Cloud Computing Concepts

CS3132

Dr. Anand Kumar Mishra

NIIT University

Privilege instruction, Sensitive instruction, and User instruction

Privileged instructions

- Privileged instructions are instructions that can only be executed by the CPU in a higher privilege mode, typically referred to as "kernel mode" or "privileged mode"
- These instructions are typically used to perform low-level tasks such as managing memory, handling interrupts, and accessing hardware devices.
- Example:
  - An example of a privilege instruction is an instruction that directly accesses or modifies a control register, such as the Control Register 0 (CR0) on x86 architecture, which controls features like paging and protection. Changing these settings requires elevated privileges, and only the kernel (operating system) should be able to execute such instructions.

Control Registers

- A control register is a processor register which changes or controls the general behavior of a CPU or other digital device
- Common tasks performed by control registers include interrupt control, switching the addressing mode, paging control, and coprocessor control
- Processors based on the Intel architecture have a set of control registers that are used for configuration of the processor at run time (such as switching between execution modes)
  - These registers are 32-bit wide on x86 and 64-bit wide on AMD64 (long mode)

Control Registers

- Control registers in x86 series
  - CR0
  - CR1 – Reserved
  - CR2 - Contains a value called Page Fault Linear Address (PFLA)
  - CR3
  - CR4
- Additional Control registers in x86-64 series
  - EFER - Extended Feature Enable Register is a register added in the AMD K6 processor

CPU Registers x86

- General Purpose Registers
- Pointer Registers
- Segment Registers
- EFLAGS Register
- Control Registers
- Extended Control Registers
- Debug Registers
- Test Registers
- Protected Mode Registers
  - GDTR - Global Descriptor Table Register
  - LDTR - Local Descriptor Table Register
  - IDTR - Interrupt Descriptor Table Register

Privileged Instructions

- The Instructions that can run only in Kernel Mode are called Privileged Instructions
- If any attempt is made to execute a Privileged Instruction in User Mode, then it will not be executed and treated as an illegal instruction - The Hardware traps it in the Operating System
- Before transferring the control to any User Program, it is the responsibility of the Operating System to ensure that the Timer is set to interrupt
  - Thus, if the timer interrupts then the Operating System regains the control.
  - Thus, any instruction which can modify the contents of the Timer is Privileged Instruction

Examples of Privileged Instructions

- Privileged Instructions are used by the Operating System in order to achieve correct operation
  - I/O instructions and Halt instructions
  - Turn off all Interrupts
  - Set the Timer
  - Context Switching
  - Clear the Memory or Remove a process from the Memory
  - Modify entries in the Device-status table

Examples of Privileged Instructions

- Turn off all Interrupts
  - This is an exception that a user prefers
  - It is applicable by the division of zero or any invalid memory access
  - It is useful for invoking a kernel routine (a system call)
  - Further, this happens due to the run of a priority function rather than the user code
- I/O instructions and Halt instructions
  - I/O (input and output) instructions are the processes between a computer and the outside world
  - Halt is an assembly language instruction
  - It halts the CPU till the upcoming external interrupt is not coming in

Examples of Privileged Instructions

- Context Switching
  - This is the process that consists of involving the context storage or state of the process
  - It happens so that the switching process will reload and execute from the same point as before
- Set the Timer
  - This is a feature that helps to set a perfect period of interruption for a computer
  - The right time is to set within the counter to finish or start all your needed instructions

Examples of Privileged Instructions

- Adapt entries in the Device-status table
  - The right device status comes with the indication of the type, state, and address for each input/output device
  - It is the entry of every device
- Remove a process or clear the memory from memory
  - A task is only executable when a memory already installed within the computer system allows it
  - This type of operation is possible in privileged instructions

Non-Privileged Instructions

- The Instructions that can run only in User Mode are called Non-Privileged Instructions
  - Reading the status of Processor
  - Reading the System Time
  - Generate any Trap Instruction
  - Sending the final printout of Printer
- Common user instructions include arithmetic and logical operations (e.g., addition, subtraction, bitwise operations), data movement (e.g., load and store), and branching (e.g., jump and conditional jump) instructions. These instructions perform fundamental tasks within a program.

Sensitive Instructions

- Sensitive instructions are those that have the potential to compromise system security or stability if misused or executed by unauthorized users or processes
- These instructions often include operations that can directly affect memory management, hardware control, or access to sensitive data
- Example:
  - An example of a sensitive instruction is an instruction that can manipulate page tables or memory protection attributes
  - For instance, on x86, instructions like `mov` can be sensitive when used to modify page table entries to change memory access permissions

Sensitive Instructions

- Sensitive instructions are not explicitly defined as a separate category in most computer architectures or instruction set architectures (ISAs)
- Sensitive instructions can be thought of as instructions that have the potential to impact system security or stability, especially when executed by an unauthorized or unprivileged process
- Whether an instruction is sensitive or not depends on the system's security model and the access control mechanisms in place

How sensitivity relates to privilege instructions

- In many computer architectures, sensitive instructions may overlap with privilege instructions because instructions that can potentially compromise system security or stability are often restricted to higher privilege levels (e.g., kernel mode)
- However, not all privilege instructions are necessarily sensitive, and not all sensitive instructions are necessarily privilege instructions.

`mov` instruction (x86 architecture)

- `mov` instruction moves data from the source to the destination
- typical syntax for the `mov` instruction in x86 assembly language is as follows:

```
mov destination, source
```

- Destination: This is where the data will be moved or copied to. It is typically a register or a memory location.
- Source: This is the data to be moved or copied. It can be a register, an immediate value (a constant), or a memory location.

`mov` instruction (x86 architecture)

- Common examples of how the mov instruction is used in x86 assembly language:
- Register-to-Register Transfer: `mov ebx, eax`
- Immediate Value to Register: `mov ecx, 42`
- Memory-to-Register Transfer: `mov edx, [ebx]`
- Register-to-Memory Transfer: `mov [edi], esi`
- Memory-to-Memory Transfer (Not Directly Supported)
  - It's important to note that the x86 architecture doesn't directly support memory-to-memory transfers using a single mov instruction. You typically need to move data between memory and a register first and then between registers and memory to achieve memory-to-memory copying.

etc.

How sensitivity relates to privilege instructions

- Example: consider an instruction like "mov" (move) in x86 assembly language
- When executed in user mode,
  - it is typically not considered a privilege instruction, and user-level programs can execute it to move data within their own memory space
- However, if a user-level program tries to use "mov" to modify critical system data or control registers, it becomes sensitive and would not be allowed due to privilege restrictions

How sensitivity relates to privilege instructions - behavior of the `mov` instruction

- User Instruction: `mov` is used for standard data copying operations within the user process's memory space

```
mov eax, ebx
```

- Privilege Instruction: `mov` may be used to access and manipulate critical system data structures and hardware control registers

```
mov cr0, eax
```

- Sensitive Instruction:
  - Memory Protection Attribute Modification
  - Changing Interrupt Descriptor Table (IDT)
  - Control Register Sensitive Modification

How sensitivity relates to privilege instructions - behavior of the `mov` instruction

- Sensitive Instruction:
  - Memory Protection Attribute Modification – Altering memory protection attributes using mov can be sensitive, especially when attempting to change page table entries to grant or restrict access to memory regions
  - Changing Interrupt Descriptor Table (IDT) - Modifying the IDT with mov can be sensitive, as it controls how the CPU responds to interrupts and exceptions

    ```
    mov idtr, edx
    ```
  - Control Register Sensitive Modification

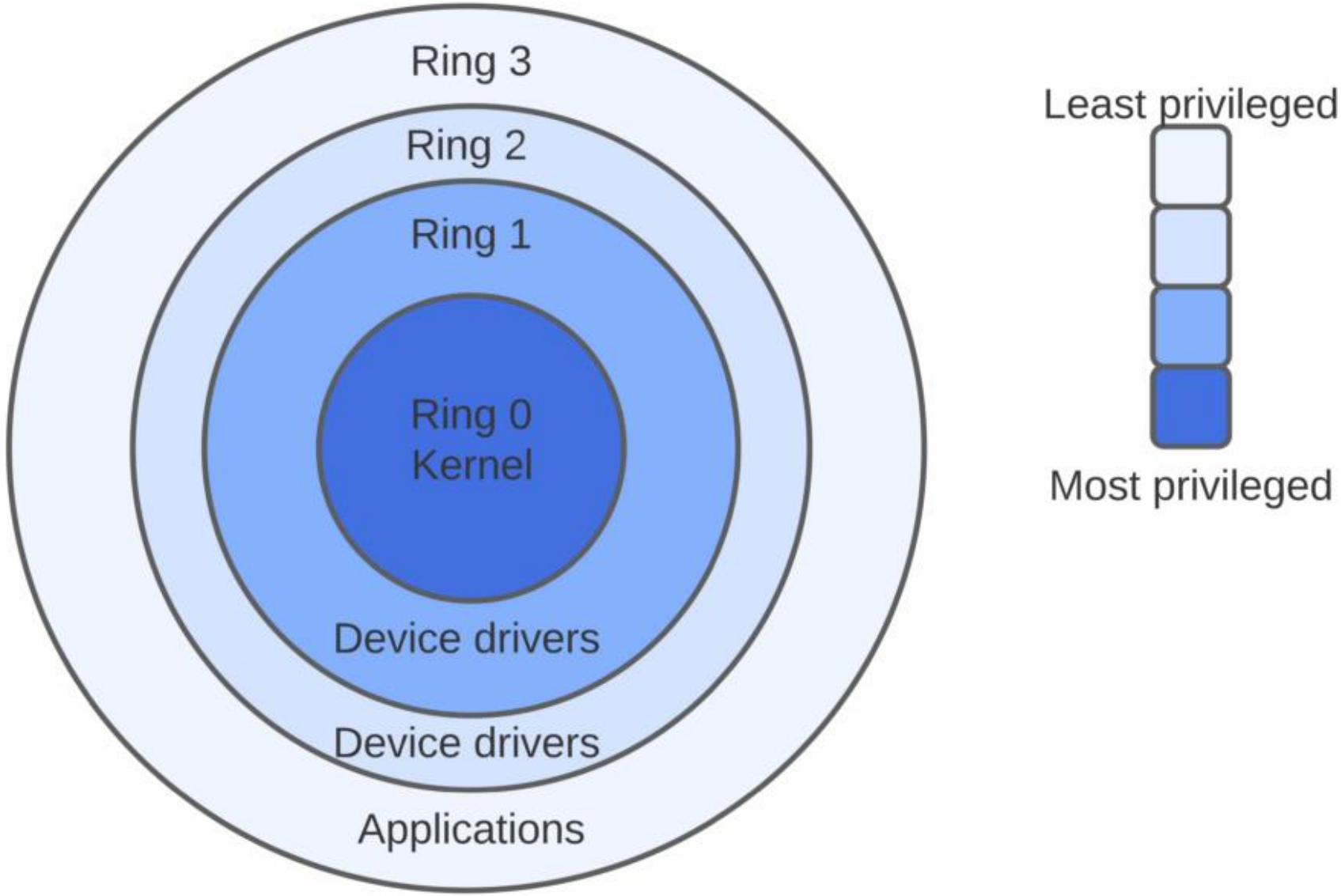
Question: Is the bootloader GRUB a hypervisor?

Question: Is the bootloader GRUB a hypervisor?

Answer: No, GRUB used in dual boot machines decides which one OS it has to boot, whereas hypervisors can run multiple guest VMs.

Rings in Operating Systems

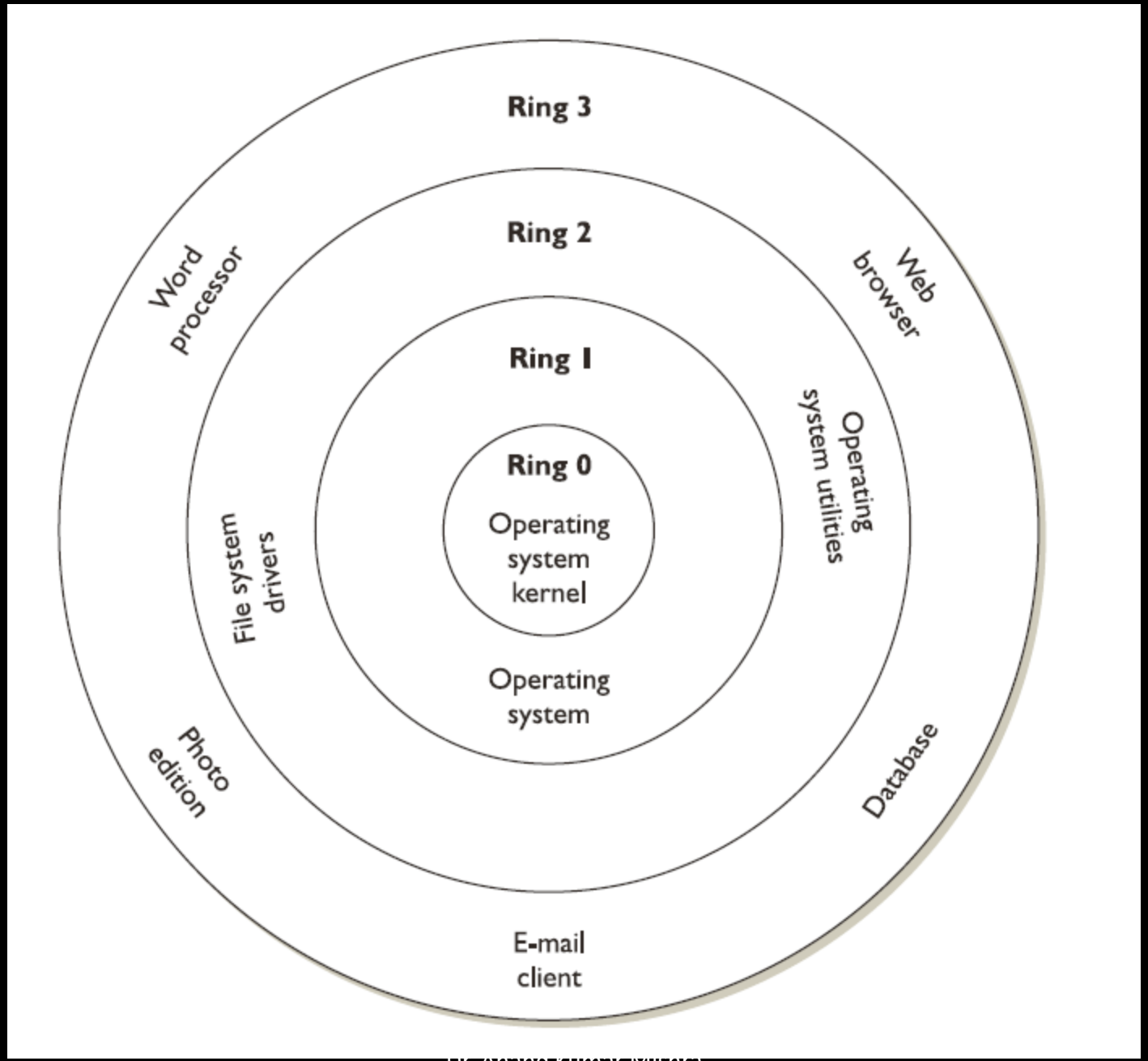
- Operating systems manage computer resources, like processing time on the CPU and accessing the memory
- Since computers run more than one software process, this will bring some issues
- Protection rings are one of the key solutions for sharing resources and hardware
- Protection rings are mechanisms to protect data and functionality from:
  - faults (by improving fault tolerance)
  - malicious behaviour (by providing computer security)



the protection rings for x86 processor architecture

Protection Rings

- Ring 0 - operating system kernel, system drivers
- Ring 1 - equipment maintenance programs, drivers, programs that work with the ports of the computer I / O
- Ring 2 - database management system, the expansion of the operating system
- Ring 3 - applications, user-run


Ring 3
Ring 2
Ring 1
Ring 0
Operating system kernel
Operating system
File system drivers
Operating system utilities
Word processor
Web browser
Photo edition
Database
E-mail client

Operating System Utilities

Windows OS Utilities

- Windows Diagnostics
- Windows Performance Monitor
- Windows Event Viewer
- Windows Registry Editor
- Windows Task Manager

Linux OS Utilities

- ps
- iostate
- vmstat

Ring 0

- The code that runs here is said to be in kernel mode
- Kernel-mode processes have the potential to affect the entire system
  - If something goes wrong here, the system would most likely crash
- This ring has direct access to both CPU and system memory

Ring 3

- User processes running in user mode have access to Ring 3
  - Therefore, this is the least privileged ring
- This is where we'll find the majority of our computer applications

Ring 1 and Ring 2

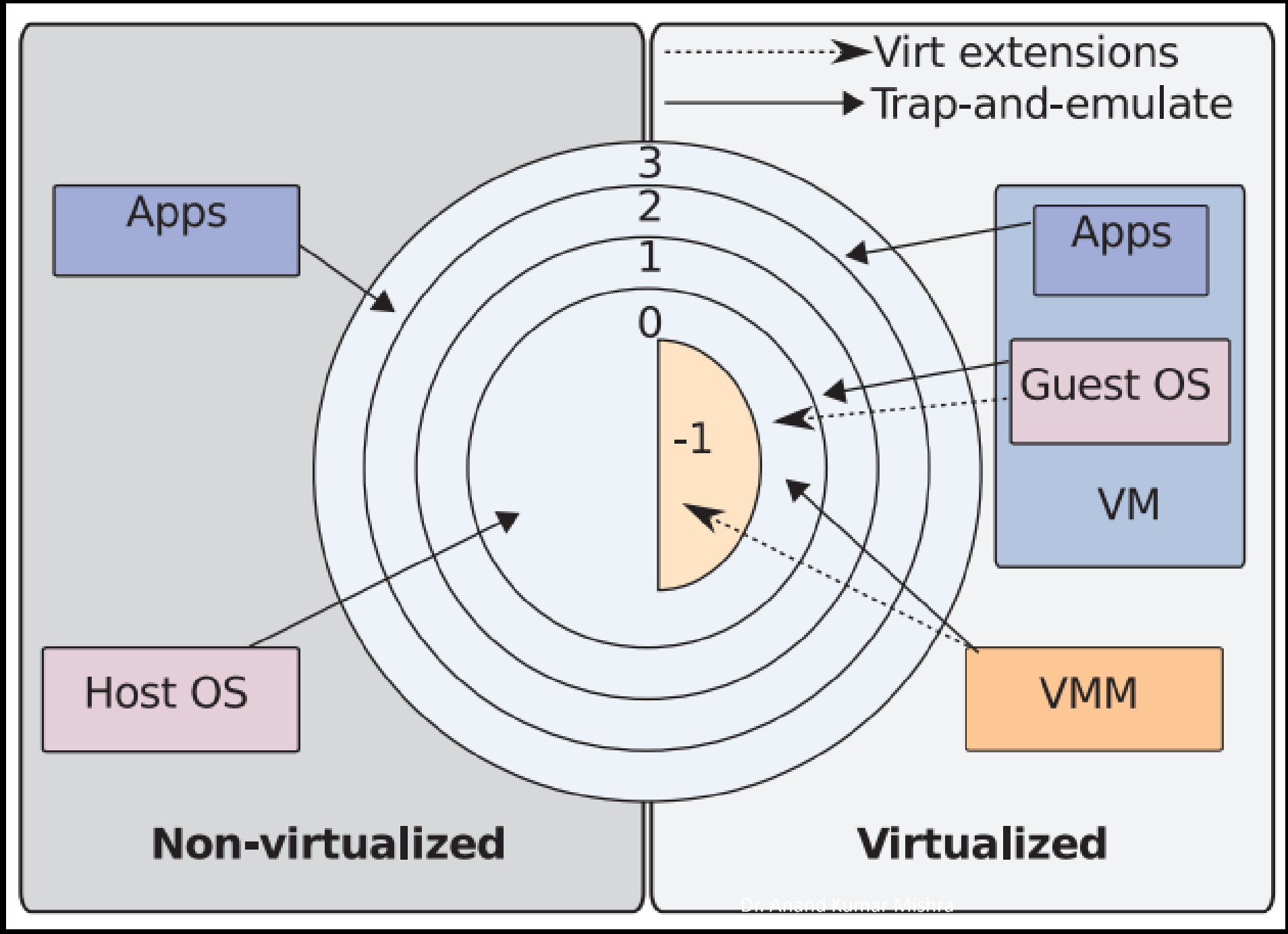
- The OS uses ring 1 to interact with the computer's hardware
  - This ring would need to run commands such as streaming a video through a camera on our monitor
- Instructions that must interact with the system storage, loading, or saving files are stored in ring 2
- These rights are known as input and output permissions because they involve transferring data into and out of working memory, RAM
  - In ring 2, for example, loading an Excel file document from storage
  - In such a case, ring 3 will be responsible for editing and saving the data

Implementation Protection Rings

- Most of the CPU architectures such as x86 include some form of protection rings
- While Linux and Windows use only ring 0 and ring 3, some other operating systems can utilize three different protection levels
- OS such as Linux, macOS, and Windows doesn't fully utilize these feature
- ARM 7 - processor architectures implements three privilege levels: application, operating system, and hypervisor
- ARM 8 implements four protection levels: **+secure monitor level**

Hypervisor Mode

- Modern CPUs offer x86 virtualization instructions for hypervisor to control “Ring 0” hardware access
- In order to help virtualization, VT and Pacifica insert new privilege level below “Ring 0” :
  - Ring −1 and intended to be used by hypervisor



CPU protection ring levels

x86 Architecture

- Its an instruction set architecture (ISA) series for computer processors
  - [An ISA specifies the behavior of machine code and defines how the software controls the CPU]
- Developed by Intel Corporation
- x86 architecture defines **how a processor handles and executes different instructions passed from the operating system (OS) and software programs**
- Designed in 1978, x86 architecture was one of the first ISAs for microprocessor-based computing
- x86 architecture is based on Intel's 8086 microprocessor

x86 architecture

- It primarily handles programmatic functions and provides services, such as:
  - memory addressing
  - software and hardware interrupt handling
  - data type
  - registers and input/output (I/O) management
- Classified by bit amount, the x86 architecture is implemented in multiple microprocessors, including 8086, 80286, 80386, Core 2, Atom and the Pentium series
- Other microprocessor manufacturers, like AMD and VIA Technologies, have adopted the x86 architecture

x86 Key features

- Provides a logical framework for executing instructions through a processor
- Allows software programs and instructions to run on any processor in the Intel 8086 family
- Provides procedures for utilizing and managing the hardware components of a central processing unit (CPU)

AMD Ryzen™ Threadripper™ PRO Processors

Professional workstation users 7nm process technology delivering an unmatched CPU core density for professional workloads and supporting 128 PCIe® 4.0 lanes.

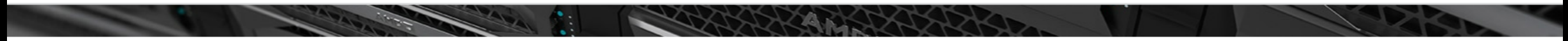
AMD Ryzen™ PRO Processors

With up to 8 cores, delivers modern performance, security features, and seamless management for the most demanding business environments.

AMD Athlon™ PRO Processors

Professional security features, performance, and manageability with exceptional value.

amd.com/en/processors



SERVER **WORKSTATION** **EMBEDDED AND SEMI-CUSTOM** **LAPTOP** **DESKTOP** **CHROMEBOOK**

AMD EPYC™

The world's highest performing x86 server processors[4] setting superior standards for performance, security and scalability.

AMD EPYC™ for Cloud Computing

With more than 170 world records across multiple platforms, AMD has a solution to suit your business needs in private and public cloud environments.[5]

AMD EPYC™ for Database and Analytics

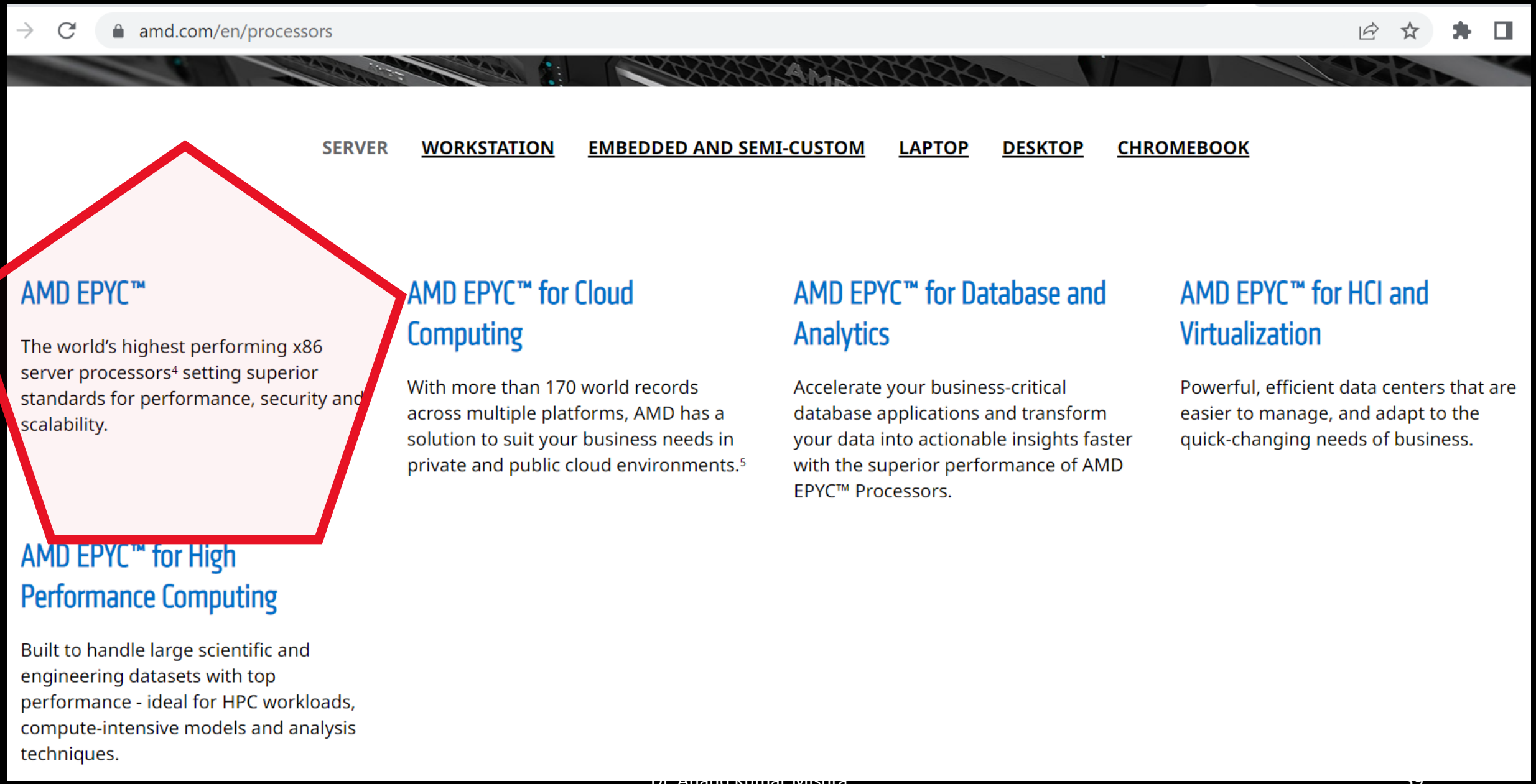
Accelerate your business-critical database applications and transform your data into actionable insights faster with the superior performance of AMD EPYC™ Processors.

AMD EPYC™ for HCI and Virtualization

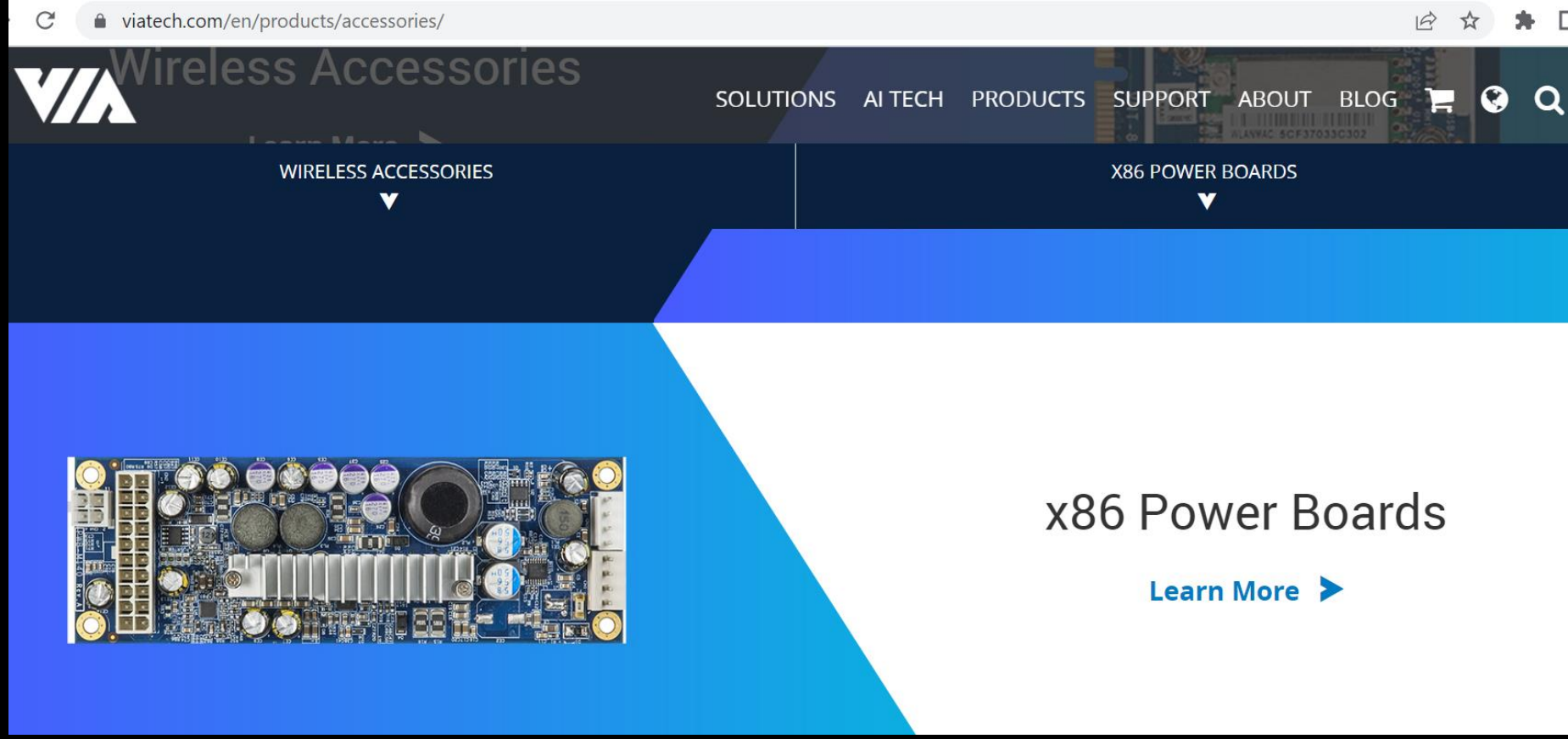
Powerful, efficient data centers that are easier to manage, and adapt to the quick-changing needs of business.

AMD EPYC™ for High Performance Computing

Built to handle large scientific and engineering datasets with top performance - ideal for HPC workloads, compute-intensive models and analysis techniques.


amd.com/en/processors
SERVER
WORKSTATION
EMBEDDED AND SEMI-CUSTOM
LAPTOP
DESKTOP
CHROMEBOOK
AMD EPYC™
The world's highest performing x86 server processors[4] setting superior standards for performance, security and scalability.
AMD EPYC™ for Cloud Computing
With more than 170 world records across multiple platforms, AMD has a solution to suit your business needs in private and public cloud environments.[5]
AMD EPYC™ for Database and Analytics
Accelerate your business-critical database applications and transform your data into actionable insights faster with the superior performance of AMD EPYC™ Processors.
AMD EPYC™ for HCI and Virtualization
Powerful, efficient data centers that are easier to manage, and adapt to the quick-changing needs of business.
AMD EPYC™ for High Performance Computing
Built to handle large scientific and engineering datasets with top performance - ideal for HPC workloads, compute-intensive models and analysis techniques.

VIA KT266A north bridge for Socket A.



- VIA Technologies Inc. is a Taiwanese manufacturer of integrated circuits, mainly motherboard chipsets, CPUs, and memory.
- It was the world's largest independent manufacturer of motherboard chipsets.

intel.in/content/www/in/en/products/details/processors/core.html

	Intel® Core™ X-series Processors	Unlocked CPUs that deliver up to 18 cores for the most extreme gaming, creative production, and multi-tasking.
	Intel® Core™ i9 Processors	Delivering up to 24 cores for seamless 4K Ultra HD and 360-degree video, robust gameplay, and multitasking performance.
	Intel® Core™ i7 Processors	This CPU packs the power of up to 16 cores for accelerated computing supporting high-end gaming, connectivity, and security.
	Intel® Core™ i5 Processors	Experience exceptional performance for home and business PCs with up to 14 cores for gaming, creativity, and multitasking.
	Intel® Core™ i3 Processors	These value-packed processors deliver outstanding performance for everyday tasks.

x86 Hardware Virtualization

- x86 operating systems are designed to run directly on the bare-metal hardware, so they naturally assume they fully ‘own’ the computer hardware
- x86 architecture offers four levels of privilege known as Ring 0, 1, 2 and 3 to operating systems and applications to manage access to the computer hardware
- Virtualizing the x86 architecture requires placing a virtualization layer under the operating system to create and manage the virtual machines that deliver shared resources
  - [Knowing that OS expects to be in the most privileged Ring 0]

VMWare: Understanding Full Virtualization, Paravirtualization, and Hardware Assist