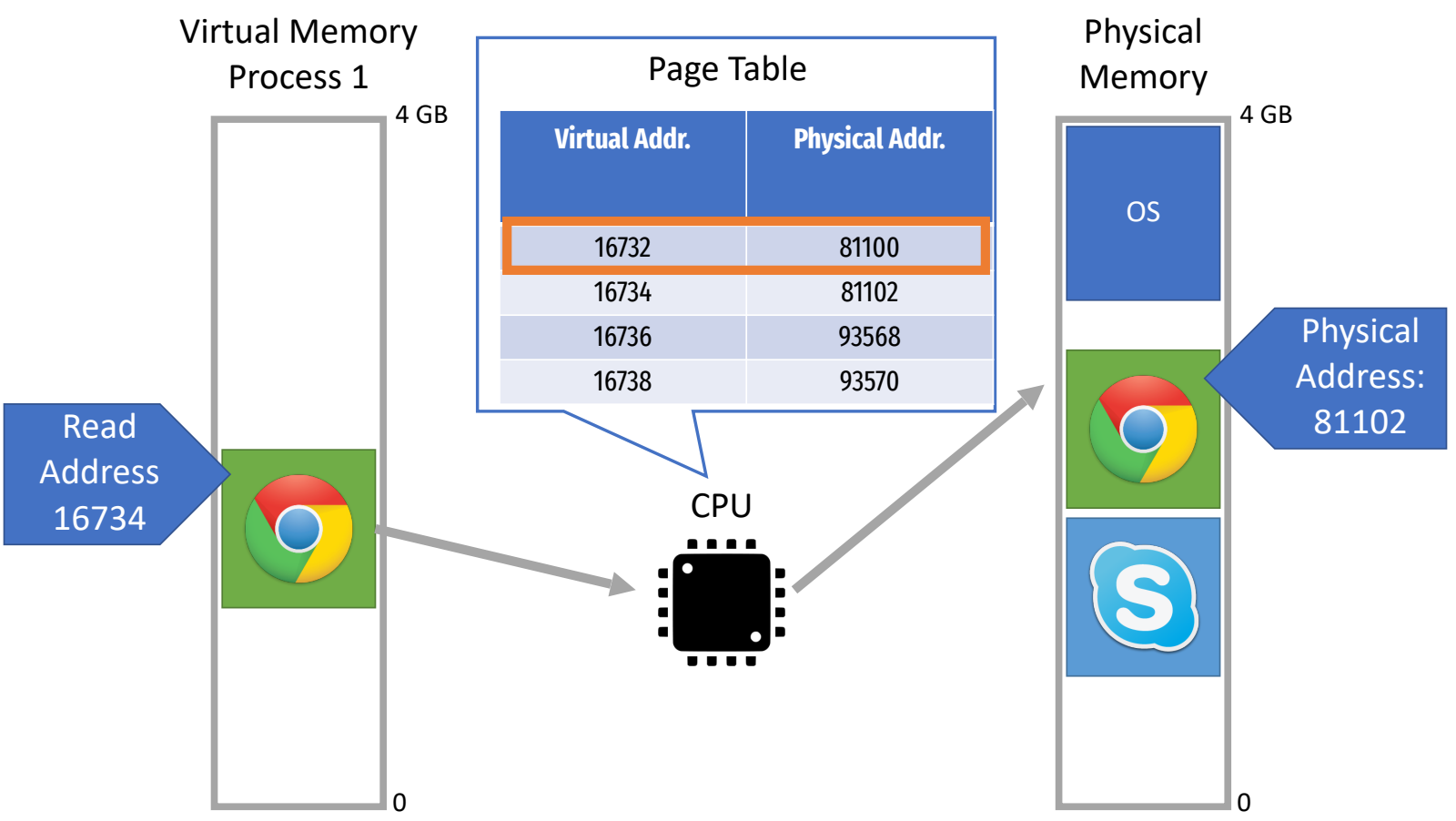


- VM addresses need to be mapped to physical addresses

INVALID


Virtual Memory
Process 1
4 GB
0
Read
Address
16734
CPU
Physical
Memory
4 GB
0
OS
Physical
Address:
81102


Virtual Memory
Process 1
4 GB
0
Read
Address
16734
Page Table
Virtual Addr.
Physical Addr.
16732
81100
16734
81102
16736
93568
16738
93570
CPU
Physical
Memory
4 GB
0
OS
Physical
Address:
81102


Virtual Memory
Process 1
4 GB
0
Read
Address
16734
Page Table
Virtual Addr.
Physical Addr.
16732
81100
16734
81102
16736
93568
16738
93570
CPU
Physical
Memory
4 GB
0
OS
Physical
Address:
81102

Virtual Memory Implementation

Each process has its own virtual memory space

- Each process has a page table that maps is virtual space into physical space
- CPU translates virtual address to physical addresses on-the-fly

- OS manages the page table to ensure process isolation.

Virtual Memory Implementation

Each process has its own virtual memory space

- Each process has a page table that maps is virtual space into physical space
- CPU translates virtual address to physical addresses on-the-fly

OS creates the page table for each process

- Installing page tables in the CPU is a protected, Ring 0 instruction
- Processes cannot modify their page tables

Virtual Memory Implementation

Each process has its own virtual memory space

- Each process has a page table that maps is virtual space into physical space
- CPU translates virtual address to physical addresses on-the-fly

OS creates the page table for each process

- Installing page tables in the CPU is a protected, Ring 0 instruction
- Processes cannot modify their page tables

What happens if a process tries to read/write memory outside its page table?

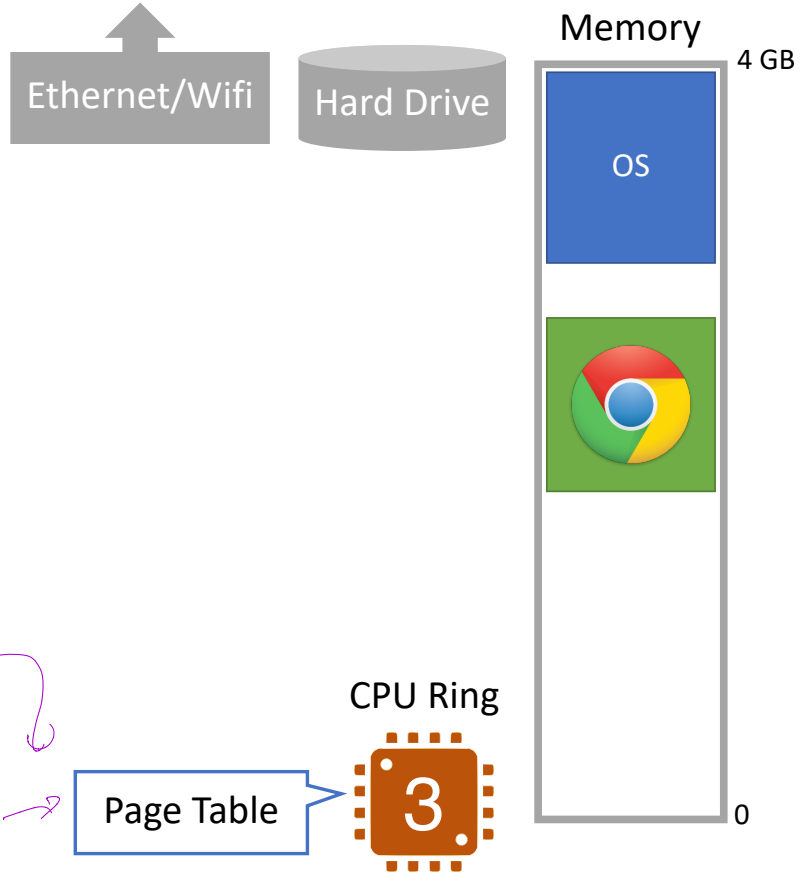
- Segmentation Fault or Page Fault
- Process crashes
- In other words, no way to escape virtual memory

VM in Action

Processes can only read/ write within their own virtual memory

Processes cannot change their own page tables

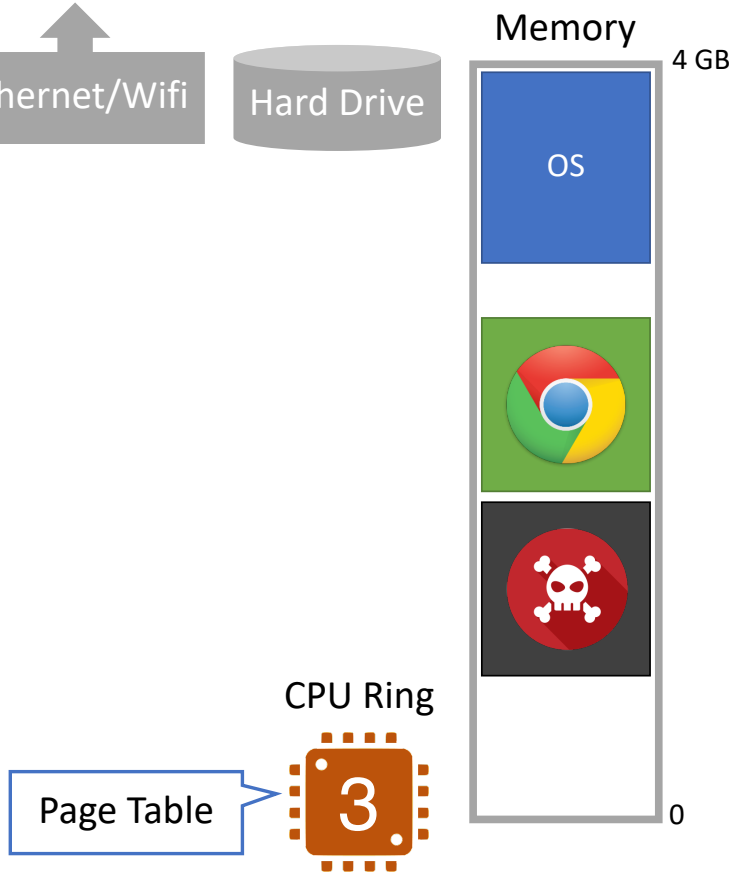
Only Ring 0 can change the page table



VM in Action

Processes can only read/ write within their own virtual memory

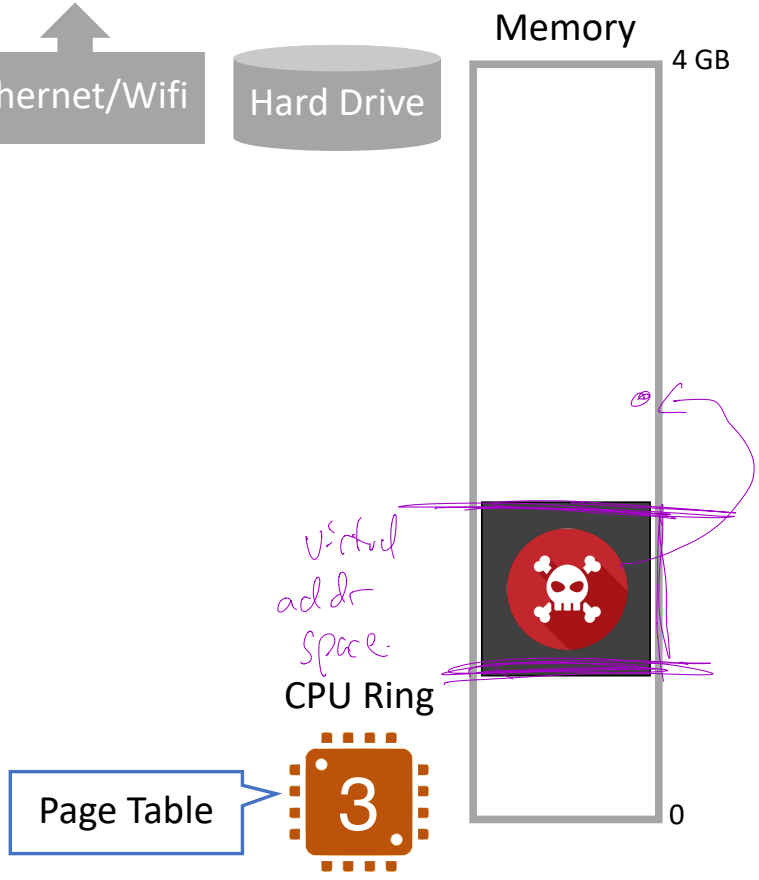
Processes cannot change their own page tables



VM in Action

Processes can only read/ write within their own virtual memory

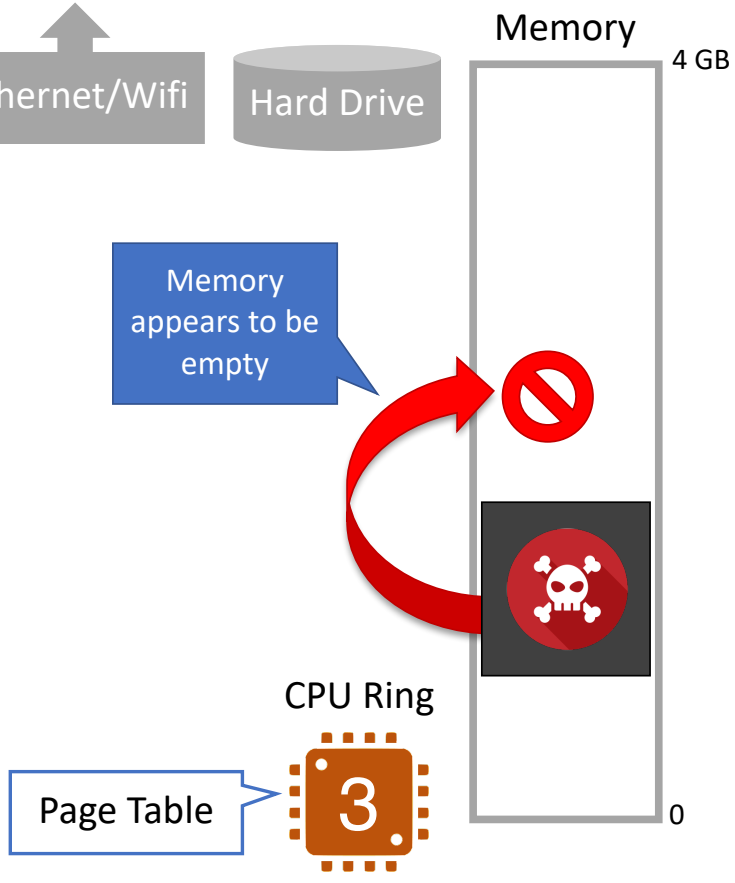
Processes cannot change their own page tables



VM in Action

Processes can only read/ write within their own virtual memory

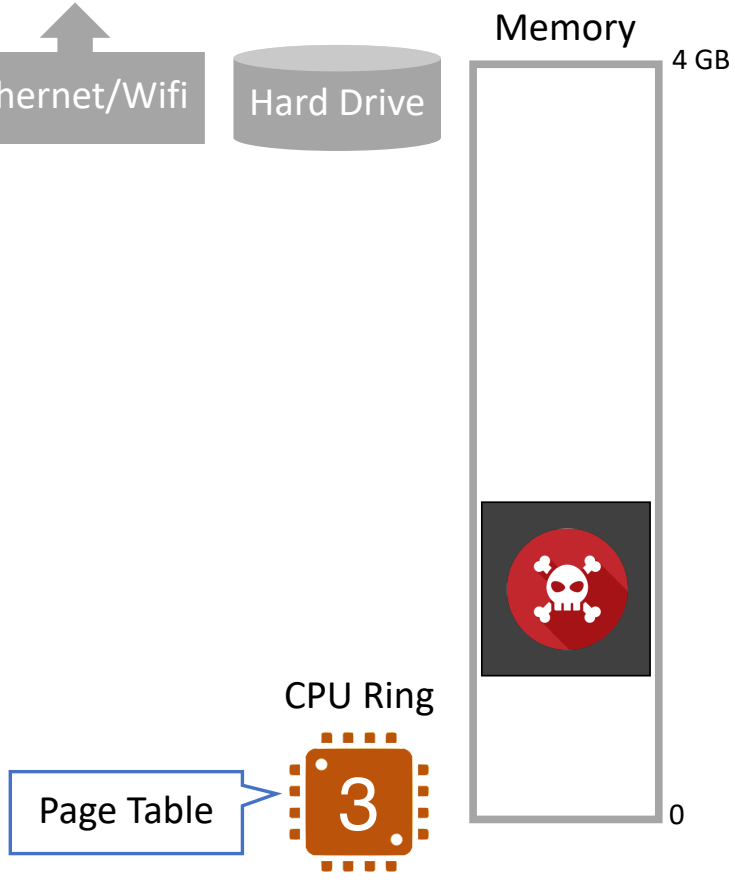
Processes cannot change their own page tables



VM in Action

Processes can only read/ write within their own virtual memory

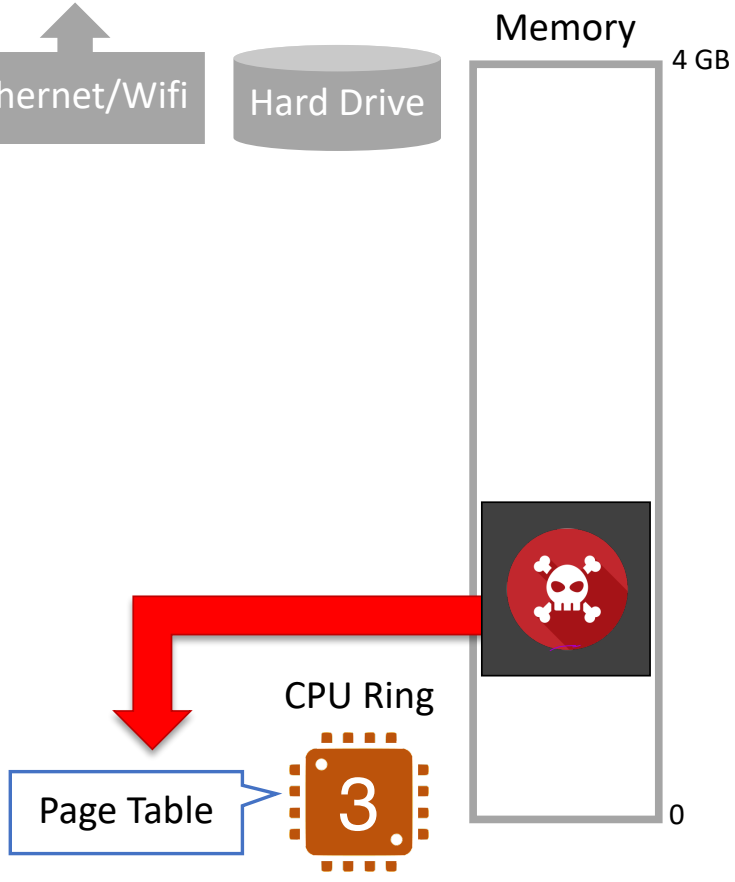
Processes cannot change their own page tables



VM in Action

Processes can only read/ write within their own virtual memory

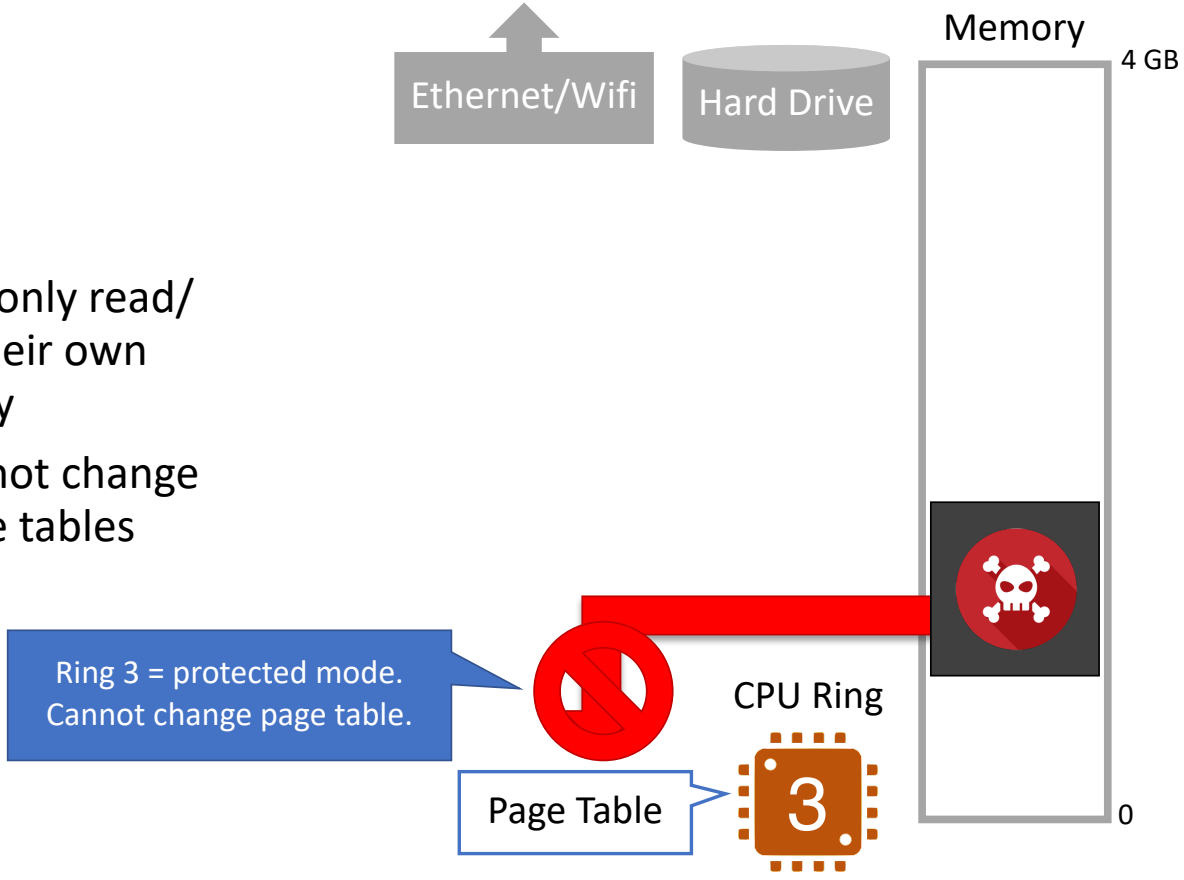
Processes cannot change their own page tables



VM in Action

Processes can only read/ write within their own virtual memory

Processes cannot change their own page tables



Threat Model

Intro to System Architecture

Hardware Support for Isolation

Examples

Principles

Review

At this point, we have achieved process isolation

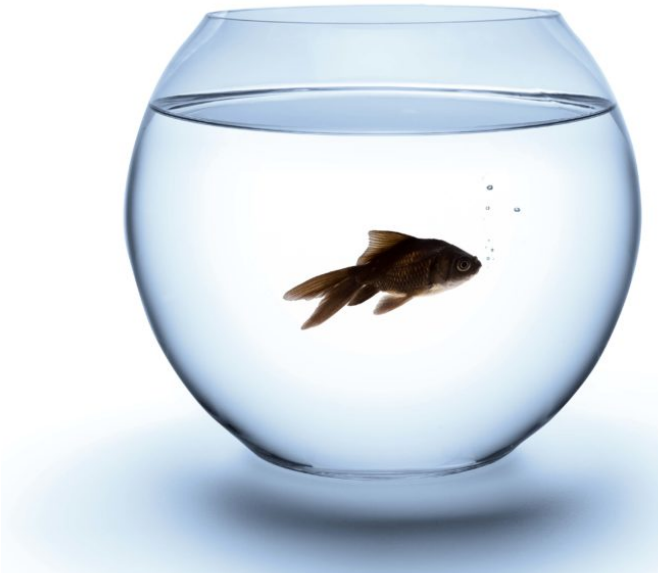
- Protected mode execution prevents direct device access
- Virtual memory prevents direct memory access

Requires CPU support

- All moderns CPUs support these techniques

Requires OS support

- All moderns OS support these techniques
- OS controls process rings and page tables



Review

At this point, we have achieved process isolation

- Protected mode execution prevents direct device access
- Virtual memory prevents direct memory access

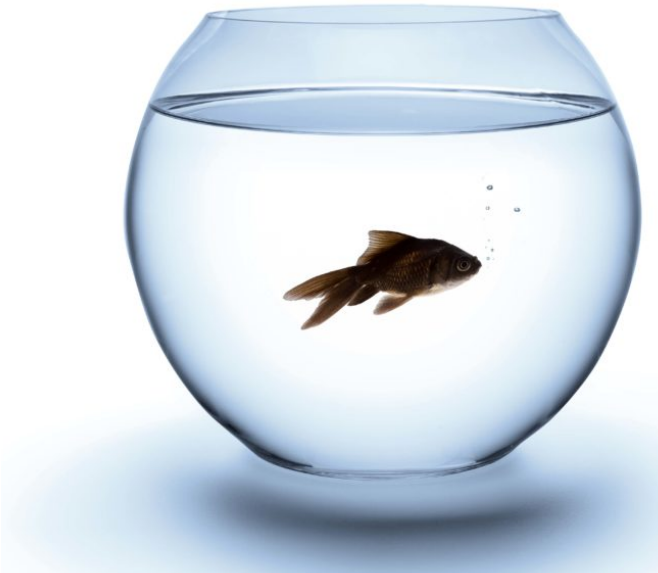
Requires CPU support

- All moderns CPUs support these techniques

Requires OS support

- All moderns OS support these techniques
- OS controls process rings and page tables

Warning: bugs in the OS may compromise process isolation



Processes on a linux machine

ps -ax

```
PID TTY      STAT   TIME COMMAND
  1 ?        Ss     0:02 /sbin/init splash
  2 ?        S      0:00 [kthreadd]
  3 ?        I<     0:00 [rcu_gp]
  4 ?        I<     0:00 [rcu_par_gp]
  6 ?        I<     0:00 [kworker/0:0H-kblockd]
  7 ?        I      0:00 [kworker/0:1-events]
  9 ?        I<     0:00 [mm_percpu_wq]
 10 ?        S      0:00 [ksoftirqd/0]
 11 ?        I      0:00 [rcu_sched]
 12 ?        S      0:00 [migration/0]
 13 ?        S      0:00 [idle_inject/0]
 14 ?        S      0:00 [cpuhp/0]
 15 ?        S      0:00 [kdevtmpfs]
 16 ?        I<     0:00 [netns]
 17 ?        S      0:00 [rcu_tasks_kthre]
 18 ?        S      0:00 [kauditd]
 19 ?        S      0:00 [khungtaskd]
 20 ?        S      0:00 [oom_reaper]
 21 ?        I<     0:00 [writeback]
 22 ?        S      0:00 [kcompactd0]
 23 ?        SN     0:00 [ksmd]
 24 ?        SN     0:00 [khugepaged]
 70 ?        I<     0:00 [kintegrityd]
 71 ?        I<     0:00 [kblockd]
 72 ?        I<     0:00 [blkcg_punt_bio]
 73 ?        I<     0:00 [tpm_dev_wq]
 74 ?        I<     0:00 [ata_sff]
 75 ?        I<     0:00 [md]
 76 ?        I<     0:00 [edac-poller]
 77 ?        I<     0:00 [devfreq_wq]
 78 ?        S      0:00 [watchdogd]
 81 ?        S      0:00 [kswapd0]
 82 ?        S      0:00 [ecryptfs-kthrea]
 84 ?        I<     0:00 [kthrotld]
 85 ?        I<     0:00 [acpi_thermal_pm]
 86 ?        S      0:00 [scsi_eh_0]
 87 ?        I<     0:00 [scsi_tmf_0]
 88 ?        S      0:00 [scsi_eh_1]
 89 ?        I<     0:00 [scsi_tmf_1]
 91 ?        I<     0:00 [vfio-irqfd-clea]
 92 ?        I      0:00 [kworker/u2:3-events_unbound]
 93 ?        I<     0:00 [ipv6_addrconf]
102 ?        I<     0:00 [kstrp]
105 ?        I<     0:00 [kworker/u3:0]
118 ?        I<     0:00 [charger_manager]
160 ?        I      0:00 [kworker/0:2-mm_percpu_wq]
163 ?        S      0:00 [scsi_eh_2]
164 ?        I<     0:00 [scsi_tmf_2]
165 ?        I<     0:00 [kworker/0:1H-kblockd]
186 ?        S      0:00 [jbd2/sda5-8]
187 ?        I<     0:00 [ext4-rsv-conver]
226 ?        S<s    0:00 /lib/systemd/systemd-journald
250 ?        Ss     0:00 /lib/systemd/systemd-udevd
251 ?        S      0:00 [irq/18-vmwgfx]
252 ?        I<     0:00 [ttm_swap]
261 ?        S<     0:00 [loop0]
268 ?        S<     0:00 [loop3]
279 ?        S<     0:00 [loop4]
283 ?        S<     0:00 [loop5]
296 ?        S<     0:00 [loop7]
297 ?        S<     0:00 [loop8]
298 ?        S<     0:00 [loop9]
300 ?        S<     0:00 [loop10]
302 ?        S<     0:00 [loop11]
303 ?        S<     0:00 [loop12]
332 ?        I<     0:00 [iprt-VBoxWQueue]
355 ?        I<     0:00 [cryptd]
547 ?        Ss     0:00 /lib/systemd/systemd-resolved
548 ?        Ssl    0:00 /lib/systemd/systemd-timesyncd
585 ?        Ssl    0:00 /usr/lib/accountsservice/accounts-daemon
586 ?        Ss     0:00 /usr/sbin/acpid
591 ?        Ss     0:00 avahi-daemon: running [abhi-VirtualBox.local]
592 ?        Ss     0:00 /usr/sbin/cron -f
596 ?        Ss     0:01 /usr/bin/dbus-daemon --system --address=systemd: --nofork --nopidfile --systemd-activa…
597 ?        Ssl    0:00 /usr/sbin/NetworkManager --no-daemon
617 ?        Ss     0:00 /usr/bin/python3 /usr/bin/networkd-dispatcher --run-startup-triggers
618 ?        Ssl    0:00 /usr/lib/policykit-1/polkitd --no-debug
625 ?        Ssl    0:00 /usr/sbin/rsyslogd -n -iNONE
628 ?        Ssl    0:12 /usr/lib/snapd/snapd
629 ?        Ssl    0:00 /usr/libexec/switcheroo-control
630 ?        Ss     0:00 /lib/systemd/systemd-logind
634 ?        Ssl    0:00 /usr/lib/udisks2/udisksd
641 ?        Ss     0:00 /sbin/wpa_supplicant -u -s -O /run/wpa_supplicant
653 ?        S      0:00 avahi-daemon: chroot helper
699 ?        Ssl    0:00 /usr/sbin/ModemManager --filter-policy=strict
703 ?        Ssl    0:00 /usr/sbin/cups-browsed
717 ?        Ss     0:00 /usr/sbin/cupsd -l
720 ?        Ssl    0:00 /usr/bin/python3 /usr/share/unattended-upgrades/unattended-upgrade-shutdown --wait-for-signal
753 ?        Ssl    0:00 /usr/bin/whoopsie -f
758 ?        Ss     0:00 /usr/sbin/kerneloops --test
763 ?        Ss     0:00 /usr/sbin/kerneloops
961 ?        Ssl    0:00 /usr/sbin/gdm3
974 ?        Sl     0:00 /usr/sbin/VBoxService --pidfile /var/run/vboxadd-service.sh
1036 ?       SNsl   0:00 /usr/libexec/rtkit-daemon
1097 ?       Ssl    0:00 /usr/lib/upower/upowerd
1297 ?       Ssl    0:00 /usr/libexec/colord
1350 ?       Sl     0:00 gdm-session-worker [pam/gdm-password]
1369 ?       Ss     0:00 /lib/systemd/systemd --user
1370 ?       S      0:00 (sd-pam)
1379 ?       Ssl    0:00 /usr/bin/pulseaudio --daemonize=no --log-target=journal
1382 ?       Sl     0:00 /usr/bin/gnome-keyring-daemon --daemonize --login
1385 ?       SNsl   0:00 /usr/libexec/tracker-miner-fs
1387 ?       Ss     0:00 /usr/bin/dbus-daemon --session --address=systemd: --nofork --nopidfile --systemd-ac…
1391 ?       Ssl    0:00 /usr/libexec/gvfsd
1396 ?       Sl     0:00 /usr/libexec/gvfsd-fuse /run/user/1000/gvfs -f -o big_writes
1409 ?       Ssl    0:00 /usr/libexec/gvfs-udisks2-volume-monitor
1420 tty2    Ssl+   0:00 /usr/lib/gdm3/gdm-x-session --run-script env GNOME_SHELL_SESSION_MODE=ubuntu /usr…
1423 tty2    Sl+    0:02 /usr/lib/xorg/Xorg vt2 -displayfd 3 -auth /run/user/1000/gdm/Xauthority …
1424 ?       Ssl    0:00 /usr/libexec/gvfs-afc-volume-monitor
1430 ?       Ssl    0:00 /usr/libexec/gvfs-gphoto2-volume-monitor
1434 ?       Ssl    0:00 /usr/libexec/gvfs-goa-volume-monitor
1438 ?       Sl     0:00 /usr/libexec/goa-daemon
1446 ?       Sl     0:00 /usr/libexec/goa-identity-service
1449 ?       Ssl    0:00 /usr/libexec/gvfs-mtp-volume-monitor
1478 tty2    Sl+    0:00 /usr/libexec/gnome-session-binary --systemd --systemd --session=ubuntu
1550 ?       S      0:00 /usr/bin/VBoxClient --clipboard
1551 ?       Sl     0:00 /usr/bin/VBoxClient --clipboard
1562 ?       S      0:00 /usr/bin/VBoxClient --seamless
1563 ?       Sl     0:00 /usr/bin/VBoxClient --seamless
1569 ?       S      0:00 /usr/bin/VBoxClient --draganddrop
1570 ?       Sl     0:02 /usr/bin/VBoxClient --draganddrop
1577 ?       S      0:00 /usr/bin/VBoxClient --vmsvga
1578 ?       Sl     0:00 /usr/bin/VBoxClient --vmsvga
1585 ?       Ss     0:00 /usr/bin/ssh-agent /usr/bin/im-launch env GNOME_SHELL_SESSION_MODE=ubuntu /usr/bin/gnome-session --systemd --session=ubuntu
1604 ?       Ssl    0:00 /usr/libexec/at-spi-bus-launcher
1610 ?       S      0:00 /usr/bin/dbus-daemon --config-file=/usr/share/defaults/at-spi2/accessibility.conf --nofork --print-address 3
1614 ?       Ssl    0:00 /usr/libexec/gnome-session-ctl --monitor
1621 ?       Ssl    0:00 /usr/libexec/gnome-session-binary --systemd-service --session=ubuntu
1635 ?       Ssl    0:07 /usr/bin/gnome-shell
1678 ?       Sl     0:00 ibus-daemon --panel disable --xim
1689 ?       Sl     0:00 /usr/libexec/ibus-dconf
1690 ?       Sl     0:00 /usr/libexec/ibus-extension-gtk3
1693 ?       Sl     0:00 /usr/libexec/ibus-x11 --kill-daemon
1695 ?       Sl     0:00 /usr/libexec/ibus-portal
1731 ?       Sl     0:00 /usr/libexec/at-spi2-registryd --use-gnome-session
1737 ?       S<     0:00 [loop13]
1740 ?       Ssl    0:00 /usr/libexec/xdg-permission-store
1745 ?       Sl     0:00 /usr/libexec/gnome-shell-calendar-server
1751 ?       Ssl    0:00 /usr/libexec/evolution-source-registry
1762 ?       Ssl    0:00 /usr/libexec/evolution-calendar-factory
1771 ?       Sl     0:00 /usr/libexec/dconf-service
1776 ?       Ssl    0:00 /usr/libexec/evolution-addressbook-factory
1804 ?       Sl     0:00 /usr/libexec/gvfsd-trash --spawner :1.3 /org/gtk/gvfs,
1807 ?       Sl     0:00 /usr/bin/gjs /usr/share/gnome-shell/org.gnome.Shell.N
1847 ?       Ssl    0:00 /usr/libexec/gsd-a11y-settings
1850 ?       Ssl    0:00 /usr/libexec/gsd-color
1851 ?       Ssl    0:00 /usr/libexec/gsd-datetime
1856 ?       Ssl    0:00 /usr/libexec/gsd-housekeeping
1858 ?       Ssl    0:00 /usr/libexec/gsd-keyboard
1861 ?       Ssl    0:00 /usr/libexec/gsd-media-keys
1863 ?       Ssl    0:00 /usr/libexec/gsd-power
1864 ?       Ssl    0:00 /usr/libexec/gsd-print-notifications
1866 ?       Ssl    0:00 /usr/libexec/gsd-rfkill
1875 ?       Ssl    0:00 /usr/libexec/gsd-screensaver-proxy
1879 ?       Ssl    0:00 /usr/libexec/gsd-sharing
1881 ?       Ssl    0:00 /usr/libexec/gsd-smartcard
1885 ?       Sl     0:00 /usr/libexec/evolution-data-server/evolution-alarm-no
1890 ?       Ssl    0:00 /usr/libexec/gsd-sound
1895 ?       Ssl    0:00 /usr/libexec/gsd-usb-protection
1899 ?       Ssl    0:00 /usr/libexec/gsd-wacom
1902 ?       Ssl    0:00 /usr/libexec/gsd-wwan
1904 ?       Ssl    0:00 /usr/libexec/gsd-xsettings
1930 ?       Sl     0:00 /usr/libexec/gsd-disk-utility-notify
1947 ?       Sl     0:00 /usr/libexec/gsd-printer
2005 ?       Sl     0:00 /usr/libexec/ibus-engine-simple
2137 ?       S<     0:00 [loop2]
2200 ?       Ssl    0:01 /usr/libexec/gnome-terminal-server
2208 pts/0   Ss     0:00 bash
2338 ?       S<     0:00 [loop1]
2389 ?       Ssl    0:00 /usr/libexec/gvfsd-metadata
2392 ?       Sl     0:00 update-notifier
2458 ?       SNl    0:02 /usr/bin/python3 /usr/bin/update-manager --no-update
2610 ?       I      0:00 [kworker/u2:0-events_unbound]
2707 ?       I      0:00 [kworker/0:0-events]
2708 ?       I      0:00 [kworker/0:3-cgroup_destroy]
3107 ?       I      0:00 [kworker/u2:1]
3248 ?       Ss     0:00 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startu
4244 ?       Ssl    0:00 /usr/lib/packagekit/packagekitd
4334 pts/0   R+     0:00 ps ax
```

Towards Secure Systems

Now that we have process isolation, we can build more complex security features

File Access Control

Anti-virus

Firewall

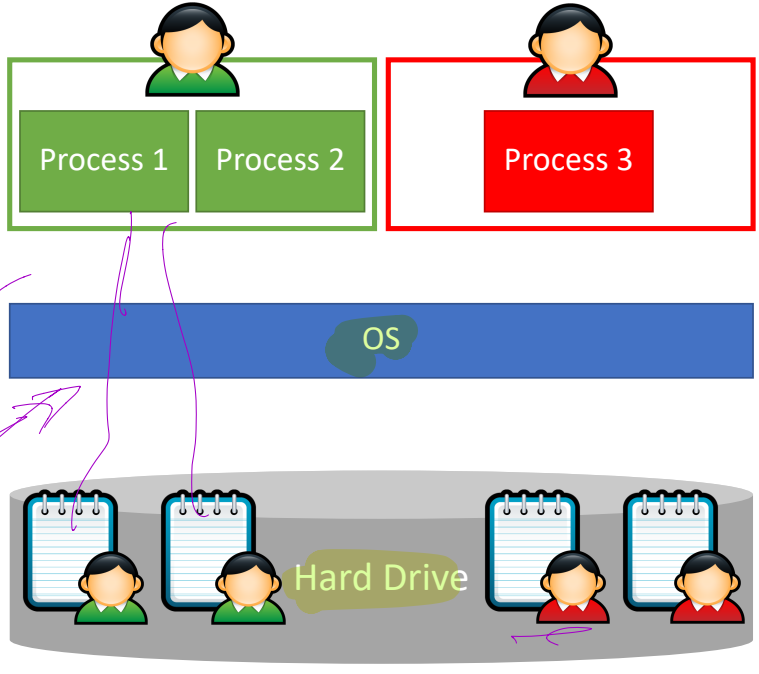
Secure Logging

File Access Control

All disk access is mediated by the OS

OS enforces access controls

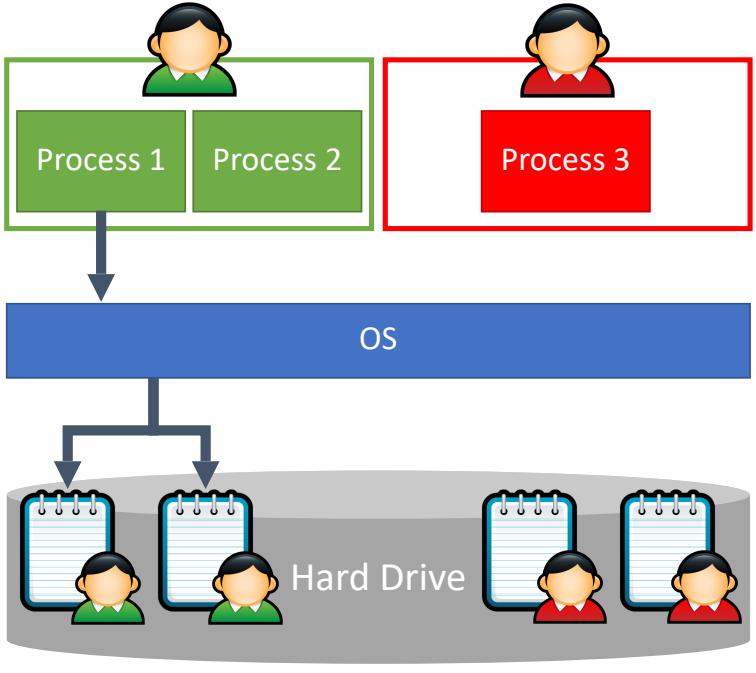
MANDITORY & DISCRETIONARY access control policies



File Access Control

All disk access is mediated by the OS

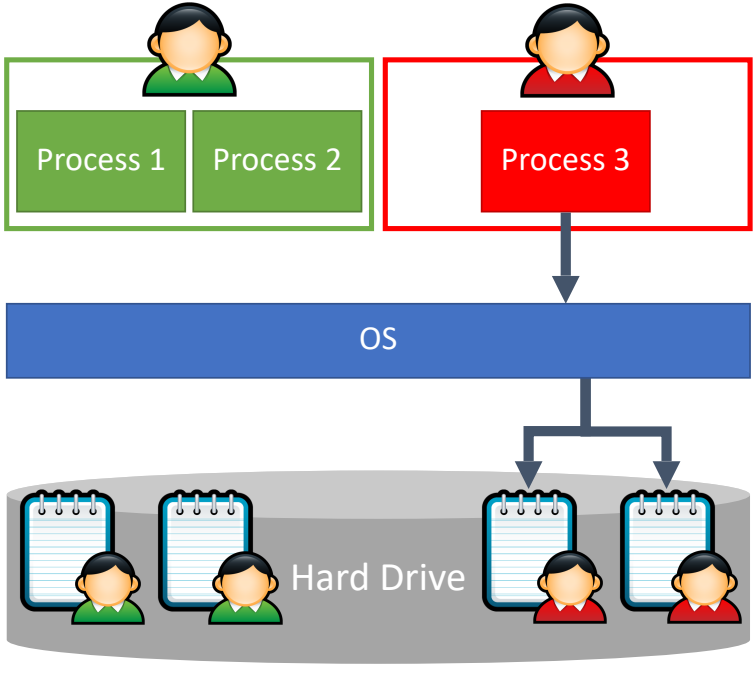
OS enforces access controls



File Access Control

All disk access is mediated by the OS

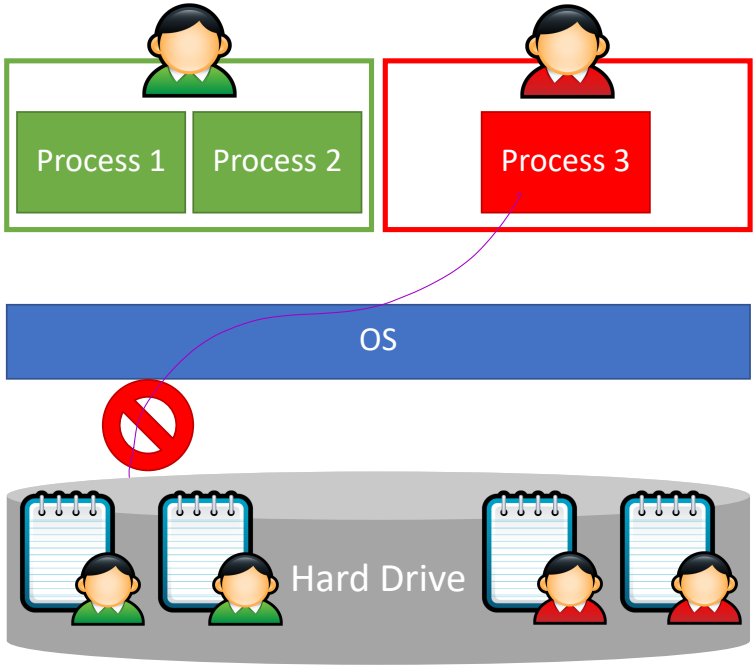
OS enforces access controls



File Access Control

All disk access is mediated by the OS

OS enforces access controls

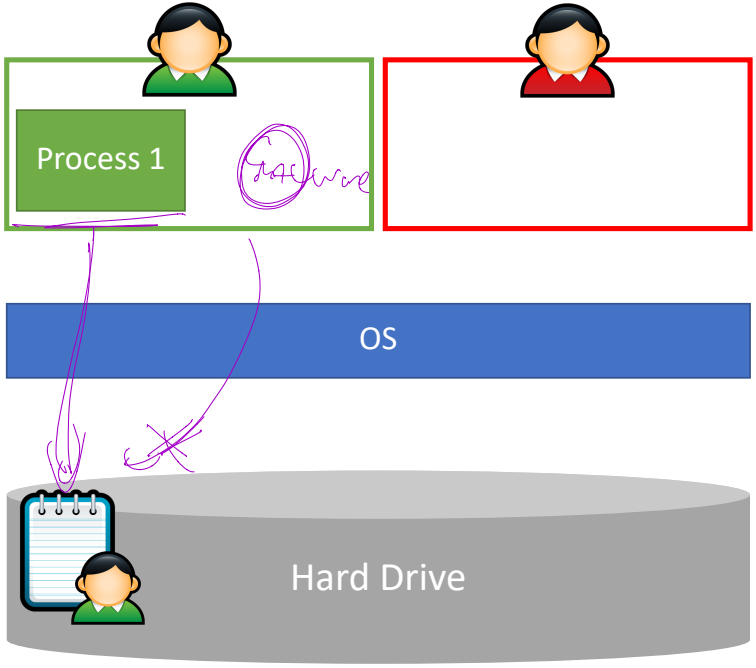


Insecure Logging

Suppose Process 1 writes information to a log file

Malware can still destroy the log

- Add or remove entries
- Add fake entries
- Delete the whole log

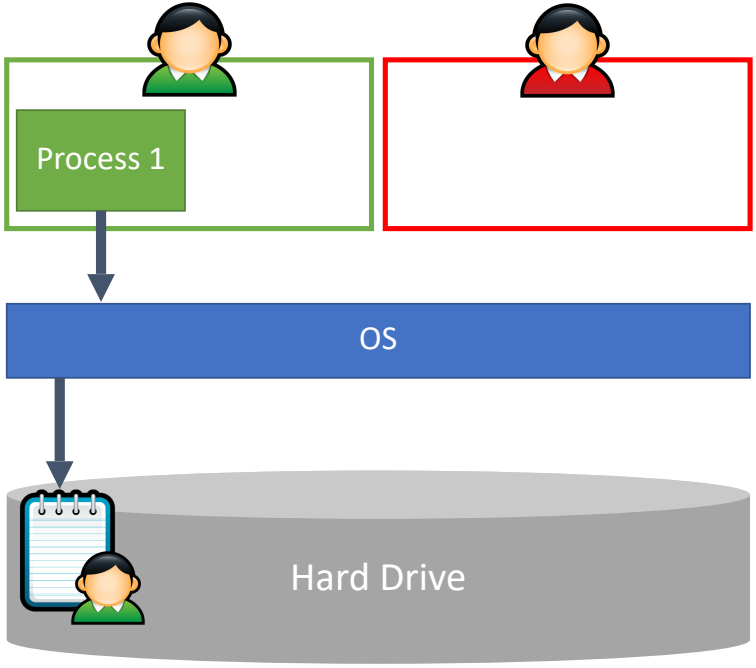


Insecure Logging

Suppose Process 1 writes information to a log file

Malware can still destroy the log

- Add or remove entries
- Add fake entries
- Delete the whole log

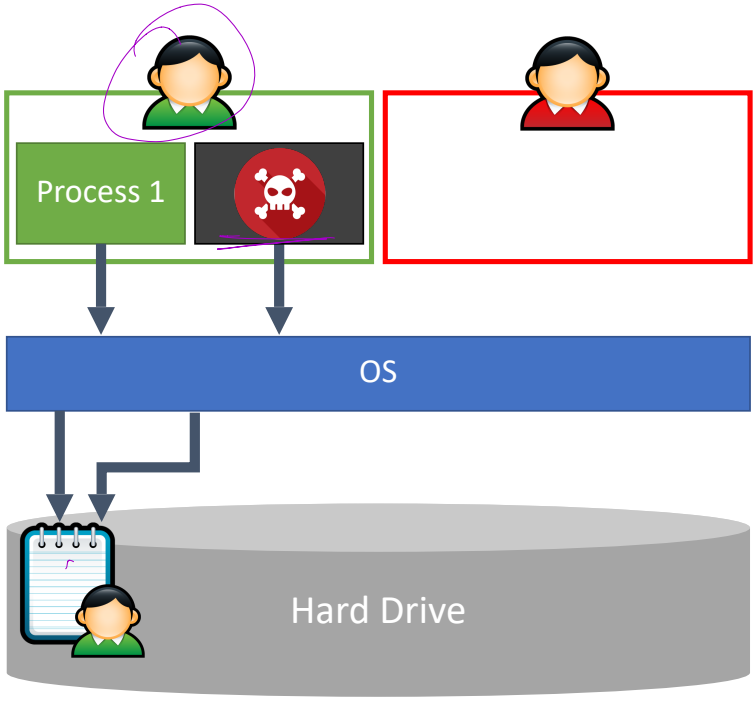


Insecure Logging

Suppose Process 1 writes information to a log file

Malware can still destroy the log

- Add or remove entries
- Add fake entries
- Delete the whole log

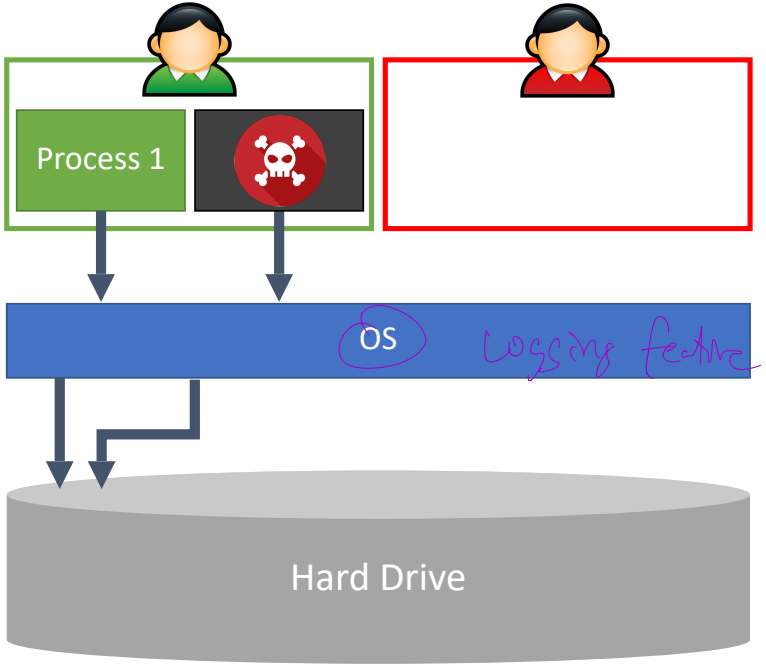


Insecure Logging

Suppose Process 1 writes information to a log file

Malware can still destroy the log

- Add or remove entries
- Add fake entries
- Delete the whole log

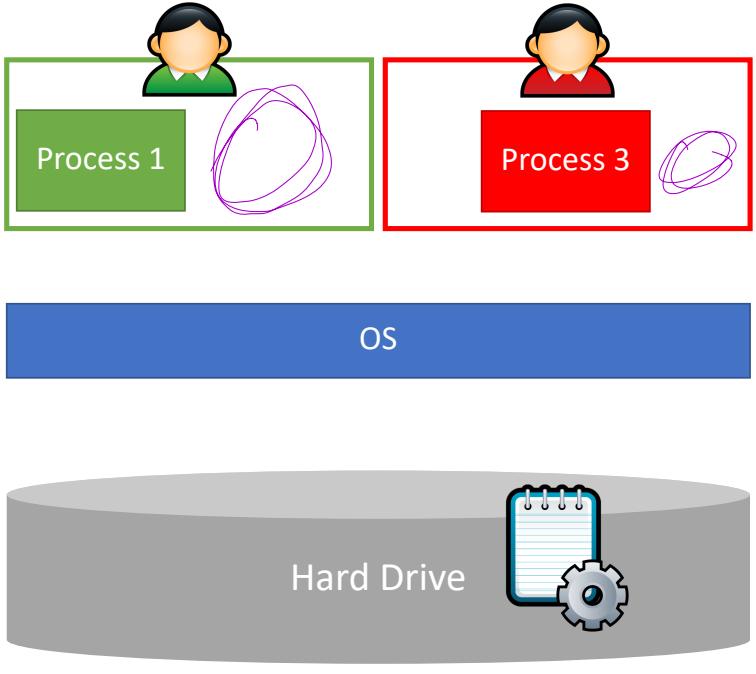


Secure Logging

OS maintains a system log

Processes may write entries to the log using an OS API

Processes may not delete entries or the log

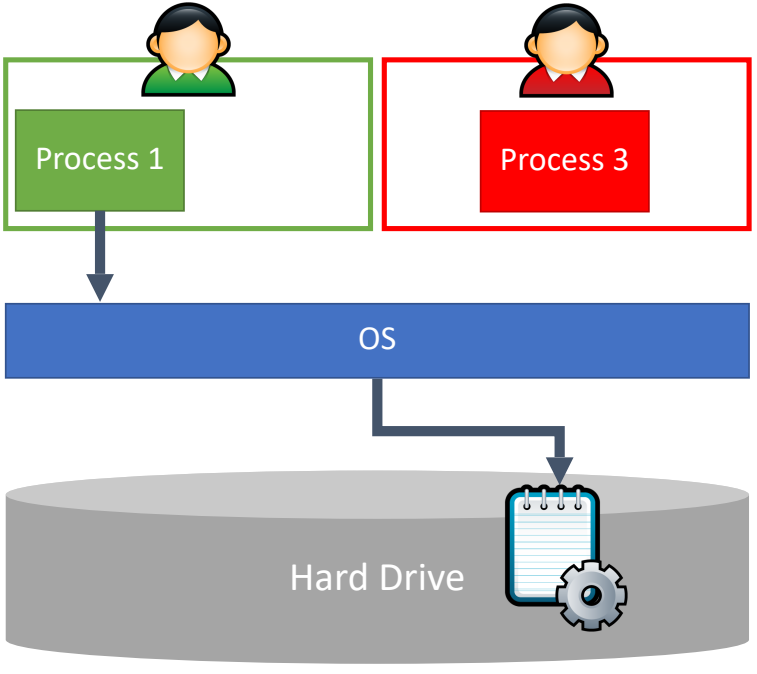


Secure Logging

OS maintains a system log

Processes may write entries to the log using an OS API

Processes may not delete entries or the log

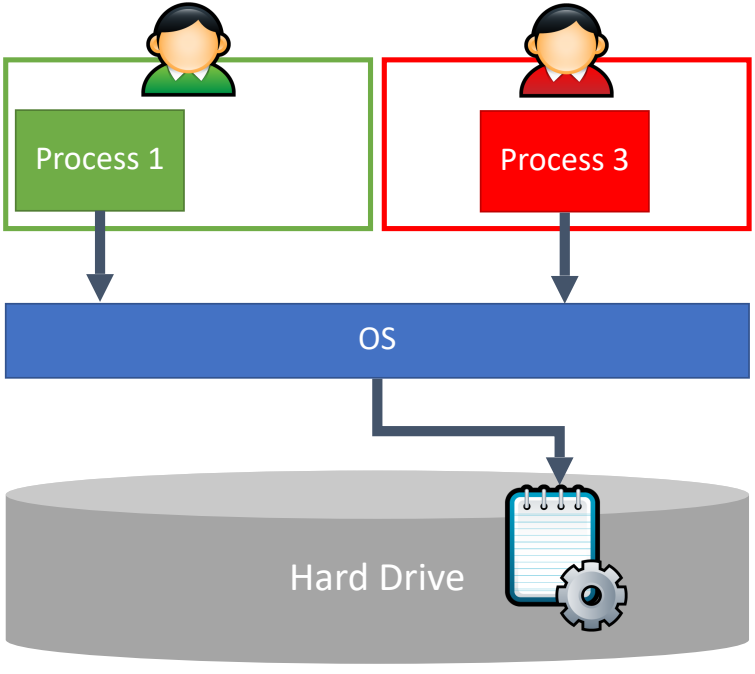


Secure Logging

OS maintains a system log

Processes may write entries to the log using an OS API

Processes may not delete entries or the log

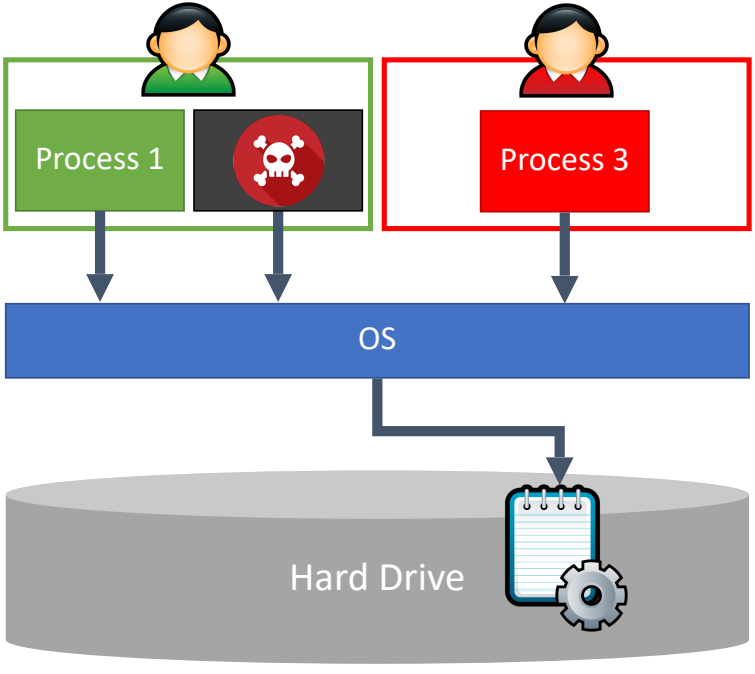


Secure Logging

OS maintains a system log

Processes may write entries to the log using an OS API

Processes may not delete entries or the log



Linux logging

```
abhi@abhi-VirtualBox:~$ ls /var/log
alternatives.log        boot.log        dist-upgrade    gdm3            private                 vboxadd-install.log
alternatives.log.1      boot.log.1      dmesg           gpu-manager.log speech-dispatcher       vboxadd-setup.log
alternatives.log.2.gz   boot.log.2      dmesg.0         hp              syslog                  vboxadd-setup.log.1
apparmor                boot.log.3      dmesg.1.gz      installer       syslog.1                vboxadd-setup.log.2
apport.log              boot.log.4      dmesg.2.gz      journal         syslog.2.gz             vboxadd-setup.log.3
apport.log.1            boot.log.5      dmesg.3.gz      kern.log        syslog.3.gz             vboxadd-setup.log.4
apt                     boot.log.6      dmesg.4.gz      kern.log.1      syslog.4.gz             vboxadd-uninstall.log
auth.log                boot.log.7      dpkg.log        kern.log.2.gz   syslog.5.gz             wtmp
auth.log.1              bootstrap.log   dpkg.log.1      kern.log.3.gz   syslog.6.gz
auth.log.2.gz           btmp            dpkg.log.2.gz   kern.log.4.gz   syslog.7.gz
auth.log.3.gz           btmp.1          faillog         lastlog         ubuntu-advantage.log
auth.log.4.gz           cups            fontconfig.log  openvpn         unattended-upgrades
abhi@abhi-VirtualBox:~$
```

Syslog

/var/log/syslog → system events

/var/log/auth.log → logins.

/var/log/kern.log → Kernel events

/var/log/cron — list of periodically running processes.

syslog

process message

```
Nov 10 10:20:24 abhi-VirtualBox systemd[1]: Reloading.
Nov 10 10:20:24 abhi-VirtualBox systemd[1]: /lib/systemd/system/dbus.socket:5: ListenStream= references a path below legacy directory /var/run/, updating /
var/run/dbus/system_bus_socket → /run/dbus/system_bus_socket; please update the unit file accordingly.
Nov 10 10:20:24 abhi-VirtualBox systemd[1]: Starting Daily apt upgrade and clean activities…
Nov 10 10:20:24 abhi-VirtualBox systemd[1]: Starting OpenBSD Secure Shell server…
Nov 10 10:20:24 abhi-VirtualBox systemd[1]: Started OpenBSD Secure Shell server.
Nov 10 10:20:24 abhi-VirtualBox systemd[1]: apt-daily-upgrade.service: Succeeded.
Nov 10 10:20:24 abhi-VirtualBox systemd[1]: Finished Daily apt upgrade and clean activities.
Nov 10 10:20:24 abhi-VirtualBox systemd[1]: Reloading.
Nov 10 10:20:24 abhi-VirtualBox systemd[1]: /lib/systemd/system/dbus.socket:5: ListenStream= references a path below legacy directory /var/run/, updating /
var/run/dbus/system_bus_socket → /run/dbus/system_bus_socket; please update the unit file accordingly.
Nov 10 10:20:28 abhi-VirtualBox dbus-daemon[596]: [system] Activating via systemd: service name='org.freedesktop.PackageKit' unit='packagekit.service'
requested by ':1.111' (uid=0 pid=4241 comm="/usr/bin/gdbus call --system --dest org.freedeskto" label="unconfined")
```

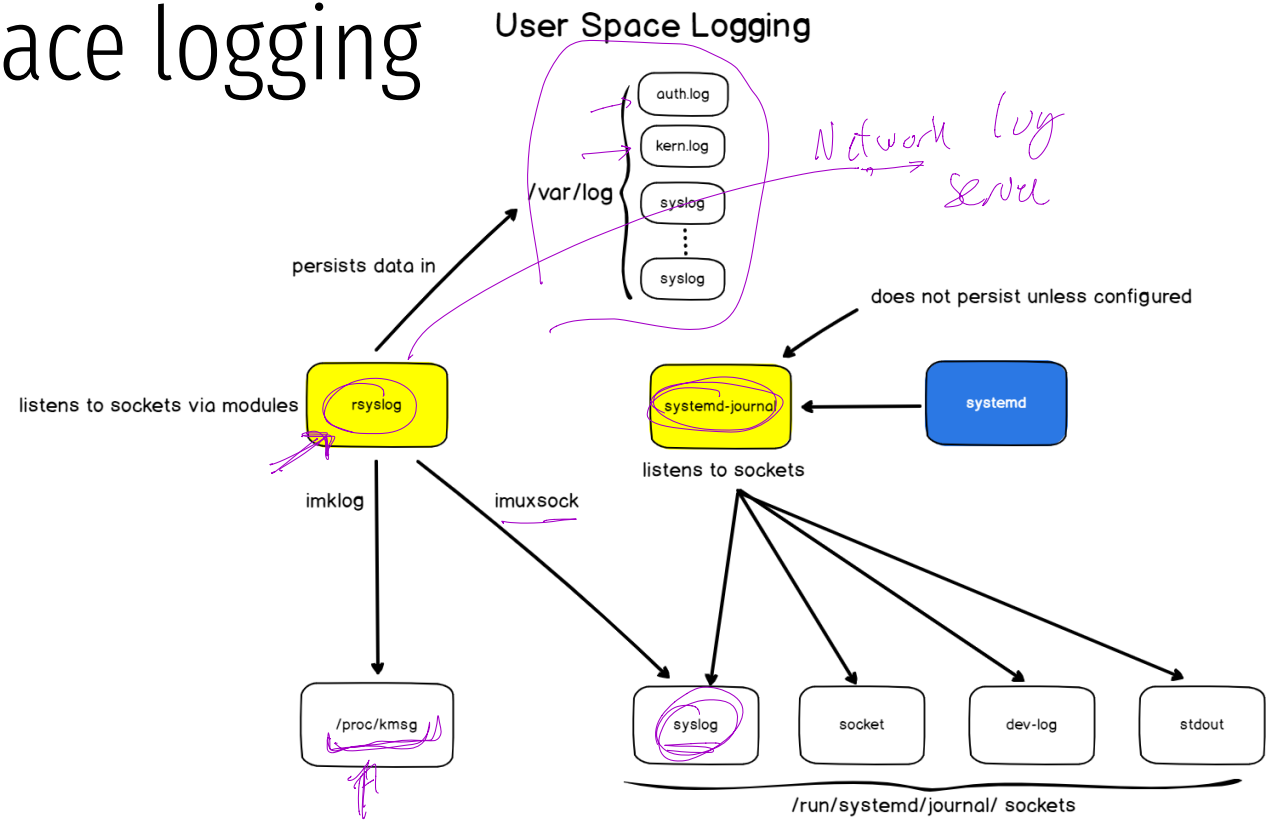
auth.log

```
Nov 10 10:18:55 abhi-VirtualBox sudo: pam_unix(sudo:session): session opened for user root by (uid=0)
Nov 10 10:18:55 abhi-VirtualBox sudo: pam_unix(sudo:session): session closed for user root
Nov 10 10:19:01 abhi-VirtualBox sudo:     abhi : TTY=pts/0 ; PWD=/home/abhi ; USER=root ; COMMAND=/usr/bin/apt install sshd
Nov 10 10:19:01 abhi-VirtualBox sudo: pam_unix(sudo:session): session opened for user root by (uid=0)
Nov 10 10:19:02 abhi-VirtualBox sudo: pam_unix(sudo:session): session closed for user root
Nov 10 10:20:16 abhi-VirtualBox sudo:     abhi : TTY=pts/0 ; PWD=/home/abhi ; USER=root ; COMMAND=/usr/bin/apt install openssh-server
Nov 10 10:20:16 abhi-VirtualBox sudo: pam_unix(sudo:session): session opened for user root by (uid=0)
Nov 10 10:20:22 abhi-VirtualBox useradd[3079]: new user: name=sshd, UID=126, GID=65534, home=/run/sshd, shell=/usr/sbin/nologin, from=none
Nov 10 10:20:22 abhi-VirtualBox usermod[3087]: change user 'sshd' password
Nov 10 10:20:22 abhi-VirtualBox chage[3094]: changed password expiry for sshd
Nov
```

Kernel logging

```
[    0.000406] MTRR variable ranges enabled:
[    0.000409]   0 base 0050000000 mask 7F80000000 uncachable
[    0.000410]   1 base 007C000000 mask 7FFC000000 uncachable
[    0.000412]   2 base 007A000000 mask 7FFE000000 uncachable
[    0.000413]   3 base 0079000000 mask 7FFF000000 uncachable
[    0.000415]   4 base 0078800000 mask 7FFF800000 uncachable
[    0.000417]   5 base 2000000000 mask 6000000000 uncachable
[    0.000418]   6 base 1000000000 mask 7000000000 uncachable
[    0.000420]   7 base 4000000000 mask 4000000000 uncachable
[    0.000421]   8 disabled
[    0.000422]   9 disabled
[    0.001256] x86/PAT: Configuration [0-7]: WB  WC  UC- UC  WB  WP  UC- WT
[    0.001570] last_pfn = 0x6fc4f max_arch_pfn = 0x400000000
[    0.021713] esrt: Reserving ESRT space from 0x0000000068146518 to 0x0000000068146550.
[    0.021729] e820: update [mem 0x68146000-0x68146fff] usable ==> reserved
[    0.021847] check: Scanning 1 areas for low memory corruption
[    0.021853] Using GB pages for direct mapping
[    0.022564] RAMDISK: [mem 0x3ce54000-0x3fffdfff]
[    0.022580] ACPI: Early table checksum verification disabled
[    0.022584] ACPI: RSDP 0x000000006F117014 000024 (v02 INTEL )
[    0.022589] ACPI: XSDT 0x000000006F116728 0000CC (v01 INTEL  NUC9i5FN 00000020 AMI  01000013)
[    0.022598] ACPI: FACP 0x000000006F0D2000 000114 (v06 INTEL  NUC9i5FN 00000020 AMI  00010013)
[    0.022606] ACPI: DSDT 0x000000006F08F000 042561 (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022610] ACPI: FACS 0x000000006F1B1000 000040
[    0.022614] ACPI: MCFG 0x000000006F0D5000 00003C (v01 INTEL  NUC9i5FN 00000020 MSFT 00000097)
[    0.022618] ACPI: SSDT 0x000000006F0D3000 001B4A (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022623] ACPI: FIDT 0x000000006F08E000 00009C (v01 INTEL  NUC9i5FN 00000020 AMI  00010013)
[    0.022627] ACPI: SSDT 0x000000006F08A000 0031C6 (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022631] ACPI: HPET 0x000000006F0D7000 000038 (v01 INTEL  NUC9i5FN 00000020 AMI  01000013)
[    0.022635] ACPI: SSDT 0x000000006F086000 0033B4 (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022639] ACPI: SSDT 0x000000006F084000 00147B (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022643] ACPI: SSDT 0x000000006F08000 0032BD (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022648] ACPI: NHLT 0x000000006F0D6000 00002D (v00 INTEL  NUC9i5FN 00000020 AMI  01000013)
[    0.022652] ACPI: LPIT 0x000000006F07F000 000094 (v01 INTEL  NUC9i5FN 00000020 AMI  01000013)
[    0.022656] ACPI: SSDT 0x000000006F07B000 002720 (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022660] ACPI: SSDT 0x000000006F07A000 00087C (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022664] ACPI: DBGP 0x000000006F079000 000034 (v01 INTEL  NUC9i5FN 00000020 AMI  01000013)
[    0.022668] ACPI: DBG2 0x000000006F078000 000054 (v00 INTEL  NUC9i5FN 00000020 AMI  01000013)
[    0.022672] ACPI: SSDT 0x000000006F076000 001B66 (v02 INTEL  NUC9i5FN 00000020 INTL 20160527)
[    0.022677] ACPI: TPM2 0x000000006F074000 00004C (v04 INTEL  NUC9i5FN 00000020 AMI  00000000)
[    0.022681] ACPI: DMAR 0x000000006F075000 0000A8 (v01 INTEL  NUC9i5FN 00000020       01000013)
[    0.022685] ACPI: WSMT 0x000000006F07E000 000028 (v01 INTEL  NUC9i5FN 00000020 AMI  00010013)
[    0.022689] ACPI: APIC 0x000000006F073000 0000F4 (v04 INTEL  NUC9i5FN 00000020 AMI  00010013)
[    0.022693] ACPI: FPDT 0x000000006F072000 000044 (v01 INTEL  NUC9i5FN 00000020 AMI  01000013)
[    0.022707] ACPI: Local APIC address 0xfee00000
[    0.023236] No NUMA configuration found
[    0.023238] Faking a node at [mem 0x0000000000000000-0x0000000880ffffff]
[    0.023254] NODE_DATA(0) allocated [mem 0x880fd5000-0x880ffffff]
[    0.023698] Zone ranges:
[    0.023700]   DMA      [mem 0x0000000000001000-0x0000000000ffffff]
[    0.023702]   DMA32    [mem 0x0000000001000000-0x00000000ffffffff]
[    0.023703]   Normal   [mem 0x0000000100000000-0x0000000880ffffff]
[    0.023705]   Device   empty
[    0.023706] Movable zone start for each node
[    0.023711] Early memory node ranges
[    0.023713]   node   0: [mem 0x0000000000001000-0x000000000009efff]
[    0.023714]   node   0: [mem 0x0000000000100000-0x000000006cf53fff]
[    0.023716]   node   0: [mem 0x000000006fc4e000-0x000000006fc4efff]
[    0.023717]   node   0: [mem 0x0000000100000000-0x0000000880ffffff]
[    0.024270] Zeroed struct page in unavailable ranges: 41229 pages
[    0.024272] Initmem setup node 0 [mem 0x0000000000001000-0x0000000880ffffff]
[    0.024274] On node 0 totalpages: 8314611
[    0.024276]   DMA zone: 64 pages used for memmap
[    0.024277]   DMA zone: 25 pages reserved
[    0.024278]   DMA zone: 3998 pages, LIFO batch:0
[    0.024379]   DMA32 zone: 6910 pages used for memmap
[    0.024380]   DMA32 zone: 442197 pages, LIFO batch:63
[    0.039990]   Normal zone: 122944 pages used for memmap
[    0.039991]   Normal zone: 7868416 pages, LIFO batch:63
```

User space logging



https://devconnected.com/linux-logging-complete-guide/

```
abhi@abhi-VirtualBox:~$ ps ax | grep log
    596 ?        Ss     0:01 /usr/bin/dbus-daemon --system --address=systemd: --nofork --nopidfile --systemd-activation --syslog-only
    625 ?        Ssl    0:00 /usr/sbin/rsyslogd -n -iNONE
    630 ?        Ss     0:00 /lib/systemd/systemd-logind
   1379 ?        S<sl   0:00 /usr/bin/pulseaudio --daemonize=no --log-target=journal
   1382 ?        Sl     0:00 /usr/bin/gnome-keyring-daemon --daemonize --login
   1387 ?        Ss     0:00 /usr/bin/dbus-daemon --session --address=systemd: --nofork --nopidfile --systemd-activation --syslog-only
   4558 pts/1    S+     0:00 grep --color=auto log
abhi@abhi-VirtualBox:~$
```

systemd-journal

systemd-journal

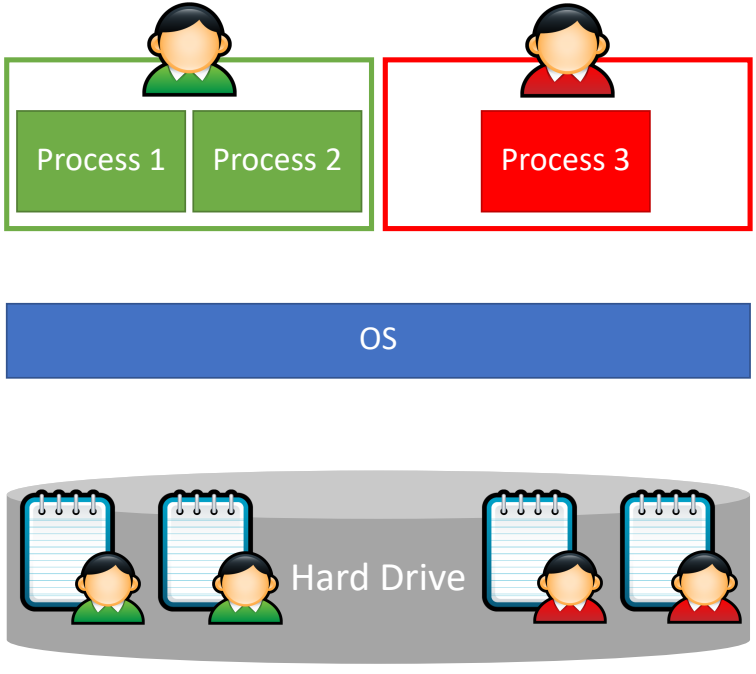
/run/log

```
abhi@abhi-VirtualBox:~$ ls -al /run/
acpid.pid           gdm3/               plymouth/           thermald/
acpid.socket        gdm3.pid            sendsigs.omit.d/    tmpfiles.d/
alsa/               initctl             shm/                udev/
avahi-daemon/       initramfs/          snapd-snap.socket   udisks2/
blkid/              lock/               snapd.socket        ufw.lock
console-setup/      log/                speech-dispatcher/  user/
crond.pid           mount/              spice-vdagentd/     utmp
crond.reboot        NetworkManager/     sshd/               uuidd/
cups/               openvpn/            sshd.pid            vboxadd-service.sh
dbus/               openvpn-client/     sudo/
fsck/               openvpn-server/     systemd/
abhi@abhi-VirtualBox:~$ ls -al /run/log/
total 0
drwxr-xr-x   3 root root              60 Nov 10 10:03 .
drwxr-xr-x  31 root root             880 Nov 10 10:20 ..
drwxr-sr-x+  2 root systemd-journal  40 Nov 10 10:03 journal
abhi@abhi-VirtualBox:~$ ls -al /run/log/journal/
total 0
drwxr-sr-x+ 2 root systemd-journal 40 Nov 10 10:03 .
drwxr-xr-x  3 root root            60 Nov 10 10:03 ..
abhi@abhi-VirtualBox:~$
```

File Access Control

All disk access is mediated by the OS

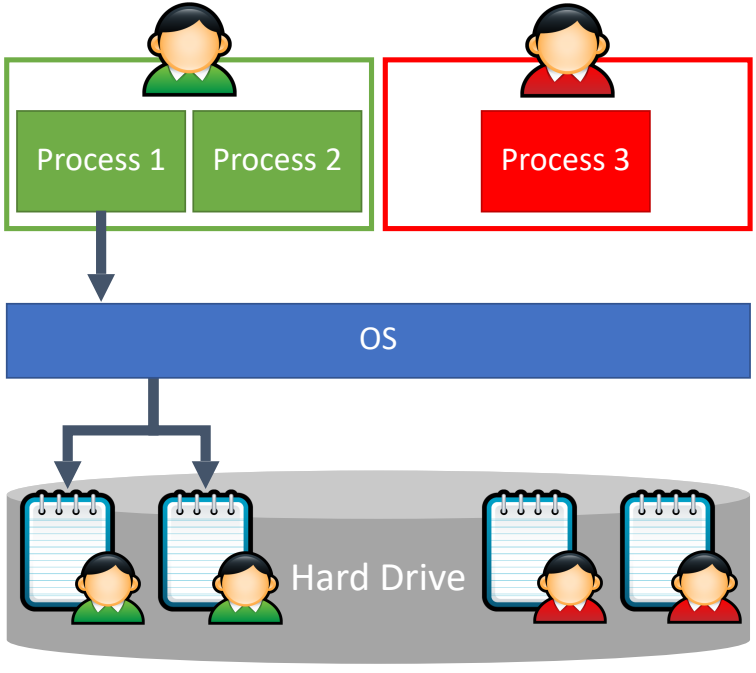
OS enforces access controls



File Access Control

All disk access is mediated by the OS

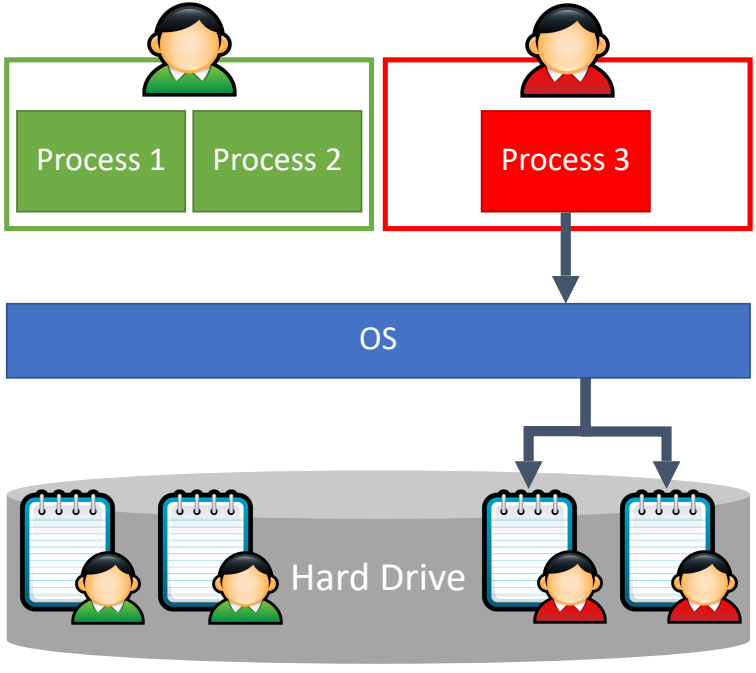
OS enforces access controls



File Access Control

All disk access is mediated by the OS

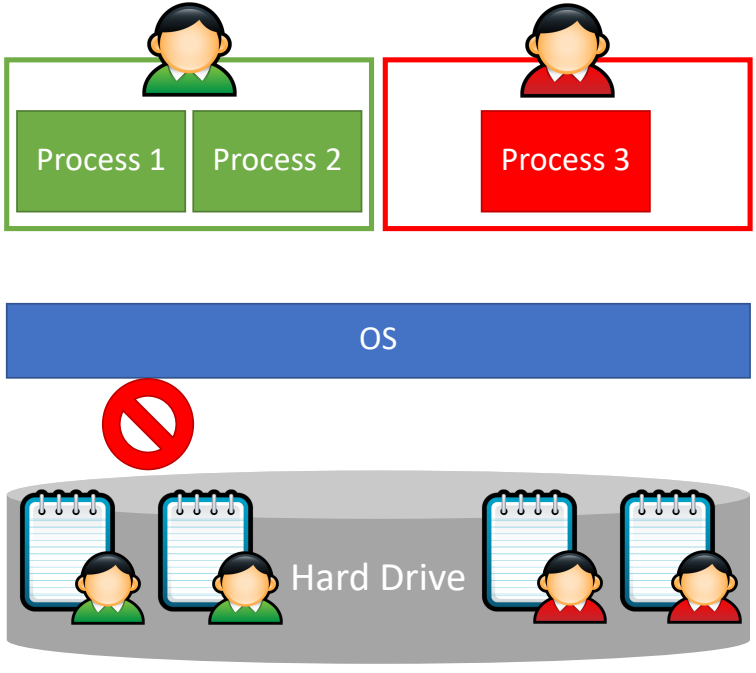
OS enforces access controls



File Access Control

All disk access is mediated by the OS

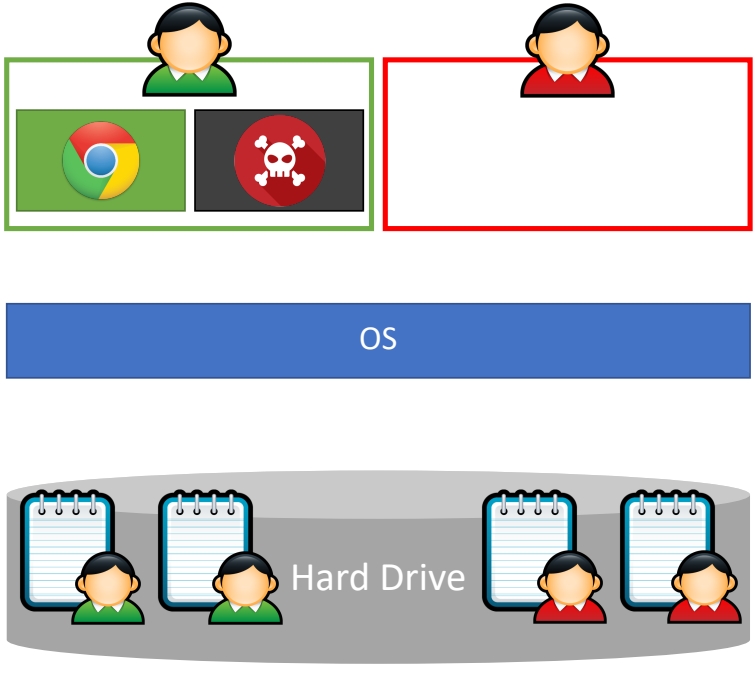
OS enforces access controls



Limitations

Malware can still cause damage

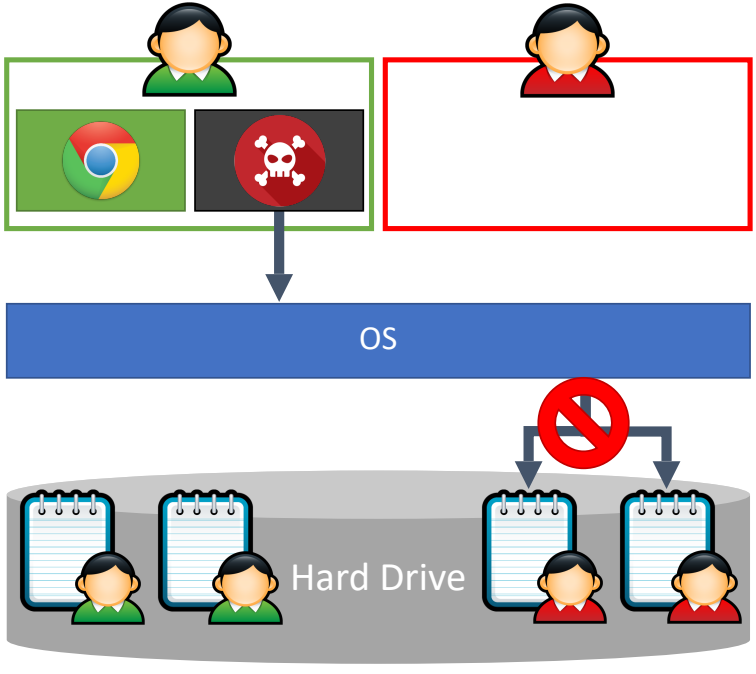
Discretionary access control means that isolation is incomplete



Limitations

Malware can still cause damage

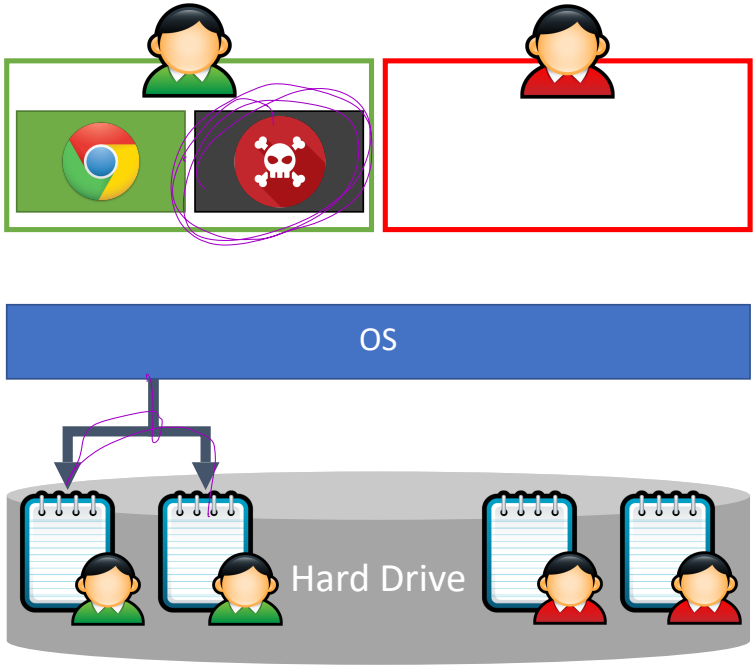
Discretionary access control means that isolation is incomplete



Limitations

Malware can still cause damage

Discretionary access control means that isolation is incomplete



Anti-virus

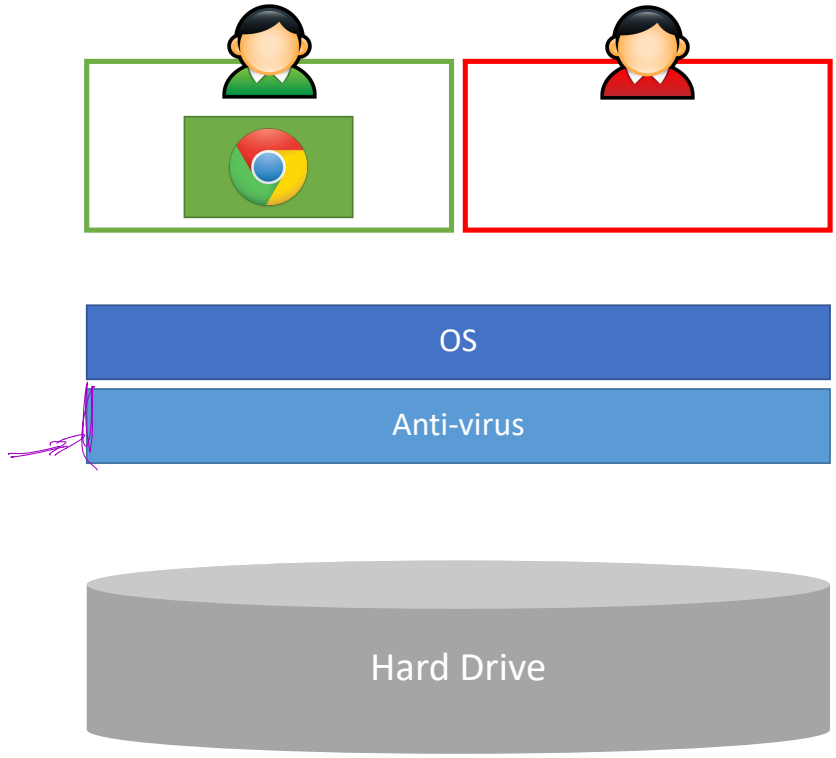
Anti-virus process is privileged

- Often runs in Ring 0

Scans all files looking for signatures

- Each signature uniquely identifies a piece of malware

Files scanned on creation and access



Anti-virus

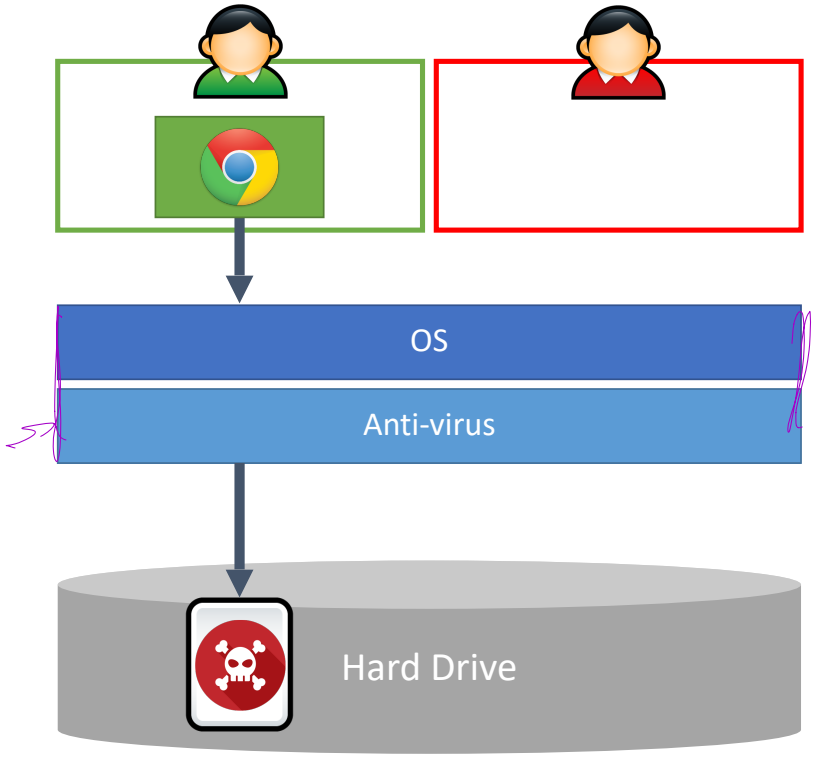
Anti-virus process is privileged

- Often runs in Ring 0

Scans all files looking for signatures

- Each signature uniquely identifies a piece of malware

Files scanned on creation and access



Anti-virus

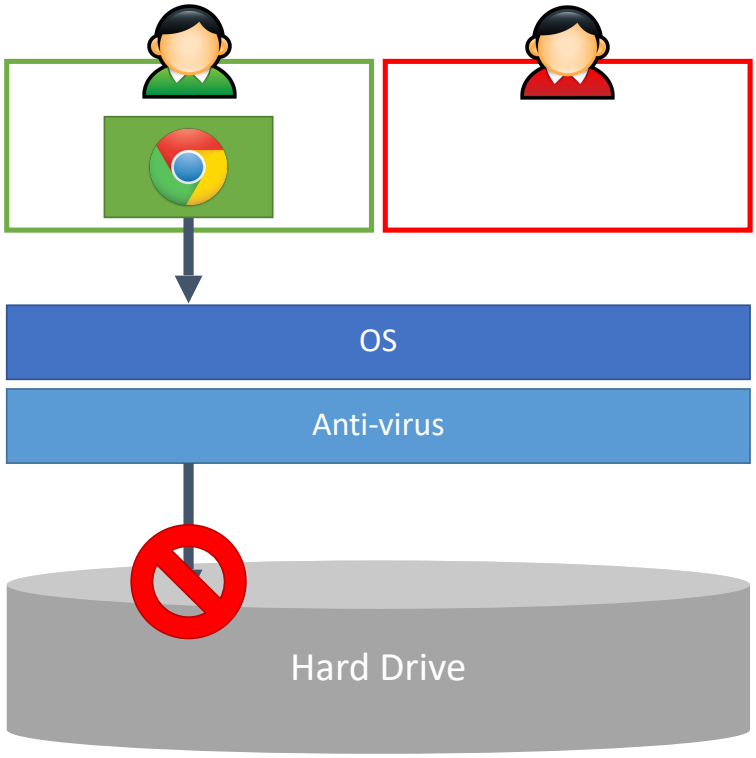
Anti-virus process is privileged

- Often runs in Ring 0

Scans all files looking for signatures

- Each signature uniquely identifies a piece of malware

Files scanned on creation and access



Anti-virus

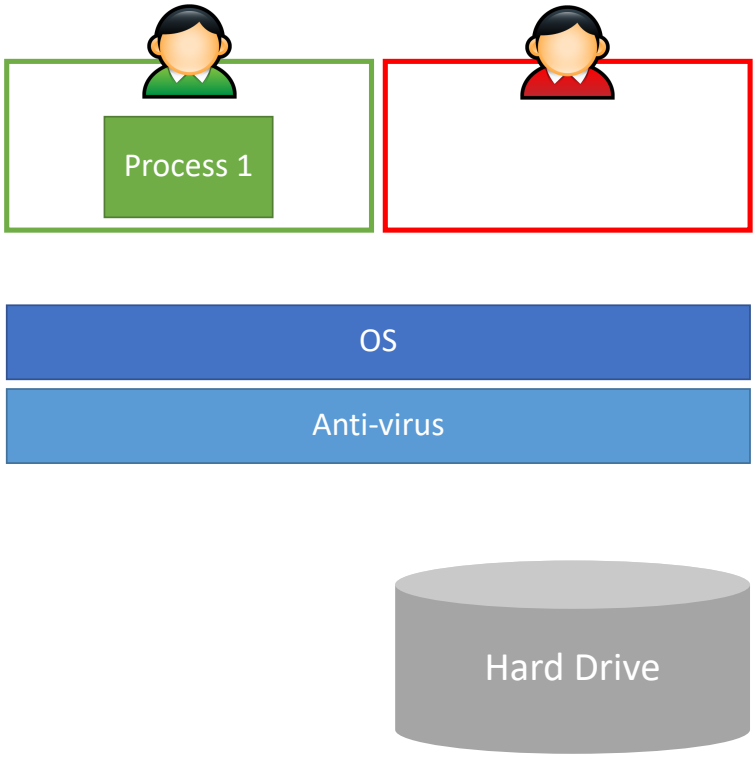
Anti-virus process is privileged

- Typically runs in Ring 0

Scans all files looking for signatures

- Each signature uniquely identifies a piece of malware

Files scanned on creation and access



Anti-virus

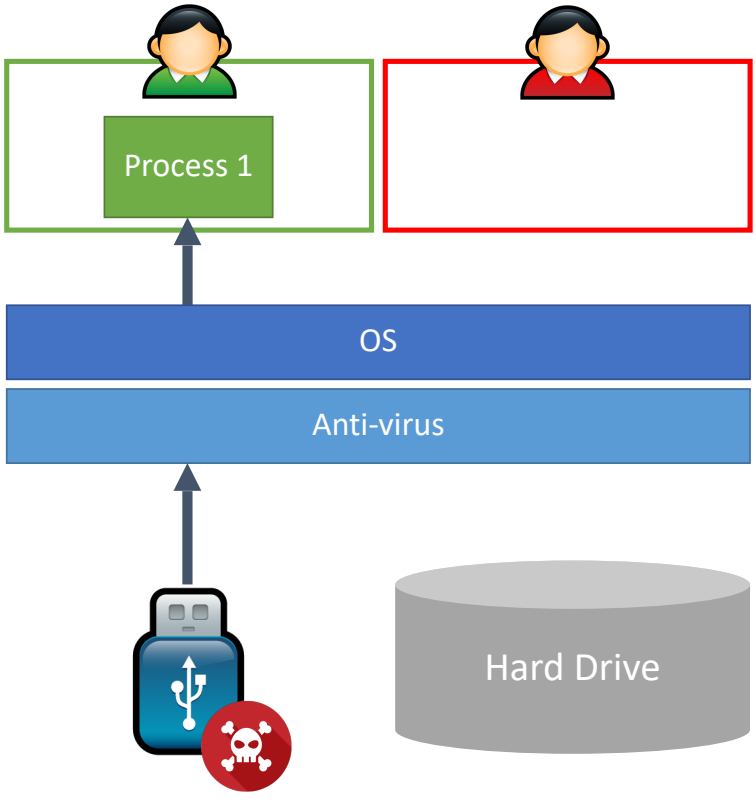
Anti-virus process is privileged

- Typically runs in Ring 0

Scans all files looking for signatures

- Each signature uniquely identifies a piece of malware

Files scanned on creation and access



Anti-virus

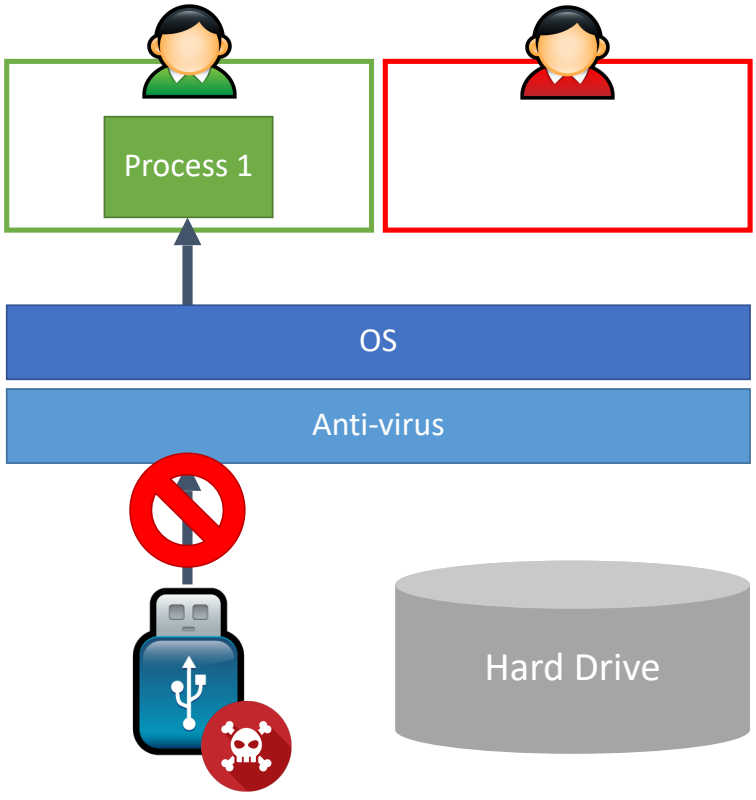
Anti-virus process is privileged

- Typically runs in Ring 0

Scans all files looking for signatures

- Each signature uniquely identifies a piece of malware

Files scanned on creation and access



Signature-based Detection

Key idea: identify invariants that correspond to malicious code or data

Example – anti-virus signatures

- List of code snippets that are unique to known malware

Problems with signatures

Signature-based Detection

Key idea: identify invariants that correspond to malicious code or data

Example – anti-virus signatures

- List of code snippets that are unique to known malware

Problems with signatures

- Must be updated frequently
- May cause false positives
  - Accidental overlaps with good programs and benign network traffic

Avast Malware Signature Update Breaks Installed Programs

Users of the free version of Avast antivirus unscathed

May 7, 2015 13:55 GMT · By Ionut Ilascu · Share:

A bad virus definition update from Avast released on Wednesday caused a lot of trouble, as it mistook various components in legitimate programs installed on the machine for malware.

The list of valid software affected by the signature update includes Firefox, iTunes, NVIDIA drivers, Google Chrome, Adobe Flash Player, Skype, Opera, TeamViewer, ATI drivers, as well as products from Corel and components of Microsoft Office.

Avoiding Anti-virus

Malware authors go to great length to avoid detection by AV

Polymorphism

- Viral code mutates after every infection

Avoiding Anti-virus

Malware authors go to great length to avoid detection by AV

Polymorphism

- Viral code mutates after every infection

```
b = a + 10
```

Avoiding Anti-virus

Malware authors go to great length to avoid detection by AV

Polymorphism

- Viral code mutates after every infection

```
b = a + 10        b = a + 5 + 5
```

Avoiding Anti-virus

Malware authors go to great length to avoid detection by AV

Polymorphism

- Viral code mutates after every infection

```
b = a + 10        b = a + 5 + 5        b = (2 * a + 20) / 2
```

same operation

different signature.

Avoiding Anti-virus

Malware authors go to great length to avoid detection by AV

Polymorphism

- Viral code mutates after every infection

```
b = a + 10        b = a + 5 + 5     b = (2 * a + 20) / 2
```

Packing

- Malware code is encrypted, key is changed every infection
- Decryption code is vulnerable to signature construction
- Polymorphism may be used to mutate the decryption code

Firewall

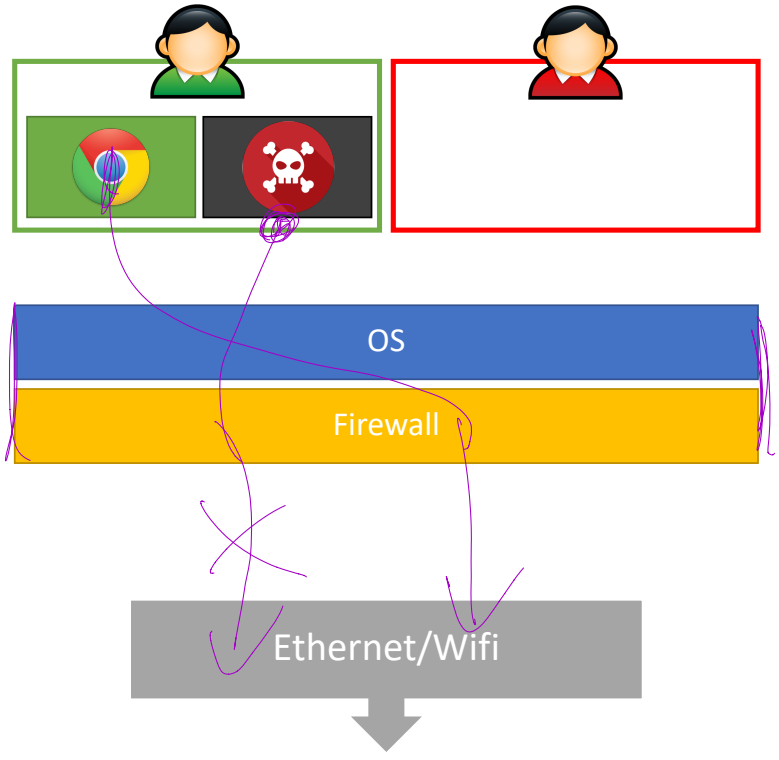
Firewall process is privileged

- Often runs in Ring 0

Selectively blocks network traffic

- By process
- By port
- By IP address
- By packet content

Inspects outgoing and incoming network traffic



Firewall

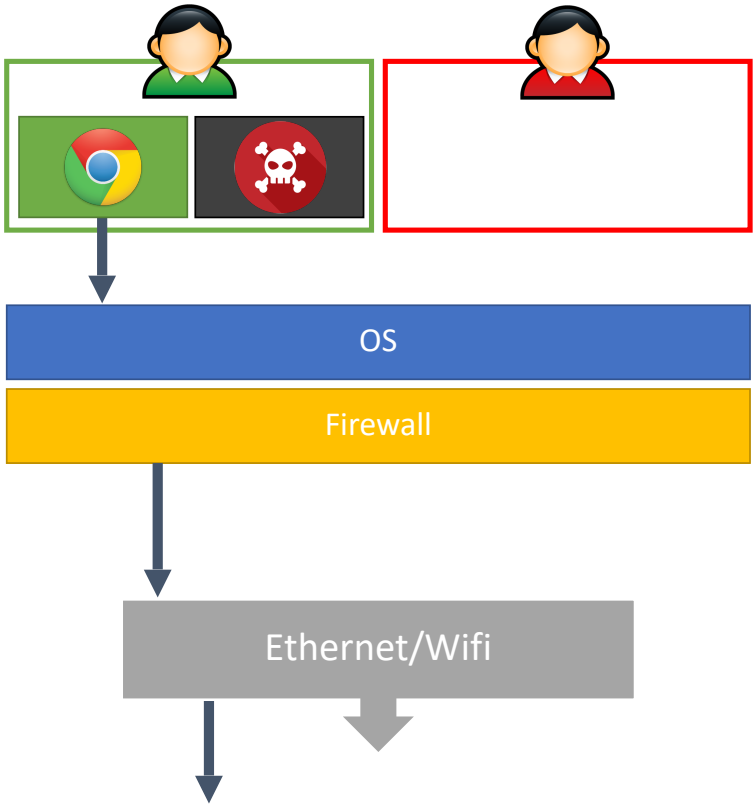
Firewall process is privileged

- Often runs in Ring 0

Selectively blocks network traffic

- By process
- By port
- By IP address
- By packet content

Inspects outgoing and incoming network traffic



Firewall

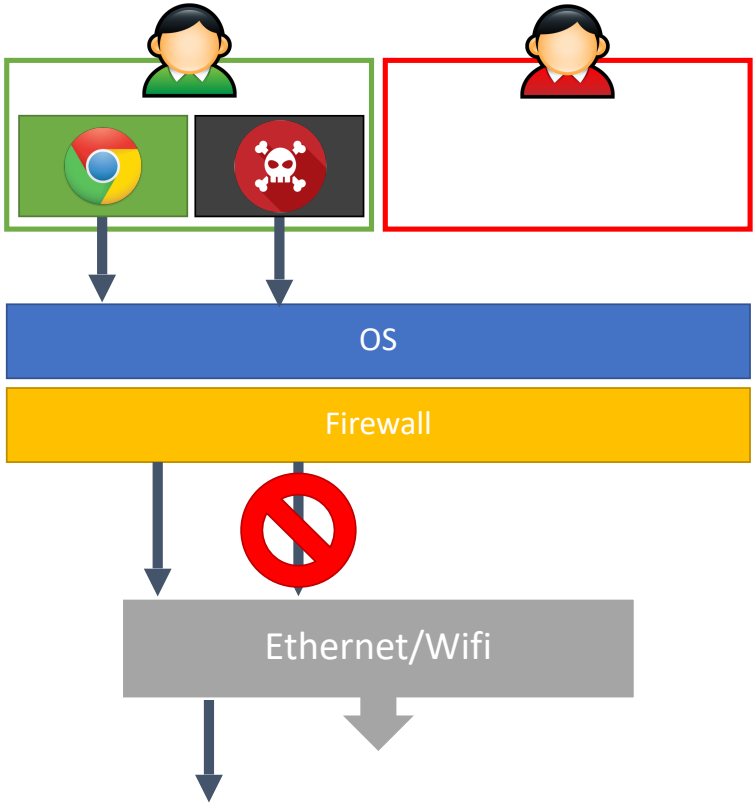
Firewall process is privileged

- Often runs in Ring 0

Selectively blocks network traffic

- By process
- By port
- By IP address
- By packet content

Inspects outgoing and incoming network traffic



Firewall

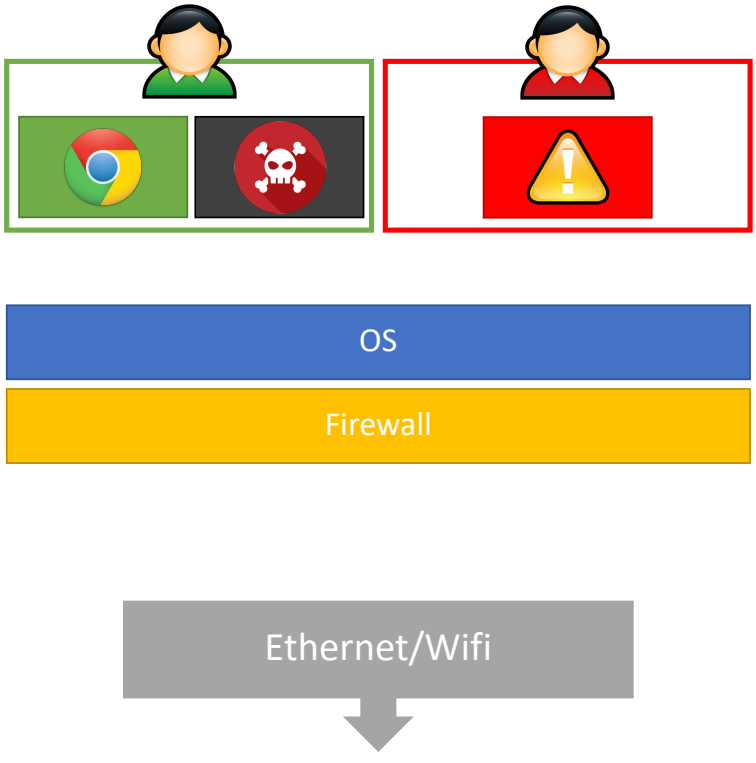
Firewall process is privileged

- Often runs in Ring 0

Selectively blocks network traffic

- By process
- By port
- By IP address
- By packet content

Inspects outgoing and incoming network traffic



Firewall

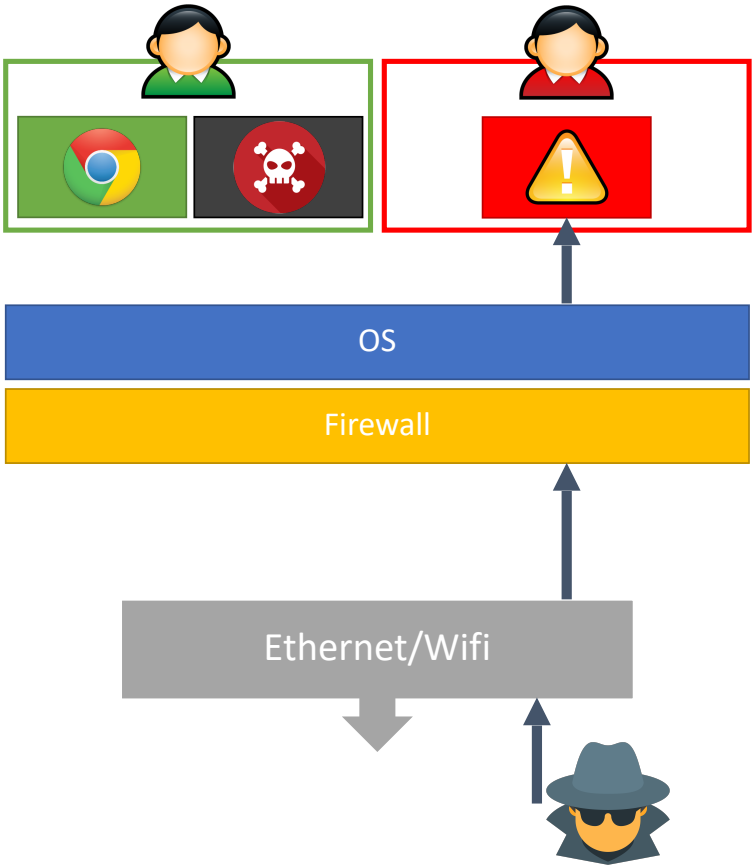
Firewall process is privileged

- Often runs in Ring 0

Selectively blocks network traffic

- By process
- By port
- By IP address
- By packet content

Inspects outgoing and incoming network traffic



Firewall

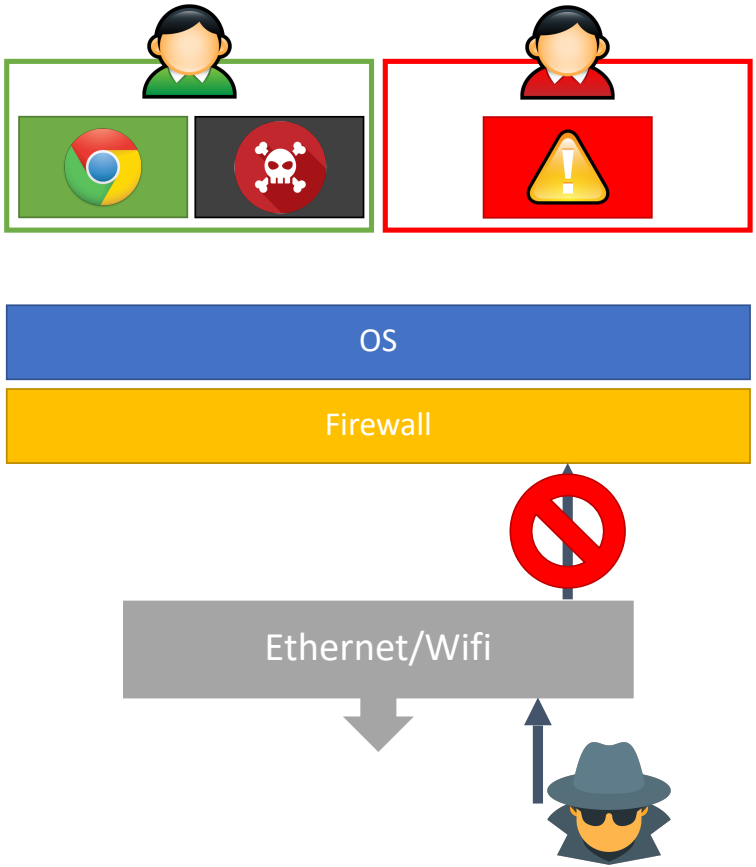
Firewall process is privileged

- Often runs in Ring 0

Selectively blocks network traffic

- By process
- By port
- By IP address
- By packet content

Inspects outgoing and incoming network traffic



Network Intrusion Detection Systems

NIDS for short

Snort

- Open source intrusion prevention system capable of real-time traffic analysis and packet logging
- Identifies malicious network traffic using signatures

Bro

- Open source network monitoring, analysis, and logging framework
- Can be used to implement signature based detection
- Capable of more complex analysis