INFO 6010 Lesson 4
Security Architecture & Engineering Part 1
Domain 3
Revision 2

Information Security Management & Network Security Architecture

FANSHAWE

Discussion Topics – Part 1

Part 1

- System architecture
  - Computer Architecture
- System Security Architecture
- Trusted computing base and security mechanisms
- Information security software models
  - Assurance evaluation criteria and ratings
- Certification and accreditation processes
- Systems Security
  - Distributed systems security
- Cloud Computing

Discussion Topics – Part 2

Part 2 (next lesson)

- Cryptography components and their relationships
- Steganography
- Public key infrastructure (PKI)

- Part 3 Will be covered later in the course
- *Site and facility design considerations*
- *Physical security risks, threats, and countermeasures*
- *Electric power issues and countermeasures*
- *Fire prevention, detection, and suppression*

System Architecture

- Security is best if it is designed and built into the foundation of anything we build and not added as an afterthought. Once security is integrated as an important part of the design, it has to be engineered, implemented, tested, evaluated, and potentially certified and accredited.
- The security of a product must be evaluated against the availability, integrity, and confidentiality it claims to provide.

System Architecture

- **Architecture** Fundamental organization of a system embodied in its components, their relationships to each other and to the environment, and the principles guiding its design and evolution.
- **Architecture description (AD)** Collection of document types to convey an architecture in a formal manner.
- **Stakeholder** Individual, team, or organization (or classes thereof) with interests in, or concerns relative to, a system.
- **View** Representation of a whole system from the perspective of a related set of concerns.
- **Viewpoint** A specification of the conventions for constructing and using a view. A template from which to develop individual views by establishing the purposes and audience for a view and the techniques for its creation and analysis.

Computer Architecture

- Computer architecture encompasses all of the parts of a computer system that are necessary for it to function
  - Operating system
  - Memory chips
  - Logic circuits
  - Storage devices
  - Input and output devices
    - Networking component
  - Data, memory and control buses

CPU

- The central processing unit (CPU) is the brain of a computer
  - Fetches instructions from memory and executes them
- An operating system is written to work with the instruction set of a specific CPU
- This is why one operating system may work on a Intel processor but not on a Motorola or SPARC processor

CPU

- The chips within the CPU contain millions of transistors
  - Each transistor holds an electrical voltage, which represents 0's and 1's to the computer
- CPU registers point to memory locations that contain the next instructions to be executed and keep status information of the data that need to be processed
  - A register is a temporary storage location internal to the CPU
- Accessing memory across the system bus to get instructions and data is much slower than accessing a register
  - Speed of processor vs speed of system bus

ALU

- Execution of instructions is done by the arithmetic logic unit (ALU)
  - The ALU performs mathematical functions and logical operations on data
- When action needs to take place on the data, the instructions and data memory addresses are passed to the CPU registers
- The results are sent back to be stored in memory space allocated for the process

Control Unit

- The control unit is the component that fetches the program code and oversees the execution of the different instructions
- It determines what application instructions get processed and assigns a priority time slice
  - Processor allocates time to each running program
  - Multitasking

CPU Registers

- *General* registers hold variables and temporary results as the ALU works through execution steps
- *Special* registers hold information such as the program counter, stack pointer, and program status word (PSW)
- *Program counter* register contains the memory address of the next instruction to be fetched
  - After that instruction is executed, the program counter is updated with the memory address of the next instruction to be processed

CPU

- Program status word (PSW) register holds condition bits for 2 modes
- User mode
  - Used by applications
  - Many CPU instructions and functions are restricted
- Privilege Mode
  - Kernel or supervisor mode
  - Trusted process or system service

CPU Registers

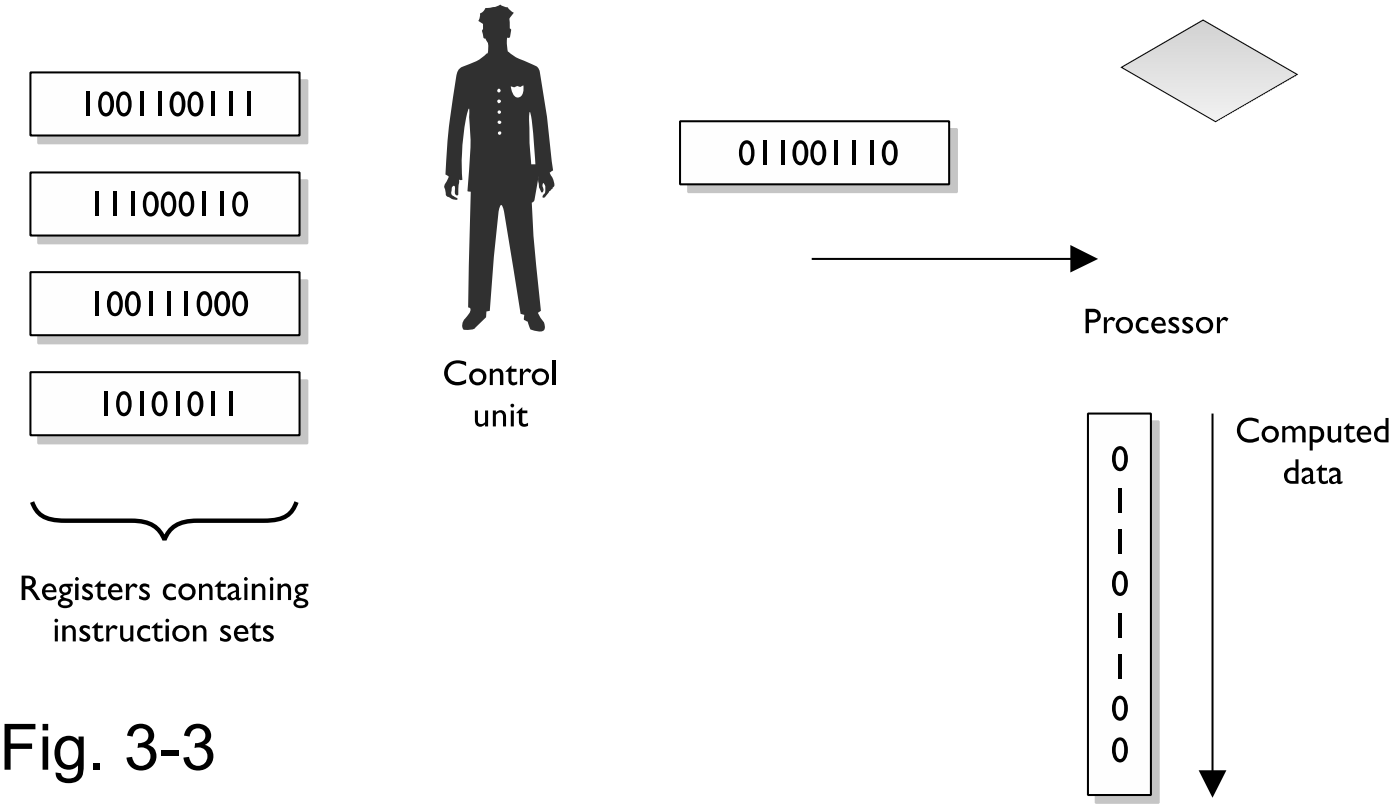


Fig. 3-3

Bus

- The CPU is connected to an *address bus* which is a hardwired connection to the RAM chips on the system and the individual input/output (I/O) devices
- If CPU needs to access data from memory it uses the address bus to open the memory location
- The RAM memory or device reads the memory address location and places the data at that location on the data bus
  - Memory addresses are reserved (assigned) to I/O devices CD-ROM, USB device, hard drive, network adapter

Multiprocessing

- Computers can have more than one CPU for increased performance
- An operating system must be developed specifically to be able to work with more than one processor
  - Symmetric mode the processors are handed work as needed
- When a process needs instructions to be executed a scheduler determines which processor is ready for more work and sends it to that CPU

Memory Types

- The operating system instructions, applications, and data are held in memory
- The basic input/output system (BIOS), device controller instructions and firmware all require memory addresses
- They do not all reside in the same memory location or even the same type of memory

Memory Types

- Random Access Memory (RAM)
  - Facility where data and program instructions can be held temporarily
- It is described as volatile because if the computer's power supply is terminated, then all information within this type of memory is lost
  - Dynamic RAM
- RAM is an integrated circuit made up of millions of transistors and capacitors
- The capacitor is where the actual charge is stored, which represents a 1 or 0 to the system

Memory Types

- Static RAM (SRAM) does not require continuous-refreshing it uses a different technology by holding bits in its memory cells without the use of capacitors
  - Requires more transistors than DRAM
  - Faster than DRAM
  - Takes up more space on the RAM chip
  - More expensive

Memory Types

- Synchronous DRAM (SDRAM)
- Synchronizes signal input and output on the RAM chip with the system CPU
- Coordinates activities with the CPU clock so the timing of the CPU and the timing of the memory activities are synchronized
- Increases the speed of transmitting and executing data.

Memory Types

- Extended data out DRAM (EDO DRAM)
- Faster than DRAM because DRAM can access only one block of data at a time
- EDO DRAM can read the next block of data while the first block is being sent to the CPU for processing
- It has a type of “look ahead” feature that speeds up memory access

Memory Types

- Burst EDO DRAM (BEDO DRAM)
- Works like EDO DRAM in that it can transmit data to the CPU as it carries out the next read option
- But can send more data at once (burst).
- Reads and sends up to four memory address locations in a small number of clock cycles.

Memory Types

- Double data rate SDRAM (DDR SDRAM)
- Carries out read operations on the rising and falling cycles of a clock pulse.
  - Instead of carrying out one operation per clock cycle, it carries out two and thus can deliver twice the throughput of SDRAM
  - Basically, it doubles the speed of memory activities, when compared to SDRAM, with a smaller number of clock cycles.

Clock Pulse

Memory Types

- **Read-Only Memory (ROM)** is a nonvolatile memory type
- When a computer's power is turned off data is still held by the memory chips
- The data in the ROM memory chips data cannot be altered
- Individual ROM chips are manufactured with the program burned in
  - Firmware

Memory Types

- **Programmable read-only memory (PROM)**
- Form of *blank* ROM that can be modified after it has been manufactured
  - Holds program instruction when power removed
- PROM can be programmed only one time because the voltage that is used to write bits into the memory cells actually burns out the fuses that connect the individual memory cells
- The instructions are “burned into” PROM using a specialized PROM programmer device

Memory Types

- **Erasable and programmable read-only memory (EPROM)**
- Can be erased, modified, and upgraded
- EPROM data that can be erased and reprogrammed
- To erase the data on the memory chip you need an ultraviolet (UV) light device

Memory Types

- **Flash memory** is a solid-state memory technology that holds data when powered down
  - Data can be altered
  - It does not have moving parts and is used hold large amounts of data
  - More as a type of hard drive but a movable storage device
- It acts as a ROM technology rather than a RAM technology

Memory Types

- **Cache memory**
  - Type of memory used for high-speed writing and reading activities
  - When the system assumes (through its programmatic logic) that it will need to access specific information many times throughout its processing activities, it will store the information in cache memory so it is easily and quickly accessible
  - Data in cache can be accessed much more quickly than data stored in RAM
  - L1 & L2 cache built into CPU

Memory Mapping

- The OS keeps track of all physical addresses
  - *Absolute* addresses
  - A process uses logical addresses as if it has the whole memory address space starting at 0 Bytes
- The OS maps each process logical address to a physical address
- A *relative* address is the offset added to the logical address to find the absolute address (physical)

Memory Mapping

- When an application makes a request for memory it is allocated a specific memory amount by the OS
  - When the application is done with the memory, it is supposed to tell the operating system to release the memory so it is available to other applications
- Some applications are written poorly and do not indicate to the OS that this memory is no longer in use
- If this happens enough times, the operating system could run out of memory to allocated to other processes
  - DoS attack

Buffer Overflows

- A buffer overflow, or buffer overrun, is a common software coding mistake that an attacker could exploit to gain access to a system.
- This error occurs when there is more data in a buffer than it can handle, causing data to overflow into adjacent storage.
- This vulnerability can cause a system crash or, worse, create an entry point for a cyberattack.
- C and C++ are more susceptible to buffer overflow.
- Secure development practices should include regular testing to detect and fix buffer overflows. These practices include automatic protection at the language level and bounds-checking at run-time.

Operating Systems - Processes

- Applications and programs work as individual units called processes
  - A program is not considered a process until it is loaded into memory and activated by the operating system
- Operating systems have different processes carrying out various types of functionality
- When a process is created the operating system assigns resources:
  - Memory segment
  - CPU time slot (interrupt)
  - Programming interfaces (APIs)
  - Files

Operating Systems - Process Scheduling

- Computers run different applications and processes at the same time
- Processes have to share resources and play nice with each other to ensure integrity
  - Some memory, data files, and variables are actually shared between different processes
- Multiple processes should not read and write to same memory addresses
  - The operating system is the master program that ensures that programs do not corrupt each other's data held in memory

Process States

- Running state
  - CPU is executing its instructions and data
- Ready state
  - Waiting to send instructions to the CPU
- Blocked state
  - Waiting for input data, such as keystrokes from a user
- Operating system is responsible for creating new processes, assigning them resources, synchronizing their communication

Process States

- The operating system keeps a process table
  - One entry per process
- Table contains
  - Process state
  - Memory allocation
  - Stack pointer
  - Program counter
  - Open files in use
- On each time slice information is moved from table to CPU registers and back to table at end of time slice

Process Isolation

- Different methods can be used to carry out process isolation:
  - Encapsulation of objects
  - Time multiplexing of shared resources
  - Naming distinctions
  - Virtual mapping

Process Isolation

- When a process is *encapsulated* no other process interacts with its internal programming code
  - Data hiding
  - Encapsulation provides an interface rule for the process
  - Processes interact through the interface
  - Allows different programming languages to interact via the interface
- *Time Multiplexing* allows multiple process to be assigned a time slice with CPU

Process Isolation

- *Naming distinctions* mean different processes have their own name or identification value
  - Processes are usually assigned process identification (PID) values
  - Operating system and other processes use the process id to call other processes
- *Virtual address space mapping* is used by operating system to assign physical address space to a process
  - All process are written to start at memory address 0
  - OS maps each process memory address to a physical memory address

Memory Stack

- Each process is assigned a stack
  - Area in memory that the process can read from and write to in a last in, first out (LIFO) fashion
  - Scratch pad to hold data variables for process
  - User input
  - Size of each memory buffer for a variable in the stack is determined by the programmer
- Stack pointer
  - Location of data in stack
- Return Pointer
  - Address in main memory to continue main program

Threads

- Applications (processes) have many different functions that can be running at same time
  - A thread is created with the instructions for each function
  - Threads are dynamically created and destroyed as needed
  - Multi-threaded application

Interrupts

- System has both hardware and software interrupts
- When a device needs to communicate with the CPU it signals an interrupt and waits
  - An interrupt informs the CPU another process has requested access to the CPU
- Each software process has an interrupt assigned
- The current process information is now stored in the process table and the process sending the interrupt gets its time to interact with the CPU

Interrupts

- Two categories of interrupts:
- Maskable interrupt is assigned to an event that may not be overly important
  - Programmer can indicate that if that interrupt calls, the program does not stop what it is doing.
  - This means the interrupt is ignored
- Non-maskable interrupts can never be overridden
  - Critical process
  - Reset button

Memory Management

- Memory management is one of the most important tasks of the operating system
- Programmers do not know the amount or type of memory in the system running the program
  - Memory manager hides all of the memory issues and just provides the application with a memory segment

Memory Management

- Goals of memory management:
  - Provide an abstraction level for programmers
    - Abstraction means the details are hidden
  - Maximize performance with the limited amount of memory available
  - Protect the operating system and applications loaded into memory

Memory Management

- Memory manager has five basic responsibilities:

1. Relocation
   - Swap contents from RAM to the hard drive as needed
   - Provide pointers for applications if their instructions and memory segment have been moved to a different location in main memory
2. Protection
   - Limit processes to interact only with the memory segments assigned to them
   - Provide access control to memory segments

Memory Management

- Memory manager has five basic responsibilities:

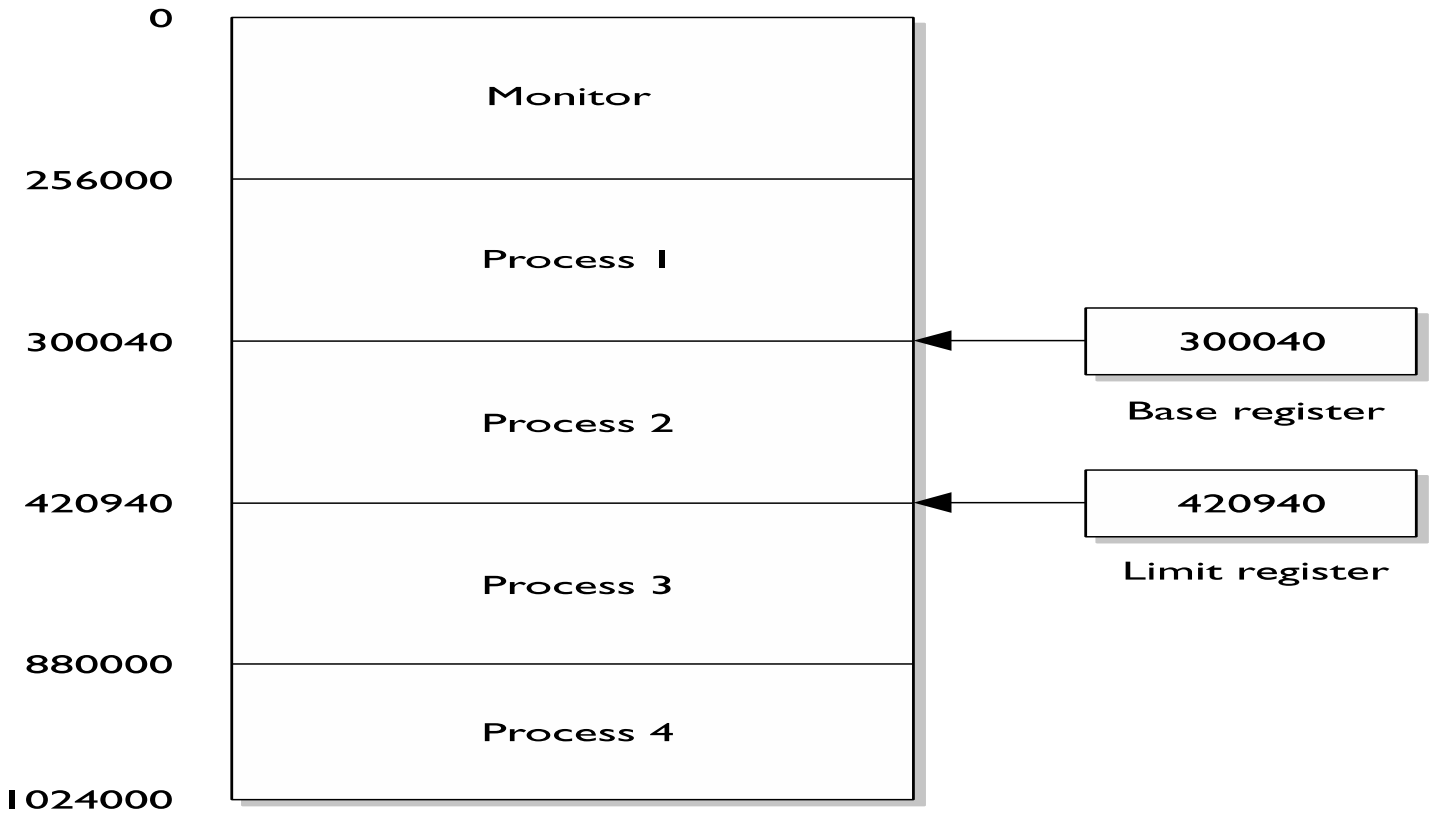
3. Sharing
   - Use complex controls to ensure integrity and confidentiality when processes need to use the same shared memory segments
   - Allow many users with different levels of access to interact with the same application running in one memory segment
4. Logical organization
   - Allow for the sharing of specific software modules such as dynamic link library (DLL) procedures
5. Physical organization
   - Segment the physical memory space for application and operating system processes

Memory Management

- To make sure a process only interacts with its memory segment the CPU uses two registers:
- Base Register
  - Contains the beginning address assigned to the process
- Limit Register
  - Contains the ending address assigned to the process

Memory Management

Fig 3-13



Base and limit registers are used to contain a process in its own memory segment.

Virtual Memory

- *Secondary* storage is nonvolatile storage media
  - Includes the computer's hard drive, floppy disks, and CD-ROMs
- The OS uses hard drive secondary storage space to extend its RAM space
  - When a system fills up its RAM space it writes data from memory onto the hard drive
- Swap space is the reserved hard drive space used to extend RAM capabilities
  - Windows systems use the **pagefile.sys** file to reserve this space
  - When a program requests access to this data it is brought from the hard drive back into memory in specific units called pages

Virtual Memory

- Security issue with using virtual swap space is that when the system is shut down, or processes that were using the swap space are terminated, the pointers to the pages are reset to “available” but the actual data written to disk is still physically written to the tracks and sectors
- Secure operating systems should wipe swap spaces after a process is done
- Should also erase this data before a system shutdown

Input/Output Devices

- Input /output devices are classified as block or character devices
- Block devices
  - Work with fixed sized blocks which have unique addresses
  - Disk drives
- Character devices
  - Work with streams of characters
  - No addressing, no fixed size
  - Printers
  - Network adapters
  - Mouse

Interrupt

- When an I/O device has completed a task it needs to inform the CPU that data is now in memory for processing
- The device's controller sends a signal to the interrupt controller
  - The interrupt controller sends a message to the CPU indicating what device needs attention
- If CPU is busy and the device's interrupt is not a higher priority than whatever job is being processed then the device has to wait

Interrupt

- The operating system has a table (called the interrupt vector) of all the I/O connected devices
- CPU compares the received number with the values within the interrupt vector so it knows which I/O device needs its services
- Table has the memory addresses of the different I/O devices so when the CPU is ready it looks in the table to find the correct memory address.

Interrupt

- One of the main goals of the operating system software that controls I/O activity is to be device independent
  - Commands do not have to included in application
- Operating systems can carry out software I/O procedures in various ways
  - Programmed I/O
  - Interrupt-driven I/O
  - I/O using DMA
  - Premapped I/O
  - Fully mapped I/O

Interrupt

- If an operating system is using *programmable* I/O the CPU sends data to an I/O device and polls the device to see if it is ready to accept more data
- If device is not ready the CPU wastes time waiting for the device to become ready

Interrupt

- With *interrupt-driven* I/O the CPU sends a character over to the printer and then goes and works on another process's request
- When the printer is done printing it sends an interrupt to the CPU
- The CPU stops what it is doing sends another character to the printer and moves to another job
  - This process continues until the whole text is printed
  - This method does waste a lot of time dealing with all the interrupts

Interrupt

- Direct Memory Access (DMA)
- Method of transferring data between I/O devices and the system's memory without using the CPU
  - Speeds up data transfer rates significantly
- The DMA controller feeds the characters to the printer without bothering the CPU
  - This method is sometimes referred to as unmapped I/O

Interrupt

- With *premapped* I/O the CPU sends the physical memory address of the requesting process to the I/O device
- The I/O device is trusted enough to interact with the contents of memory directly
- CPU does not control the interactions between the I/O device and memory

Interrupt

- With *fully mapped* I/O the operating system does not fully trust the I/O device
  - The physical address is not given to the I/O device instead the device works purely with logical addresses and works on behalf (under the security context) of the requesting process
- The operating system does not trust the process or device and acts as the broker to control how they communicate with each other

CPU Modes & Protection Rings

- Stable operating systems must be protected from users and applications
- Must distinguish between operations performed by OS and operations performed by users or applications
- Operating system must keep track of all of these events and ensure none of them violate system's overall security

CPU Modes & Protection Rings

- Protection rings provide strict boundaries and definitions for what the processes within each ring can access and what operations they can execute
- Processes that operate within the inner rings have more privileges than the processes operating in the outer rings
- Inner rings only permit the most trusted components and processes to operate within them
  - Processes that execute within the inner rings are usually referred to as privileged or supervisor mode
  - Processes working in the outer rings are said to execute in user mode

CPU Modes & Protection Rings

- Most commonly used architecture provides four protection rings:
  - Ring 0 Operating system kernel
  - Ring 1 Remaining parts of the operating system
  - Ring 2 I/O drivers and utilities
  - Ring 3 Applications and user activity

CPU Modes & Protection Rings

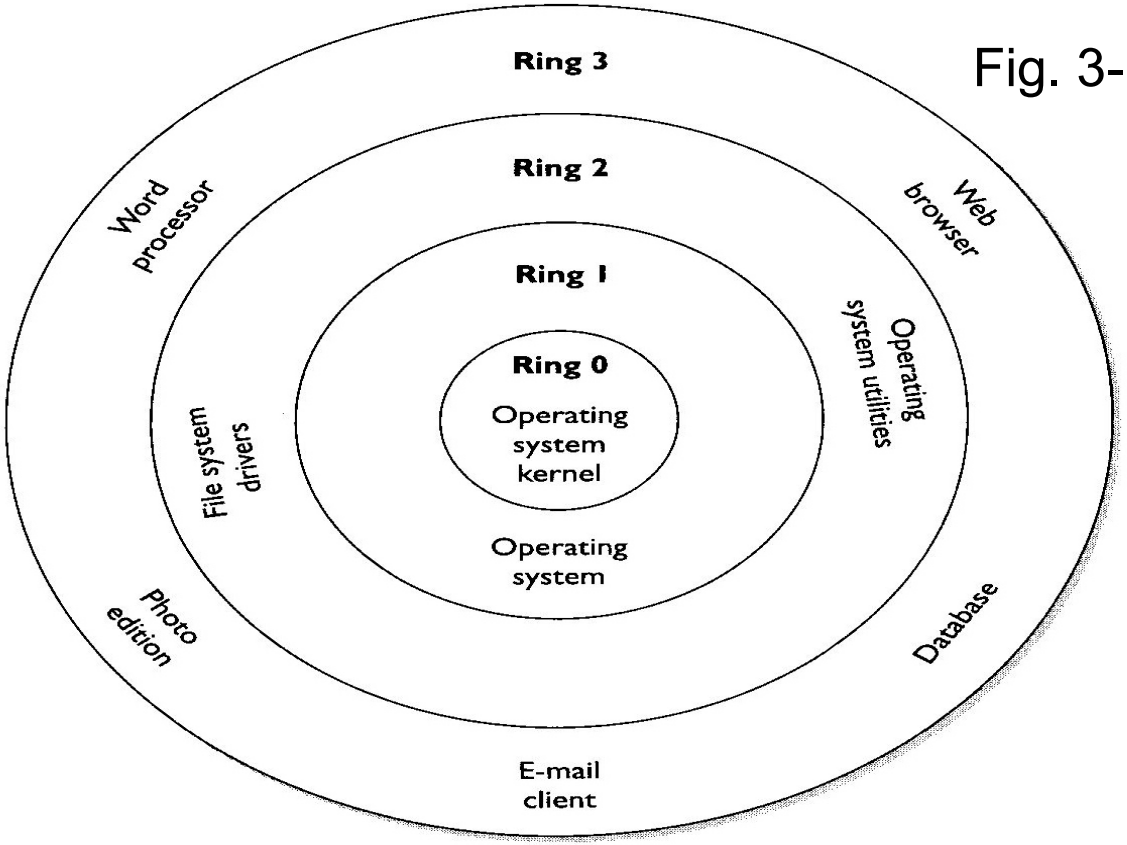


Fig. 3-15

More trusted processes operate within lower numbered rings.

OS Architecture

- A monolithic operating system architecture is commonly referred to as “The Big Mess” because of its lack of structure
  - Operating system is mainly made up of various procedures that can call upon each other in a haphazard manner
- Communication between the different modules is not structured and controlled
- Data hiding is not provided
- MS-DOS is an example of a monolithic operating system

OS Architecture

- A *Layered* operating system architecture separates system functionality into hierarchical layers
- Called THE
  - Technische Hogeschool Eindhoven multiprogramming system
  - 5 layers
- Processes at the different layers have interfaces to be used by processes in layers below and above them.

OS Architecture

- Five layers of functionality
  - Layer 0 controlled access to the processor and provided multiprogramming functionality
  - Layer 1 carried out memory management
  - Layer 2 provided inter-process communication
  - Layer 3 deals with I/O devices
  - Layer 4 applications
  - Layer 5 user layer

OS Architecture

- A *monolithic* operating system provides only one layer of security
- With a *Layered* system each layer provides its own security and access control
- Modularizing software and its code increases the assurance level of the system
- If one module is compromised it does not mean all other modules are now vulnerable
- Examples of layered operating systems:
- VAX/VMS, Multics, and Unix

Virtual Machines

- Original virtual machine was a 16 bit operating system environment created by a 32 bit operating system
- Allowed 32 bit Windows NT to run older DOS applications
- Multiple 16 bit virtual machines could be running on one host operating system
- Backwards compatibility was continued with the introduction of 64 bit operating systems.

Virtual Machines

- Today a virtual instance of an operating system is known as a virtual machine
- A single *host* computer can execute multiple *guest* operating systems
  - Guest operating system share resources such as RAM, processors and storage
- Virtual machines do not directly access these resources
- They communicate with the host environment responsible for managing system resources

Virtual Machines

- Virtual machines can be used to consolidate the workloads of several under utilized servers to fewer machines
- Related benefits
  - Savings on hardware
    - Under utilized server (DHCP)
  - Environmental costs
    - Rack space, power consumption
  - Management and administration
- Legacy applications can run in virtual machines

Virtual Machines

- Virtual machines can be used to provide secure, isolated sandboxes for running untrusted applications
  - Virtualization is an important concept in building secure computing platforms
- Virtual machines can be used to run multiple and different operating systems simultaneously
- Virtualization can make tasks such as system migration, backup, and recovery easier

Virtual Machines

- Virtual machines can provide the illusion of hardware
  - Sometimes hardware that you do not have (SCSI devices, floppy drives or multiple processors)
- Virtualization can also be used to simulate networks of independent computers

Virtual Machines

- Virtual machines allow for testing, debugging and performance monitoring
- Virtual machines can isolate what they run and provide fault and error containment
  - You can inject faults proactively into software to study its subsequent behaviour
- Virtual machines are great tools for research and academic experiments
- Since they provide isolation, they are safer to work with

System Security Architecture

FANSHAWE

System Security Architecture - Security Policy

Security starts at a policy level, with high-level directives that provide the foundational goals for a system overall and the components that make it up from a security perspective.

- A *security policy* is a strategic tool that dictates how sensitive information and resources are to be managed and protected. It expresses the security level by setting the security mechanisms goals.
- This is an important element that has a major role in defining the architecture and design of the system.
- The security policy is a foundation for the specifications of a system and provides the baseline for evaluating a system after it is built.
- The evaluation is carried out to make sure that the goals that were laid out in the security policy were accomplished.

Security Policy

If the following goals are enshrined in policy it will help to achieve secure systems:

- Discretionary access control–based operating system
- Provides role-based access control functionality
- Capability of protecting data classified at “public” and “confidential” levels
- Does not allow unauthorized access to sensitive data or critical system functions
- Enforces least privilege and separation of duties
- Provides auditing capabilities
- Implements trusted paths and trusted shells for sensitive processing activities
- Enforces identification, authentication, and authorization of trusted subjects
- Implements a capability-based authentication methodology
- Does not contain covert channels
- Enforces integrity rules on critical files

System Architecture Requirements

- Not all components need to be trusted and therefore not all components fall within the trusted computing base (TCB).
- The TCB is defined as the total combination of protection mechanisms within a computer system
- The TCB includes hardware, software, and firmware
- These are part of the TCB because the system is sure these components will enforce the security policy and not violate it.

Trusted Computing Base

- Originated from the Orange Book
  - Trusted Computer Security Evaluation Criteria book
  - National Computer Security Center (NCSC)
- Does not address the level of *security* a system provides
- Measures the level of *trustworthiness* a system provides
  - This is because no computer system can be totally secure
- If a system meets a certain criteria it is looked upon as providing a certain level of trust meaning it will react *predictably* in different types of situations
  - Types of attacks and vulnerabilities change and evolve over time and with enough time and resources, most attacks become successful

Trusted Computing Base

- TCB is defined as the total combination of protection mechanisms on a computer system
  - TCB does not address all system components
  - Not all need to be trusted
- TCB does include hardware, software components, and firmware
  - Each can affect the computer's environment in a negative or positive manner, and each has a responsibility to support and enforce the security policy of that particular system
- Some components and mechanisms have direct responsibilities in supporting the security policy, such as firmware that will not let a user boot a computer from a floppy disk, or the memory manager that will not let processes overwrite other processes' data

Trusted Computing Base

- If TCB is enabled then the system has a trusted path, a trusted shell, and system integrity checking capabilities
  - A trusted path is a communication channel between the user, or program, and the kernel
- TCB provides protection resources to ensure this channel cannot be compromised

Security Perimeter

- Not every process and resource falls within the TCB
- Some components fall outside of an imaginary boundary referred to as the *security perimeter*
  - A security perimeter is a boundary that divides the trusted from the untrusted
- For the system to stay in a secure and trusted state, precise communication standards must be developed to ensure that when a component *within* the TCB needs to communicate with a component *outside* the TCB, the communication cannot expose the system to unexpected security compromises
  - This type of communication is handled and controlled through interfaces

Reference Monitor

- The reference monitor has rules for what access subjects have to objects
- Ensure that the subjects have the necessary access rights to protect the objects from unauthorized access
  - For a system to achieve a higher level of trust it must require subjects (programs, users, or processes) to be fully authorized prior to accessing an object (file, program, or resource)
- The reference monitor is an access control concept not an actual physical component

Security Kernel

- The security kernel is the core of the TCB
- The security kernel control all access and functions between subjects and objects
  - The security kernel enforces the reference monitor rules
- Security kernel has three main requirements:
  - It must provide isolation for the processes carrying out the reference monitor rules
  - It must be invoked for every access attempt and must be impossible to circumvent.
  - It must be small enough to be tested and verified in a complete and comprehensive manner.

Security Models

- Security models incorporates the security policy that should be enforced in the system
  - A security model is a symbolic representation of a policy
- It maps the goals of the policy into a set of rules that a computer system must follow
- Specifies explicit data structures and techniques necessary to enforce the security policy

Security Models

- The security model is represented by mathematical formulas, and analytical ideas which are mapped to system specification
- Specifications are developed per operating system type (Unix, Windows, Macintosh)
- Individual vendors can decide how they are going to implement mechanisms that meet these necessary specifications

Security Models

- **The *Bell-LaPadula* security model** enforce rules to provide *confidentiality* protection
- **The *Biba* security model** enforce rules to provide *integrity* protection
- Formal security models, such as Bell-LaPadula and Biba, are used to provide *high assurance* in security
- Informal models, such as *Clark-Wilson*, are used more as a *framework* to describe how security policies should be expressed and executed

Bell-LaPadula Model

- Developed in the 1970s
  - U.S. military used time-sharing mainframe systems and was concerned about the security of these systems and leakage of classified information
- The first mathematical model of a multilevel security policy used to define the concept of a secure state machine and modes of access and outlined rules of access
- The Bell-LaPadula model is a state machine model that enforces the *confidentiality* aspects of access control

Bell-LaPadula Model

- A system that employs the Bell-LaPadula model is called a multilevel security system
- Users with different clearances use the system
- System processes data with different classifications
- Level at which information is classified determines the handling procedures
  - The subject's clearance is compared to the object's classification and then specific rules are applied to control how subject-to-object interactions can take place

Bell-LaPadula Model

- The Bell-LaPudula model is a subject-to-object model
  - Uses subjects, objects, access operations (read, write, and read/write) and security levels
- Subjects and objects can reside at different security levels and will have relationships and rules dictating the acceptable activities between them
- Uses a lattice of security levels (top secret, secret, sensitive, and so on)

Bell-LaPadula Model

- Three main rules are used and enforced in the Bell-LaPadula model:
- The simple security rule
  - Subjects at a given security level cannot read data that reside at a higher security level (no read up rule)
- The *-property (star property) rule
  - Subjects in a given security level cannot write information to a lower security level (no write down rule)
- The strong star property rule
  - Subjects that have read and write permissions at the same security level, nothing higher and nothing lower

Bell-LaPadula Model

- The Bell-LaPadula model was developed to provide *confidentiality*
- This model does not address the *integrity* of the data the system maintains
- Only who can and cannot access the data and what operations can be carried out

Multilevel Security Mode

- Permits two or more classification levels of information to be processed at the same time
- Not all of the users have the same clearance level or formal approval to access all the information being processed by the system
- User can only access certain files
  - Must have formal approval, NDA, need to know, and the necessary clearance to access the data that they need to carry out their jobs
- The **Bell-LaPadula** model is an example of a multilevel security model because it handles multiple information classifications at a number of different security levels within one system simultaneously

The Biba Model

- Biba addresses the *integrity* of data within applications
- The Biba model is not concerned with security levels and confidentiality, so it uses a lattice of integrity levels
  - The Biba model was developed after the Bell-LaPadula model
  - It is a state machine model and is very similar to the Bell-LaPadula model

The Biba Model

- The Biba model prevents data from any integrity level from flowing to a higher integrity level
- Biba has three main rules to provide this type of protection:
- *-integrity axiom
  - A subject cannot write data to an object at a higher integrity level (referred to as “no write up”). *For “star” think **writing**.*
- Simple integrity axiom
  - A subject cannot read data from a lower integrity level (referred to as “no read down”). *For “Simple” think **reading**.*
- Invocation property
  - A subject cannot request service (invoke) to subjects of higher integrity

Clark-Wilson Model

- The Clark-Wilson model was developed after Biba and takes some different approaches to protecting the integrity of information
- Separates data into subsets
  - Highly protected data is referred to as a **constrained data item** (CDI)
  - data that does not require a high level of protection which is called an **unconstrained data item** (UDI)
- Users cannot modify critical data (CDI) directly

Clark-Wilson Model

- This model uses the following elements:
- Users
  - Active agents
- Transformation procedures (TPs)
  - Programmed abstract operations, such as read, write, and modify
- Constrained data items (CDIs)
  - Can be manipulated only by TPs
- Unconstrained data items (UDIs)
  - Can be manipulated by users via primitive read and write operations
- Integrity verification procedures (IVPs)
  - Check the consistency of CDIs with external reality

Clark-Wilson Model

- The subject (user) must be authenticated to a piece of software
- The software procedures (TPs) will carry out the operations on behalf of the user
- Using TPs to modify CDIs is referred to as a well-formed transaction
- The Clark-Wilson model also outlines how to incorporate separation of duties into the architecture of an application

Non-Interference Model

- This type of model does not concern itself with the flow of data but rather with what a subject knows about the state of the system
  - If a lower-level entity was aware of a certain activity that took place by an entity at a higher level and the state of the system changed for this lower-level entity, the entity might be able to deduce too much information about the activities of the higher state which in turn is a way of leaking information
  - Users at a lower security level should not be aware of the commands executed by users at a higher level and should not be affected by those commands in any way

Covert Channels

- A covert channel is a way for an entity to receive information in an unauthorized manner
  - Information flow is not controlled by a security mechanism
  - This type of information path was not developed for communication thus the system does not properly protect this path
- Receiving information in this manner clearly violates system's security policy

Covert Channels

- Developers never envisioned information being passed in this way
- The channel to transfer this unauthorized data is the result of one of the following conditions:
  - Improper oversight in the development of the product
  - Improper implementation of access controls within the software
  - Existence of a shared resource between the two entities
- Not all covert channels can be eliminated

Brewer & Nash Model

- Also called the Chinese Wall model was created to provide access controls that can change dynamically depending upon a user's previous actions
  - The main goal of the model is to protect against conflicts of interest by users' access attempts
- User A works on confidential files in directory Z
  - User A should not have access to files in Directory X
  - If user A does get read access to directory X it cannot write to directory Z

Graham-Denning Model

- Addresses and defines a set of basic rights in terms of commands that a specific subject can execute on an object
- These things may sound insignificant but when you're building a secure system they are critical

Graham-Denning Model

- This model has eight primitive protection rights or rules of how these types of functionalities should take place securely:
  - How to securely create an object
  - How to securely create a subject
  - How to securely delete an object
  - How to securely delete a subject
  - How to securely provide the read access right
  - How to securely provide the grant access right
  - How to securely provide the delete access right
  - How to securely provide transfer access rights

Harrison-Ruzzo-Ullman Model

- The HRU security model (Harrison, Ruzzo, Ullman model) is an operating system level, computer security model, which deals with the integrity of access rights in the system.
- It is an extension of the Graham-Denning model, based around the idea of a finite set of procedures being available to edit the access rights of a subject on an object .
- It is named after its three authors, Michael A. Harrison, Walter L. Ruzzo and Jeffrey D. Ullman.

System Evaluation Methods

- A security evaluation examines the security-relevant parts of a system, meaning the TCB:
  - Access control mechanisms
  - Reference monitor
  - Kernel
  - Protection mechanisms
- The relationship and interaction between these components are also evaluated

System Evaluation Methods

- There have been different methods of evaluating and assigning assurance levels to systems.
- Methods and ideologies have evolved over time.
- Historically there were different approaches to evaluating and assigning assurance levels to a system.
- Now there is a framework known as the Common Criteria which is the only one of global significance.

Common Criteria

- In 1990 the International Organization for Standardization (ISO) identified the need of international standard evaluation criteria to be used globally
- The Common Criteria project started in 1993
- Several organizations came together to combine and align existing and emerging evaluation criteria
  - Organizations within the United States, Canada, France, Germany, the United Kingdom, and the Netherlands

Common Criteria

- Under the Common Criteria model
- Evaluation is carried out on a product and it is assigned an Evaluation Assurance Level (EAL)
- The Common Criteria has seven assurance levels
- The range is from EAL1, where functionality testing takes place to EAL7 where thorough testing is performed and the system design is verified

Common Criteria

- EAL1 – Functionally tested
- EAL2 – Structurally tested
- EAL3 – Methodically tested and checked
- EAL4 – Methodically designed, tested, and reviewed
- EAL5 – Semi formally designed and tested
- EAL6 – Semi formally verified design and tested
- EAL7 – Formally verified design and tested

Common Criteria

- Common Criteria works to answer two basic questions about products being evaluated:
  - What does the security mechanisms do (functionality)
  - How sure are you that the mechanisms work (assurance)
- This system sets up a framework that enables
  - Consumers to clearly specify their security issues and problems
  - Developers to specify their security solution to those problems
  - Evaluators to unequivocally determine what the product actually accomplishes

Certification

- Certification is the comprehensive technical evaluation of the security components and their *compliance* for the purpose of *accreditation*
- To assess the appropriateness of a specific system a certification process may use
  - safeguard evaluation
  - risk analysis
  - verification
  - testing
  - auditing techniques

Certification

- A company that specializes in certification will perform the necessary procedures to certify the systems
- The evaluation team will perform tests on the software configurations, hardware, firmware, design, implementation, system procedures, and physical and communication controls

Accreditation

- Accreditation is the formal acceptance of the adequacy of a system's overall security and functionality by management
- The certification information is presented to the responsible body (management) and they ask questions, review the reports and findings, and decide whether to accept the product and whether any corrective action needs to take place

Accreditation

- Once satisfied with the system's overall security as presented management makes a formal accreditation statement
  - Sign off on the document
- By doing this management is stating it understands the level of protection the system will provide in its current environment and understands the security risks associated with installing and maintaining this system

Open Systems

- Systems described as open are built upon standards, protocols, and interfaces that have published specifications,
  - Third-party vendors can develop add-on components and devices
- This type of architecture provides interoperability between products created by different vendors
  - Vendors follow specific standards and provide interfaces that enable each system to easily communicate with other systems and allow add-ons to hook into the system easily

Open Systems

- A majority of the systems in use today are open systems.
- The reason an administrator can have Windows, Macintosh, and Unix computers communicating easily on the same network is because these platforms are open
- If a software vendor creates a closed system, it is restricting its potential sales to proprietary environments.

Closed Systems

- Systems referred to as closed use an architecture that does not follow industry standards.
- Closed systems are proprietary
  - Interoperability and standard interfaces are not employed to enable easy communication between different types of systems and add-on features
  - The system can only communicate with like systems
- A closed architecture can provide more security because it does not have as many doorways in, and it operates in a more secluded environment than open environments

Closed Systems

- Because a closed system is proprietary, there are not as many tools to thwart the security mechanisms and not as many people who understand its design, language, and security weaknesses and thus exploit them
- A majority of the systems today are built with open architecture to enable them to work with other types of systems, easily share information, and take advantage of the functionality that third-party add-ons bring

Client Based Systems

- Client-based systems are combined into applications that execute entirely on one user device (such as a workstation or smartphone). The software is installed on a specific computer and the user can interact with it with no network connectivity.

- One of the main vulnerabilities of client-based systems is that they tend to have weak authentication mechanisms. This means an adversary who gains access to the application would be able to access its data on local or even remote stores.

- The data is usually stored in plaintext, which means that even without using the application, the adversary could easily read the data.

Client Server Systems

- A client/server (also called server-based) system requires that two (or more) separate applications interact with each other across a network connection in order for users to benefit from them. A common example of a client-server application is a web browser, which is designed to connect to a web server.
- Websites are simply HTML files served up to your browser (client). The difference between a client vs server based application is where the files are stored.

Distributed systems

- A **distributed system** is a **system** whose components are located on different networked computers, which communicate and coordinate their actions by passing messages to one another. The components interact with one another in order to accomplish collective tasks or common goals.

Cloud Computing

FANSHAWE

Cloud Computing

- Cloud Computing is a general term used to describe a class of network based computing that takes place over the Internet,
  - basically a step on from Utility Computing
  - a collection/group of integrated and networked hardware, software and Internet infrastructure (called a platform).
  - Using the Internet for communication and transport provides hardware, software and networking services to clients
- These platforms hide the complexity and details of the underlying infrastructure from users and applications by providing very simple graphical interface or API (Applications Programming Interface).

What is Cloud Computing?

- In addition, the platform provides on demand services, that are always on, anywhere, anytime and any place.
- Pay for use and as needed, elastic
  - scale up and down in capacity and functionalities
- The hardware and software services are available to
  - general public, enterprises, corporations and businesses markets

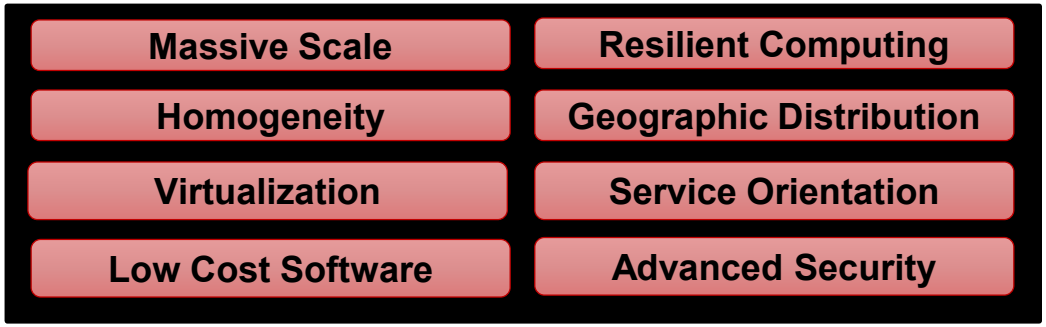
Cloud Summary

- Cloud computing is an umbrella term used to refer to Internet based development and services

- A number of characteristics define cloud data, applications services and infrastructure:
  - Remotely hosted: Services or data are hosted on remote infrastructure.
  - Ubiquitous: Services or data are available from anywhere.
  - Commodified: The result is a utility computing model similar to traditional that of traditional utilities, like gas and electricity - you pay for what you would want!

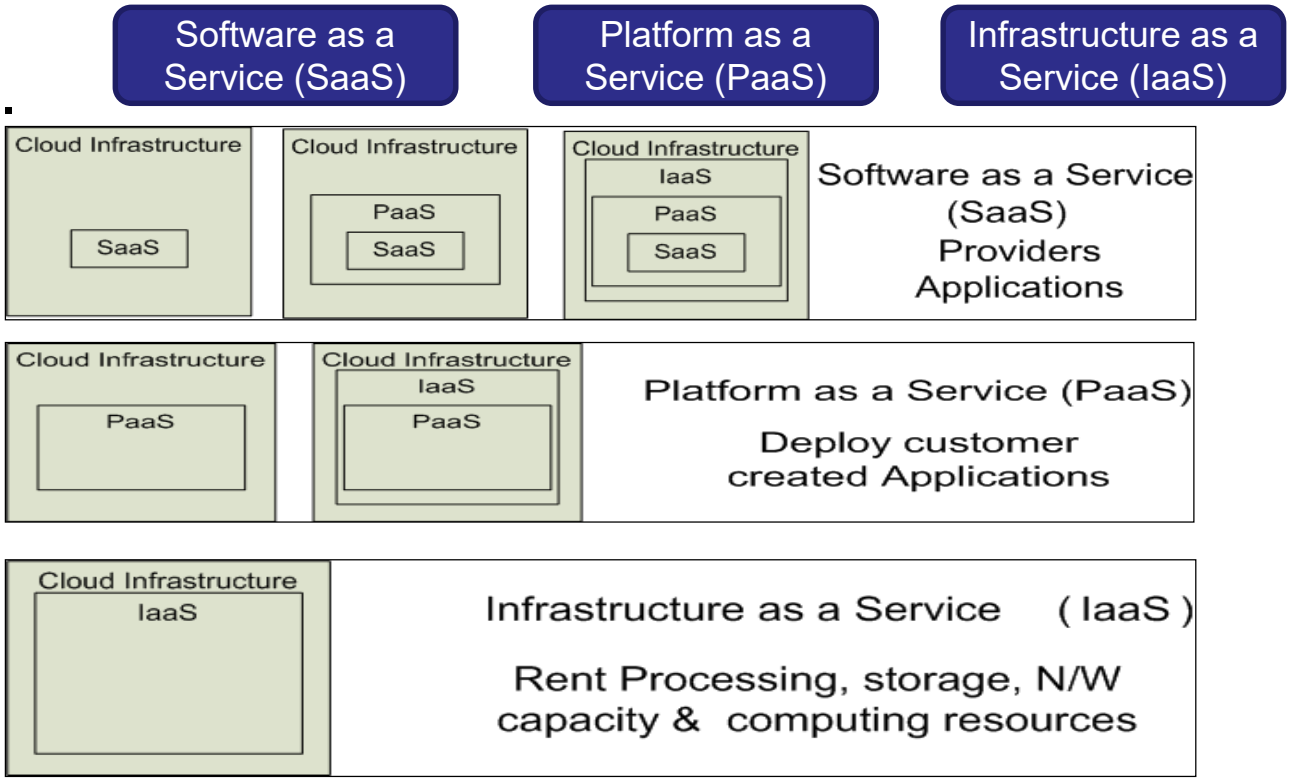
Cloud Computing Characteristics

Common Characteristics:



Essential Characteristics:

Cloud Service Models



Advantages of Cloud Computing

- Easier group collaboration:
  - Sharing documents leads directly to better collaboration.
  - Many users do this as it is an important advantages of cloud computing
    - multiple users can collaborate easily on documents and projects
- Device independence.
  - You are no longer tethered to a single computer or network.
  - Changes to computers, applications and documents follow you through the cloud.
  - Move to a portable device, and your applications and documents are still available.

Disadvantages of Cloud Computing

- Requires a constant Internet connection:
  - Cloud computing is impossible if you cannot connect to the Internet.
  - Since you use the Internet to connect to both your applications and documents, if you do not have an Internet connection you cannot access anything, even your own documents.
  - A dead Internet connection means no work and in areas where Internet connections are few or inherently unreliable, this could be a deal-breaker.

Disadvantages of Cloud Computing

- Does not work well with low-speed connections:
  - Similarly, a low-speed Internet connection, such as that found with dial-up services, makes cloud computing painful at best and often impossible.
  - Web-based applications require a lot of bandwidth to download, as do large documents.
- Features might be limited:
  - This situation is bound to change, but today many web-based applications simply are not as full-featured as their desktop-based applications.
    - For example, you can do a lot more with Microsoft PowerPoint than with Google Presentation's web-based offering

Disadvantages of Cloud Computing

- Can be slow:
  - Even with a fast connection, web-based applications can sometimes be slower than accessing a similar software program on your desktop PC.
  - Everything about the program, from the interface to the current document, has to be sent back and forth from your computer to the computers in the cloud.
  - If the cloud servers happen to be backed up at that moment, or if the Internet is having a slow day, you would not get the instantaneous access you might expect from desktop applications.

Disadvantages of Cloud Computing

- Stored data might not be secure:
  - With cloud computing, all your data is stored on the cloud.
    - The questions is How secure is the cloud?
  - Can unauthorised users gain access to your confidential data?
- Stored data can be lost:
  - Theoretically, data stored in the cloud is safe, replicated across multiple machines.
  - But on the off chance that your data goes missing, you have no physical or local backup.
    - Put simply, relying on the cloud puts you at risk if the cloud lets you down.

Parallel Computing

- Most CPUs can only execute a single instruction at a time, which can cause bottlenecks. ***Parallel computing*** is the simultaneous use of multiple computers to solve a specific task by dividing it among the available computers.

- This division of labour can happen at one of three levels: bit, instruction, or task.
- ***Bit-level parallelism*** takes place in every computing device. When the CPU performs an instruction on a value stored in a register, each bit is processed separately through a set of parallel gates.

- ***Instruction-level parallelism*** allows two or more program instructions to be executed simultaneously.

- ***Task-level parallelism*** takes place at a higher level of abstraction. In it we divide a program into tasks or threads and run each in parallel.

- ***Data parallelism*** describes the distribution of data among different nodes that then process it in parallel. It is related to task parallelism, but is focused on the data.

Database Systems

Database Security Issues

- 2 database security issues are *aggregation* and *inference*
- *Aggregation* happens when a user only has clearance or permission to access some information
  - Can then figure out other data by combining information it can access
- *Inference* is the intended result of aggregation
  - The *inference* problem happens when a subject deduces the full story from the pieces learned of through *aggregation*

Database Security Issues

- Content-dependent access control
  - Based on the sensitivity of the data
  - The more sensitive the data the smaller the subset of subjects (individuals) who can gain access to the data
- Context-dependent access control
  - Software “understands” what actions should be allowed based upon the sequence of the request
  - Blocks access to prevent aggregation of information
- Cell suppression
  - Used to hide specific cells that contain information that could be used in inference attacks

Database Security Issues

- Partitioning a database
  - Dividing the database into different parts which makes it much harder for an unauthorized individual to find connecting pieces of data that can be brought together and other information that can be deduced or uncovered
- Noise and perturbation
  - Inserting bogus or false information into database
  - Misdirecting an attacker or confusing the matter enough that the actual attack will not get useful information

Web Based Systems

- Websites can be exploited via vulnerabilities arising from thoughtless programming.
- The clearer and simpler a website is, the easier it is to analyze its various security aspects. Once a website has been strategically analyzed, the user-generated input fed into the website also needs to be critically scrutinized.
- All input must be considered unsafe, and ought to be sanitized before being processed. All output generated by the system should also be filtered to ensure private or sensitive data is not being disclosed.

Encryption helps secure the input/output operations of a web application. That data may be intercepted by malicious users, but should only be readable, or modifiable, by those with the secret key used to encrypt it.

- In the event of an error, websites ought to be designed to behave in a predictable and noncompromising manner. This is also generally referred to as ***failing securely***.
- Web application firewalls (WAFs) inspect traffic going to (or coming from) a web application in order to filter out potentially malicious content. As WAF's are separate from the web application, it provides an added layer of defense that can be independently tuned without having to rewrite or reconfigure the web application.

Mobile Systems

- Mobile devices are small computers that can connect to websites and various devices, and are entry points for malicious activities.
- They are penetrated through various attack vectors, compromised by malware, sensitive data is stolen from them, denial-of service attacks can take place.
- The largest hurdle of securing any mobile or portable device is that people do not always equate them to providing as much risk as a workstation or laptop.
- The following are some of the issues with using mobile devices in an enterprise:
  - False base stations can be created.
  - Confidential data can be stolen.
  - Camera and microphone functionality can be used improperly.
  - Internet sites can be accessed in violation of company policies.
  - Malicious code can be downloaded.
  - Encryption can be weak and not end to end.

Cyber Physical Systems

Cyber-Physical Systems

- It is often easier, better, faster, and cheaper for computers to control physical devices and systems than for people to do so.

- Any system in which computers and physical devices collaborate via the exchange of inputs and outputs to accomplish a task or objective is a *cyber-physical system*. These systems fall into two classes as follows:

Embedded Systems

- These systems are cheap, rugged, small, and use very little power. The computing device is part of (or embedded into) a mechanical or electrical device or system.

- Embedded systems are usually built around microcontrollers, which are specialized devices that consist of a CPU, memory, and peripheral control interfaces.

- Some of the very features that make embedded systems useful are also the source of many vulnerabilities.

Internet Of Things

The Internet of Things (**IoT**) is a system of interrelated computing devices, mechanical and digital machines, objects, that are provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction.

Challenges include:

- Authentication is seldom present.
- Encryption not often used as it requires too much processor power
- Updates are not often provided

Industrial Systems

Industrial Control Systems

- Industrial control systems (ICS) consist of information technology that is specifically designed to control physical devices in industrial processes. ICS exist on factory floors to control conveyor belts and industrial robots.

SCADA

SCADA is an acronym for **supervisory control and data acquisition**, a computer system for gathering and analyzing real time data. **SCADA** systems are used to monitor and control a plant or equipment in industries such as telecommunications, water and waste control, energy, oil and gas refining and transportation.

ICS (Industrial Control systems) Security recommendations:

- Apply a risk management process to ICS.
- Segment the network to place IDS/IPS at the subnet boundaries.
- Disable unneeded ports and services on all ICS devices.
- Implement least privilege through the ICS.
- Use encryption wherever feasible.
- Ensure there is a process for patch management.
- Monitor audit trails regularly.

Maintenance Hooks

Maintenance hooks are a type of backdoor; they are shortcuts installed by system designers and programmers to allow developers to bypass normal system checks during development, such as requiring users to authenticate. Maintenance hooks become a security issue if they are left in production systems.

Time of Check Attacks

- Some attacks take advantage of the way a system processes requests and performs tasks. A time-of-check to time-of-use (**TOCTOU**, TOCTTOU or **TOC/TOU**) is a class of software bugs caused by a race condition involving the checking of the state of a part of a system (such as a security credential) and the use of the results of that check

Summary

- The CISSP exam will test the following areas
- System architecture
- Operating system and hardware
  - CPU, registers, RAM, ROM interrupts & I/O functions
- Information security software models
  - Bella-LaPadula, Biba etc
- Trusted computing base and security mechanisms
- Assurance evaluation criteria and ratings
- Certification and accreditation processes
- Distributed systems security

Homework

- Read the relevant chapter in the set book 'All In One CISSP Exam Guide' – by Shon Harris.

- Depending on which edition you have the relevant sections will be in different places – so use the index.

- Then identify and do the practice m/c questions relating to this subject.

Questions

- ?