



Mississippi State
UNIVERSITY

J. A. “Drew” Hamilton, Jr., Ph.D.
Director, Distributed Analytics & Security Institute
Director, Center for Cyber Innovation
Professor, Computer Science & Engineering

CCI
Post Office Box 9627
Mississippi State, MS 39762

Voice: (662) 325-2294
Fax: (662) 325-7692
hamilton@cci.msstate.edu



Mississippi State University Center for Cyber Innovation

Domain 3 Security Engineering



Outline

(Engineering Management of Security) 13%

- **Engineering processes using secure design principles**
- **Security models fundamental concepts**
- **Security evaluation models**
- **Security capabilities of information systems**
- **Security architectures, designs, and solution elements vulnerabilities**
- **Web-based systems vulnerabilities**
- **Mobile systems vulnerabilities**
- **Embedded devices and cyber-physical systems vulnerabilities**
- **Cryptography**
- **Site and facility design secure principles**
- **Physical security**



Engineering processes using secure design principles

Dr. Patrick Pape
Shon Harris



Mississippi State University Center for Cyber Innovation

Domain 3 Security Engineering

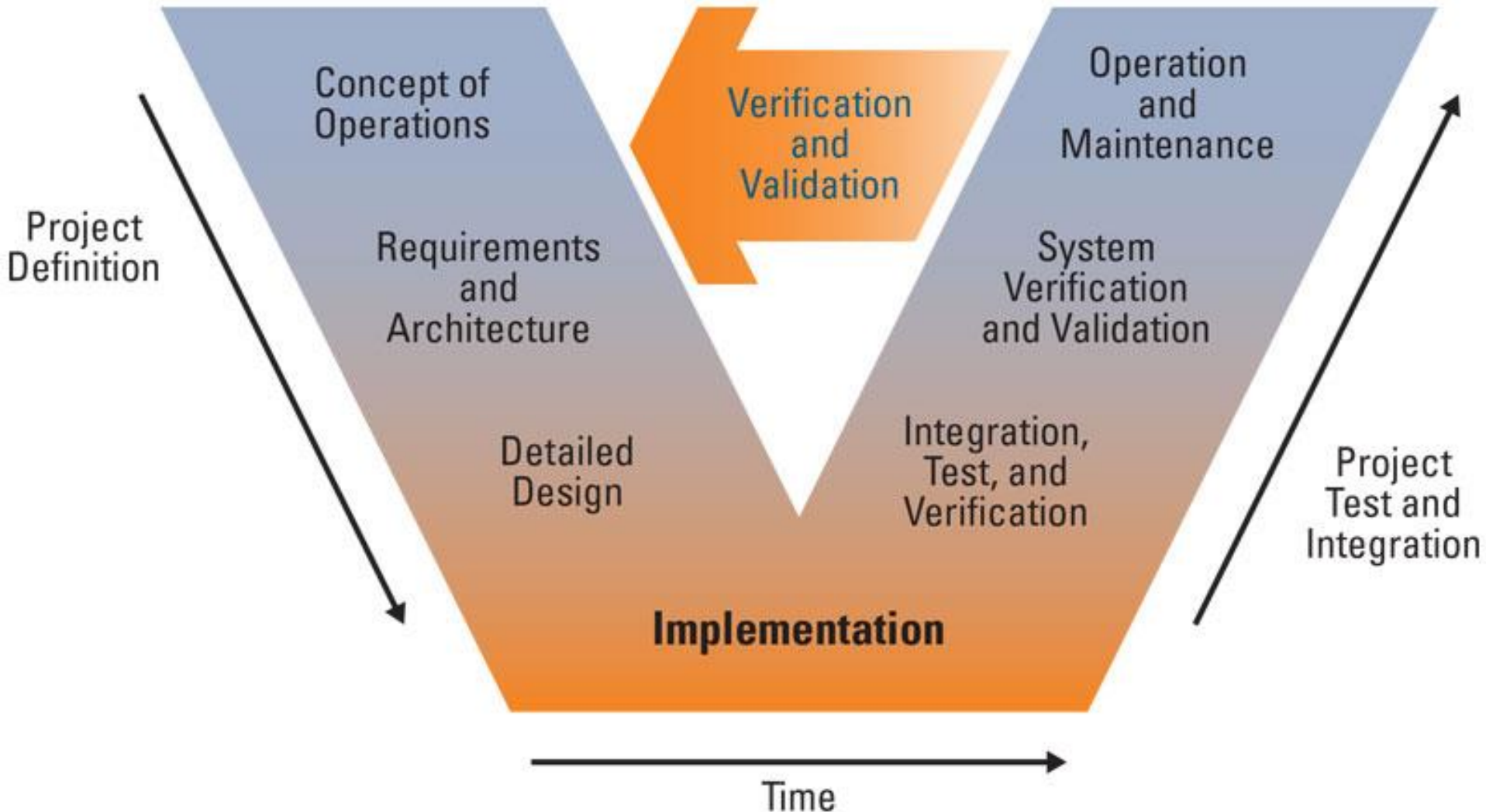


Chapter Outline

- **Computer hardware architecture**
- **Operating system architectures**
- **Trusted computing base and security mechanisms**
- **Protection mechanisms within an operating system**
- **Various security models**
- **Assurance evaluation criteria and ratings**
- **Certification and accreditation processes**
- **Attack types**



V- Model



Security Concepts

- **Security policy** – statement that outlines how entities access each other, what operations they can perform, the level of protection required, and the actions taken in the event that the previous requirements are not met. Applies to software and hardware compliance
- **Security model** – outlines the requirements needed to support and implement a security policy. Provides details on how the policy is upheld.
 - **Note: Individual systems and devices can have their own security policies and dictate the level of security that should be provided.**



Computer Security

- **Availability – Prevention of loss of, or loss of access to, data and resources**
- **Integrity – Prevention of unauthorized modification of data and resources**
- **Confidentiality – Prevention of unauthorized disclosure of data and resources**
- **The provided level of security of a system is determined based on these principals.**
- **Security is best implemented into the design of a system and not integrated as an afterthought.**



Architecture – CPU

- **The CPU fetches instructions from memory and executes them, it is the brain of the computer. Each type of CPU has a specific set of instructions and architecture and the OS must be compliant.**
- **Arithmetic logic unit (ALU) – actually executes the instructions, including math functions and logical operations on data located in memory.**
- **Control unit – manages and synchronizes the system while application code and OS instructions are executed, fetches**



CPU

- **Registers**

- **General** – used to hold variables and temporary data while the ALU executes the instructions
- **Special** – dedicated registers for holding specific data, such as the stack pointer, program counter
- **Program counter** – contains the memory address of the next instruction to be fetched, updated after each instruction is executed
- **Stack pointer** – directs the CPU to the information located on the stack as the stack is traversed



CPU

- **Program status word (PSW)** – register that holds different condition bits. For example, one bit determines if the computer should be running in user or kernel mode
- Computer in the user mode runs in a lower privilege mode to prevent malicious applications from gaining access beyond it's scope
- CPU uses *address buses* to indicate the location of instructions and the memory or I/O device sends the data residing at the address location back via *data bus*
- After execution the address and result are sent back along the buses to the requesting program's memory



Multiprocessing

- **Symmetric mode** – processors are handed work as needed, used to balance the workload, a scheduler determines which processor is ready to work
- **Asymmetric mode** – when a processor is dedicated to a specific type of task, usually used for time-sensitive instructions



Process Management

- **Process** – set of instructions that are running, programs become processes when they are loaded into memory and activated by the OS. Each process has its own system resource (time slot, memory, etc.)
- **Multiprogramming** – when more than one program can be loaded into memory at a time.
- **Note:** multiprogramming is replaced by multitasking in newer OS, meaning multiple programs can be loaded into memory at a time AND the OS can handle requests from multiple programs simultaneously as well. Saves CPU time by handling requests for other programs while waiting on



Process Management

- **Cooperative multitasking** – when an OS requires that processes release their resources voluntarily. Can lead to problems with poor programming allowing a resource to be held indefinitely.
- **Preemptive multitasking** – when an OS places a time limit on how long a resource can be held by a process so, a process can be suspended and another process can use the CPU, through *time sharing*.
- Each OS has it's own process model. For example, *forking* is used in UNIX and LINUX systems. This is when children processes are formed which have the same characteristics as the parent, but their own memory space, stack and program counter.



Process Management

- **Process states (Figure 5-5):**
 - **Running – CPU is executing instructions and data**
 - **Ready – waiting to send instructions to CPU**
 - **Blocked – waiting for some input data**
- **Process table (Figure 5-6) – OS keeps track of each process' information to be loaded into registers when the CPU needs it**
- **Processes know when it is able to communicate with the CPU with *interrupts*. The OS isn't actually running applications at the same time, but each process is run in turn when their interrupt occurs. Note: some processes should not be interrupted and thus the OS assigns various levels of priority.**

Interrupts

- **Maskable interrupts** – usually assigned to less important events, the current process can finish before the event takes place, essentially ignoring interrupt.
- **Nonmaskable interrupts** – can not be overridden and is used for critical events
- **Watchdog timer** – a process that resets the system with a warm boot when the OS hangs and cannot recover.



Thread Management

- When the process needs to send something to the CPU a thread is generated.
- Thread – an individual instruction set and the data that is to be worked on by the CPU
- Example: opening, saving, printing, etc. in a word processor requires a thread to be created dynamically
- A program that is designed to complete several tasks simultaneously requires several threads to be generated at the same time. This is called *multithreading*
- Note: each thread shares resources with the process that created it, so each thread works in the memory space of the original process and has access to all the same files and resources.



Process Scheduling

- The OS creates data structures and dedicates necessary processes to handle process requests for resources. When the process is complete the data structures are broken down and resources are returned, failure to do so ends in a loss of critical resources.
- Deals with *deadlocks*, which is when multiple processes need each others resource to complete. For example , if process A holds resource 1 and needs resource 2 to complete, but process B has resource 2 and needs resource 1 to complete.
- Different OS handle deadlocks differently, such as requiring processes to have all their resources before starting or forcing one of the deadlocked processes to relinquish their resource.



Process Activity

- In order to protect processes from each other, *process isolation* is used. This prevents processes from trying to read/write to the same locations at the same time, corrupting other processes' data.
- Encapsulation of Objects – no other process understands or interacts with the internal code of the process, but instead deals with an interface. Provides *data hiding*, meaning that outside software won't know how a process works. Enforces modularity in programming code.
- Time multiplexing – allows processes to use the same resources, by creating a single channel of requests from the processes and feeding them to the CPU one at a time, thus splitting up time shares with the processes.
- Naming distinctions – different processes have their own process identification value (PID), isolated processes all have their own PID.
- Virtual address space mapping – allows each process to have it's own memory space from RAM. Provides integrity and confidentiality

Memory Management

- Provide abstraction level for programmers, meaning the details are hidden
- Maximize performance with limited available memory
- Protect the operating system and applications loaded into memory
- Each computer has a memory hierarchy from fast and expensive memory (registers, cache) to slow and cheap (RAM, hard drive).
- The *memory manager* allocates and deallocates different memory segments, enforces access control to protect process memory segments, and swap memory contents from RAM to the hard drive.
- The CPU uses a *base register* and *limit register* containing the start and stop address, respectively, for the available process memory. So, a thread cannot reference outside memory addresses.



Memory Types

- **Random access memory (RAM) – temporary storage where data and program instructions are held and altered. Used for read/write activities and is volatile, meaning when the computer powers down the data stored here is lost.**
- **Dynamic RAM (DRAM) – a memory controller recharges the memory by continually reading and writing the same values to memory, slower than SRAM**
- **Static RAM (SRAM) – does not require continuous refreshing, does not use capacitors, faster but takes up more space**
- **Thrashing – when a computer spends more time moving data around small portions of memory than processing the data**
- **The processor, memory type and amount and bus speeds are all factors in system performance.**



Memory Types

- **Synchronous DRAM (SDRAM)** – synchronizes itself with the system's CPU and synchronizes signal input and output on the RAM chip, timing of the CPU clock and memory activities are synchronized, increases speed of transmitting and executing data.
- **Extended data out DRAM (EDO DRAM)** – faster than DRAM, because it can capture the next block of data while the first memory block is sent to the CPU, this is referred to as the “look ahead feature”
- **Burst EDO DRAM (BEDO RAM)** – works like EDO RAM, but it is able to send more data at once. It reads and sends up to four memory addresses in a small number of cycles.
- **Double data rate SDRAM (DDR SDRAM)** – carries out reads on rising/falling cycles of clock pulse, thus carrying out two operations per clock cycle, essentially doubling speed of memory compared to SDRAM.



Memory Types

- **Read only memory (ROM)** – nonvolatile memory, the data is held within the memory chips without power, data cannot be altered, programs and routines designed into the ROM is called *firmware*
- **Programmable read-only memory (PROM)** – type of ROM that can be modified after it has been manufactured one time.
- **Erasable PROM (EPROM)** – can be modified, erased and upgraded, requires a UV light device and erases all the data
- **Electrically EPROM (EEPROM)** – allows for data to be stored and erased electrically one byte at a time, slow
- **Flash memory** – similar to EEPROM, but faster, works by moving around varying levels of voltage to indicate 1 or 0 in address space, erasing occurs in blocks instead of by byte



Memory Types

- **Cache memory** – used for high speed writing and reading activities, when the system assumes that the data will need to be accessed again.
- **Protection Issues:**
 - **Every address reference is validated for protection**
 - **Multiple processes can share access to the same segment with different access rights**
 - **Different instruction and data types can be assigned different levels of protection**
 - **Processes cannot generate an unpermitted address or gain access to an unpermitted segment**
 - **Higher complexity, usually means more exploitable vulnerabilities**
- **Hardware segmentation** – memory is separated physically, instead of just logically, thus helping to keep lower-privileged processes from accessing or modifying a higher-level process's memory space.



Memory Mapping

- The CPU uses physical addresses when dealing with memory, but software uses logical addresses, giving the computer an access control level between software and memory for protection and efficiency
- When a program attempts to access memory, it's access rights are checked and then the instruction and commands are carried out, allowing processes access to only their allocated memory.
- Absolute addresses – physical memory address used by the CPU
- Logical addresses – indexed memory addresses used by the software
- Relative addresses – based on a known address with an offset value applied
 - Though the memory manager sets aside physical memory for a process, the process uses it's own addressing scheme without regard to the actual address allocated. The manager has to interpret these logical addresses into absolute addresses.



Memory Leak

- Applications are allocated specific memory by the OS when a request is made. The application is supposed to tell the OS when it is done, so that the memory gets released and becomes available to other applications.
- If this happens too much, the OS becomes starved for memory which affects performance.
- Leaks are often the cause of Denial-of-Service attacks. This can be done by continually having the bad process run until the memory is all “used up” and the system hangs.
- Countermeasures
 - Garbage collector – software that identifies unused committed memory and alerts the OS to release it
 - Developing better code that properly releases memory

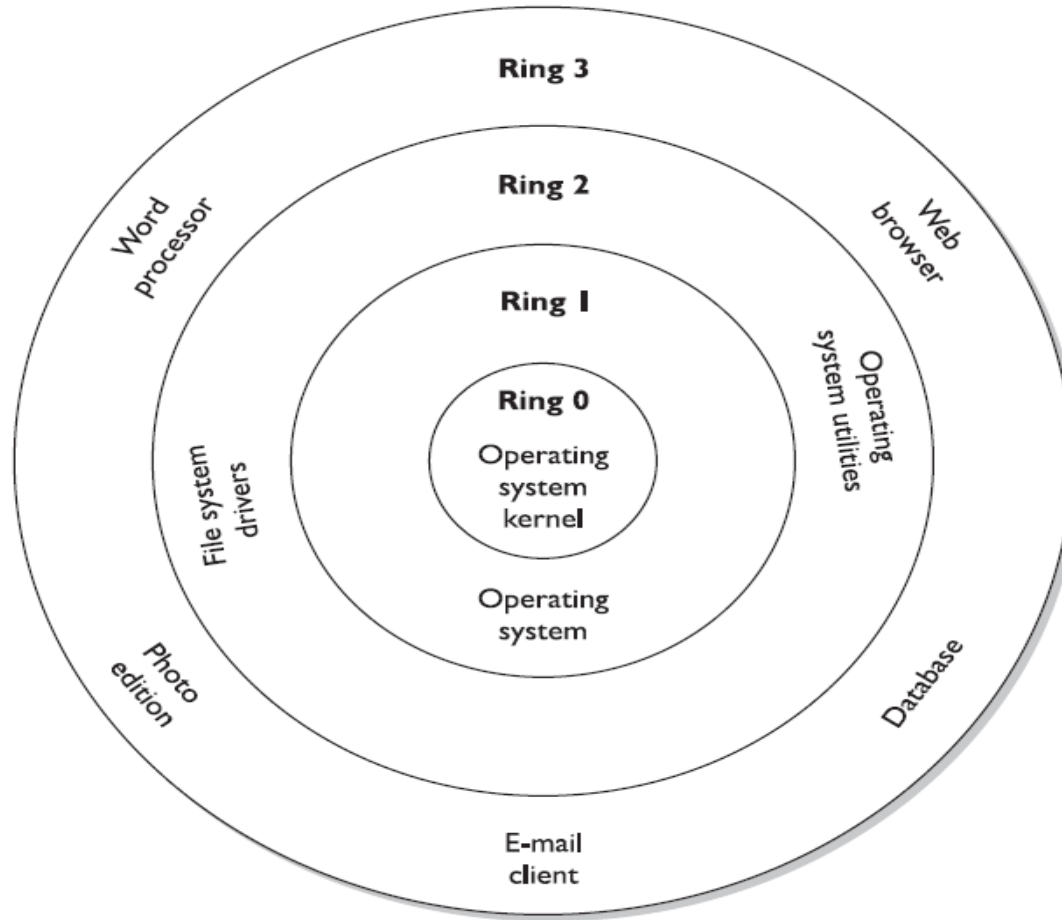


Virtual Memory

- **Secondary storage** – considered nonvolatile storage media and includes hard drive, floppy disk, etc.
- **Virtual memory** – when secondary storage and RAM are combined. Hard drive space is used to expand RAM space. *Swap space* is the reserved hard drive space used to extend the RAM space.
- When a system fills up volatile memory, data is written to the hard drive. When the data is requested, it is moved from the hard drive, back into memory units, called *pages*. This is called *virtual memory paging*. This takes longer than accessing actual RAM memory space.
- **Note:** when data is encrypted and saved on the hard drive, they will be decrypted when used. While the unencrypted data is in RAM, the system could write the data to the swap space in it unencrypted state, creating a vulnerability.



CPU Modes and Protection Rings



OS Architecture

- **Vocabulary:**
 - **Layered operating system – architecture separates system functionality into hierarchical layers**
 - **Data hiding – instructions and data at the various layers do not have direct access to the instructions and data at any other layer**
 - **Domain – a set of objects that a subject is able to access**
 - **Execution domain – the domain where a privileged process needs to be able to execute its instructions and process its data with assurance that programs in a different domain cannot negatively affect its environment.**



OS Architecture

- **Layering and data hiding provide protection by implementing layers of abstraction. Access to sensitive processes and data can only take place through properly formatted requests that are sent to system APIs. Meaning the communication between different layers of trust only happens through well-defined interfaces. Creating and maintaining these different layers helps protect data from other processes that are not authorized to access it.**
- **If a process does not have an interface to communicate with a process in another layer, it cannot have access to its data.**
- **The protection ring architecture allows for processes to run in either kernel or user mode.**



OS Architecture

- **Processes with a higher trust level have a larger domain than those in a lower trust level.**
- **Execution domains allow for the isolation of process activity, which provides protection and system stability.**
- **Monolithic systems have all kernel activities running in supervisory mode, whereas micro kernels have only a small subset of kernel activities running in this privileged mode. All other kernel activities run in user mode. Most modern operating systems take a hybrid approach between monolithic and microkernel in design.**



Virtual Machines

- **Virtualization enables single hardware equipment to run multiple operating system environments synchronously, increasing processing power utilization. These virtual instances of operating systems, applications and storage devices are called *virtual machines*. And is referred to as a guest executed on a host machine.**
- **A single host environment is capable of running several guests at the same time, with each VM pooling from the same set of resources. These resources are emulated through the host environment, the guests do not directly interact with the resources.**
- **Cheaper alternative to a full physical system for each OS. List on page 319-320 useful for exam prep.**



I/O Device Management

- **OS has to control all input/output devices.**
 - It sends commands to devices, handles interrupts and provides the interface between the devices and applications.
- **The OS needs to access and release devices and resources properly.**
 - Different OS have different methods of handling devices.
 - For example, in Windows NT applications cannot make direct requests to hardware devices.
 - Methods such as this provide a level of protection that helps ensure resource integrity and availability.
- **Interrupts are generated when the device has completed its task and the data is in memory for processing.**
 - The device signals the interrupt controller, which works similarly to interrupts mentioned previously.
 - If the CPU is busy and the priority level of the device is lower than the current task, then the device waits.
 - The *interrupt vector* table is used by the CPU to determine which of the devices needs attention.



I/O Procedures

- **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 it isn't the CPU wastes time waiting. Very slow way of working and wastes CPU time.
- **Interrupt-Driven I/O** – The CPU sends data to the device and goes on to service another process while it waits for the device to finish and send an interrupt. Then the CPU sends more data and returns to servicing other processing until another interrupt is received. Time is wasted on interrupts.
- **I/O Using DMA** – Direct memory access is a way of transferring data between devices and the system's memory without using the CPU, allowing for faster data transfer rates. Also called *unmapped I/O*
- **Premapped I/O** – The CPU sends the physical memory address of the requesting process to the I/O device and the device is trusted to interact with the contents of memory directly. The CPU is not involved and the OS trusts the device to work correctly, which would lead to security vulnerabilities.
- **Fully Mapped I/O** – similar to premapped, but the OS does not fully trust the device. The physical address is not given, but the logical addresses are used instead. The OS acts as the middle man and controls how the device and the process interact.

System Architecture

- **Creating the proper security protocols in an OS requires the question of where the protection should be placed and how the protection should work. Security mechanisms can be placed in the kernel, hardware, OS, services and program layers. Hardware security is more broad and provides a baseline for higher layers, whereas security in the user layer is more detailed and focused.**
- **A higher complexity often means a decrease in the assurance gained through security mechanisms, because of the higher threshold of technical knowledge needed to fully utilize the mechanisms. But simpler mechanisms are easier to handle and provide less functionality.**
- **The first step is to establish the security measures in ring 0 and then decide which code can be interacted with in a secure manner. Trusting all components in the system causes too much overhead and complexity. To be trusted, the mechanisms must perform in a predictable and secure manner and not adversely affect other mechanisms. In return, the mechanisms receive a higher privilege and priority level. So, it is vital to identify the trust subjects and objects and place them into subsets.**



Trusted Computing Base

- **Trusted computer base (TCB) – the total combination of protection mechanisms within a computer system, including hardware, software and firmware. Items in the TCB will enforce the security policy and not violate it.**
- **Addresses the level of trust a system provides, as measured originally by the Orange Book. No system is ever totally secure, but is instead measured by how predictably it will react to different types of situations.**
- **TCB includes mechanisms that directly enforce the policy and those that act appropriately and not violate the trust of the system.**
- **Used in systems with a hardened kernel that compartmentalizes system processes. So, in a sense, the TCB is the kernel with other components like trusted commands, programs and configuration files and can interact directly with the kernel.**



Trusted Computing Base

- **Trusted path** – a communication channel between the user, or program, and the kernel.
- **Trusted shell** – means that someone who is working in that shell cannot “bust out of it”, and other processes can not “bust into it”.
- The four basic functions of the TCB are process activation, execution domain switching, memory protection, and I/O operations.
- Process activation deals with the activities that must take place when a process is going to have its instructions and data processed by the CPU.
- Execution domain switching occurs when a process needs to call upon a process in a higher protection ring. This refers to when the CPU goes from executing instructions in user mode to privileged mode and back.
- The TCB compartmentalizes the memory and I/O operations into discrete units, which are the processes that make up the kernel, to make them secure.
- Trust is built into a system and measured against a criteria as a guideline.



Security Perimeter

- **A boundary that divides the trusted from the untrusted.**
- **When a trusted component needs to communicate with an untrusted component, a component that lies outside the security perimeter, the communication must not expose the system to unexpected security compromises.**
- **Interfaces are used to control communication across the perimeter.**
- **Outer components should not be able to access critical system resources through communication with a component inside the perimeter.**



Systems Evaluation Methods

- **Security Evaluation** – examines the security-relevant parts of a system, TCB, access control mechanisms, reference monitor, kernel, protection mechanisms and the relationship and interaction between these components.
- **Many assurance evaluation processes exist because methods and ideologies evolved over time, and various parts of the world view computer security differently and rate differently based on their priorities.**
- **Evaluation is done so that users will know which systems meet their needs based on their assurance ratings. These ratings are obtained by an independent third party and are not marketing by the developer, giving more confidence in the rating.**

- **Security policy** – the policy must be explicit and well defined and enforced by the mechanisms within the system
- **Identification** – individual subjects must be uniquely identified
- **Labels** – access control labels must be associated properly with objects
- **Documentation** – must be provided, including test, design, and specification documents, user guides, and manuals
- **Accountability** – audit data must be captured and protected to enforce accountability
- **Life-cycle assurance** – software, hardware, and firmware must be able to be tested individually to ensure that each enforces the security policy in an effective manner throughout their lifetimes
- **Continuous protection** – the security mechanisms and the system as a whole must perform predictable and acceptably in different situations continuously

The Orange Book

- **Developed by the U.S. DoD, named the Trusted Computer System Evaluation Criteria (TCSEC), which is used to evaluate operating systems, applications, and different products.**
- **Used to determine if the product meets the security properties the vendor claims it does and whether the product is appropriate for a specific function.**
- **Classification**
 - **Security policy**
 - **Accountability**
 - **Assurance**
 - **Documentation**



The Orange Book

- **Division D: Minimal Protection** – failed to meet criteria and requirements of higher divisions
- **Division C: Discretionary Protection**
 - **C1: Discretionary Security Protection** – users are processing information at the same sensitivity level; thus, strict access control and auditing measures are not required.
 - **C2: Controlled Access Protection** – users are trusted but a certain level of accountability is required. Seen as the most reasonable class for commercial applications, but protection level is weak



The Orange Book

- **Division B: Mandatory Protection**
 - **B1: Labeled Security** – intended for environments that require systems to handle classified data
 - **B2: Structured Protection** – environment that processes sensitive data that require a higher degree of security, requires systems that are relatively resistant to penetration and compromise
 - **B3: Security Domains** – highly secured environment that processes very sensitive information, requires systems that are highly resistant to penetration



The Orange Book

- **Division A: Verified Protection**
 - **A1: Verified Design – the most secure of the secured environments, deals with top-secret information and cannot adequately trust anyone using the systems without strict authentication, restrictions, and auditing**



Red Book

- **The Orange book addressed single-system security and focused on which users could access a system and not what they could do with the objects.**
- **The Trusted Network Interpretation (TNI), or Red Book, addresses security evaluation topics for networks, network components, and isolated LAN and WAN systems.**
- **Forgoes specific details about how to implement security mechanisms, instead providing a framework for securing different types of networks.**
- **The Red Book rates confidentiality of data and operations that happen within a network and network products. Measures functionality, strength and assurance of encryption and protocols.**

Red Book

- **Communication integrity**
 - **Authentication – Protects against masquerading and playback attacks. Mechanisms include digital signatures, encryption, timestamp, and passwords.**
 - **Message integrity – Protects the protocol header, routing information, and packet payload from being modified. Mechanisms include message authentication and encryption**
 - **Nonrepudiation – Ensures that a sender cannot deny sending a message. Mechanisms include encryption, digital signatures, and notarization.**
- **Denial-of-service prevention**
 - **Continuity of operations – Ensures that the network is available even if attacked. Mechanisms include fault-tolerant and redundant systems and capability to reconfigure network parameters in case of an emergency**
 - **Network management – Monitors network performance and identifies attacks and failures. Mechanisms include components that enable network administrators to monitor and restrict resource access.**

Red Book

- **Compromise protection**
 - **Data confidentiality** – Protects data from being accessed in an unauthorized method during transmission. Mechanisms include access controls, encryption, and physical protection of cables.
 - **Traffic flow confidentiality** – ensures that unauthorized entities are not aware of routing information or frequency of communication via traffic analysis. Mechanisms include padding messages, sending noise, or sending false messages.
 - **Selective routing** – routes messages in a way to avoid specific threats. Mechanisms include network configuration and routing tables
- **Assurance is measured by how things actually work on the network compared to a theory of how things should work.**



Information Technology Security Evaluation Criteria

- **ITSEC – first attempt at establishing a single standard for evaluating security attributes of computer systems and products by many European countries.**
- **Evaluates two main attributes of a system's protection mechanism, functionality and assurance. Services provided to subjects are examined and measured.**
- **Protection functionality is tested to see if the product delivers on what the vendors says it should. Assurance is the measure of confidence in the protection mechanisms and their effectiveness and capability to perform consistently.**
- **Possible to get the same functionality rating and two very different assurance ratings. Page 365 has a long list of the items tested during an evaluation.**

Common Criteria

- **Developed through a collaboration among national security standard organization across the world.**
- **ITSEC is flexible, but was too complex and had too many classifications. The Rainbow Series is too rigid for the business world. The Common Criteria is designed to fit right in the middle.**
- **Evaluates a product against a protection profile, which is structured to address real-world security needs. The evaluation is carried out on a product and assigned an Evaluation Assurance Level (EAL)**
 - **EA1 Functionally tested**
 - **EA2 Structurally tested**
 - **EA3 Methodically tested and checked**
 - **EA4 Methodically designed, tested, and reviewed**
 - **EA5 Semiformally designed and tested**
 - **EA6 Semiformally verified design and tested**
 - **EA7 Formally verified design and tested**

- **Protection profiles – mechanism used to describe a real-world need of a product that is not currently on the market. Contains the set of security requirements, their meaning and reasoning, and the corresponding EAL rating that the intended product will require.**
 - **Descriptive elements – Provides the name of the profile and a description of the security problem to be solved**
 - **Rationale – Justifies the profile and gives more detailed description of the real-world problem to be solved. The environment, usage assumptions, and threats are illustrated along with guidance on the security policies that can be supported by products and systems that conform to this profile**
 - **Functional requirements – Establishes a protection boundary, meaning the threats or compromises within this boundary to be countered. The product or system must enforce the boundary established in this section**
 - **Development assurance requirements – Identifies the specific requirements the products or system must meet during the development phases, from design to implementation**
 - **Evaluation assurance requirements – Establishes the type and intensity of the evaluation**

Certification vs. Accreditation

- **Certification – the comprehensive technical evaluation of the security components and their compliance for the purpose of accreditation.**
 - **Ensure that a system, product, or network is right for the customer's purpose**
- **Accreditation – formal acceptance of the adequacy of a system's overall security and functionality by management.**
 - **Certification information is presented to management and changes are made.**
 - **Once satisfied with the system's overall security as presented, management makes a formal accreditation statement.**



Open vs. Closed Systems

- **Open systems are built upon standards, protocols, and interfaces that have published specifications, which allows third-party vendors to develop add-on components and devices. Provides interoperability between products by different vendors.**
- **Closed systems use an architecture that does not follow industry standards. Interoperability and standard interfaces are not employed to enable easy communication between different types of systems and add-ons. Closed systems can provide more security because it has less doorways in and operates in a more secluded manner. Also, there are less tools available to defeat the security mechanisms and fewer people understand the design.**

Enterprise Architecture

- **System security architecture is a high-level design that works as a framework. The architecture outlines what needs to be in place to ensure that the system's security policy and protection level are met.**
- **An *enterprise security architecture* defines the information security strategy that consists of a layer of policy, standards, solutions, and procedures and the way they are linked across the enterprise strategically, tactically, and operationally. Infrastructure is the underlying technology and hardware needed to support enterprise security architecture.**
- **Complete a checklist (page 374) to see if an organization lacks an enterprise security architecture. If the answer is yes to most of the questions then there is not a useful architecture in place.**
- **Shifts from technology-oriented to business-centric security processes. Administrative, technical, and physical controls are linked to properly manage risk. Refer to chapter 3 or page 375 for steps to setting up a security program. These steps are the basic steps of setting up an enterprise security architecture.**

Zachman Framework

- **Not security oriented, but it is a good template to work with because it offers direction on how to understand and actual enterprise in a modular fashion.**
- **Table 5-3 Zachman framework for enterprise architecture**
- **Strategic alignment – the business drivers and the regulatory and legal requirements are being met by the security architecture.**



Threats

- **Maintenance hooks – type of backdoor, instructions within software that only the developer knows about and can invoke, and give easy access to the code. Preventive measures against backdoors: host intrusion detection system, file system encryption and auditing to detect backdoor usage.**
- **Time-of-Check/Time-of-Use – deals with the sequence of steps a system uses to complete a task, takes advantage of the dependency on the timing of events that take place in a multitasking system. Prevention: apply software locks to prevent race conditions**
- **Buffer overflow**



Security models fundamental concepts

Dr. Drew Hamilton

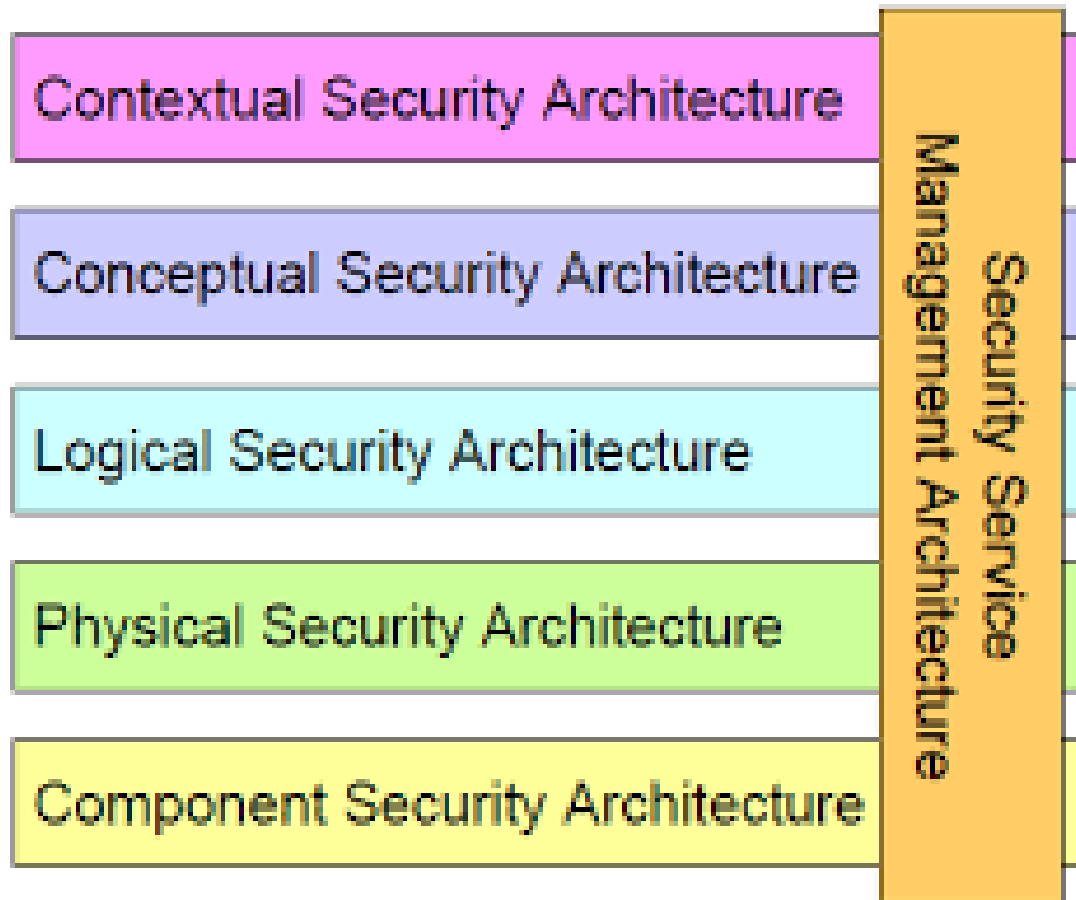


Mississippi State University Center for Cyber Innovation

Domain 3 Security Engineering



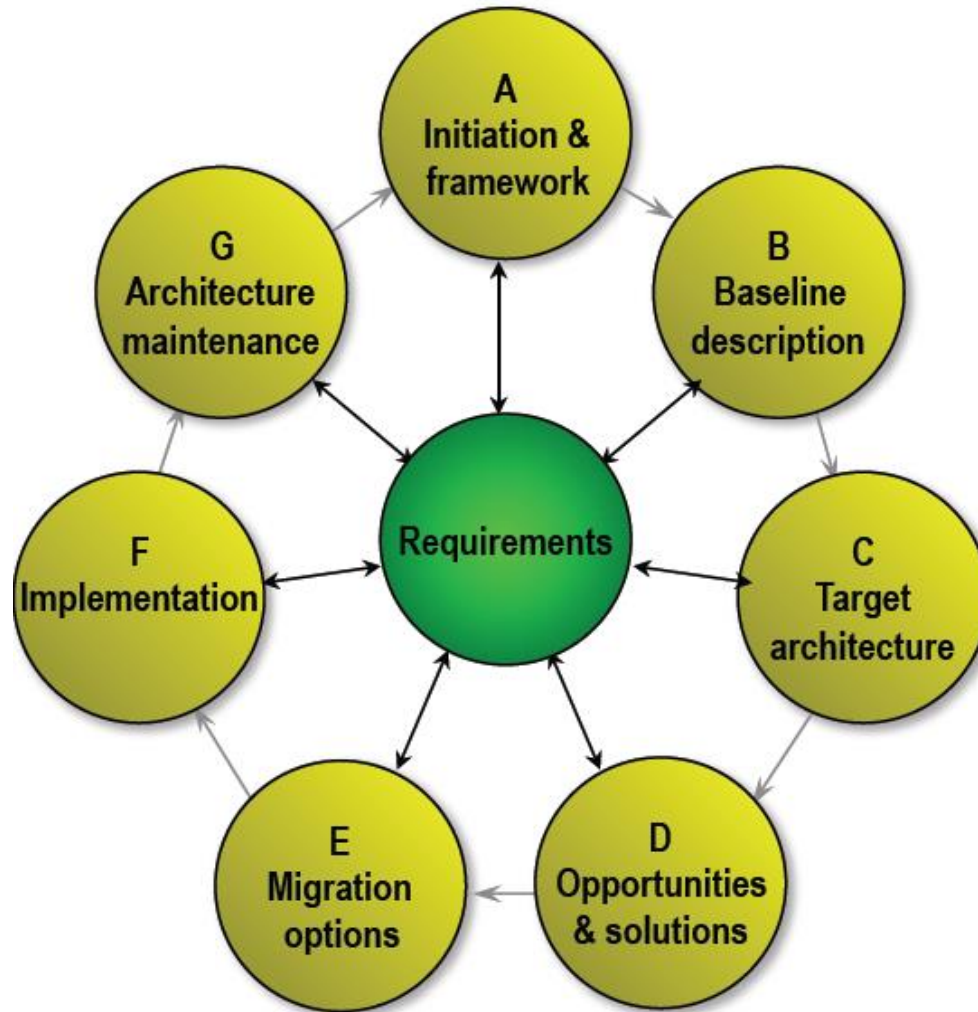
SABSA Model for Security Architecture



Sherwood Applied Business Security Architecture

	Assets (What)	Motivation (Why)	Process (How)	People (Who)	Location (Where)	Time (When)
Contextual	The business	Business risk model	Business process model	Business organization and relationships	Business geography	Business time dependencies
Conceptual	Business attributes profile	Control objectives	Security strategies and architectural layering	Security entity model and trust framework	Security domain model	Security-related lifetime and deadlines
Logical	Business information model	Security policies	Security services	Entity schema and privilege profiles	Security domain definitions and associations	Security processing cycle
Physical	Business data model	Security rules, practices and procedures	Security mechanisms	Users, applications and user interface	Platform and network infrastructure	Control structure execution
Component	Detailed data structures	Security standards	Security products and tools	Identities, functions, actions and ACLs	Processes, nodes, addresses and protocols	Security step timing and sequencing
Operational	Assurance of operational continuity	Operational risk management	Security service management and support	Application and user management and support	Security of sites and platforms	Security operations schedule

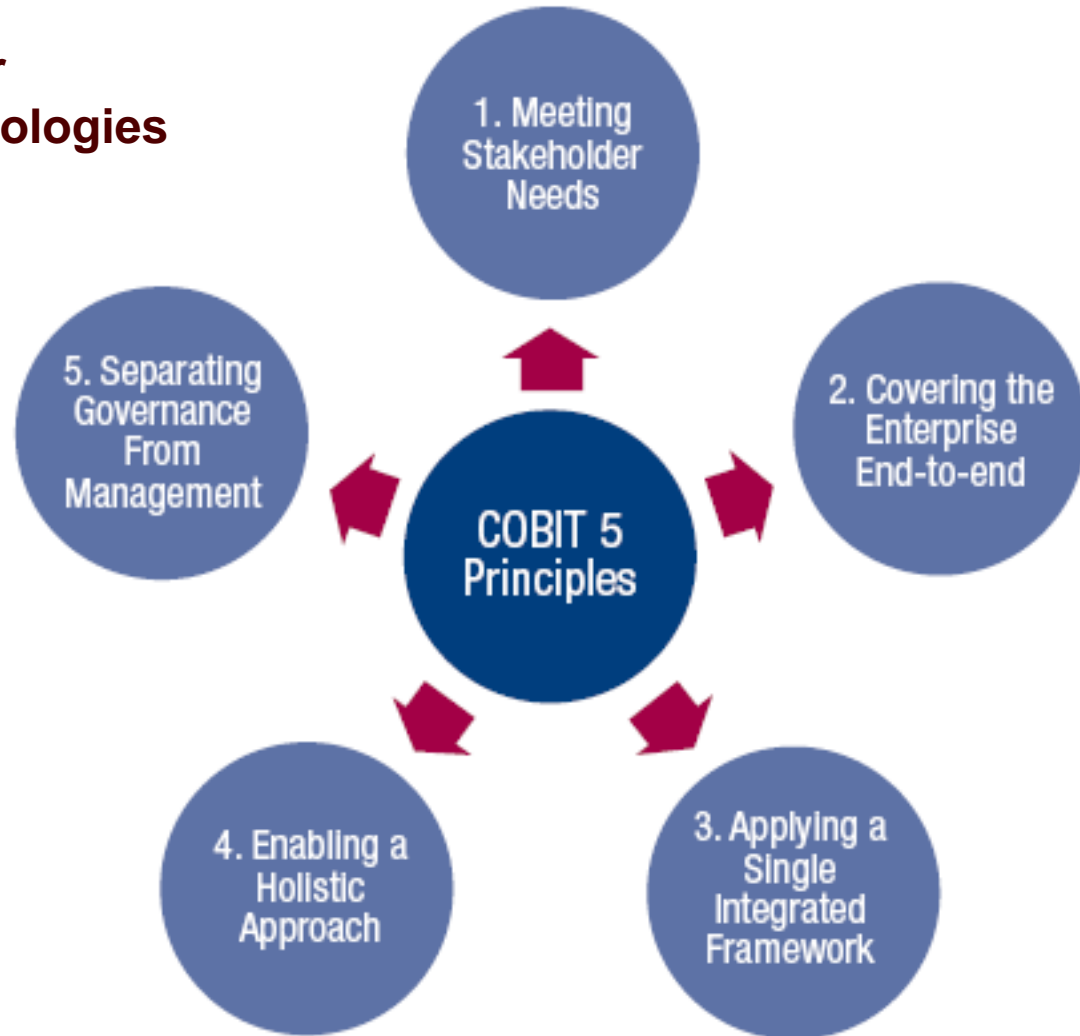
The Open Group Architecture Framework (TOGAF)



COBIT 5

COBIT = Control Objectives for Information and Related Technologies

ISACA is an international professional association focused on IT Governance. Previously known as the Information Systems Audit and Control Association, ISACA now goes by its acronym only, to reflect the broad range of IT governance professionals it serves.

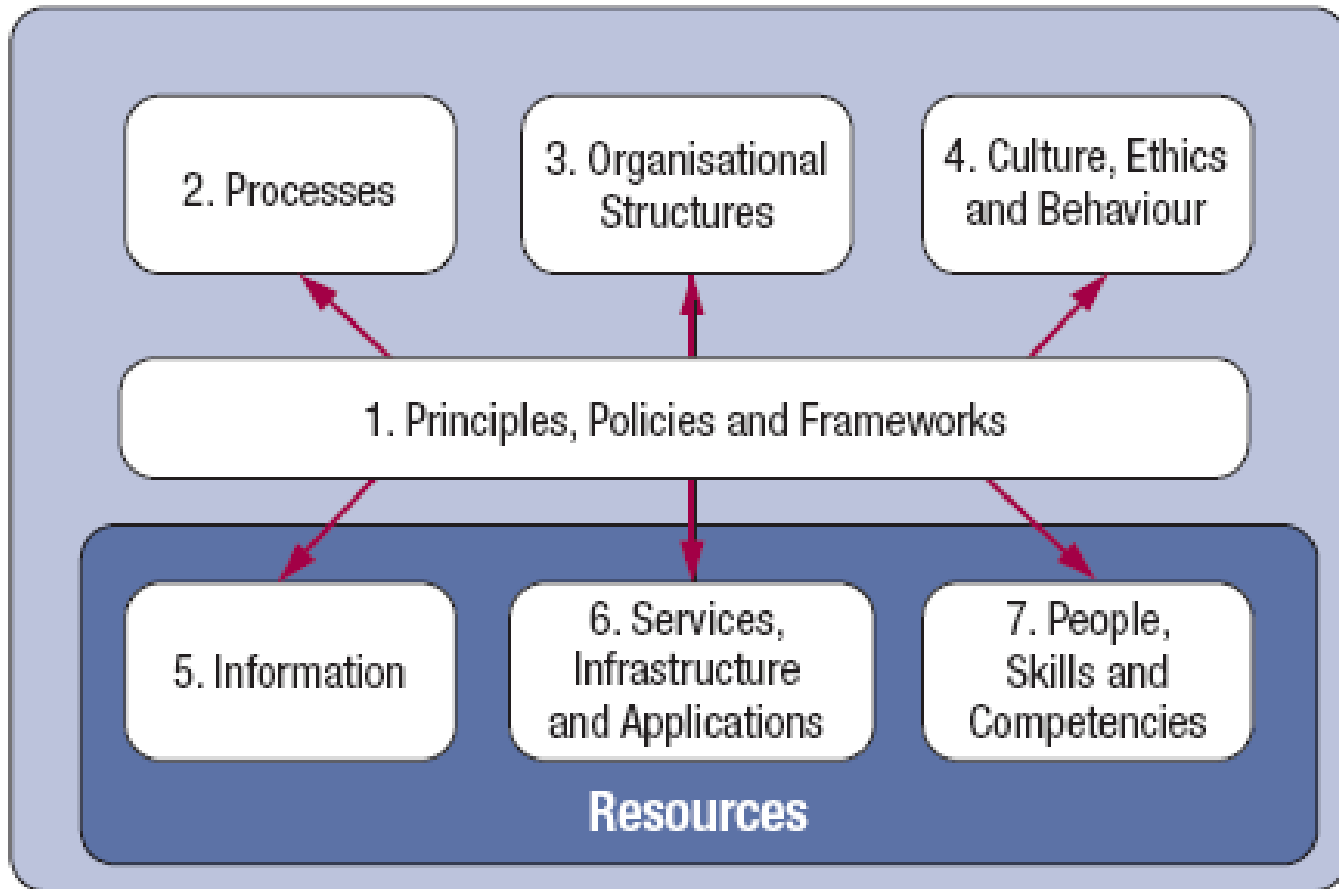


COBIT 5

- **Divided into Governance and Management domains**
 - **Governance: Contains five governance processes; within each process, evaluate, direct and monitor (EDM)**
 - **Management: Contains four domains, in line with the responsibility areas of plan, build, run and monitor (PBRM)**
 - **Align, Plan and Organize (APO)**
 - **Build, Acquire and Implement (BAI)**
 - **Deliver, Service and Support (DSS)**
 - **Monitor, Evaluate and Assess (MEA)**



COBIT 5 Enablers



Payment Card Industry Data Security Standards

- **The PCI DSS represents a common set of industry tools and measurements to help ensure the safe handling of sensitive information.**
- **The standard provides an actionable framework for developing a robust account data security process - including preventing, detecting and reacting to security incidents.**
- **Applies to any entity that stores, processes and/or transmits CHD.**



PCI DSS Six Goals

- **Build and Maintain a Secure Network**
- **Protect Card Holder Data**
- **Maintain a Vulnerability Management Program**
- **Implement Strong Access Control Measures**
- **Regularly Monitor and Test Networks**
- **Maintain an Information Security Policy**



PCI DSS 12 Requirements

- **1) Install and Maintain a firewall configuration to protect Card Holder Data (CHD)**
 - Firewall and Router configuration standards
 - Review Network Diagram
 - Firewall and Router connections are restricted (inbound/outbound traffic)
 - No direct internet connection to CHD (DMZ)
- **2) Do not use vendor supplied defaults**
 - Attempt to sign on with defaults
 - Hardening standards and system configuration
 - Non-console admin access is encrypted



PCI DSS 12 Requirements

- **3) Protect stored CHD**
 - Retention Policy and Procedures
 - Quarterly process for deleting stored CHD
 - Sample incoming transactions, logs, history files, trace files, database schemas and content
 - Do not store full track, CVV or PIN
 - Render PAN unreadable (mask/truncate)
 - Encryption and key management
- **4) Encrypt transmission of CHD**
 - Verify encryption and encryption strength
 - Verify wireless is industry best practice (no WEP)



PCI DSS 12 Requirements

- **5) Use and regularly update Antivirus software**
 - All system have AV
 - AV is current, actively running and logging
- **6) Develop and maintain secure systems and applications**
 - Patch management – current within one month
 - ID new security vulnerabilities with risk rating
 - Custom code is reviewed prior to release
 - Change management process
 - Developers are trained in secure coding techniques



PCI DSS 12 Requirements

- **7) Restrict access to CHD by need-to-know**
 - Review access policies
 - Confirm access rights for privileged users
 - Confirm access controls are in place
 - Confirm access controls default with “deny-all”
- **8) Assign a unique ID to each user**
 - Verify all users have a unique ID
 - Verify authentication with ID/PW combination
 - Verify two-factor authentication for remote access
 - Verify terminated users are deleted
 - Inspect configurations for PW controls



PCI DSS 12 Requirements

- **9) Restrict physical access to CHD**
 - Access to computer rooms and data centers
 - Video cameras are in place and video is secure
 - Network jacks are secure – not in visitor area
 - Process for assigning badges
 - Storage locations are secure (offsite media)
- **10) Track and monitor all access to network resources**
 - Review audit trails – actions, time, date, user, etc.
 - Time server updates and distribution
 - Process to review security logs



PCI DSS 12 Requirements

- **11) Regularly test security systems**
 - Test for wireless access points
 - Internal and external network vulnerability scans
 - Internal and external penetration testing annually
 - File integrity monitoring tools are used
- **12) Maintain security policies**
 - Policies are reviewed at least annually
 - Explicit approval is required for access
 - Auto disconnect for inactivity-internal and remote
 - Security awareness program is in place
 - Incident Response Plan



Build and Maintain a Secure Network

Requirement 1: Install and maintain a firewall configuration to protect cardholder data

Firewalls are devices that control computer traffic allowed between an entity's networks (internal) and untrusted networks (external), as well as traffic into and out of more sensitive areas within an entity's internal trusted networks. The cardholder data environment is an example of a more sensitive area within an entity's trusted network.

A firewall examines all network traffic and blocks those transmissions that do not meet the specified security criteria.

All systems must be protected from unauthorized access from untrusted networks, whether entering the system via the Internet as e-commerce, employee Internet access through desktop browsers, employee e-mail access, dedicated connections such as business-to-business connections, via wireless networks, or via other sources. Often, seemingly insignificant paths to and from untrusted networks can provide unprotected pathways into key systems. Firewalls are a key protection mechanism for any computer network.

Other system components may provide firewall functionality, provided they meet the minimum requirements for firewalls as provided in Requirement 1. Where other system components are used within the cardholder data environment to provide firewall functionality, these devices must be included within the scope and assessment of Requirement 1.

PCI DSS Requirements	Testing Procedures	In Place	Not in Place	Target Date/Comments
1.1 Establish firewall and router configuration standards that include the following:	1.1 Obtain and inspect the firewall and router configuration standards and other documentation specified below to verify that standards are complete. Complete the following:			
1.1.1 A formal process for approving and testing all network connections and changes to the firewall and router configurations	1.1.1 Verify that there is a formal process for testing and approval of all network connections and changes to firewall and router configurations.			
1.1.2 Current network diagram with all connections to cardholder data, including any wireless networks	1.1.2.a Verify that a current network diagram (for example, one that shows cardholder data flows over the network) exists and that it documents all connections to cardholder data, including any wireless networks.			
	1.1.2.b Verify that the diagram is kept current.			
1.1.3 Requirements for a firewall at each Internet connection and between any demilitarized zone (DMZ) and the internal network zone	1.1.3.a Verify that firewall configuration standards include requirements for a firewall at each Internet connection and between any DMZ and the internal network zone.			
	1.1.3.b Verify that the current network diagram is consistent with the firewall configuration standards.			

Reference Monitor and Security Kernel

- **Reference monitor** – an abstract machine that mediates all access subjects have to objects, to ensure that the subjects have the necessary access rights and to protect the objects from unauthorized access and destructive modifications. Subjects must be fully authorized to use a resource before accessing an object.
- **Security kernel** – made up of hardware, software, and firmware within the TCB and implements and enforces the reference monitor concept. Mediates all access and functions between subjects and objects.
 - Isolation must be provided for the processes carrying out the reference monitor concept and the processes must be tamperproof.
 - Must be invoked for every access attempt and must be impossible to circumvent, complete and foolproof implementation.
 - Small enough to be tested and verified in a complete and comprehensive manner.
- The TCB is the totality of protection mechanisms within a system that work to enforce the security policy, and contains the security kernel. The security kernel implements the concept of the reference monitor.



Security Policy

- A set of rules and practices that dictates how sensitive information and resources are managed, protected and distributed.
- The security policy clearly defines the security level that the security mechanisms are to accomplish and provides the framework for the system's security architecture.
- For a system to obtain an acceptable level of trust, it must be based on an architecture that provides the capabilities to protect itself from untrusted processes, compromises and attacks at different layers.
- Multilevel security policies – security policies that prevent information from flowing from a high security level to a lower security level. Subject can access objects if the subject's security level is greater than or equal to the object's classification.
- Least privilege is enforced when resources and processes are properly isolated. This means that a process has no more privileges than necessary to be able to fulfill its functions.



Security Model

- **The symbolic representation of the security policy. It maps the abstract goals of the policy to information system terms by specifying explicit data structures and techniques necessary to enforce the security policy.**
- **Contains the necessary mathematical formulas, relationships, and structure to follow to accomplish the goals created by the policy. For example, the model provides the means by which x can access y through specific methods.**
- **Some models enforce rules for confidentiality protection, Bell-LaPadula, and others for integrity protection, Biba. Formal models like these are used to provide a high assurance in security and informal models are used to provide a framework for how the policies should be expressed.**
- **The model gives form to the goals outlined in the policy and solves security access problems.**



State Machine Models

- A model that uses the state of the system to verify the security of a system. Meaning that all current permissions and all current instances of subjects accessing objects must be captured. If the subjects can access objects only by means that are concurrent with the security policy, the system is secure.
- The state of the system is a snapshot of the system in time. Activities that alter the state of the system are called *state transitions*.
- The developers must review the system when using the state machine model to determine if the system can, at any point, go from a secure state to an insecure state. List all starting states and outline how these states can change. “if condition then update”
- Each state must be secure, so when system fails it is able to “save itself” and not become vulnerable.
- Initially the developers must define where the state variables are, this could be any and all data variables. Next, the developer must define a secure state for each variable.



Security evaluation models

Dr. Patrick Pape
Shon Harris



Mississippi State University Center for Cyber Innovation

Domain 3 Security Engineering



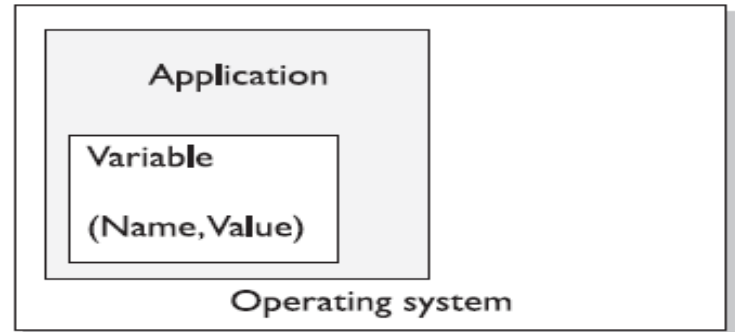
75

State Machine Model

1. Default values of state variable must be safe.
2. User attempts to change variable default value.
3. System checks this subject's authentication.
4. System ensures that change will not put system into an insecure state.
5. System allows the variable values to change = STATE CHANGE.



1.

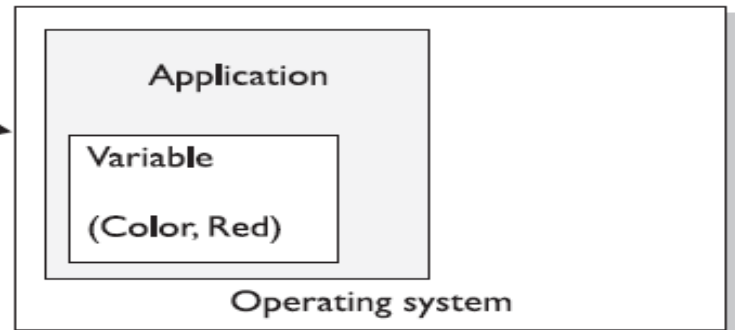


2.

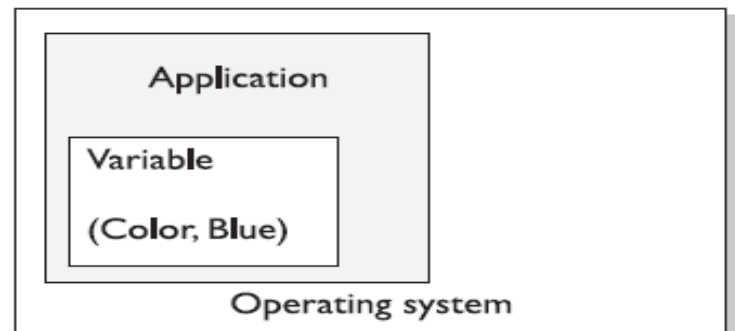
3.

4.

5.



Steps repeat, which causes another state change



Bell-LaPadula Model

- **Rules to Know for exam**
 - **Simple security rule – a subject cannot read data within an object that resides at a higher security level (the “no read up” rule).**
 - ***-property rule – a subject cannot write to an object at a lower security level (the “no write down” rule)**
 - **Strong star property rule – For a subject to be able to read and write to an object, the subject's clearance and the object's classification must be equal**
- **Basic Security Theorem – if a system initializes in a secure state and all allowed state transitions are secure, then every subsequent state will be secure no matter what inputs occur.**



Biba Model

- **Rules:**
 - ***-integrity axiom – a subject cannot write data to an object at a higher integrity level (referred to as “no write up”)**
 - **Simple integrity axiom – a subject cannot read data from a lower integrity level (referred to as “no read down”)**
 - **Invocation property - A subject cannot request service (invoke) to subject of higher intensity.**
- **The Bell-LaPadula and Biba models are both information flow models because they are concerned about data flowing from one level to another.**
- **Bell-LaPadula uses security levels and is most often used in the government. Biba uses integrity levels and is most often used in commercial settings, where data integrity is more important than who can and has seen what.**
- **Test tip: if the word “simple” is used, the rule is talking about reading and if the rule uses * or “star”, it is talking about writing.**

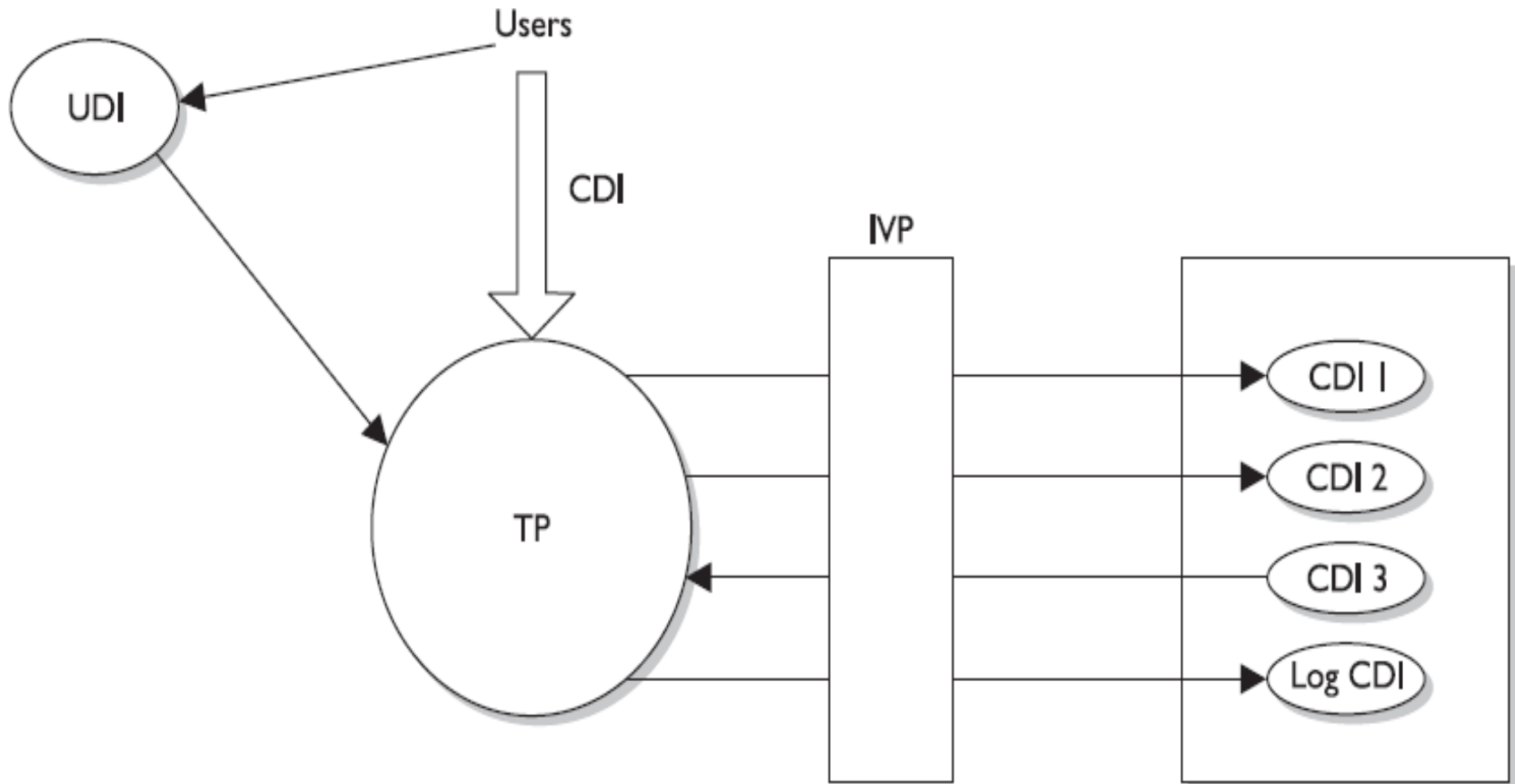


Clark-Wilson Model

- **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**
- **Access triple – subject (user), program (TP) and object (CDI)**
- **Well-formed transaction – using TPs to modify CDIs, a series of operations that are carried out to transfer the data from one consistent state to the other.**
- **Separation of duties – a countermeasure to potential fraudulent activities, tasks may require authentication by a higher level subject.**



Clark-Wilson Model



Integrity Models

- **Goals:**
 - Prevent unauthorized users from making modifications
 - Prevent unauthorized users from making improper modifications (seperation of duties)
 - Maintain internal and external consistency (well-formed transaction, IVP)
- **Clark-Wilson addresses all three, Biba addresses only the first.**



Information Flow Model

- **Models dealing with the flow of any kind of information from one level to another, not limited to security or integrity levels.**
- **Information is held in discrete compartments, subjects and objects are labeled with clearance and classification levels. Information is compartmentalized by classification and need to know, as discussed in chapter 4 with MAC systems.**
- **Intended to aid developers in designing a system that does not allow information to flow in a way that jeopardizes the system, like with covert channels.**



Covert Channels

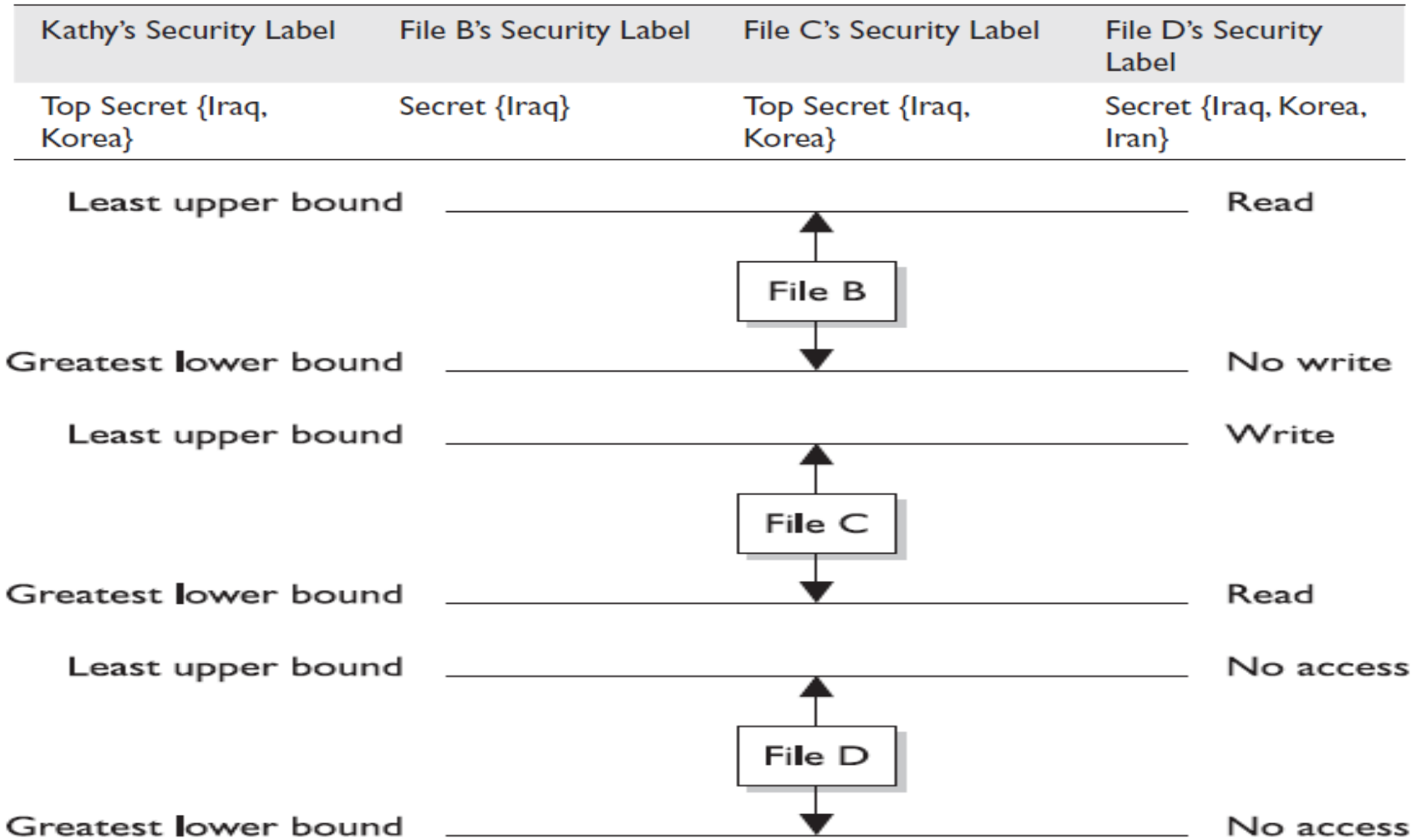
- **Covert channels are a way for an entity to receive information in an unauthorized manner, not regulated by a security measure.**
 - **Improper oversight in the development of the product**
 - **Improper implementation of access controls within the software**
 - **Existence of a shared resource between the two entities**
- **Covert storage channel – processes are able to communicate through some type of storage space on the system. For example, a system is infected with a Trojan horse that installed software that is able to communicate with another process in a limited way. The software is able to read from a sensitive file and sends the information back to the attacker by locking and unlocking a specific file to signify 1 and 0 respectively. Also through creating files and checking if the file exists to send 1/0s.**
- **Covert timing channel – one process relays information to another by modulating its use of system resources. The processes that are communicating to each other are using the same shared resource, time. A process is installed via Trojan horse and in a multitasked system either accepts the opportunity to interact with the CPU or rejects it, to symbolize a 0 or 1, similar to Morse code.**

The Noninterference Model

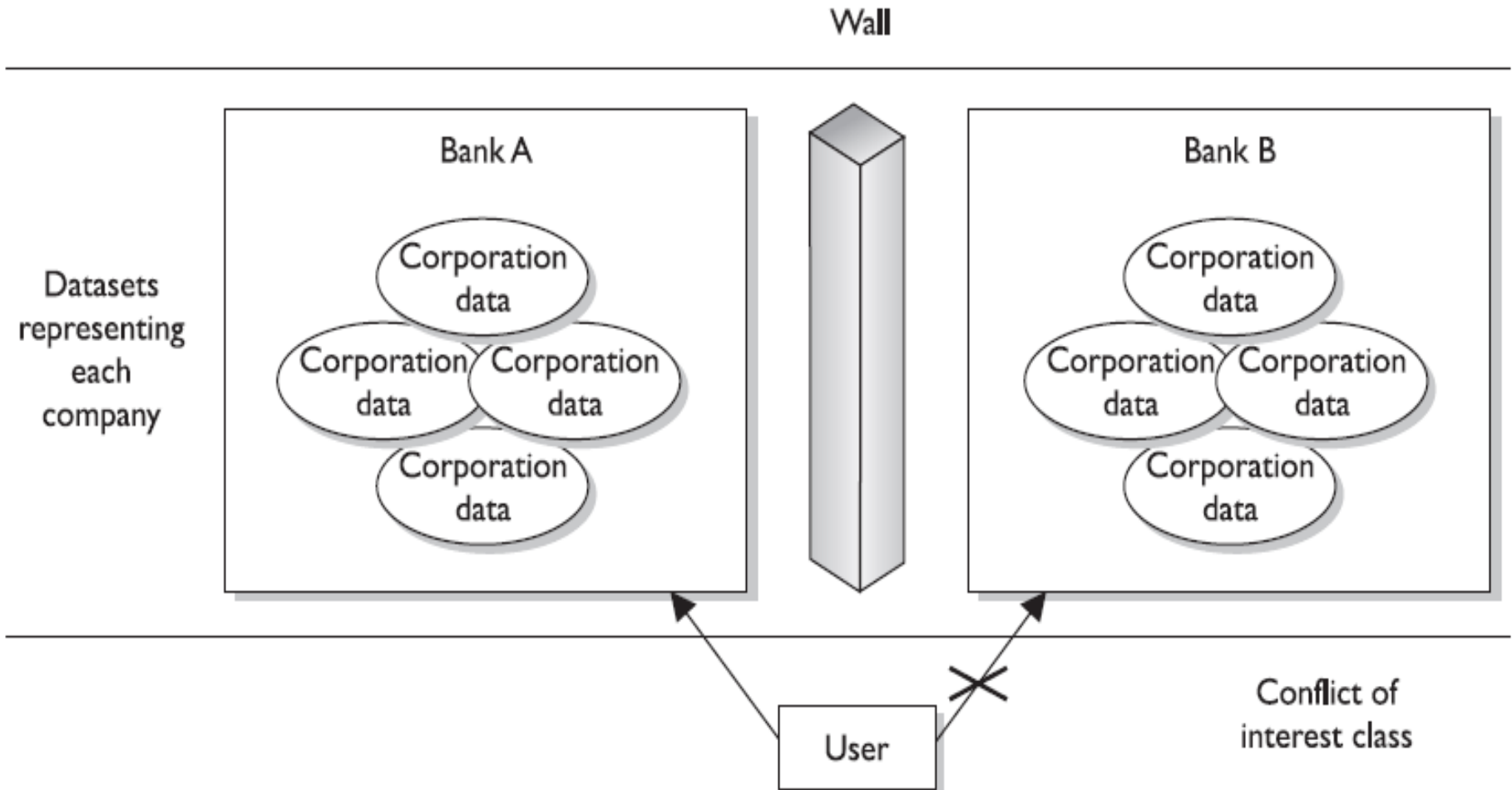
- This model is implemented to ensure any actions that take place at a higher security level do not affect, or interfere with, actions that take place at a lower level.
- Is not concerned with information flow, but what a subject knows about the state of a system. If an entity at a higher level performs an action, it cannot change the state for the entity at a lower level.
- If lower level entities knew about particular activity that occurred at a higher level that affect it at the lower level, the entity could deduce information about the higher levels, an information leak then occurs.
- Main focus is on combating covert channels and *inference attacks*. These are attacks that occur when someone has access to some type of information and can infer something that he does not have the clearance level or authority to know.



Lattice Model



The Brewer and Nash Model (Chinese Wall)



The Graham-Denning Model

- **Defines a set of basic rights in terms of commands that a specific subject can execute on an object.**
- **Rules – How to securely:**
 - **Create/delete an object/subject**
 - **Provide the read access rights**
 - **Provide the grant access rights**
 - **Provide the delete access rights**
 - **Provide the transfer access rights**



Other Models

- **Lattice Model** – a structure consisting of a finite partially ordered set together with least upper and greatest lower bound operators on the set
- **Chinese Wall Model** – allows for dynamically changing access controls that protect against conflicts of interest
- **Harrison-Ruzzo-Ullman Model** – deals with access rights of subjects and the integrity of those rights. Subjects can carry out only a finite set of operations on an object.



Security Modes of Operation

- **Several things come into play when determining the mode the OS should be working in:**
 - **The types of users who will be directly or indirectly connecting to the system**
 - **The type of data (classification levels, compartments, and categories) processed on the system**
 - **The clearance levels, need to know, and formal access approvals the users will have**
- **Guards – software or hardware interfaces between low level security and high level security processes to prevent vulnerabilities**



Security Modes

- **Dedicated Security Mode, All users must have...**
 - **Proper clearance for all information on the system**
 - **Formal access approval for all information on the system**
 - **A signed NDA for all information on the system**
 - **A valid need to know for *all* information on the system**
 - **All users can access all data**
- **System High-Security Mode, All users must have...**
 - **Proper clearance for all information on the system**
 - **Formal access approval for all information on the system**
 - **A signed NDA for all information on the system**
 - **A valid need to know for *some* information on the system**
 - **All users can access some data, based on their need to know**



Security Models

- **Compartmented Security Mode, All users must have...**
 - Proper clearance for the highest level of data classification
 - Formal access approval for all information they will access
 - A signed NDA for all information they will access on the system
 - A valid need to know for *some* information on the system
 - All users can access some data, based on their need to know and formal access approval
- **Multilevel Security Mode, All users must have...**
 - Proper clearance for all information they will access
 - Formal access approval for all information they will access
 - A signed NDA for all information they will access on the system
 - A valid need to know for *some* of the information on the system
 - All users can access some data, based on their need to know, clearance, and formal access approval



Trust and Assurance

- As mentioned earlier in the TCB section, no system is really ever totally secure, given enough resources. But, systems can provide certain levels of trust. The level of trust tells the customer how much protection he can expect out of this system and the *assurance* that the system will act in a correct and predictable manner in each and every computing situation.
- An assurance rating is given after a thorough inspection of the system's development, maintenance, and delivery, which indicates the level of trust in the system.
- In a trusted system, all protection mechanisms work together to process sensitive data and provide the necessary production, whereas assurance looks in more detail. The higher the scrutiny of the design and build, the higher the assurance rating.



Security capabilities of information systems

Dr. Drew Hamilton

MSGT Alex Applegate, USAF (ret)

Dr. C.W. Perr, Sandia Labs



Mississippi State University Center for Cyber Innovation

Domain 3 Security Engineering



Availability & Reliability

- **Availability:** A measurement of whether a system is *ready to be used immediately*
 - System is up and running at any given moment
- **Reliability:** A measurement of whether a system can *run continuously without failure*
 - System continues to function for a long period of time
- A system goes down 1ms/hr has an availability of more than 99.99%, but is unreliable
- A system that never crashes but is shut down for a week once every year is 100% reliable but only 98% available



Safety & Maintainability

- **Safety: A measurement of *how safe failures are***
 - System fails, nothing serious happens
 - For instance, high degree of safety is required for systems controlling nuclear power plants
- **Maintainability: A measurement of *how easy it is to repair a system***
 - A highly maintainable system may also show a high degree of availability
 - Failures can be detected and repaired automatically? Self-healing systems?



Faults

- A system fails when it cannot meet its promises (specifications)
- An error is part of a system state that may lead to a failure
- A fault is the cause of the error
- Fault-Tolerance: the system can provide services even in the presence of faults
- Faults can be:
 - Transient (appear once and disappear)
 - Intermittent (appear-disappear-reappear behavior)
 - A loose contact on a connector → intermittent fault
 - Permanent (appear and persist until repaired)



Failure Models

Type of failure	Description
Crash failure	A server halts, but is working correctly until it halts
Omission failure <i>Receive omission</i> <i>Send omission</i>	A server fails to respond to incoming requests A server fails to receive incoming messages A server fails to send messages
Timing failure	A server's response lies outside the specified time interval
Response failure <i>Value failure</i> <i>State transition failure</i>	The server's response is incorrect The value of the response is wrong The server deviates from the correct flow of control
Arbitrary failure (Byzantine failure)	A server may produce arbitrary responses at arbitrary times

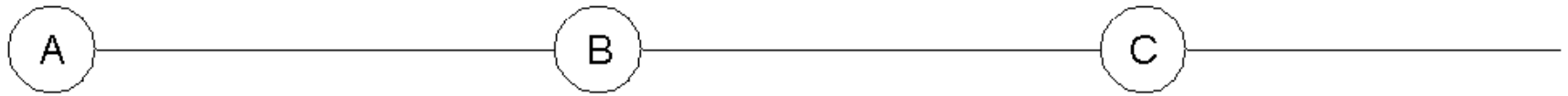


Failure Masking

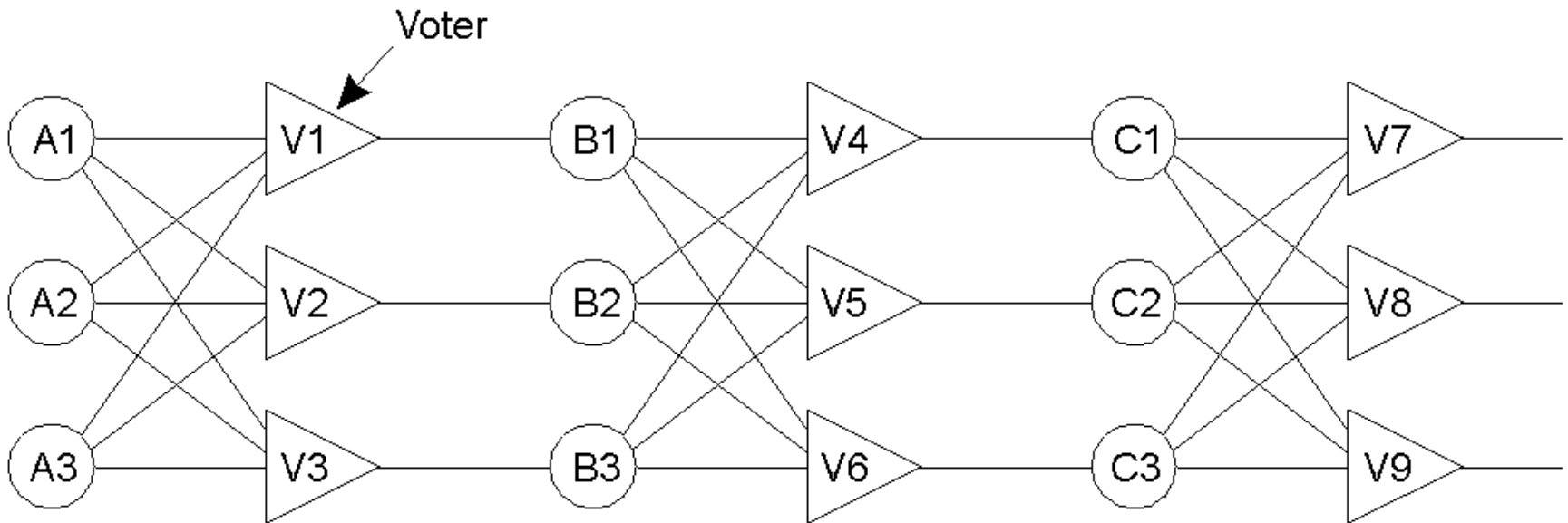
- **Redundancy is key technique for hiding failures**
- **Redundancy types:**
 - 1. Information: add extra (control) information**
 - **Error-correction codes in messages**
 - 2. Time: perform an action persistently until it succeeds:**
 - **Transactions**
 - 3. Physical: add extra components (S/W & H/W)**
 - **Process replication, electronic circuits**



Example – Redundancy in Circuits



(a)

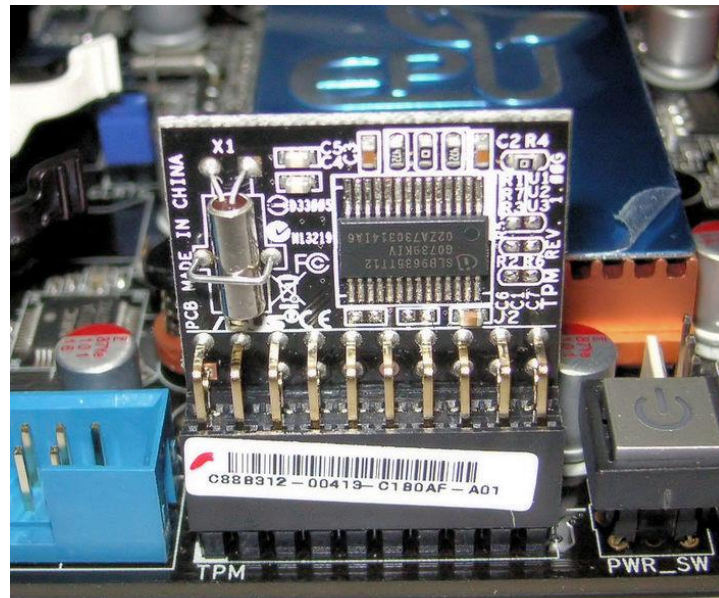
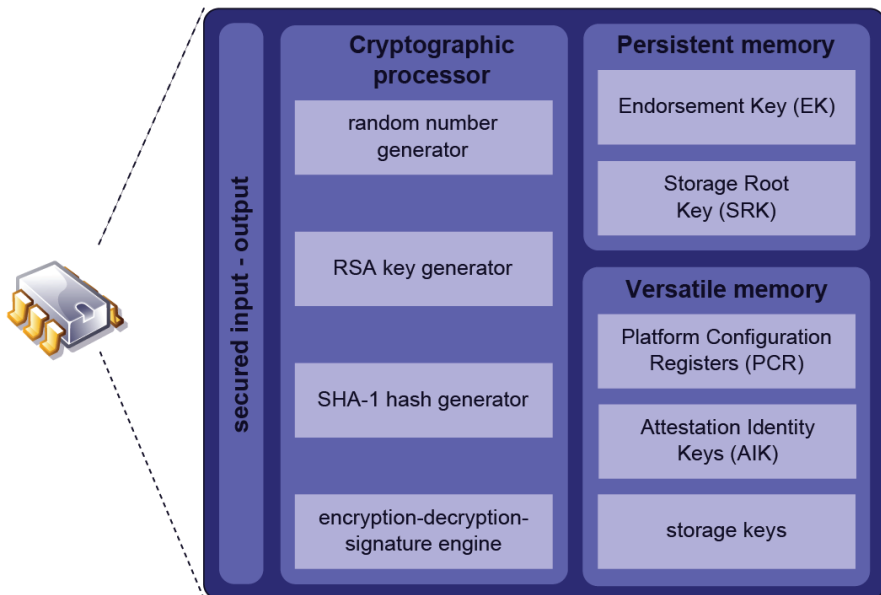


(b)



Trusted Platform Module for Windows

- Bitlocker requires TPM hardware
- Encryption key stored on removable USB drive
 - Not in all versions of Windows 7 / Vista - only enterprise/ultimate versions
 - Limited availability of motherboards with TPM chips
- How good are TPMs?
 - Banned in Russia, China, Belarus and Kazakhstan



The Virtual Machine Concept

- Using special emulation software, an entire operating system can be run inside another nearly as if an independent system
- “Guest” operating system has it’s own hard disk allocation, memory, and even (emulated) CPU/registers
- With enough resources, multiple guest operating machines can run on the same host and behave like a private network



How is a Virtual Machine Helpful?

- **While running in as a guest operating system, a virtual machine is isolated and cannot demonstrably affect the host**
 - Provides a safe forensic work place
- **Process Repeatability**
 - Every time a new forensic investigation or process is initiated, the environment can be completely reset to an initial state



How is a Virtual Machine Helpful? (cont'd)

- **Bootable images (such as forensic copies) can be run directly as a virtual machine**
 - **Prove ability to boot**
 - **Containment if booby trapped**
 - **Ability to reconstruct some process threads and memory structures**
 - **Ability to take snapshots, test, revert, and replay at will**



Additional Capabilities

- **Virtual Machines have the file system, RAM, CPU/registers, and BIOS in a readable state in real time**
- **Using snapshots, an investigator can start, stop, replay, or revert the state of the machine, analyze processes, review the registry, or check network traffic at any time and as often as necessary**



Drawbacks

- **Can be a significant drain of resources**
 - Each virtual machine must be allocated space on the hard drive
 - Each snapshot taken increases that space requirement
 - Requires significant CPU and RAM. A lower-end workstation may not be capable of running more than one virtual guest or doing other work while the virtual machine is running
- **Complexity causes some brittleness, may not always work**



Cloud Computing Background

- **Features**
 - Use of internet-based services to support business process
 - Rent IT-services on a utility-like basis
- **Attributes**
 - Rapid deployment
 - Low startup costs/ capital investments
 - Costs based on usage or subscription
 - Multi-tenant sharing of services/ resources
- **Essential characteristics**
 - On demand self-service
 - Ubiquitous network access
 - Location independent resource pooling
 - Rapid elasticity
 - Measured service
- **“Cloud computing is a compilation of existing techniques and technologies, packaged within a new infrastructure paradigm that offers improved scalability, elasticity, business agility, faster startup time, reduced management costs, and just-in-time availability of resources”**



Cloud Delivery Models

- **SaaS**
 - Software as a service is a software licensing and delivery model in which software is licensed on a subscription basis and is centrally hosted.
 - It is sometimes referred to as "on-demand software", and was formerly referred to as "software plus services" by Microsoft.
- **PaaS**
 - Platform as a service (PaaS) or application platform as a service (aPaaS) is a category of cloud computing services that provides a platform allowing customers to develop, run, and manage applications without the complexity of building and maintaining the infrastructure typically associated with developing and launching an app.
- **IaaS**
 - Infrastructure as a Service (IaaS) is a form of cloud computing that provides virtualized computing resources over the Internet. IaaS is one of three main categories of cloud computing services, alongside Software as a Service (SaaS) and Platform as a Service (PaaS)



Cloud Deployment Models

	Type	Properties
1.	Private cloud	<ul style="list-style-type: none">• Outsource or own• Lease or buy• Separate or virtual data center
2.	Community cloud	<ul style="list-style-type: none">• Private cloud for a set of users with specific demands• Several stakeholders
3.	Public cloud	<ul style="list-style-type: none">• Mega scaleable infrastructure• Available for all
4.	Hybrid cloud	<ul style="list-style-type: none">• Combination of two clouds• Usually private for sensitive data and strategic applications



Security architectures, designs, and solution elements vulnerabilities

Dr. Drew Hamilton



Mississippi State University Center for Cyber Innovation

Domain 3 Security Engineering



109

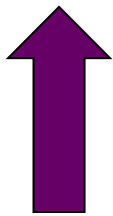
Scale

Handheld

PalmOS
Linux
WIN CE



USB
Serial
IR



Commercial
Model

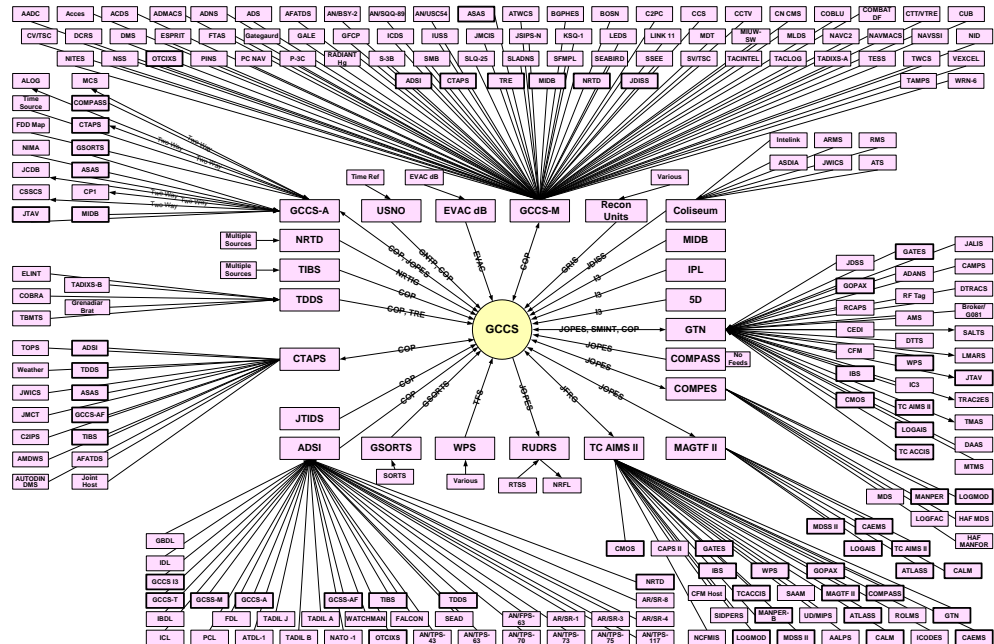
Desktop

Solaris
Windows
Mac OS



Versus

Military Model

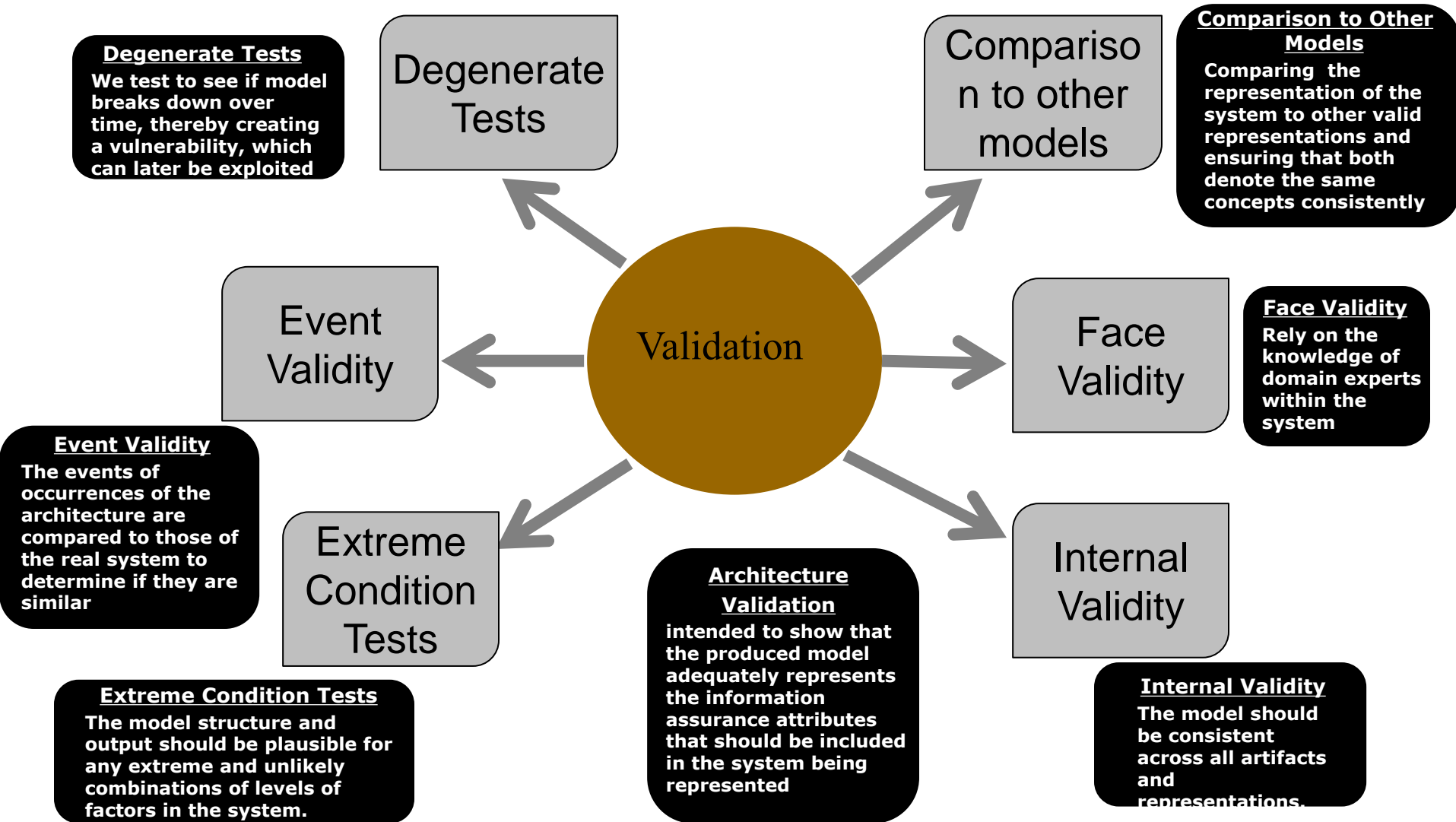


Architectural Definitions

- The DODAF is not merely a “computer systems architecture.” It is broader and consistent with the Institute for Electrical and Electronics Engineers (IEEE) definition:
 - An architecture is composed of “the structures or components, their relationships, and the principles and guidelines governing their design and evolution over time.”



Overview of Model Validation Techniques



An Architectural Approach to Security Metrics

- An initial linkage between DODAF and DIACAP

IA Control Subject Area	IA Control Number	IA Control Name	Inherited? (Y/N)	C/NC/NA	Impact Code	Related Views
Access Control	AC-1	Access Control Policies and Procedures	N	C	Medium	OV-6a
Access Control	AC-2	Account Management	N	NC	High	Not implemented
Access Control	AC-2 (1)	Account Management	N	C	Medium	OV-5 Maintenance
Access Control	AC-2 (2)	Account Management	N	C	Low	OV-6a, SV-10a
Access Control	AC-2 (3)	Account Management	N	NC	Medium	Not implemented
Access Control	AC-2 (4)	Account Management	N	NC	Low	Not implemented
Access Control	AC-2 (5)	Account Management	N	C	Medium	OV-4, OV-5 Maintenance
Access Control	AC-3	Access Enforcement	Y	C	Medium	OV-4
Access Control	AC-3 (1)	Access Enforcement	Y	C	Medium	OV-4
Access Control	AC-3 (2)	Access Enforcement	Y	C	Medium	OV-4
Access Control	AC-4	Information Flow Enforcement	N	C	Medium	SV-2,OV-3
Access Control	AC-4 (2)	Information Flow Enforcement	N	NC	Low	Not implemented
Access Control	AC-5	Separation of Duties	N	C	High	OV-4
Access Control	AC-6	Least Privilege	N	C	Medium	OV-4
Access Control	AC-6 (1)	Least Privilege	N	C	Medium	OV-4
Access Control	AC-7	Unsuccessful Login Attempts	Y	C	Medium	SV-10a, SV-10b, SV-10c
Access Control	AC-7 (1)	Unsuccessful Login Attempts	Y	C	Medium	SV-10a, SV-10b, SV-10c
Access Control	AC-8	System Use Notification	Y	C	Low	SV-10a
Access Control	AC-9	Previous Logon Notification	N	NC	Low	Not implemented
Access Control	AC-10	Concurrent Session Control	N	NC	Medium	Not implemented
Access Control	AC-11	Session Lock	N	C	Medium	SV-10a, SV-10b
Access Control	AC-11 (1)	Session Lock	N	NC	Medium	Not implemented
Access Control	AC-12	Session Termination	Y	C	High	SV-10a, SV-10c
Access Control	AC-12 (1)	Session Termination	Y	NC	Medium	Not implemented
Access Control	AC-12 (2)	Session Termination	Y	C	Medium	SV-10a, SV-10c
Access Control	AC-13	Supervision and Review - Access Control	N	NC	Medium	Not implemented
Access Control	AC-13 (1)	Supervision and Review - Access Control	N	NC	Medium	Not implemented



DODAF 2.0 Information Assurance Architecture – Traceability

IA Control Subject Area	IA Control Number	IA Control Name	Inherited? (Y/N)	C/N/C/NA	Impact Code	Related Views	Approximate Cost of Non-Compliance	Approximate cost of implementation
Access Control	AC-1	Account Management	N	NC	High	OV-6a	100.00	
Access Control	AC-2 (1)	Account Management	N	C	Medium	Not implemented	200.00	
Access Control	AC-2 (2)	Account Management	N	C	Low	OV-6a, SV-10a		
Access Control	AC-2 (3)	Account Management	N	NC	Low	Not implemented	250.00	
Access Control	AC-2 (4)	Account Management	N	NC	Low	Not implemented	100.00	
Access Control	AC-2 (5)	Account Management	N	C	Medium	OV-4, OV-5 Maintenance		
Access Control	AC-2 (6)	Account Management	N	C	Medium	OV-4, OV-5 Maintenance		
Access Control	AC-3 (1)	Access Enforcement	Y	C	Medium	OV-4		200.00
Access Control	AC-3 (2)	Access Enforcement	Y	C	Medium	OV-4		200.00
Access Control	AC-4	Information Flow Enforcement	N	C	Medium	SV-2(1-3)		200.00
Access Control	AC-4 (2)	Information Flow Enforcement	N	NC	Low	Not implemented		
Access Control	AC-5	Separation of Duties	N	C	High	OV-4		300.00
Access Control	AC-6	Least Privilege	N	C	Medium	OV-4		100.00

Control Sets Scorecard

The DIACAP process provides a scorecard that can be used to evaluate a system against a set of standard control sets. In this example, we will show how the Account Management control is modeled and evaluated using the DODAF IA architecture.

Account Management Control

Control: The organization manages information system accounts, including establishing, activating, modifying, reviewing, disabling, and removing accounts. The organization will:

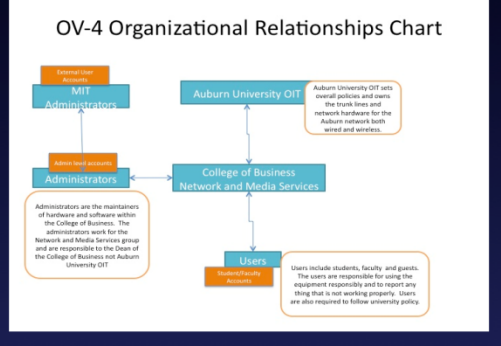
- a. Reviews information system accounts [Assignment: Organization-defined frequency] or at least annually.
- b. Identifies authorized users of the information system and specifies access rights/privileges.
- c. Requires proper identification for requests to establish information system accounts and approves all such requests.
- d. Authorizes and monitors the use of guest/anonymous accounts and removes, disables, or otherwise secures unnecessary accounts.
- e. Notifies account managers when information system users are terminated or transferred and associated accounts are removed, disabled, or otherwise secured.
- f. Notifies account managers when users' information system usage, privileges or access authorizations change.

Control Enhancements:

- (1) The organization employs automated mechanisms to support the management of information system accounts.
- (2) The information system automatically terminates temporary and emergency accounts after [Assignment: organization-defined time period for each type of account], not to exceed 72 hours.
- (3) The information system automatically disables inactive accounts after [Assignment: organization-defined timeperiod], not to exceed 30 days.
- (4) The organization employs automated mechanisms to audit account creation, modification, disabling, and termination actions and to notify, as required, appropriate individuals.
- (5) The organization establishes and administrators all privileged user accounts in accordance with a role-based access scheme that organizes all system and network privileges into roles (e.g., key management, network system administration, database administration, web administration). The Information System Security Manager (ISSM), Information Assurance Manager (IAM) tracks privileged role assignments.

OV-6a Operational Rules Model

- Before any workstation can be used a proper authentication must be performed.
 - This authentication is accomplished by using the AU username and password or a COB username and password.
 - There are several account levels in the system that correspond to user groups as shown in the OV-4. These are administrators, faculty, students, and external users. These accounts are controlled both manually and automatically. Administrators can install applications on workstations, configure and patch updates and servers, and change permissions of other users. External users can only access the shared portion of the COB server through standard protocols (e.g., ftp).
 - There are no specific software usage restrictions enforced. If a user can authenticate to a server/workstation, then they are authorized to use any software on that machine.
 - Software can only be loaded by administrators. If additional software is needed then the appropriate application is installed by administrators.
- Any temporary accounts that are created are deleted within 24 hours.
 - Any information is available to anyone able to authenticate to a workstation.
- Printing is done by attempting to print from any office automation suite at a workstation.
 - The print request is sent to the print server and when authorized the print job is then sent to the networked printer.
- Information gathering is authorized for anyone able to authenticate to a workstation.
 - Each individual can access their own email accounts from any workstation.
 - All internet use policies enforced by Auburn University are enforced at all workstations.
 - All incoming content is scanned for spyware and viruses and dealt with immediately.
 - All workstations are mapped and can transfer files within the network only. No file transfers from outside sources are authorized. (Such as home computers.)
- Maintenance is performed by administrators from the College of Business.
 - This includes any hardware maintenance or software updates, additions or deletions.
- The organization maintains several policies related to information assurance, that can be found in the information assurance policy documentation maintained separately from the software description.



SV-10a Systems Rules Model cont.

Authentication Rules:

Accounts:

- There are 3 different levels of groupings: Students, Faculty, Administration.
- Each of these groupings has associated permissions levels on the Lab workstation.
- Temporary accounts are automatically deleted 30 days after student graduation or termination.
- Temporary accounts are deleted after 24 hours using automated tools on College of Business Server.

Password Creation Strength:

- If user has university approved password (credentials) Action is allowed.
- If user does not have university approved password (credentials) Action is not allowed.

Login Rules:

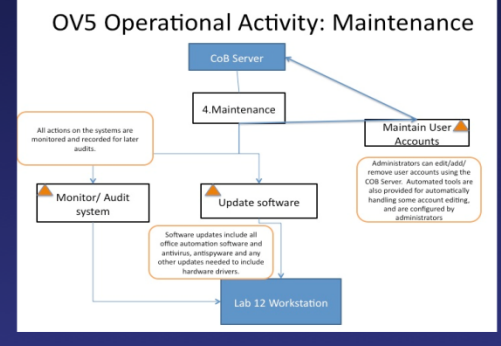
- If user attempts local login AND password authorized by authentication server Action is allowed.
- If user attempts remote login Action is not allowed.
- If password rejected by authentication server Action is not allowed.
- If the user attempts to login 3 times and fails the account will be locked.
- The user will have to have the account unlocked by the COB administrator.
- The user will be displayed a disclaimer upon login notifying them of system use parameters, including the monitoring of all activity possible including keystrokes.
- The workstation will lock after 15 minutes or inactivity. The user will have to login again to use that workstation.
- No actions may be performed on the system without a successful login.
- The system authenticates the hardware being connected by IP and MAC address before authenticating the user.

Maintenance:

- COB administrators are the only personnel allowed to perform maintenance on Lab 12 workstations.
- Maintenance is performed after hours if at all possible to prevent reduction in productivity.
- Anti-virus software/hardware is updated at least daily sometimes more than once a day.
- All other software updates are performed on a weekly basis unless a special need is indicated. (Such as a critical security patch for Microsoft Windows).
- Administrators are responsible for checking maintenance tools and updates for any malicious content.

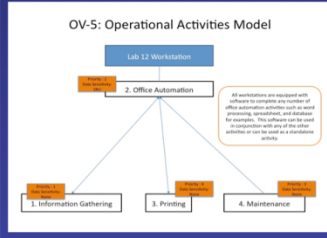
These items are directly related to Information Assurance.

- College has virtual machines for remote VPN login for users located on office automation server.



DODAF artifacts

The DODAF artifacts demonstrate the application (or lack thereof) of each control set. We can see that control AC-2 (2) is explicitly covered in the OV-6a and SV-10a, while controls AC-2 (3) and (4) are not implemented in this system.



Risk Model

Each control set is analyzed to assess the related vulnerabilities in the system. Residual risk that is not covered by the control set as well as any risk introduced by the control set is determined. The operational risk is determined as the operational activities put at risk by the vulnerabilities, and is linked to the operational views.

Understanding Security Architecture

Dr. Drew Hamilton



Mississippi State University Center for Cyber Innovation

Domain 3 Security Engineering



115

Agenda

- **Security Architecture Common Elements**
- **Principles of Security Architecture**
- **Security Architecture Process**
- **Models Capturing Security Architecture**
- **The Institute for Critical Information Infrastructure Protection (ICIIP) MODEL**
- **Security Architecture Comparison and Assessment**
- **Conclusion**



Security Architecture

- In February 2003, the White House released the National Strategy to Secure Cyber Space that responded to an urgent need for users, operators, and vendors of networked data and communications systems from both public and private sectors to work together to improve the security of the nation's information infrastructure.
- The National Strategy proposed the following goals:
 - 1) preventing cyber attacks against America's critical information infrastructures,
 - 2) reducing national vulnerability to cyber attacks, and 3) minimizing damage and recovery time from cyber attacks that may actually occur.



Security Architecture Common Elements (1/8)

1) Network security architecture:

Network security may be achieved by:

- Eliminating network components that use shared Ethernet.
- Implement the concept of defense
- Use multiple firewalls within network.
- Implement intrusion detection systems at key points within network to monitor threats and attacks.
- Measure and report network traffic statistics for the computers on the network.



Security Architecture Common Elements (2/8)

2) Host based security architecture:

This can be achieved through good system administration practices such as:

- Maintain up to date virus protection.
- make sure that system software are configured properly, and latest patches are installed.
- Perform risk assessment to identify the most important computers to protect.
- Disable network services that are not needed and run host-based firewall on computers to block unwanted network traffic.
- Monitor security alerts and develop mechanism for quickly patching systems.
- Create centralized system logging service.
- Develop central authentication service to replace host-based password files



Security Architecture Common Elements (3/8)

3) Application Security Architecture:

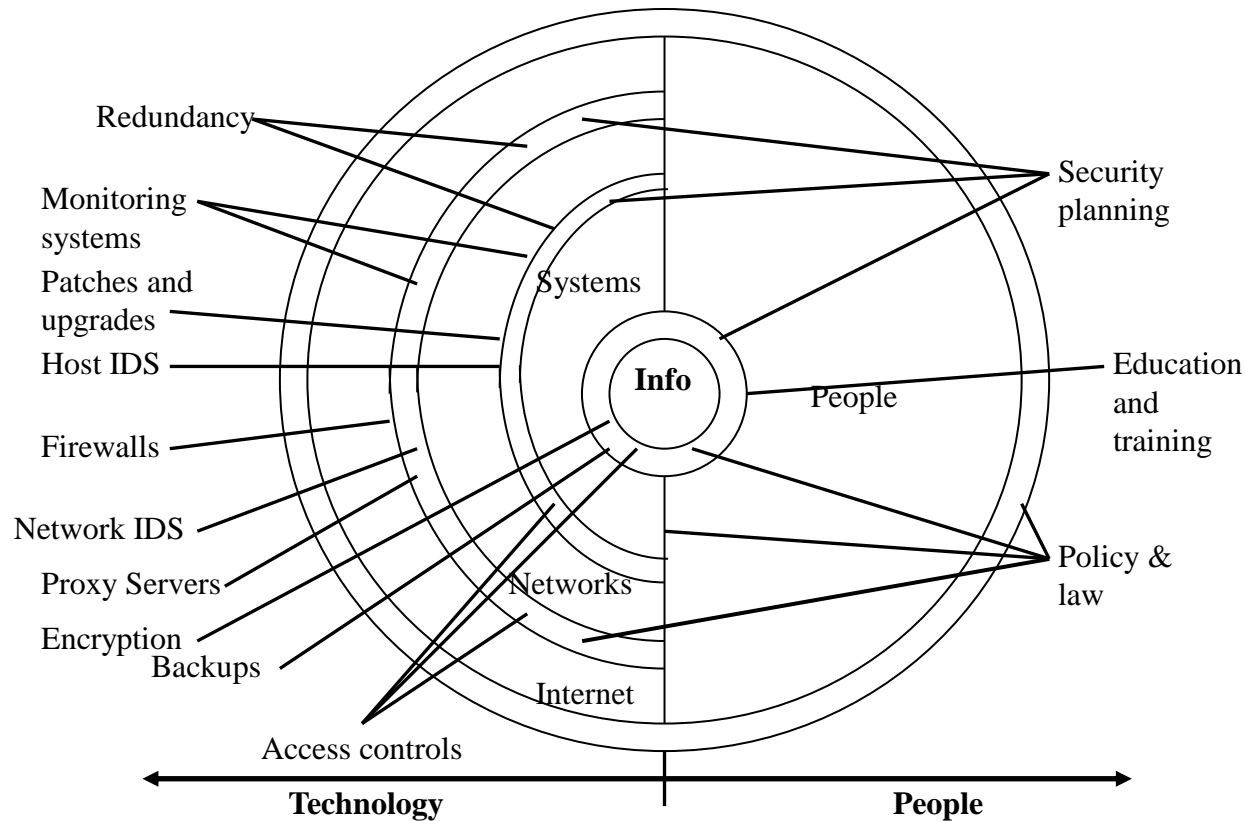
Application security deals with two main concerns:

- Protecting the code and services running on the system, protecting who is connecting to them and protecting what is output from the programs.
- Delivering reusable application security services such as reusable authentication, authorization, and auditing services enabling developers to build security into their system.



Security Architecture Common Elements (4/8)

4) Data and information Security Architecture



Security Architecture Common Elements (5/8)

5) Software Security Architecture:

- Some examples of software vulnerabilities are:
 - software deletion,
 - software modifications by using Logic bomb,
 - Trojan horse,
 - viruses, worms, and Trapdoor,
 - Information leaks,
 - inserting malicious code
- There are two advantages of Software Protection:
 - Portability which ensures that a user-level software-product must coexist with a variety of operating systems. For example, it allows a browser to have platform-independent security mechanisms.
 - Performance since it offers significantly cheaper cross-domain calls whereas if they were implemented in hardware they would slow programs to an unacceptable level [Wallach et al. 1997].



Security Architecture Common Elements (6/8)

6) Hardware

Attacks against hardware:

- Since it is easy to identify and see the devices that are connected to the system, it is easy to attack by adding, changing, removing, intercepting traffic to and flooding with traffic the devices connected to the system.
- Hardware may suffer accidental acts that are not intentional “involuntary machine slaughter” where it can be drenched with water, burned, frozen, gassed or electrocuted with power surges.
- “Voluntary machine slaughter” which a person actually wants to harm the hardware of a system.



Security Architecture Common Elements (7/8)

7) Database

Security can be addressed by:

- Operating system integrity control and recovery procedures.
- Element integrity is achieved by using the proper access control to protect a specify data element from being changed or written by unauthorized users.
- Element accuracy is ensured by using checks on the values of elements that can be used to prevent the insertion of improper values.
- Constraint conditions can be used to detect incorrect values.
- Two-phase update is used to ensure that an update operation is performed on the complete record and that no part of the data was updated before the operation is aborted for what ever reason.
- The database recover data by maintaining a log of users' accesses and what they have changed.
- The concurrency/consistency problem resulting from many users accessing or sharing the same database can be solved by using different kinds of locks.
- In multilevel databases, two levels of security for individual elements that are sensitive and non-sensitive are inadequate; therefore, each element should be associated with a related sensitivity level.

Security Architecture Common Elements (8/8)

8) Physical Security

- It is in general used to describe the security needed outside the computer system.
- Some examples of the natural disasters that may affect a system are flood and fire.
- Damage may also result from power loss that can be because of an uninterruptible power supply or surge suppressors.
- Human vandals may physically attack systems which can be easily prevented by employing guards or using locks.

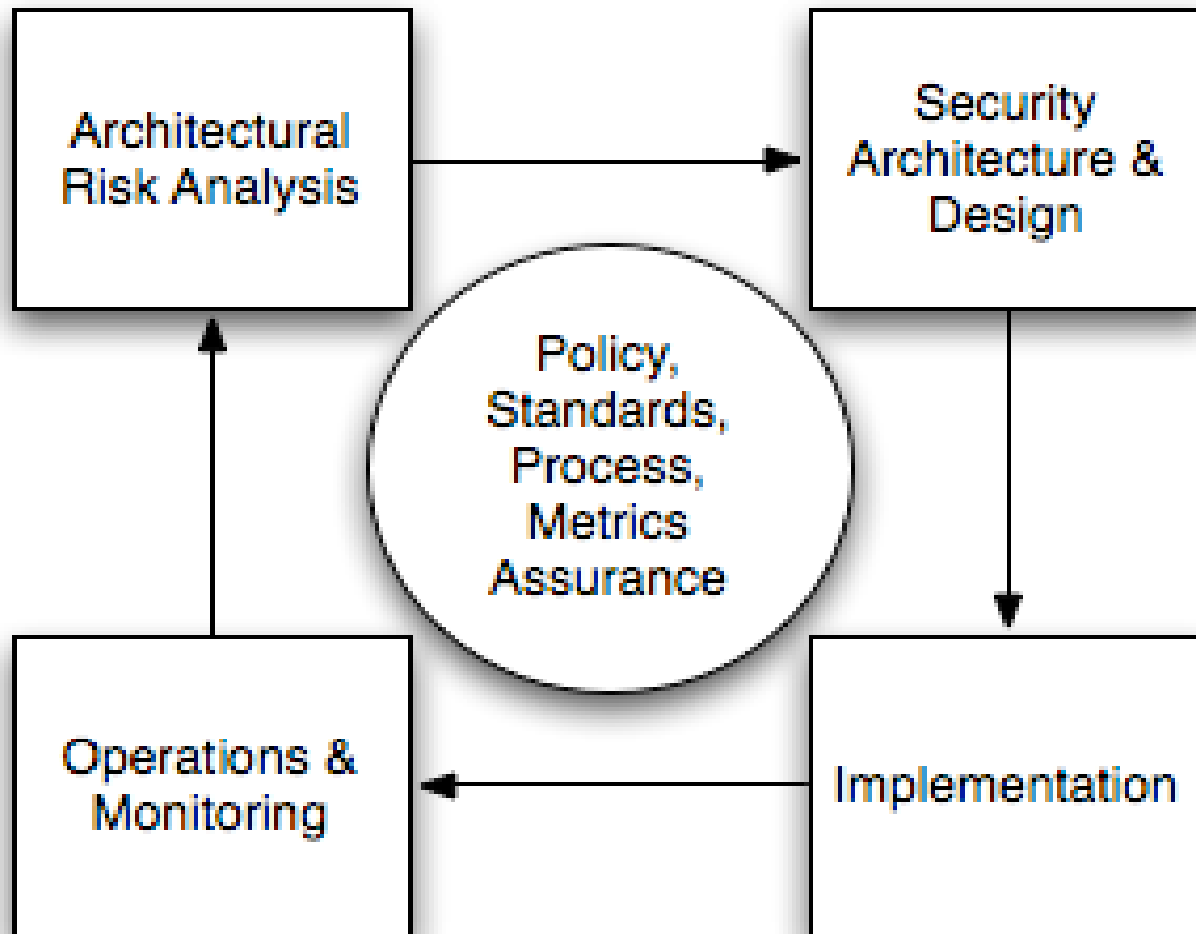


Principles of Security Architecture

- **Set a security policy for the system and know what's on it.**
- **Verify actions.**
- **Always give the least privilege practical.**
- **Practice defense in depth and not rely on one form of security precaution.**
- **Auditing the system and keep (and review) system logs.**
- **Build the system to contain intrusions and minimize the consequences when a system is cracked.**
- **A system is only as strong as its weakest link and the more defenses a system has, the less likely that the weakest one will leave it vulnerable.**
- **The only way to be reliably certain that the system is secure after being successfully attacked is to reinstall the BIOS, reformat the hard drive, and restore files from a backup taken before the system was compromised.**
- **Practice full disclosure. When a system is successfully attacked, or is known to be vulnerable, let users know as soon as possible.**



Security Architecture Process



Models Capturing Security Architecture

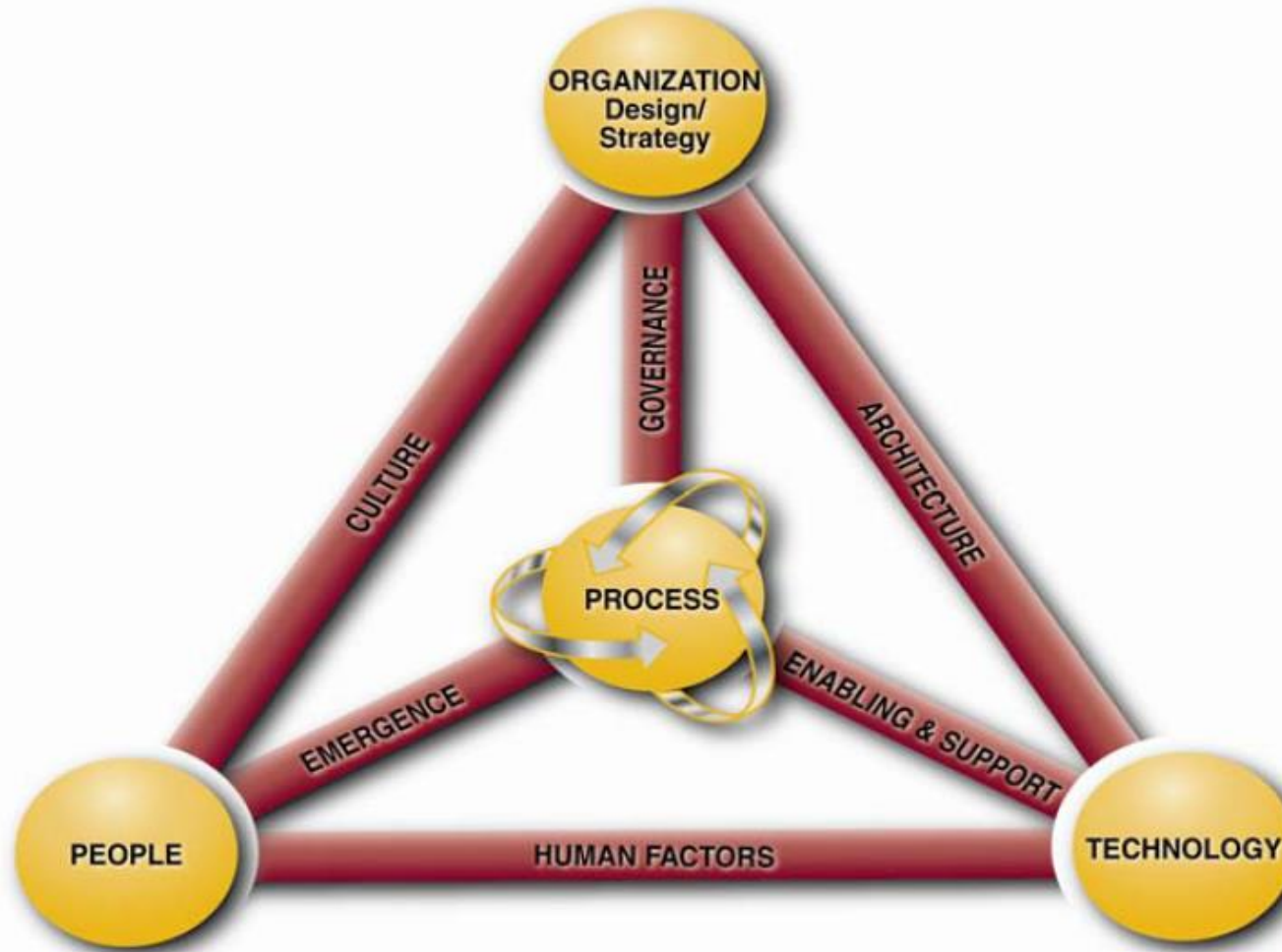
- **Designing a security architecture requires capturing the architecture in an appropriate way.**
- **The representation should be clear, concise and consistent to facilitate easy analysis and comparison of architectures.**

Models for capturing architecture:

- 1) **The Domain Approach is easy to understand and would allow a concise representation of an organization's discrete information sets along with any appropriate physical elements such as buildings, server rooms, and printers.**
- 2) **The Defense Architecture Framework (DoDAF) does not deal specifically with information security, and is likely too broad to be ideally suited to architecture capture.**
- 3) **3) The International Common Criteria's Protection Profiles are formal documents that could certainly capture security architecture, but perhaps at an unnecessary level of detail.**



The Institute for Critical Information Infrastructure Protection (ICIIP) MODEL



Security Architecture Comparison and Assessment

Techniques used to compare among and assess different security architectures:

- Bayesian networks allow considering the effect of countermeasures on potential attacks. However, justifying the data used in Bayesian networks is a serious issue that needs to be considered.
- Simulation has dynamic nature, giving decision-makers knowledge of the architecture. However, it relies on the existence of an accurate model, which is hard to obtain in the information security domain. With risk analysis, unavailable or inaccurate data can reduce their effectiveness.
- The IATF robustness strategy provides minimum requirements on architectures, but the incompleteness of the strategy and its US specific requirements are issues to be considered.
- Game theory could theoretically provide optimal designs for security architectures. Unfortunately, it is not well developed enough for the information security domain to be relied upon.
- Survivability analysis techniques are useful for architecture assessment, but are restricted to architectures containing networks.
- Economic models have practical, non-technical uses, incorporating a human factors and system view into the security architecture analysis. However, they do not provide the most important answers for government and Defense information systems.



Conclusion

- **To insure security, it is important to build-in security in both plan and design phases.**
- **There is no single solution for a security architecture that will work across all organizations and the infrastructure is constantly evolving.**
- **Common elements of a security architecture are not the only important factor to be considered.**
- **We need to consider, for example, other security architecture models, comparison and assessment techniques.**



Web-based systems vulnerabilities

Dr. Drew Hamilton



Mississippi State University Center for Cyber Innovation

Domain 3 Security Engineering



132

Background

- **Separation of program and data is once more abolished by web-based applications**
 - **Content providers embed executable content in documents to create interactive web pages that can process user input**
- **Computation is moved back to the client**
 - **Documents include executable code**
 - **Clients run on quite powerful machines**
 - **Servers free themselves by offloading computation to clients**
 - **Clients need protection from rogue content providers**
- **Mobile code moves from machine to machine collecting information from different places or looking for spare computing resources**
- **Users are forced to become sys admins and make policy decisions**



Browsers

- Provide “bells and whistles” to support attractive presentation of content
- Is a service layer for web applications
- Includes the protocols to communicate with web servers
- Manage security relevant information for the client



Browsers in the Trusted Computer Base

- **Browsers handle web traffic**
 - has to indicate return address as a minimum
 - lack of privacy protection as the server can build up a database about its clients
- **Browsers manage default settings and preferences**
 - default settings include the location of executables
 - security prefs indicate the protection clients want to apply to their web session
- **Browsers keep a cache of recently visited pages**
 - this is convenient for the user
 - consider using a terminal in an airport lounge
- **Browsers often run in “system mode” with full access to all system resources**



Browsers in the Trusted Computer Base (cont.)

- **Web security applications use encryption and digital signatures**
 - When performed for the client, browser entrusted with the client's private keys
 - Browsers today come with the root verification keys of major certification bodies
 - Browser must protect
 - verification keys from modification
 - signature keys from disclosure
 - encryption keys from disclosure
- **Browsers integrate other comm services like email**
 - Unnecessary use of a complex program to run email
 - Email messages can exploit browser bugs
 - Unexpected interactions
- **Overall, browsers being used for functions they were not intended for**



CGI Scripts

- **Common Gateway Interface**
 - **Metalanguage (middleware)**
 - **translates URLs or HTML forms into runnable programs**
 - **Scripting languages used**
 - **Perl, TCL, etc.**
 - **Server Side Includes (SSIs)**
 - **SSI in-lines**
 - **example: page counter**



Sample CGI Attack

- A script for sending a file to a client may look like:

```
cat thefile | mail clientaddress
```

- where thefile is the name of the file and clientaddress is the mail address of the client

- When a malicious user enters:

```
cat thefile | mail user@address | rm -rf/
```

- as the mail address the server will execute and after mailing the file to the user, delete all files the script has permission to delete



Options for aliasing CGI on a UNIX Server

- **Script-aliased CGI:**
 - all CGI scripts are put into one directory
 - e.g. `./cgi-bin` in the web server root directory, e.g. `/var/httpd`
 - EASIER to find and track all CGI scripts
- **Non-script-aliased CGI:**
 - all CGI scripts are identified by their extension, e.g. `.cgi`.



Securing CGI Scripts on UNIX

- **Need UID for web server program**
 - do not run web server program as “root”
- **Create a special web server UID and carefully control its access rights**
 - Do not share UID with other services
- **Conduct code review of installed CGI scripts**
 - use public resources for checking
 - Different issues between say ENS and Earthlink
- **EXEC operator with argument cmd**
`<!#exec cmd = “myprogram myparamters” ->`
 - passes the string myprogram myparameters to /bin/sh for execution
 - malice can come from the program or the parameters, particularly if myparameters contains a shell escape
 - Options Includes NOEXEC
 - Unescape operation gets rid of shell escapes in input coming from the client by commenting out escape characters



Cookies

- **Where web servers store information about their customers**
 - searching large customer databases on server costly
- **HTTP requests do NOT automatically identify individual users**
 - Thus easier to use a cooperating browsers' customer side
 - Server requests browser to store a cookie that contains information the server will use the next time the client calls
 - **.netscape/cookies**
- **Cookies give browsers the chance to create stateful HTTP sessions**
- **Privacy**
 - cookies stored by the browser create client profiles



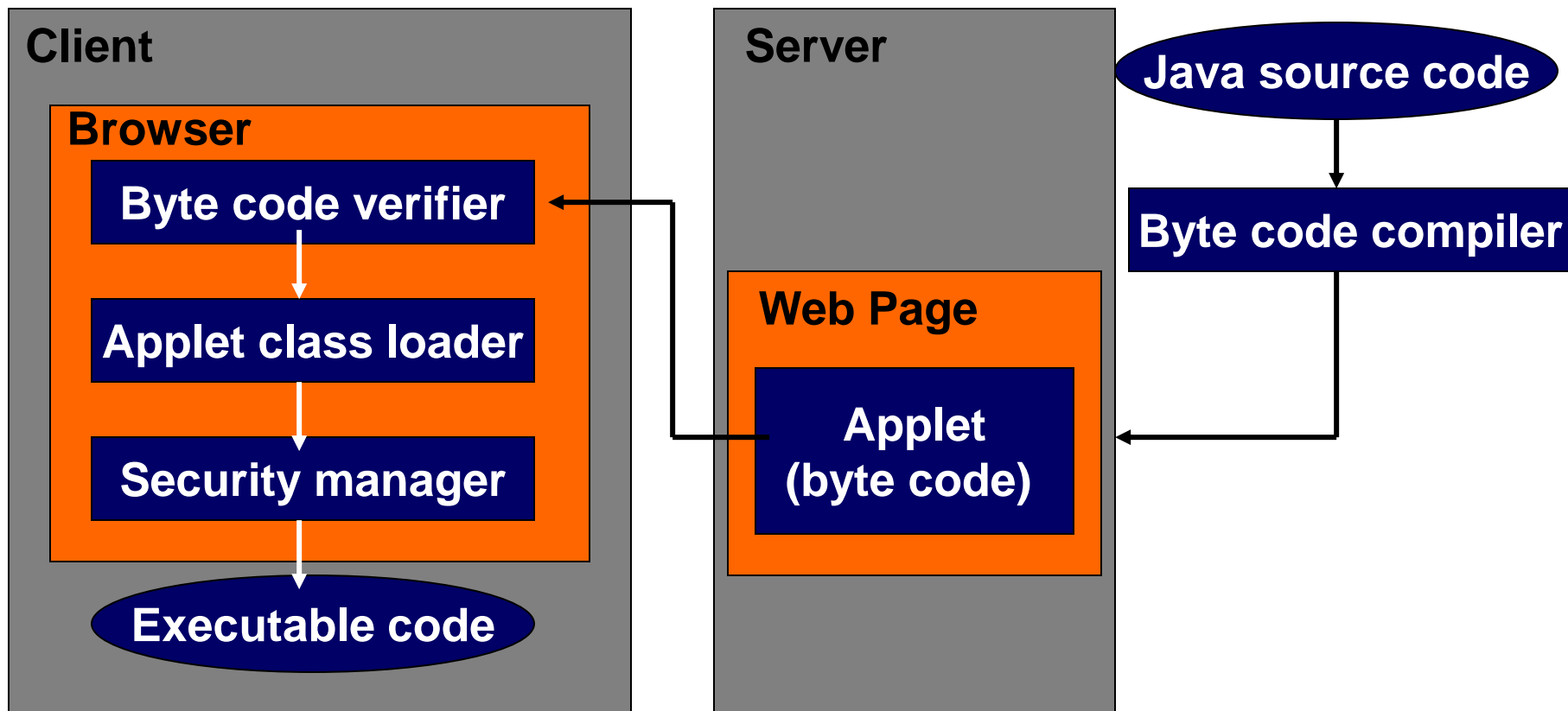
UNIX Cookie Example (.netscape/cookies)

- **www.marion-institute.org FALSE /
FALSE 2137622427 CFID 475137**
- **www.marion-institute.org FALSE /
FALSE 2137622427 CFTOKEN 2642479**
- **.bravenet.com TRUE / FALSE 1373583329
HASCOOKIES 1**
- **.bravenet.com TRUE / FALSE 1293837163
BNUC366777 1058309425**
- **marionmilitary.edu FALSE / FALSE
2137622427 CFID 475161**
- **marionmilitary.edu FALSE / FALSE
2137622427 CFTOKEN 92849103**



Java Sandbox

- Executable Content (applets) from remote web sites



Environment for Applets

- Users cannot rely on prior acquaintance and trust relationship with the source of an applet
- Few users are willing to rule personally on each access request made by an applet
- Client's operating system cannot be expected to offer any protection



Language Design Decisions (Java)

- The language itself should make it more difficult for programs to create damage.
- The execution environment provides mechanisms for access control
- The security policies enforced by the execution environment have to be set correctly



Security for Executable Java Applets (Objectives)

- Applets do not get access to the user's file system
- Applets cannot obtain information about the user's name, email address, machine configuration, etc.
- Applets may make outward connections only back to the server they came from
- Applets can only pop-up windows that are marked "untrusted"
- Applets cannot reconfigure the system, e.g. by creating a new class loader or a new security manager



Byte Code Verifier

- **Checks for:**
 - the class file is in the proper format
 - stacks will not overflow
 - all operands have the correct type
 - there will be no data conversion between types
 - all references to other classes are legal
- **Byte code verifier reduces the workload on the interpreter**
 - guaranteed code properties do not have to be checked again
- *However, security still depends on the run-time environment*



Applet Class Loader

- **Class loader protects the integrity of the run-time environment**
- **Applets must not be allowed to create their own class loaders**
 - Applets are handled by the applet class loader
- **Java comes with its own class library**
 - The CLASSPATH environment variable specifies the location of built-in classes
 - The security issues associated with altering CLASSPATH should be obvious
- **“Spoofing” of the CLASSPATH can be avoided by:**
 - If the applet class loader first searches the built-in classes in the local name space
 - Then expand search to the class making the request



Security Manager

- **Reference Monitor of the Java Security Model**
 - Performs run-time checks on ‘dangerous’ methods
- **Java classes are grouped into packages**
 - packages facilitate rudimentary access control to classes
- **Variables and methods can be declared as follows:**
 - **Private:** only the class creating the variable or method has access
 - **Protected:** only the class creating the variable or method and its subclasses have access
 - **Public:** all classes have access
 - **None of the above:** only classes within the same package have access



Summation of Web Threats

	Threats	Consequences	Countermeasures
Integrity	<ul style="list-style-type: none"> •Modification of user data •Trojan horse browser •Modification of memory •Modification of message traffic in transit 	<ul style="list-style-type: none"> •Loss of information •Compromise of machine •Vulnerability to all other threats 	<ul style="list-style-type: none"> •Cryptographic checksums
Confidentiality	<ul style="list-style-type: none"> •Eavesdropping on the Net •Info theft from server •Info theft from client •Info about network configuration •Info about which clients talk to server 	<ul style="list-style-type: none"> •Loss of Information •Loss of Privacy 	<ul style="list-style-type: none"> •Encryption, •Web Proxy
Denial of Service	<ul style="list-style-type: none"> •Killing of user threads •Flooding machine with bogus threats •Filling up disk or memory •Isolating machines by DNS attack 	<ul style="list-style-type: none"> •Disruptive •Annoying •Prevent user from getting work done 	<ul style="list-style-type: none"> •Difficult to prevent
Authentication	<ul style="list-style-type: none"> •Impersonation of legitimate users •Data Forgery 	<ul style="list-style-type: none"> •Misrepresentation of user •Belief that false information is valid 	<ul style="list-style-type: none"> •Cryptographic techniques



Cookies

- Where web servers store information about their customers
 - searching large customer databases on server costly
- HTTP requests do NOT automatically identify individual users
 - Thus easier to use a cooperating browsers' customer side
 - Server requests browser to store a cookie that contains information the server will use the next time the client calls
 - **.netscape/cookies**
- Cookies give browsers the chance to create stateful HTTP sessions
- Privacy
 - cookies stored by the browser create client profiles



Language Design Decisions (Java)

- The language itself should make it more difficult for programs to create damage.
- The execution environment provides mechanisms for access control
- The security policies enforced by the execution environment have to be set correctly



Java Review: Applets vs. Applications

From: *Java in a Nutshell – Flanagan*

- “A **program** in Java consists of one or more class definitions, each of which has been **compiled** into its own .class file of **Java Virtual Machine object code**.”
 - One of these classes must define a method `main()`, which is where the program starts running.
 - To invoke a Java program you run the Java interpreter, *java*, and specify the name of the class that contains the `main()` method.
- A Java applet is NOT an application – it is a Java class that is loaded and run by an already running Java application such as a web browser or an applet viewer.
- Note: Ada 95 has this capability – i.e. “Adapplets.”



Security for Executable Java Applets (Objectives)

- Applets do not get access to the user's file system
- Applets cannot obtain information about the user's name, email address, machine configuration, etc.
- Applets may make outward connections only back to the server they came from
- Applets can only pop-up windows that are marked "untrusted"
- Applets cannot reconfigure the system, e.g. by creating a new class loader or a new security manager

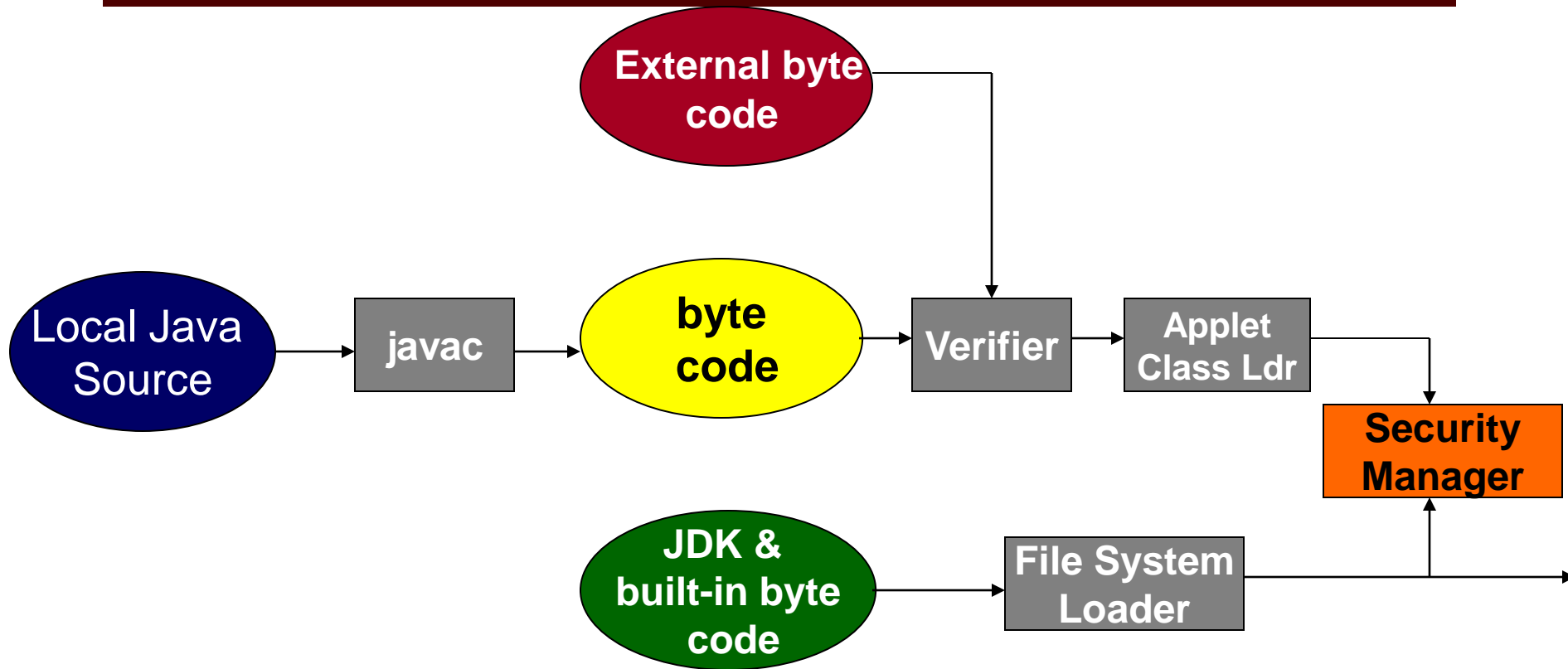


Applications versus Applets

- An applet loaded across the network it is not allowed to:
 - read/write/delete files on the client file system
 - no use of File.delete() method or sys calls rm or del
 - rename files on the client file system
 - no use of File.renameTo() or mv or rename commands
 - conduct directory operations
 - content listing
 - check for existence of a file
 - obtain file information – size, type and modification time stamp
 - conduct network operations
 - create a network connection to any computer other than the host from which it originated
 - listen for or accept network connections on any port in the client system
 - specify any network control functions – SocketImplFactory, etc.....
 - read or define any system properties
 - run or exit any program
 - no use of Runtime.exec(), System.exit() or Runtime.exit() methods
 - load dlls on the client system using load() or loadLibrary()
 - thread creation or manipulation
 - create a new ClassLoader or SecurityManager
 - define classes that are part of packages on the client system



Three roads for Java byte code

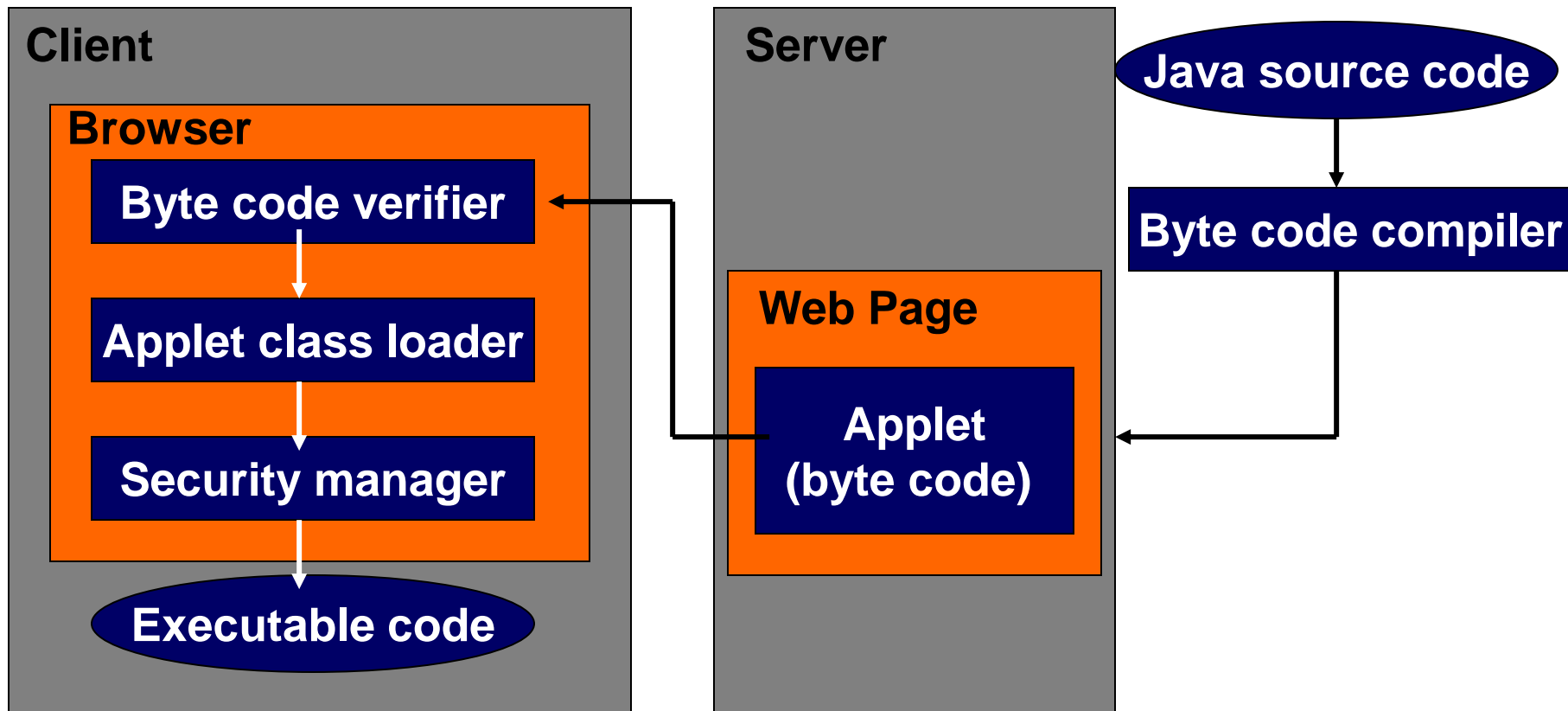


- **External byte code** loaded across the network must be verified
- Locally developed **byte code** is subject to the same checks unless it is part of the CLASSPATH
- **Byte code from the JDK distribution (and other classes in CLASSPATH)** does NOT pass through the verifier, **may** be checked by Security Manager



Java Sandbox

- Executable Content (applets) from remote web sites

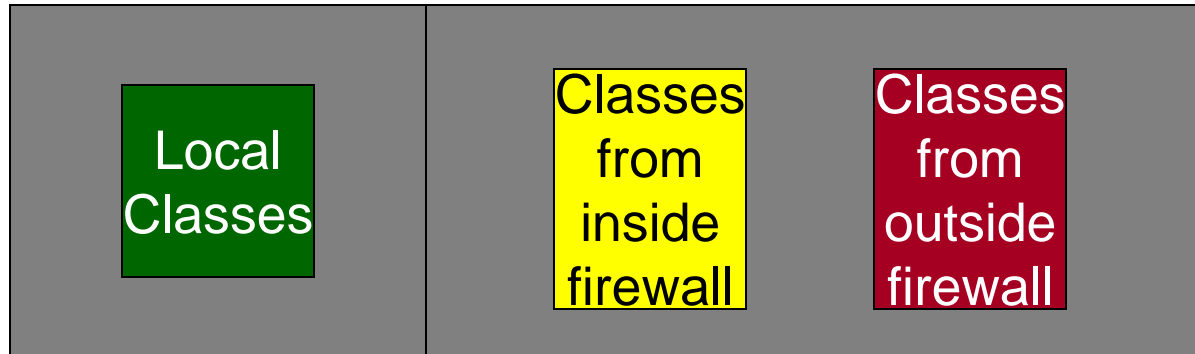


Byte Code Verifier

- **Checks for:**
 - the class file is in the proper format
 - stacks will not overflow
 - all operands have the correct type
 - there will be no data conversion between types
 - all references to other classes are legal
- **Byte code verifier reduces the workload on the interpreter**
 - guaranteed code properties do not have to be checked again
- *However, security still depends on the run-time environment*



Class Loader in a Dynamic Environment



- The Java environment has classes arriving and departing dynamically
- The class loader divides classes that it loads into several distinct name spaces according to where the classes came from
- Local classes are kept distinct from classes loaded from other machines
- Furthermore, these outside classes are protected from each other.



Applet Class Loader

- **Class loader protects the integrity of the run-time environment**
- **Applets must not be allowed to create their own class loaders**
 - Applets are handled by the applet class loader
- **Java comes with its own class library**
 - The CLASSPATH environment variable specifies the location of built-in classes
 - The security issues associated with altering CLASSPATH should be obvious
- **“Spoofing” of the CLASSPATH can be avoided by:**
 - If the applet class loader first searches the built-in classes in the local name space
 - Then expand search to the class making the request



Security Manager

- **Reference Monitor of the Java Security Model**
 - Performs run-time checks on ‘dangerous’ methods
- **Java classes are grouped into packages**
 - packages facilitate rudimentary access control to classes
- **Variables and methods can be declared as follows:**
 - **Private:** only the class creating the variable or method has access
 - **Protected:** only the class creating the variable or method and its subclasses have access
 - **Public:** all classes have access
 - **None of the above:** only classes within the same package have access



Security Manager Functions

- Prevent installation of new Class Loaders.
 - The Class Loader's job is to keep the name spaces properly organized.
 - Because things like file I/O permission will depend on whether or not a class is local, the Class Loader has an important job.
 - **Must not be subject to spoofing**
- Protecting threads and thread groups
 - Not fully functional....
- Controlling the creation of OS programs
- Controlling access to OS processes
- Controlling file system operations such as read & write
 - access to local files strictly controlled
- Controlling socket operations such as connect and accept
- Controlling access to Java packages (or groups of classes)



Some Compromises of the “Sandbox”

- **MSIE Cache Exploit**
- **Advanced Type Systems in Computing**
 - **Naval Postgraduate School**
- **Mark LaDue’s “Public Enemy”**
- **Others I chose not to experiment with:**
 - **diskhog.java**
 - **triplethreat.java**
 - **mutator.java**



Microsoft Internet Explorer - "Where do you want (your data) to go today?"

- The object of the exercise here is to open a connection to a port on the local machine, and provide a two-way pipe back to a remote machine on the Internet.
 - This is achieved by using the Java `net.socket` class to talk to the local machine, and the `showDocument()` thingy for the remote.
 - This exploit relies on the fact that Java behaves differently when loaded across the net, to a load from local hard disk.
 - When loaded across the net, the applet is not allowed to open a network socket to anything other than the server that delivered it in the first place
 - (see <http://www.javasoft.com/sfaq/#socket> for details).
 - This is enforced by the centralized security manager class. However, if the applet is loaded from local disk, this limitation is relaxed, allowing a socket to be opened on the browsing machine.



Type Systems for Secure Remote Evaluation

- The project aims to improve our understanding of the role of type systems in programming languages.
 - Type systems provide a very elegant separation of concerns.
 - Static analyses are typically much easier to reason about when captured by a logical framework such as a type system.
 - Implementations of the analyses are separate algorithmic issues that have their own soundness and completeness proof obligations.
 - This project is concerned with developing new type systems and techniques for formal proofs of semantic soundness, algorithmic issues, and computational lower bounds for these systems.
- This effort aims to identify the rudiments of a provably-secure programming language.
 - It requires formulation of appropriate security and safety properties so that one can prove with respect to a formal semantics that every well-typed program cannot violate these properties.
 - For example, it would be nice to prove that every well-typed Java Applet when executed by a browser does not cause the browser to crash. Clearly, there isn't such a proof as evidenced by enabling Java in them and [clicking here](#) to run a tiny (killerApp)let.



PublicEnemy.java by Mark LaDue

This Java application directly attacks Java class files. Given a target directory, it searches that directory and all subdirectories for Java class files. Once a class file is located, PublicEnemy alters the contents of its "access_flags" for the class, its fields, and its methods. The results are the following:

1. The class becomes public.
2. Any "static" or "volatile" fields remain as such; "final" fields become "non-final"; "transient" fields become "non-transient;" and "private" or "protected" fields become "public," while "public" fields remain so.
3. Any "abstract, "native," "synchronized," or "static" methods remain as such; "final" methods become "non-final;" and "private" or "protected" methods become "public," while "public" methods remain so.

This should open the class to the maximum amount of inspection and abuse without directly affecting its ability to run. Note that the size of the resulting class is the same as the original. The ability to modify Java class files on the fly is just the skill that a Java Platform Virus will require. The fact that it's this easy bodes ill....



Summation of Web Threats

	Threats	Consequences	Countermeasures
Integrity	<ul style="list-style-type: none"> •Modification of user data •Trojan horse browser •Modification of memory •Modification of message traffic in transit 	<ul style="list-style-type: none"> •Loss of information •Compromise of machine •Vulnerability to all other threats 	<ul style="list-style-type: none"> •Cryptographic checksums
Confidentiality	<ul style="list-style-type: none"> •Eavesdropping on the Net •Info theft from server •Info theft from client •Info about network configuration •Info about which clients talk to server 	<ul style="list-style-type: none"> •Loss of Information •Loss of Privacy 	<ul style="list-style-type: none"> •Encryption, •Web Proxy
Denial of Service	<ul style="list-style-type: none"> •Killing of user threads •Flooding machine with bogus threats •Filling up disk or memory •Isolating machines by DNS attack 	<ul style="list-style-type: none"> •Disruptive •Annoying •Prevent user from getting work done 	<ul style="list-style-type: none"> •Difficult to prevent
Authentication	<ul style="list-style-type: none"> •Impersonation of legitimate users •Data Forgery 	<ul style="list-style-type: none"> •Misrepresentation of user •Belief that false information is valid 	<ul style="list-style-type: none"> •Cryptographic techniques



Static versus Dynamic Type Checking

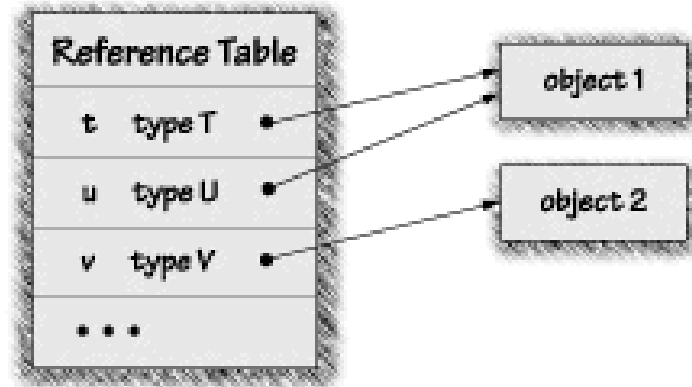
- **Java has no dynamic memory allocation**
 - does not allow you to cast object or array references into integers or vice-versa
 - does not allow “pointer arithmetic”
 - does not allow you to compute the size in bytes of any primitive type or object
- **Dynamic type checking is inefficient**
 - to improve performance, Java uses static type checking
 - faster, but less secure than say, Ada
- **In a type-confusion attack, a malicious applet creates two pointers to the same object-with incompatible type tags.**
 - When this happens, the Java system is in trouble.
 - The applet can write into that memory address through one pointer, and read it through another pointer.
 - The result is that the applet can bypass the typing rules of Java, completely undermining its security.



Type Confusion Attack Example

- The applet has two pointers to the same memory: one pointer tagged with type T and one tagged with type U. Suppose that T and U are defined like this:

```
class T {  
    SecurityManager x;  
}  
class U {  
    MyObject x;  
}
```

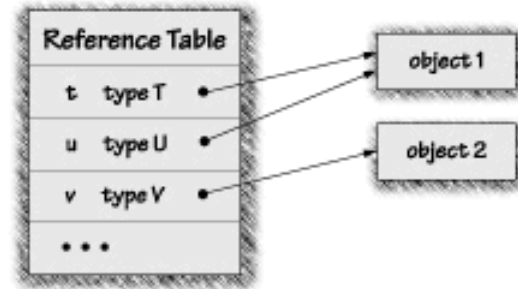


- Now the applet can run code like this:
T t = the pointer tagged T;
U u = the pointer tagged U;
t.x = System.getSecurity(); // the Security Manager
MyObject m = u.x;



Exploit Results

- The result is that the object ends up with a pointer, tagged as having type `MyObject`, to the memory representing Java's Security Manager object.
- By changing the fields of `m`, the applet can then change the Security Manager, even though the Security Manager's fields have been declared private.
- While this example showed how type confusion can be used to corrupt the Security Manager, the tactic may be exploited to corrupt virtually any part of the running Java system.




<http://www.securingjava.com/chapter-five/chapter-five-7.html>



Conclusion

- **Fundamental engineering tradeoff:**
 - **Functionality versus Security**
 - **Always an inverse relationship**
 - **Evident in many aspects of security**



You don't know her. But she might know you. The fact is, hackers are virtually everywhere. And if your network isn't protected by a BorderWare firewall, it's not as secure as it should be.

BorderWare is the first firewall designed to be as smart about time and money as it is about security. It's the only firewall that's up and running in hours, not days.

It offers total transparency to all authorized users, so existing software and procedures don't need to be modified, and no one needs retraining. It runs its own diagnostics.

What's more, it doesn't require expensive workstations. In fact, BorderWare operates on an Intel® processor.

As well, it has everything you need to link to the Internet: Mail, News, WWW, FTP and DNS. And it combines packet filtering with both application-level and circuit-level gateways.

BorderWare also enables you to define proxies for secure and specialized applications through the firewall. It responds to attacks and initiates alarms. And is configured to grow as you and your enterprise do.

It's an impressive list of features. But it has to be. Because the whole world, unfortunately, is watching.

<http://www.borderware.com>

BORDERWARE
Nobody Comes Close

All trademarks are the property of their respective owners.

Call us at 1-800-334-8195 or +1(416) 368-7157. Fax +1(416) 368-7789. 20 Toronto Street, Suite 400, Toronto, Ontario, Canada M5C 2B8.



Mobile systems vulnerabilities

Dr. Drew Hamilton

**Rushing Attack: Reference: Yih-Chun Hu,
Adrian Perrig & David B. Johnson**

Reference: ISC2 CBK



Connected Organizations

- **Several organizations may be connected through a computer network**
- **Policies for all organizations involved must be considered**
- **Special focus should be placed on boundaries between organizations**
- **Consider data/information flow and sensitivity level**



Importance of MOU/MOA

- **Memorandum of Understanding (MOU) or Memorandum of Agreement can be used to define requirements for both parties**
- **Important to define accreditation requirements to meet expectations of all parties involved**
- **MOU/MOA can be used to establish accreditation parameters in a joint accreditation**



Development & Execution of MOA/MOU

- **Coordination**
- **Access**
- **Control**
- **Enclave Definition**



Connectivity Involved in Communications

- **Communications require connectivity of some sort**
- **System to System Connectivity**
 - **Hardware**
 - **Software**
- **Personal Connectivity**
 - **Offline communications**
 - **Telephone**
- **Important to consider all forms of connectivity involved**



Threats from Emissions Security (EMSEC)

- **System emissions can release sensitive information if not handled correctly**
- **Intercepted data may be analyzed to determine information in a system or a transmission line**



Threats from TEMPEST Failures

- **Information may be compromised**
- **Systems may malfunction**
- **Interconnects may malfunction**
- **Possible damage to other systems due to emissions**
- **TEMPEST Emanations may come from:**
 - **Cables**
 - **Power Lines**
 - **Displays**
 - **Sound**



Tempest Requirements

- **Shielding**
- **TEMPEST zoning**
- **Zone of control/zoning**
- **Grounding**
- **Filtered Power**
- **Banding**
- **Cabling**



Threats from Electronic Emanations

- **Susceptible to Active and Passive Attacks**
- **Emanations may be intercepted and interpreted without the knowledge of the system being compromised**
- **Attackers could also use emanations to interrupt the correct operation of the system**
 - **Bugs**
 - **TEMPEST Viruses**
 - **Telephone snooping**
 - **Glitching**



Mobile Threats

- With the popularity of mobile devices like smartphones and tablets, many additional security threats have emerged
- Individuals lose their mobile devices and don't have capabilities to remotely wipe data from the device
- Individuals keep sensitive data on mobile devices and do not use passcodes
- Individuals "jailbreaking" their mobile phones
- Individuals use poorly designed mobile applications that can have security vulnerabilities
- Individuals use insecure wireless networks, leaving their devices vulnerable to different types of attacks



McAfee iOS Security Tips

- **Tip 1: Enable Passcode Lock on Your iPhone**
- **Tip 2: Disable Features that Could Be Accessed Without Entering the Passcode**
- **Tip 3: Overcoming Privacy Issues Due to the Inherent Design of the iPhone**
- **Tip 4: Erase All the Data Before Return, Repair, or Resale of Your iPhone**
- **Tip 5: Regularly Update Your iPhone's Firmware**
- **Tip 6: To Jailbreak or Not to Jailbreak**
- **Tip 7: Enable Safari's Privacy and Security Settings on the iPhone**
- **Tip 8: Using Bluetooth, Wi-Fi, and Email Securely**
- **Tip 9: Enable Restrictions**
- **Tip 10: Enable Find My iPhone**

Infoworld: Android Fragmentation

- **The Android platform suffers from fragmentation**
 - **Multiple versions of Android in the market, even on current devices.**
 - **Manufacturers often make their own changes to Android, so they could be behind Google's current reference release.**
 - **In addition, carriers and manufacturers may not update their devices' Android version when Google does, or they take months or even years to do so.**
- **As a result, many people within the same organization might be using outdated versions that could be riddled with security vulnerabilities.**
 - **"People focus on malware risks of Android, but arguably the greater risk is that fragmentation creates different user experiences," says Ojas Rege, vice president of strategy at MobileIron, a provider of enterprise mobility management products. "This variety of user experiences makes it hard to educate your employees about how to take security measures, because the experience on each device is different."**



BGP Misconfiguration

- It is well-known that simple, accidental BGP configuration errors can disrupt Internet connectivity.
- Yet little is known about the frequency of misconfiguration or its causes, except for the few spectacular incidents of widespread outages.
 - In this paper, we present the first quantitative study of BGP misconfiguration.
 - Over a three week period, we analyzed routing table advertisements from 23 vantage points across the Internet backbone to detect incidents of misconfiguration.
 - For each incident we polled the ISP operators involved to verify whether it was a misconfiguration, and to learn the cause of the incident.



Misconfiguring BGP

- We also actively probed the Internet to determine the impact of misconfiguration on connectivity.
- Surprisingly, we find that configuration errors are pervasive, with 200-1200 prefixes (0.2-1.0\% of the BGP table size) suffering from misconfiguration each day.
- Close to 3 in 4 of all new prefix advertisements were results of misconfiguration.
- Fortunately, the connectivity seen by end users is surprisingly robust to misconfigurations.
- While misconfigurations can substantially increase router processing overhead, only one in twenty five affects connectivity.
- While the causes of misconfiguration are diverse, we argue that most could be prevented through better router design.



What is BGP? RFC 1771, 1772

- In the Internet environment. BGP is an inter-Autonomous System routing protocol. The network reachability information exchanged via BGP provides sufficient information to detect routing loops and enforce routing decisions based on performance preference and policy constraints as outlined in RFC 1104
- In particular, BGP exchanges routing information containing full AS paths and enforces routing policies based on configuration information. Factors driving this RFC:
 1. Exhaustion of the class-B network address space. One fundamental cause of this problem is the lack of a network class of a size which is appropriate for mid-sized organization; class-C, with a maximum of 254 host addresses, is too small while class-B, which allows up to 65534 addresses, is too large to be densely populated.
 2. Growth of routing tables in Internet routers are beyond the ability of current software (and people) to effectively manage.
 3. Eventual exhaustion of the 32-bit IP address space.



Classless inter-domain routing (CIDR)

- **Classless inter-domain routing (CIDR) attempts to deal with these problems by proposing a mechanism to slow the growth of the routing table and the need for allocating new IP network numbers.**
- **It does not attempt to solve the third problem, which is of a more long-term nature, but instead endeavors to ease enough of the short to mid-term difficulties to allow the Internet to continue to function efficiently while progress is made on a longer-term solution.**
- **BGP-4 is an extension of BGP-3 that provides support for routing information aggregation and reduction based on the Classless inter-domain routing architecture (CIDR)**



CIDR Architecture RFC 1519

- **The proposed solution is to topologically allocate future IP address assignment, by allocating segments of the IP address space to the transit routing domains.**
- **There are two basic components of this addressing and routing plan: one, to distribute the allocation of Internet address space and two, to provide a mechanism for the aggregation of routing information.**



Aggregation Routing

- **Aggregation and its limitations**
 - One major goal of the CIDR addressing plan is to allocate Internet address space in such a manner as to allow aggregation of routing information along topological lines.
 - For simple, single-homed clients, the allocation of their address space out of a transit routing domain's space will accomplish this automatically -rather than advertise a separate route for each such client, the transit domain may advertise a single aggregate route which describes all of the destinations connected to it.
 - Unfortunately, not all sites are singly-connected to the network, so some loss of ability to aggregate is realized for the non-trivial cases.



Autonomous Systems

- All of the discussions in RFC 1772 are based on the assumption that the Internet is a collection of arbitrarily connected Autonomous Systems.
- That is, the Internet will be modeled as a general graph whose nodes are AS's and whose edges are connections between pairs of AS's.
- The classic definition of an Autonomous System is a set of routers under a single technical administration, using an interior gateway protocol and common metrics to route packets within the AS and using an exterior gateway protocol to route packets to other AS's.
- Since this classic definition was developed, it has become common for a single AS to use several interior gateway protocols and sometimes several sets of metrics within an AS.
- The use of the term Autonomous System here stresses the fact that, even when multiple IGP's and metrics are used, the administration of an AS appears to other AS's to have a single coherent interior routing plan and presents a consistent picture of which destinations are reachable through it.
- AS's are assumed to be administered by a single administrative entity, at least for the purposes of representation of routing information to systems outside of the AS



BGP Topological Model

- When we say that a connection exists between two AS's (autonomous systems), we mean two things:
- **Physical connection:** There is a shared Data Link subnet at work between the two AS's, and on this shared subnetwork each AS has at least one border gateway belonging to that AS. Thus the border gateway of each AS can forward packets to the border gateway of the other AS without resorting to Inter-AS or Intra-AS routing.
- **BGP connection:** There is a BGP session between BGP speakers in each of the AS's, and this session communicates those routes that can be used for specific destinations via the advertising AS.



On-Demand Protocols

- There are two categories of routing protocols: table-driven and on demand-routing.
- In **table-driven routing protocols** routing information is periodically advertised to all nodes so all nodes have an up-to-date view of the network.
- Alternatively, **on-demand routing protocols** only discovers a new route when it is required to.
- Hybrid routing protocols also exist and they try to achieve an efficient balance between both categories of protocols



Comparison between Table-Drive Routing and On-Demand Routing

	Table-driven Routing	On-Demand Routing
Availability of Routing Information	Immediately from route table	After a route discovery
Route Updates	Periodic Advertisements	When requested
Routing Overhead	Proportional to the size of the network regardless of network traffic	Proportional to the number of communicating nodes and increases with increased node mobility

It is clear that on-demand protocols are more suited for mobile handheld devices as network bandwidth and battery power is limited.



Ad hoc On-demand Distance Vector Routing (AODV)

- Ad hoc On-demand Distance Vector Routing (AODV) is an on-demand version of the table-driven Dynamic Destination-Sequenced Distance-Vector (DSDV) protocol
- To find a route to the destination, the source broadcasts a route request packet.
- This broadcast message propagates through the network until it reaches an intermediate node that has recent route information about the destination or until it reaches the destination.
- When intermediate nodes forwards the route request packet it records in its own tables which node the route request came from.
- This information is used to form the reply path for the route reply packet as AODV uses only symmetric links.
- As the route reply packet traverses back to the source, the nodes along the reverse path enter the routing information into their tables.
- When ever a link failure occurs, the source is notified and a route discovery can be requested again if needed.



Dynamic Source Routing

- The Dynamic Source Routing (DSR) protocol is a source-routed on-demand protocol.
- There are two major phases for the protocol: route discovery and route maintenance.
- The key difference between DSR and other protocols is the routing information is contained in the packet header.
- Since the routing information is contained in the packet header then the intermediate nodes do not need to maintain routing information.
- An intermediate node may wish to record the routing information in its tables to improve performance but it is not mandatory.
- Another feature of DSR is that it supports asymmetric links as a route reply can be piggybacked onto a new route request packet.
- DSR is suited for small to medium sized networks as its overhead can scale all the way down to zero.
- The overhead will increase significantly for networks with larger hop diameters as more routing information will be contained in the packet headers



Rushing Attack

- Disseminates ROUTEREQUESTs quickly throughout the network, suppressing any later legitimate ROUTEREQUESTs when nodes drop them due to the duplicate suppression.
- Practical only on on-demand routing protocols using duplicate suppression at each node (AODV)
- Threats: Failure of route discovery



Rushing Attack Example

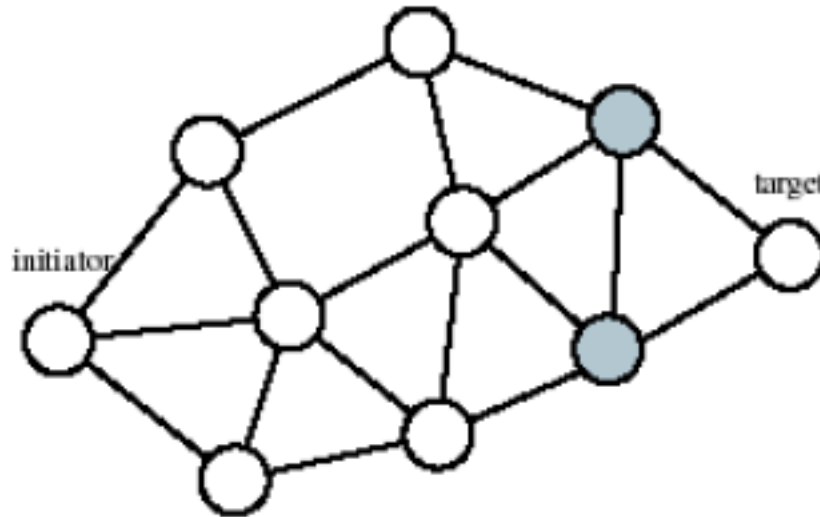


Figure 1: Example network illustrating the rushing attack.

- The initiator node initiates a Route Discovery for the target node.
- If the Route Requests for this Discovery forwarded by the attacker are the first to reach each neighbor of the target, then any route discovered by this Route Discovery will include a hop through the attacker.

Secure Route Delegation

- **S-BGP uses Route Attestations to ensure that each AS listed in the BGP AS path is indeed a valid AS**
- **In S-BGP, before sending a route update to its neighbor, the AS signs a route attestation delegating it the right to further propagate the update.**
- **Used to make sure that BOTH neighbors believe that they are within transmission range.**



Significance of Wireless Security

- **Wireless transmissions are vulnerable in much the same manner as electronic emissions**
- **Any wireless transmissions must be carefully controlled**
 - **Signal Strength**
 - **Encryption**
 - **Limited Access Points**
- **Any usage of wireless transmissions must be justified with strong system requirements do to risk involved**



Risks with Handheld Devices

- **Hard to control configuration because of several separate personal users**
- **Must monitor connections to networks**
- **Electronic or wireless emanations risked to attack**
- **Prone to physical compromise**
 - **Generally extremely portable**
 - **Much easier to misplace or have stolen than traditional device**



Wireless Vulnerabilities

- **Active Attacks**
 - **Signal Jamming**
 - **Virus**
 - **Network Infiltration**
 - **Impersonation**
- **Passive Attacks**
 - **Snooping**
 - **Packet Recording**



Summary

- **Communications and boundaries between systems must be carefully considered**
- **Electronic emanations and wireless transmissions can cause system compromise if not controlled**
- **Handheld devices require special security measures due to enhanced risk**



Embedded devices and cyber-physical systems vulnerabilities

Reference: Karen Mercedes Goertzel,
CISSP, BAH



Safety Critical Systems

- **Goal of cyber security:**
- **To prevent the cyber system from becoming the means by which the information created, processed, transmitted, or stored by that system can be inappropriately disclosed, modified, or destroyed**
 - **This goal is achieved by: Protecting systems and information against realized**
 - **Detecting indications of realized threats**
 - **Reacting to verified threats**
 - **Recovering from successfully realized threats**



Safety Critical Systems

- **Main concern: hazards**
 - Realized as mishaps (causing software or hardware faults and failures)
 - Unintentional causation: human error, accident, or “act of god”
 - Partly or wholly non-deterministic (stochastic): occurrence may be unpredictable, but outcome is generally predictable
 - Straightforward: single, localized event leads to simple fault or failure
- **Potential consequences if realized:**
 - Physical injury or death
 - Destruction of the means of survival (food, water, shelter, medicine)
 - Environmental damage, with potential to damage health
 - Destruction of means of financial livelihood or support
 - Very difficult to characterize in terms of actual (vs. arbitrarily-defined/actuarial) financial loss



Cyber-Physical Systems

- **“Touch” the physical world in some way**
- **Main functions: Monitoring, diagnosis, and control of physical processes**
- **Subject to safety hazards (because they’re physical) and security threats (because they’re cyber) Need to avoid hazards and tolerate faults drives the imperative for safety**
- **Need to protect against threats and tolerate intrusions drives the imperative for security**
- **Can be embedded, non-embedded, or a hybrid of the two**



Embedded Cyber Physical Systems

- **Vehicle controllers/computers (accessible via vehicle telematics, e.g., GM OnStar)**
- **Network-connected (incl. RFID networks) medical devices**
 - Embedded components of industrial and infrastructure control systems process controllers*
 - intelligent electronic devices (IEDs)
 - remote telemetry units (RTUs)
 - Sensors
- **Remote-controlled robots**
- **Weapon systems**
- **Unmanned aerial vehicles**
- **Maritime and avionic navigation devices**



Non-Embedded Cyber Physical Systems

- Air traffic control systems
- Avionic and space flight control systems
- Non-embedded components of industrial and infrastructure control systems
 - Supervisory Control And Data Acquisition (SCADA) systems
 - Distributed control systems (DCS)
- SmartGrid
- Railway signalling and track switching systems



“Software Errors” in CPS

- **Medical devices (acc. to FDA, 33% of all recalls due to software failures)**
- **Weapon systems**
- **Military planes and helicopters**
- **Telephone switches**
- **Space vehicles**
- **Soviet early warning system**
- **Automobile recalls**
- **Environmental monitoring systems**
- **Prisoner monitoring system**
- **Industrial Control Systems**
 - (buffer overflows, heap overflows, memory corruption vulnerabilities in multiple SCADA products in U.S., Europe, and China)



Critical Infrastructure at Risk

- Industrial control systems combine data processing, command and control, and physical process control.
- Use open networking protocols, remote access, even Internet connectivity
- Combine old legacy software never meant for networked use with new components on mobile devices using wireless for remote access/control
- Numerous vulnerabilities at system and software levels exposed to insider and outsider threats
 - Outcomes of threats and hazards are so similar, the root cause may be impossible to determine *Example: Systematic sabotage of Maroochy Shire (Australia) waste water treatment plant by Vitek Boden, ex-IT consultant, using stolen SCADA software, laptop, and WiFi. Most of Boden's 46 separate "break-ins" over 4 months were mistaken for stochastic failures*
 - Outcome was the same: Millions of litres of raw sewage released into surrounding rivers, resort grounds, and nature reserves



Security of safety-critical cyber-physical systems

- **Safety *always* trumps security** Safety remains the #1 imperative in safety-critical cyber-physical systems
- **Security is only the means to the end of guaranteeing safety**
- **Security that does not guarantee safety has no value**
- **Security must NEVER impede safety**
- **Security threats represent a new, unprecedented type of “intentional hazard”** Safety engineering takes physical into account, but not cyber
- **Safety requirements are based on models of hazards, faults, and errors, not threats and attacks**
- **Events typical of cyber attacks are not captured in most hazard models**
- **No distinction between unintentional and intentional causality**



CPS Exposure

- **Growing dependence on software alone**
 - No manual/mechanical backup in case of failure
- **Increasing use of commodity/open source components**
 - Larger, more complex, harder to analyze, more dormant code
 - Unknown pedigree/provenance (supply chain issues)
 - Prone to more flaws, including those exploitable as vulnerabilities
 - Many components used are general-purpose, e.g., Windows, never engineered for safety-critical and cyber-physical stresses
- **Increased connectivity via exposed networks**
 - Leveraging the Internet
 - Leveraging wireless (cell, satellite, RFID)
 - Limited processor/memory in embedded systems



Security of Safety-Critical Systems

- **Functional (including usage) and non-functional (interface, data, constraint, and quality) requirements**
 - **Need to capture requirements and test cases based on use, misuse, and abuse cases, threat/attack models and hazard/mishap models**
- **Deployed executable**
 - **Need to assure deployed executable's integrity**
 - **Need to assure executing software's availability**
- **Data created, stored, transmitted, or used by the software**
 - **Need to assure confidentiality, integrity, and availability of inputs and outputs**
- **Software's interfaces and execution environment**
 - **Need to assure availability, integrity, and accessibility of environment (hardware, software, network) resources and services relied on by the software**
 - **Need to assure access to the deployed software by authorized users and only authorized users (human and other software)**



Safety versus Security

1. When in mode V , the system shall not cause accidental harm of type W to valuable assets of type X at an average rate $>$ asset value Y per Z time duration.
2. When in mode W , the system shall not cause mishaps of type X with an average rate of more than Y mishaps per Z trips.
3. When in mode X , the system shall not cause hazard Y to exist more than an average of Z percent of the time.
4. Upon detecting an accident of type W when in mode X , the system shall react by performing function Y an average of at least Z percent of the time.

1. When in mode V , the system shall limit the occurrence of malicious harm of type W to valuable assets of type X at an average rate $>$ asset value Y per Z time duration.
2. When in mode W , the system shall prevent the first successful attacks of type X for a minimum of Z time duration.
3. When in mode X , the system shall not have security vulnerability Y for more than an average of Z percent of the time.
4. Upon detecting a misuse of type W when in mode X , the system shall react by performing function Y an average of at least Z percent of the time.



Detect-Protect-React-Recover

- **Detect log (events)**
 - audit (usage)
 - sense (anomalies, intrusions)
 - monitor (execution, interactions [inputs/outputs])
- **Protect through defense in depth and defense in breadth**
- **React, in hopes of minimizing extent, intensity, duration of impact**
 - minimizing likelihood of recurrence by blocking certain types of inputs
 - terminating user sessions
 - terminating some or all functionality
 - disconnecting some or all network connections
 - assessing damage, attempting recovery to pre-incident state



Surviving Cyber Attacks on CPS

- **Safety-critical cyber-physical systems have no tolerance for delays associated with post-incident recovery**
- **Need ability to tolerate attacks and survive threats in the same way they tolerate faults and survive hazards**
- **Need ability to survive even persistent, high-intensity intrusions and attacks**
- **System level *Redundancy*: hot sparing, decoupling and replication high-consequence components**
- ***Rapid recovery*: automatic backups, automatic swap-over of high-consequence components**



Survivable CPS Software

- **Software level Diversity, redundancy, fault-tolerance**
 - Exception-handling logic that is purpose-written, not generic
 - typically exceeds quantity of associated program logic
 - minimizes ability of faults to aggregate/escalate into failures
 - enables continued operation of critical functions rather than failure graceful degradation: lower priority functions terminate while critical functions keep running
 - Prevents software from simply crashing, entering non-secure state, or dumping core(including cache, temp files)



Cryptography

Dr. Drew Hamilton



Mississippi State University Center for Cyber Innovation

Domain 3 Security Engineering



218

Cryptographic Shortfalls - Enigma

- The machine has several variable settings that affect the operation of the machine. The user must select three rotors from a set of rotors to be used in the machine. A rotor contains one-to-one mappings of all the letters. Some Enigma machines had more than 3 rotors which just added to the number of possible encryption combinations. The other variable element in the machine is the plug board. The plug board allowed for pairs of letters to be remapped before the encryption process started and after it ended.



<http://www.ugrad.cs.jhu.edu/~russell/classes/enigma/>



Mississippi State University Center for Cyber Innovation

Domain 3 Security Engineering



219

More on Enigma

- When a key is pressed, an electrical current is sent through the machine. The current first passes through the plug board, then through the three rotors, through the reflector which reverses the current, back through the three rotors, back through the plug board and then the encrypted letter is lit on the display. After the display is lit up, the rotors rotate. The rotors rotate similar to an odometer where the right most rotor must complete one revolution before the middle rotor rotated one position and so on.
- As the current passes through each component in the Enigma machine, the letter gets remapped to another letter. The plug board performed the first remapping. If there is a connection between two letters, the letters are remapped to each other. For example if there is a connection between "A" and "F", "A" would get remapped to "F" and "F" would get remapped to "A". If this isn't a connection for a particular letter, the letter doesn't get remapped. After the plug board, the letters are remapped through the rotors. Each rotor contains one-to-one mappings of letters but since the rotors rotate on each key press, the mappings of the rotors change on every key press. Once the current passes through the rotors, it goes into the reflector. The reflector is very similar to a rotor except that it doesn't rotate so the one-to-one mappings are always the same. The whole encryption process for a single letter contains a minimum of 7 remappings (the current passes through the rotors twice) and a maximum of 9 remappings (if the letter has a connection in the plug board).



- In order to decrypt a message, the receiver must have the encrypted message, and know which rotors were used, the connections on the plug board and the initial settings of the rotors. To decrypt a message, the receiver would set up the machine identically to the way the sender initially had it and would type in the encrypted message. The output of typing in the encrypted message would be the original message. Without the knowledge of the state of the machine when the original message was typed in, it is extremely difficult to decode a message.



The Venona Intercepts

- The U.S. Army's Signal Intelligence Service, the precursor to the National Security Agency, began a secret program in February 1943 later codenamed VENONA. The mission of this small program was to examine and exploit Soviet diplomatic communications but after the program began, the message traffic included espionage efforts as well.
- Although it took almost two years before American cryptologists were able to break the KGB encryption, the information gained through these transactions provided U.S. leadership insight into Soviet intentions and treasonous activities of government employees until the program was canceled in 1980.
- The VENONA files are most famous for exposing Julius (code named LIBERAL) and Ethel Rosenberg and help give indisputable evidence of their involvement with the Soviet spy ring.
- The first of six public releases of translated VENONA messages was made in July 1995 and included 49 messages about the Soviet's efforts to gain information on the U.S. atomic bomb research and the Manhattan Project. Over the course of five more releases, all of the approximately 3,000 VENONA translations were made public.

Cover-name	Message number	Date	Publication reference
			3/ or 3/HBF/
AKIM (Cont.)	N.Y. to M.	059 08072	MB 42-21 T900 ✓ 2A-0096 2843 ^{6/74}
		899 11063	JKM 11 T1424 ✓
		969 22063	JKQ 05 T1385 ✓
		1047 02073	JKT 50 T406 ✓
		579 28044	JIG 02 T507 ✓
		770 30054	JIF 38 T38 ✓
		854 16064	JIJ 47 T58 ✓
		863 16064	JHP 18 T328 ✓
		864 16064	JHE 47 T1168 ✓
		928 01074	JHE 47 T62 ✓
		1071 28074	JHK 15 T529 ✓
		1090 30074	JHM 07 (NSA)
		1133 07084	JHM 19 T50.1 ✓
		1140 09084	JHM 17 T370 ✓
		1200 23084	JHB 17 T50.2 ✓
		1207 23084	JHB 26 T80 ✓
		1208 23084	JHB 27 T167 ✓
		1504 21104	JHQ 46 T433 ✓
		1559 06114	JHY 22 T421 ✓
		1622 20114	JIB 38 T422.1 ✓
	1638 21114	JHS 05 T423 ✓	
	1661 28114	JHS 16 T422.2 ✓	
	1730 08124	JHZ 28 T168.2 ✓	
	1828 28124	JHU 38	
	2101 24123	J30 42	
AL	N.Y. to M.	596 01054	JIG 16 T417 ✓ 2A-0114 0013A
ALAN RALPH BOWEN	M. to N.Y.	227 13035	JKN 38 T737 ✓ 2A-0114 0147 0094A 0115 4077
ALEKSANDR	N.Y. to M.	899 11063	JKM 12 T1414 ✓ 2A-0118 0095A
		968 22063	JKQ 02 T788 ✓ 1B-9050
		148 29014	(NSA)
		620 08054	JHD 39 T466 ✓
		720 16064	JTB 06 T121 ✓



<http://www.nsa.gov/venona>

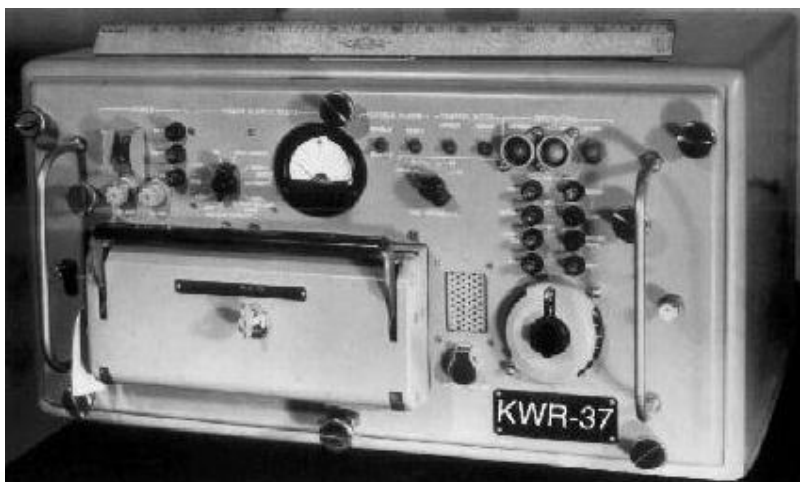


USS Pueblo - Crypto Compromise



John Walker & the USS Pueblo

- The KW-37, code named JASON, was an encryption system developed in the 1950s by the U.S. National Security Agency to protect fleet broadcasts of the U.S. Navy.
- KW-37s fell into North Korean hands when the USS Pueblo was captured in 1968. New keying material was issued to ships throughout the world to limit the ongoing damage. In 1985 it was revealed that the Walker spy ring had been selling key lists and cards to the Soviet Union for decades. KW-37 systems were taken out of service by the early 1990s.
- John Walker's last "drop" to the KGB pictured below. The KGB attempted to pay Walker \$200,000.00 for that drop.



Authentication

- **Requirements - must be able to verify that:**
 - 1. Message came from apparent source or author,**
 - 2. Contents have not been altered,**
 - 3. Sometimes, it was sent at a certain time or sequence.**
- **Protection against active attack (falsification of data and transactions)**



Approaches to Message Authentication

- **Authentication Using Conventional Encryption**
 - Only the sender and receiver should share a key
- **Message Authentication without Message Encryption**
 - An authentication tag is generated and appended to each message
- **Message Authentication Code**
 - Calculate the MAC as a function of the message and the key. $MAC = F(K, M)$



Cryptographic Hash

- Producing *hash values* for accessing data or for security.
- A hash value (or simply *hash*) is a number generated from a string of text.
 - The hash is substantially smaller than the text itself, and is generated by a formula in such a way that it is extremely unlikely that some other text will produce the same hash value.
- Hashes play a role in security systems where they're used to ensure that transmitted messages have not been tampered with.
 - The sender generates a hash of the message, encrypts it, and sends it with the message itself.
 - The recipient then decrypts both the message and the hash, produces another hash from the received message, and compares the two hashes.
 - If they're the same, there is a very high probability that the message was transmitted intact.

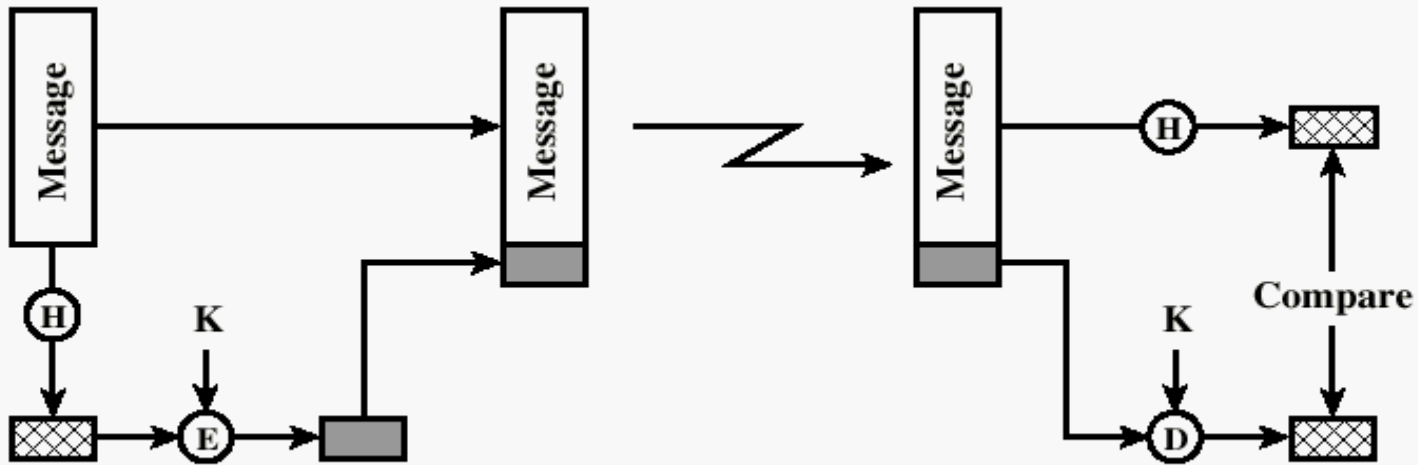


Trivial Hashing Example

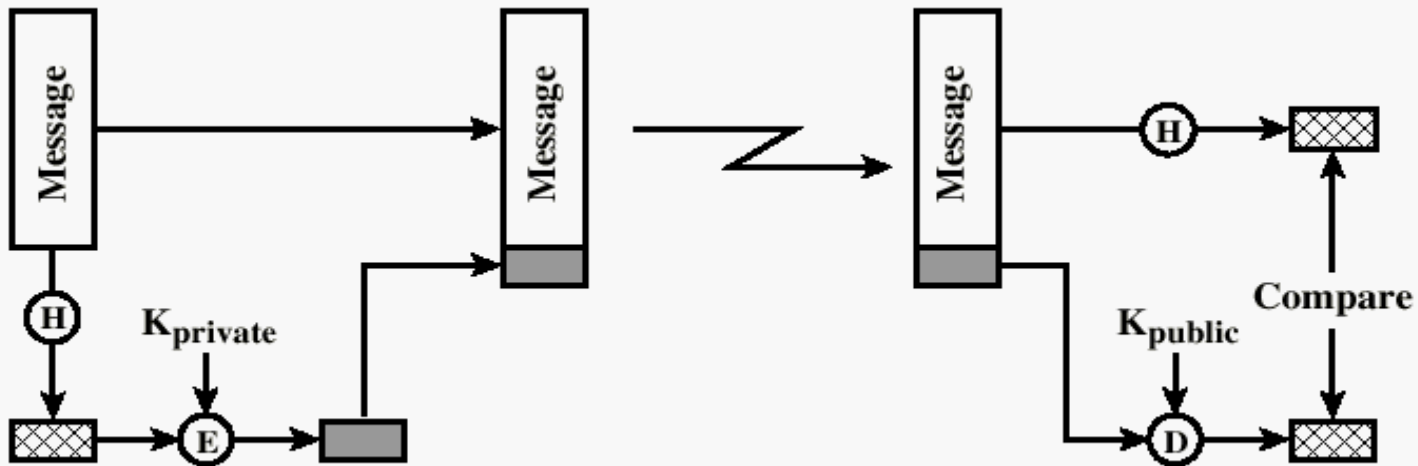
- Hashing is also a common method of accessing data records. Consider, for example, a list of names:
 - John Smith
 - Sarah Jones
 - Roger Adams
- To create an index, called a *hash table*, for these records, you would apply a formula to each name to produce a unique numeric value. So you might get something like:
 - 1345873 John Smith
 - 3097905 Sarah Jones
 - 4060964 Roger Adams
- Then to search for the record containing *Sarah Jones*, you just need to reapply the formula, which directly yields the index key to the record.
- This is much more efficient than searching through all the records till the matching record is found.



One-way HASH function



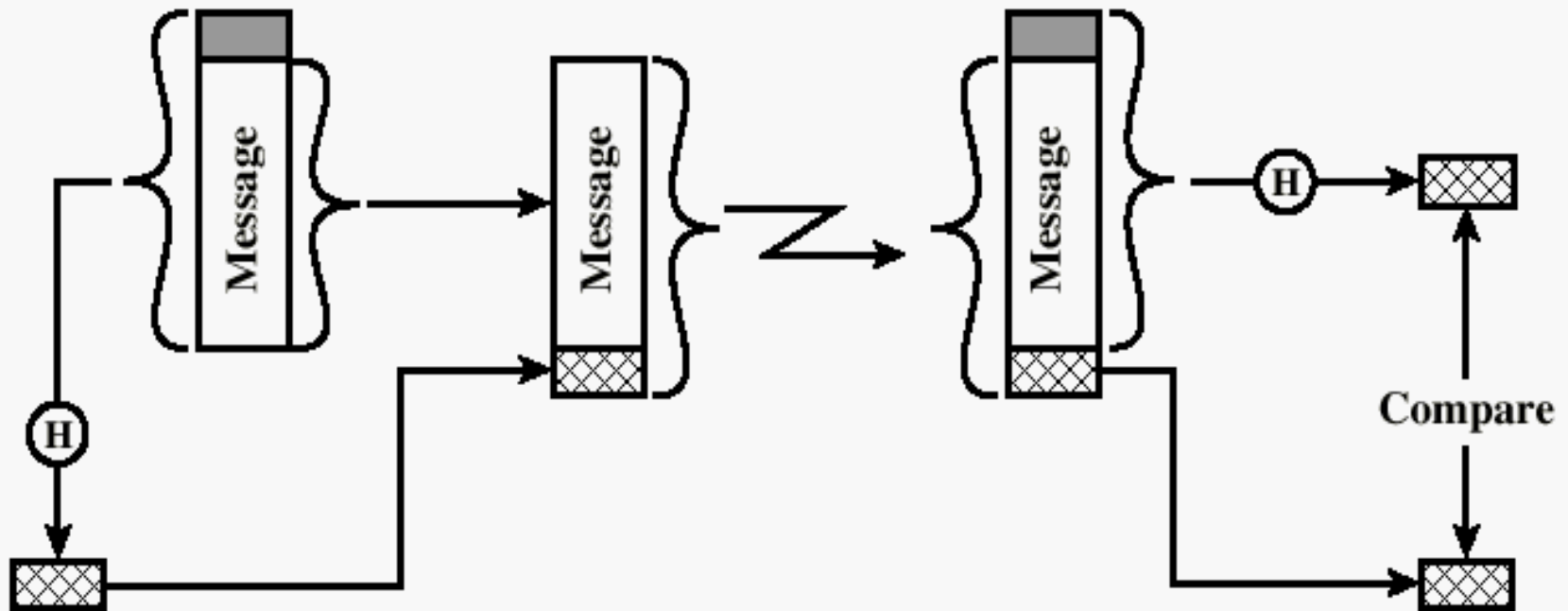
(a) Using conventional encryption



(b) Using public-key encryption

One-way HASH function

- Secret value is added before the hash and removed before transmission.



(c) Using secret value

Secure HASH Functions

- Purpose of the HASH function is to produce a “fingerprint.”
- Properties of a HASH function H :
 1. H can be applied to a block of data at any size
 2. H produces a fixed length output
 3. $H(x)$ is easy to compute for any given x .
 4. For any given block x , it is computationally infeasible to find x such that $H(x) = h$
 5. For any given block x , it is computationally infeasible to find y with $H(y) = H(x)$.
 6. It is computationally infeasible to find any pair (x, y) such that $H(x) = H(y)$



Hash Attacks

- **Dictionary Attacks**
 - Based on known lists of common words
- **Birthday attacks** – group of 23 people, 50% chance 2 will have same birthday. 60 people, 99% chance. Relevant because it describes the amount of effort that must be made to determine when 2 randomly-chosen values will be the same (collisions). Weak hash causes many collisions
 - Attack the hash value
 - Attack the initialization vector
- **Rainbow table attacks**
 - Hash reductions
 - Salts



Public-Key Cryptography Principles

- **The use of two keys has consequences in: key distribution, confidentiality and authentication.**
- **The scheme has six ingredients**
 - **Plaintext**
 - **Encryption algorithm**
 - **Public**
 - **Private key**
 - **Ciphertext**
 - **Decryption algorithm**

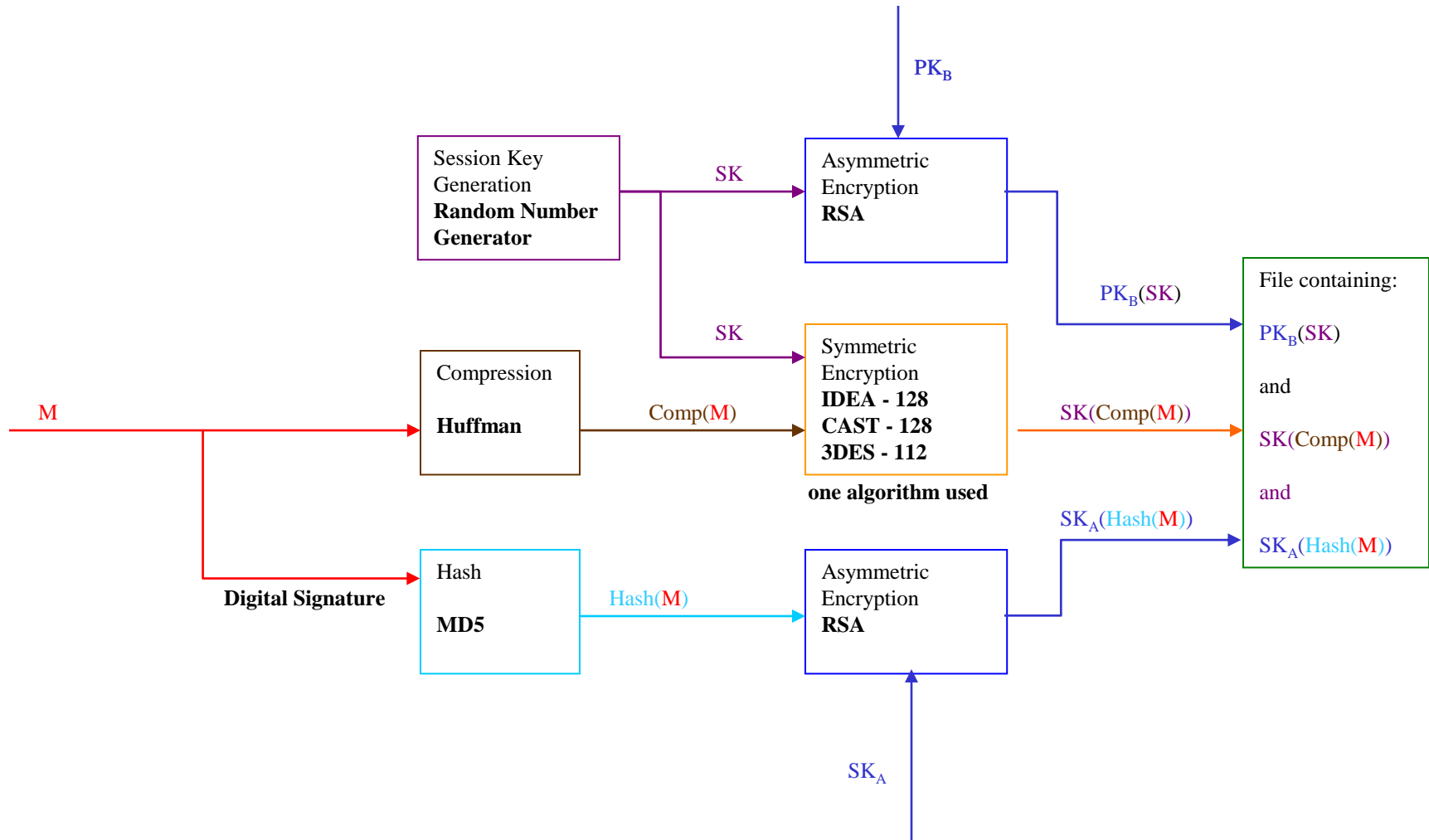


Assumptions

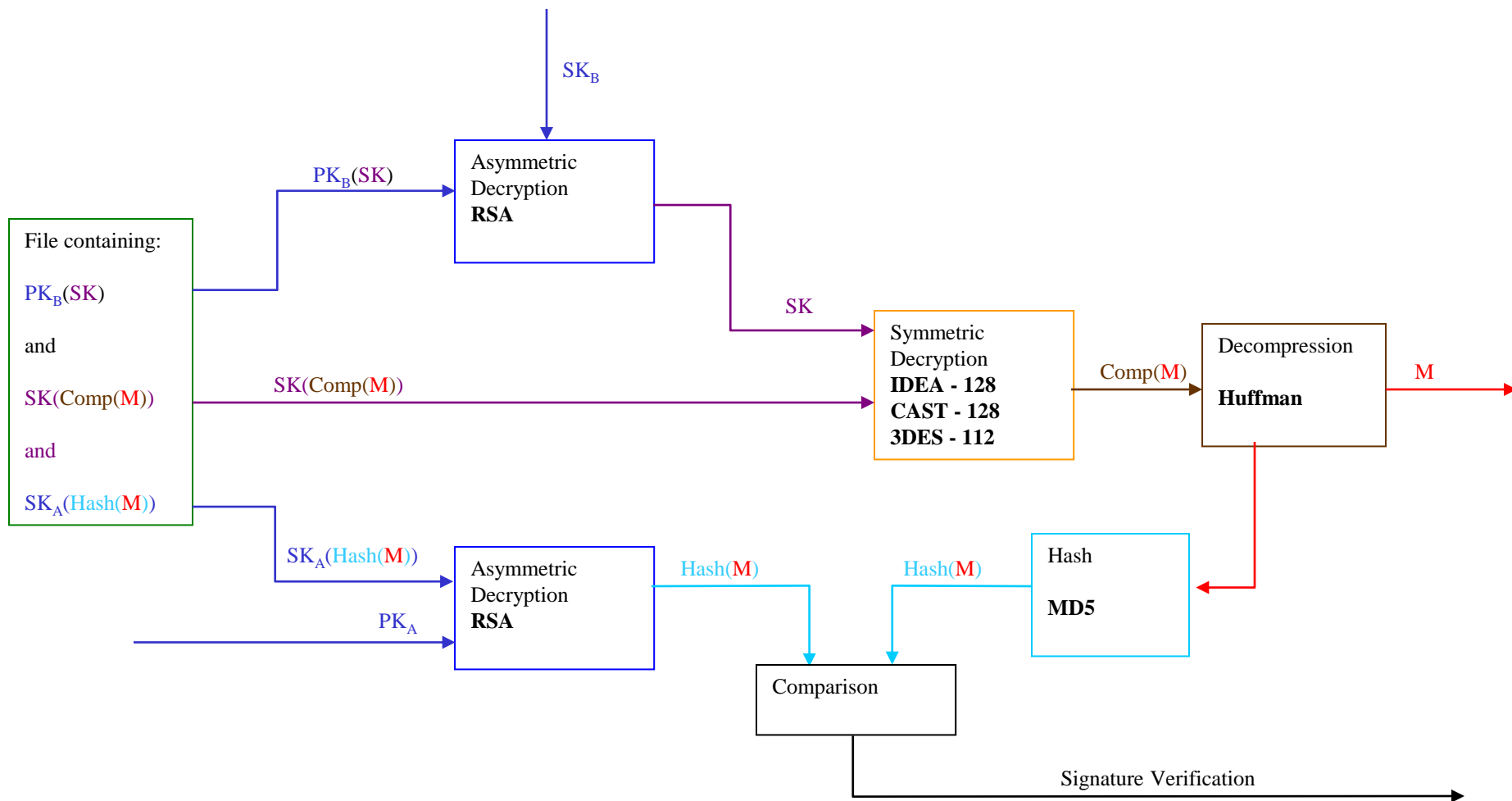
- Requirement exists for secure email to transmit sensitive but unclassified information for joint and combined operations in the Pacific Theater.
- Commercial Internet preferred method of transferring messages.
- No special encryption hardware should be required.
- Cost is a factor.
- Source code required to evaluate encryption evaluation.
- PGP
 - Freeware available from MIT and Network Associates.
 - Source code publicly available.
 - PGP versions have been publicly available and tested since 1991.
 - However, as with most reported results, insufficient information is provided to replicate the tests.



PGP Sender



PGP Receiver



Endpoint Concerns

- **Physical Security**
- **Tempest**
- **Compromise (pass phrase, Session Key_a, secret keys)**
- **Public Key Tampering**
- **Viruses or Trojan Horses (PGP)**
- **Social Engineering**
- **Audio/Video Surveillance**
- **Key Management**



Encryption Attacks

- **Symmetric Encryption**
 - **Brute force**
 - **Flaw in implementation**
- **Asymmetric Encryption**
 - **Brute force**
 - **Factoring (RSA)**
 - **Flaw in implementation**



PGP Attack Methodology

- **Brute Force Key space Search:** Compromise sensitive email traffic by attempting to decrypt the message using every possible key.
- **Prime Number Sieve:** Compromise the public key component of the encryption algorithm by determining the underlying prime numbers used in the encryption algorithm.
- **Endpoint Management:** Use social engineering or other techniques such as dumpster diving to determine the encryption key.



Effects on Search Time

(Number of Bits) (Number of Systems)

- Each time the key length is increased by one bit the size of the key space doubles which doubles the search time
 - When the key has 8 bits, the size of the search space is $2^8 = 256$ possible keys
 - When the key has 9 bits, the size of the search space is $2^9 = 512$ possible keys
- In the actual test runs, the search times follow this 2^{NumBits} rule: for each 1-bit increase in key size the search time doubles
- Each time the number of systems is doubled the search time is halved
 - When the key has 8 bits and the number of systems is 16, it will take $2^8 / 16 = 16$ units of time.
 - When the key has 8 bits and the number of systems is 32, it will take $2^8 / 32 = 8$ units of time.
- In the actual test runs the search times follow this **SearchTime/NumSystems** rule: for each doubling of the number of systems the search time is halved



RC5-64

- On 14-Jul-2002, a relatively characterless PIII-450 in Tokyo returned the winning key to the distributed.net key servers.
 - The key 0x63DE7DC154F4D039 produces the plaintext output:
 - The unknown message is: some things are better left unread
 - So, after 1,757 days and 58,747,597,657 work units tested the winning key was found!
- While it's debatable that the duration of this project does much to devalue the security of a 64-bit RC5 key by much, we can say with confidence that RC5-64 is not an appropriate algorithm to use for data that will still be sensitive in more than several years' time.
 - The next time someone bemoans the public's short attention span or need for instant gratification you should remind them what **331,252** people were able to accomplish by joining together and working for nearly **five years**. distributed.net's RC5-64 project clearly shows that even the most ambitious projects can be completed by volunteers thanks to the combined power of the internet and distributed computing.

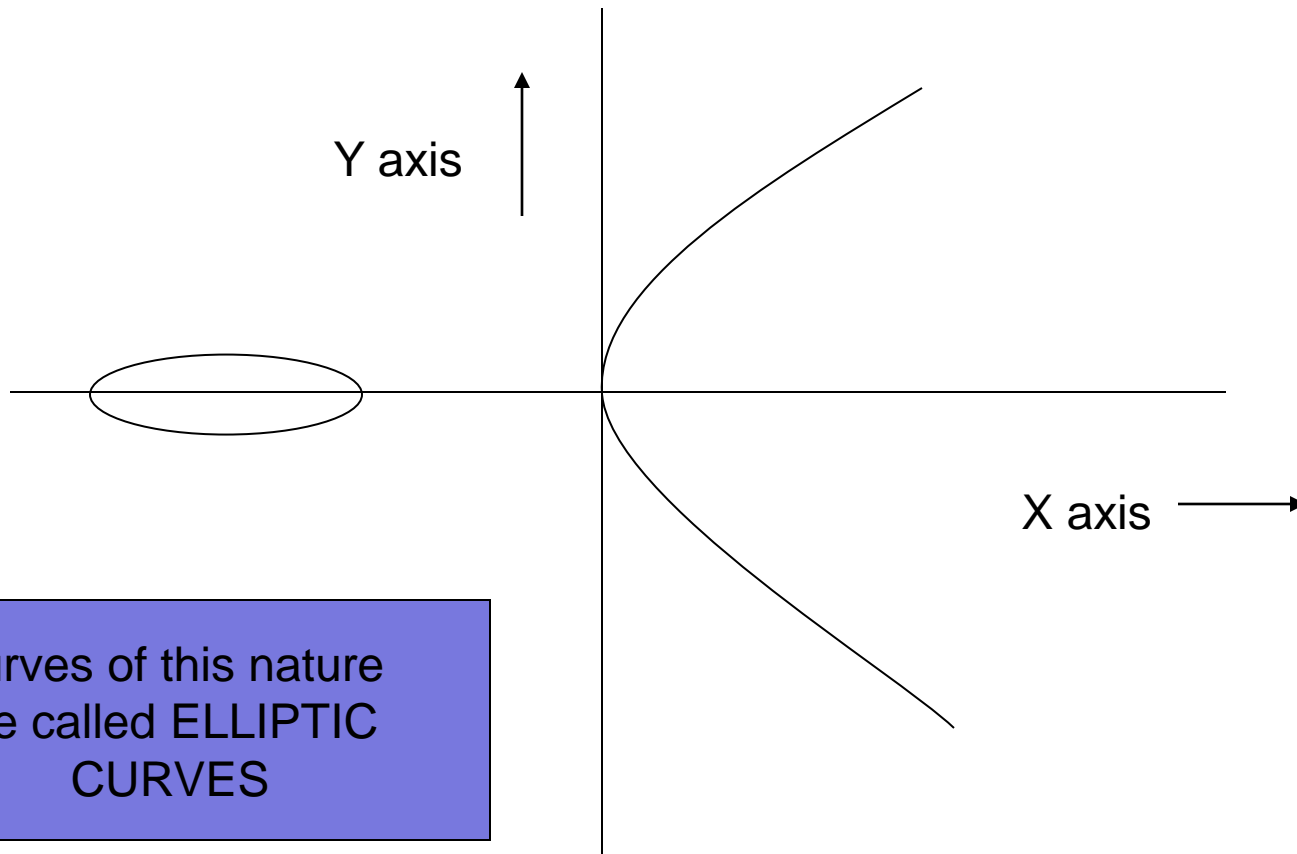


RC5 Stats

- Ignoring artificially high numbers resulting from network difficulties, we completed 86,950,894 work units on our best day.
 - This is 0.12% of the total key space meaning that at our peak rate we could expect to exhaust the key space in 790 days.
- Our peak rate of 270,147,024 keys/sec is equivalent to 32,504 800MHz Apple PowerBook G4 laptops or 45,998 2GHz AMD Athlon XP machines or (to use some rc5-56 numbers) nearly a half million Pentium Pro 200s.
- Over the course of the RC5-64 project, 331,252 individuals participated. We tested 15,769,938,165,961,326,592 keys.
- Sources: Charles Iser
 - <http://www.distributed.net/pressroom/news-20020926.html>
 - http://www.rsasecurity.com/news/releases/pr.asp?doc_id=1400



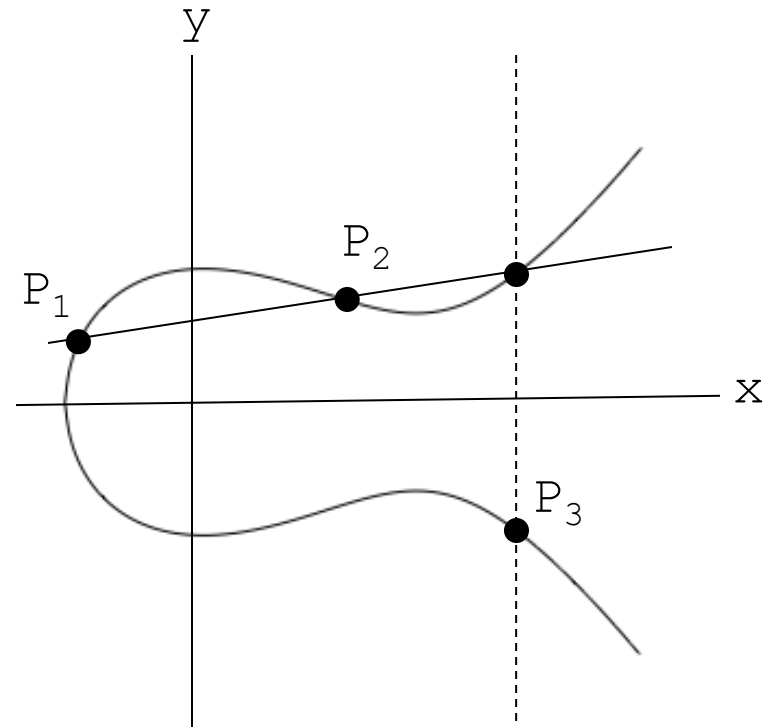
Graphical of an Elliptic Curve



Curves of this nature
are called ELLIPTIC
CURVES



Elliptic Curve Picture



- Consider elliptic curve
 $E: y^2 = x^3 - x + 1$
- If P_1 and P_2 are on E , we can define
 $P_3 = P_1 + P_2$
as shown in picture
- Addition is all we need

Steganography

- **Steganography means “to hide in plain sight” and is derived from the Greek term for *covered writing*. – Kruse & Heiser, 2004**
- **Automated steganographic tools exist for images, sound files, video, MP3s, documents, and other forms of transport.**



1.4MB Source

+



400KB Message

=



1.4MB Composite



Site and facility design secure principles

Reference: Casey Cegielski, Auburn
Reference: ISC2



Physical Security Requirements

Life Safety

Safety of people is the primary concern.

Physical Security Requirements

1. Deter
2. Delay
3. Detect
4. Assess
5. Respond

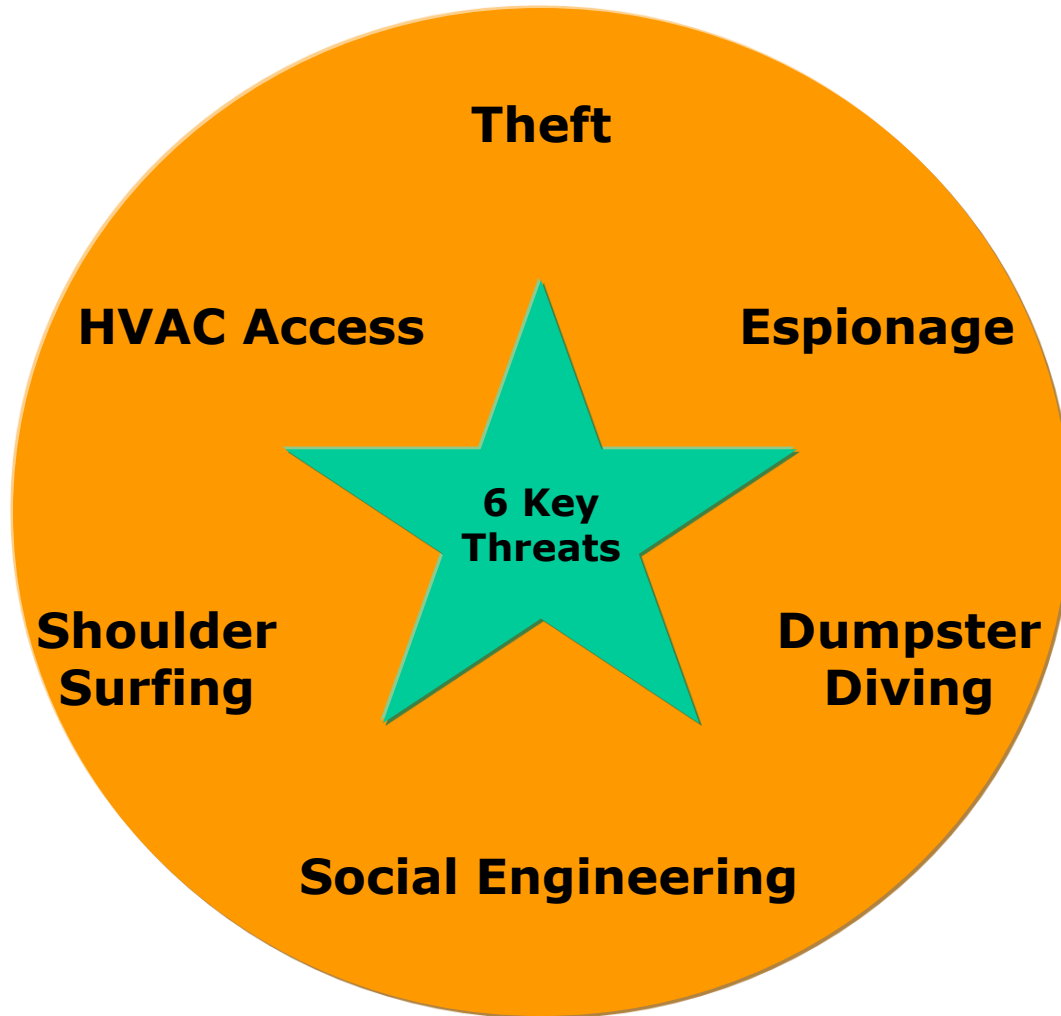


Threats to Physical Security

- **Natural/Environmental (e.g., earthquakes, floods, storms, hurricanes, fires)**
- **Utility Systems (e.g., communication outages, power outages)**
- **Human-Made/Political Events (e.g., explosions, vandalism, theft, terrorist attacks, riots)**



ISC2 Malicious Threats



Theft

- Internal/external results in increased costs

Espionage

- Loss of intellectual property & market share

Dumpster Diving

- Access to sensitive corporate information

Social Engineering

- Intelligence Attack

Shoulder Surfing

- Results in unauthorized access

HVAC

- Access via HVAC vents



Countermeasures

Theft

- **IDS & Locked Doors and Keys**
- **Access Control**

Espionage

- **Employee Tracking & Job Rotation**
- **Strict Internal Controls**

Dumpster Diving

- **Layered Defense System**
- **Disposal Policy**



Countermeasures

Social Engineering

- **Employee Accountability**
- **Employee Security Awareness**

Shoulder Surfing

- **Keyboard Keystroke Placement**
- **Awareness of your Surroundings**

HVAC Access

- **Narrow Shafts**
- **Section Lock Downs**

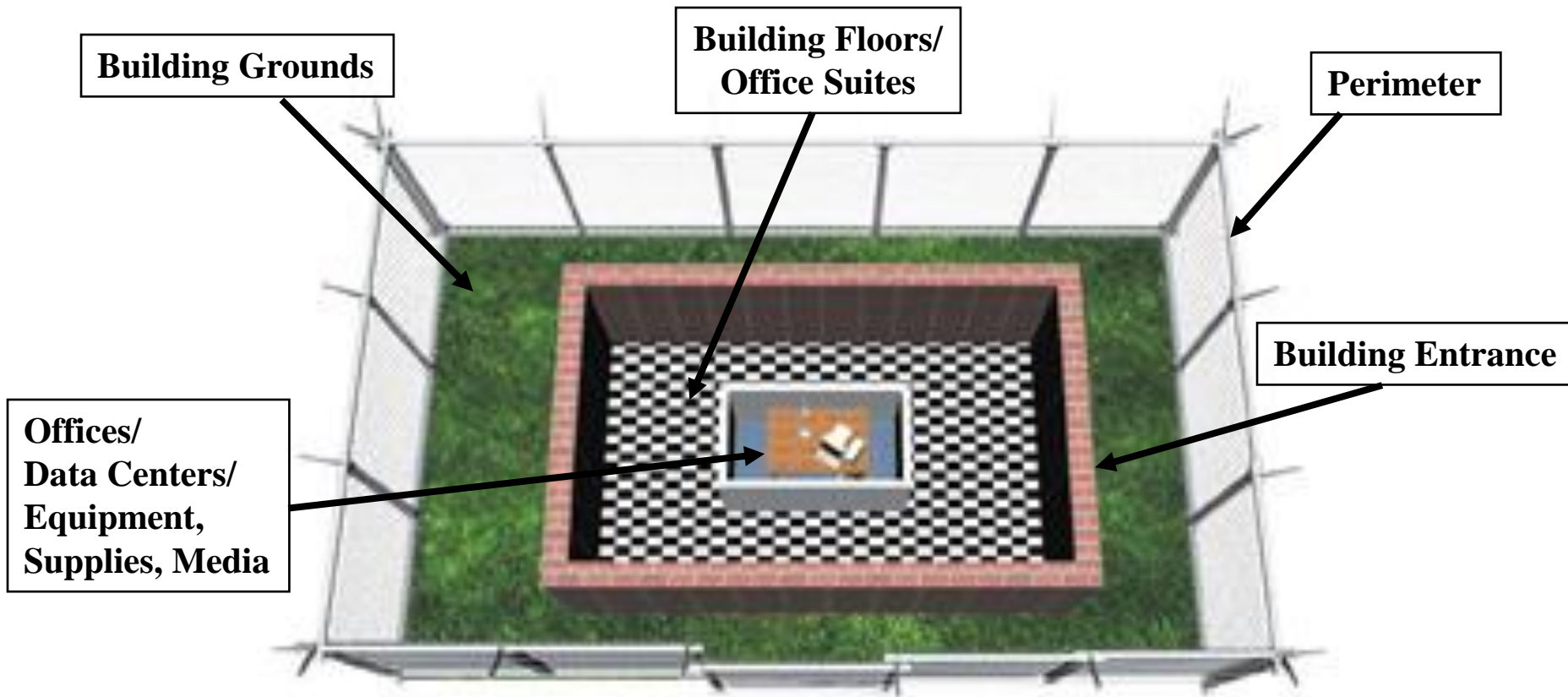


Physical Security Outline

- **Layered Defense Model**
- **Crime Prevention Through Environmental Design**
- **Site Location**
- **Facility Construction Impacts**
- **Infrastructure Support Systems**



Layered Defense Model



Crime Prevention Through Environmental Design

- The physical environment of a building is changed or managed to produce behavioral effects that will assist in reducing the incidence and fear of crime.
- Focuses on the relationships between the social behavior of people and the environments.

Three Key Strategies

1. **Territoriality** - people protect territory that is their own
2. **Surveillance** - high degree of visual control
3. **Access Control** - limit access and control the flow of access



Site Location

Security should include where the building is and how it should be built.

Crime?

Riots?

Natural disasters?

Adjacent buildings?

Airport?

Highway?

Military Base?

Emergency support systems?



Facility/Construction Issues

- **Walls, Windows, and Doors**
- **Entry Points**
 - Primary & secondary entrances
 - Windows
 - Roof access
 - Maintenance entrance
 - Emergency exits
 - Loading docks



Doors

- **Hollow-core versus solid-core**
- **Isolation of critical areas**
- **Lighting of doorways**
- **Contact Devices (switches)**
- **Mantraps (double door systems)**
- **Door Safety**
 - **Do not block exit doors**
 - **Provide sufficient and appropriate lock mechanics**
 - **Hinges securely fixed to the frames**
 - **Frame securely fixed to the adjoining wall.**

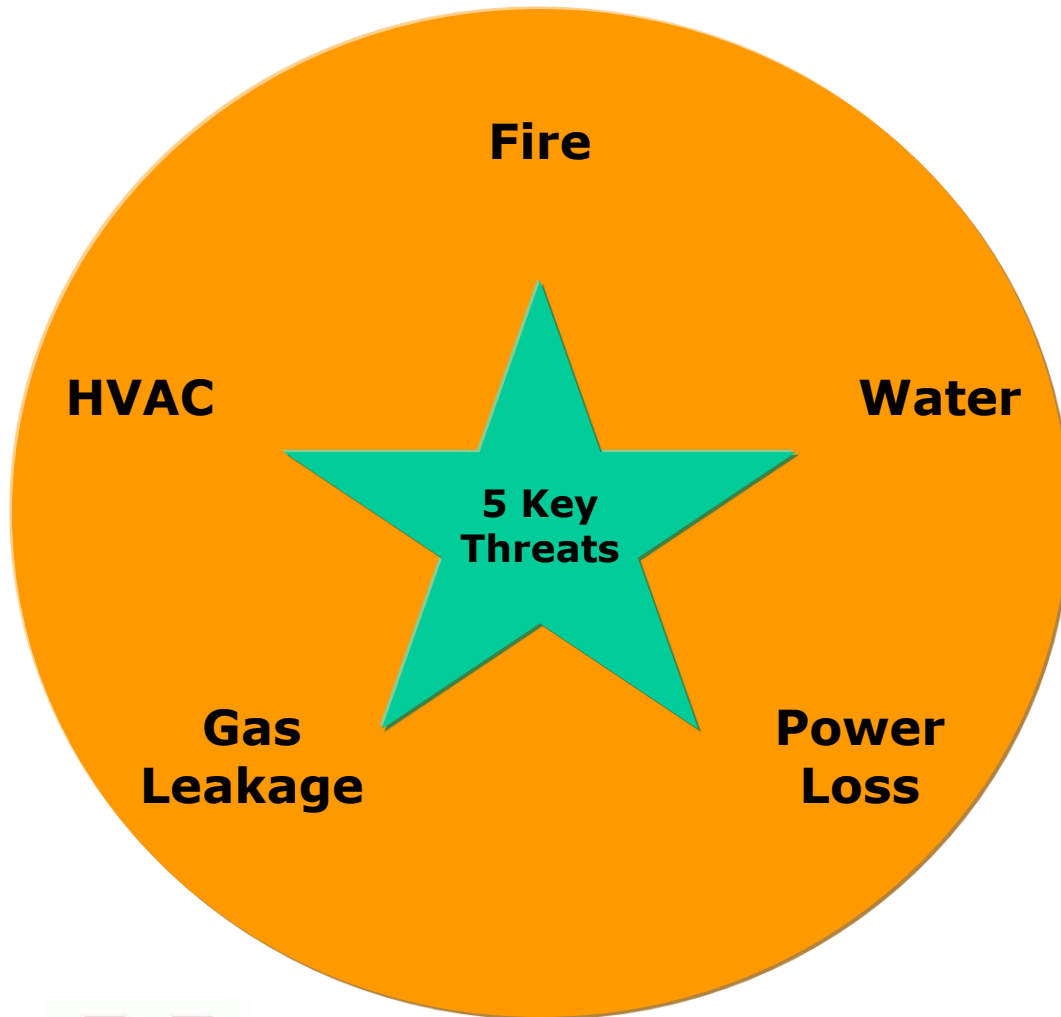


Windows

- **Standard plate glass**
- **Tempered glass**
- **Acrylic materials**
- **Polycarbonate windows - glass and polycarbonate combinations combine the best quality of glass and acrylics**
- **Laminated Glass**
- **Wired Glass**
- **Solar Window Films**
- **Window Security Films**
- **Glass Breakage Sensors**



ISC2 Support Systems Threats



Fire

- Damage & destruction of facilities/equipment

Water

- Flooding/dripping

Power Loss

- Disruption/stop in operations

Gas Leakage

- Explosion

HVAC

- Overheating/overcooling



Fire

Best Practices

- Fire Containment System (floors, vents, HVAC)
- Fire Extinguishing System (permanent & mobile)
- Abiding by the Fire Codes
- Fire Prevention Training and Drills



Fire Prevention

- **Use fire resistant materials for walls, doors, furnishings, etc.**
- **Reduce the amount of combustible papers around electrical equipment.**
- **Provide fire-prevention training to employees - remember, life safety is the most important issue.**
- **Conduct fire drills on all shifts so that personnel know how to safely exit a building.**



Fire Detection

- **Ionization-type smoke detectors detect charged particles in smoke.**
- **Optical (Photoelectric) detectors react to light blockage caused by smoke.**
- **Fixed or rate-of-rise temperature sensors - heat detectors that react to the heat of a fire.**
- **Combinations are usually used for the best effectiveness in detecting a fire.**



Fire Types and Suppression

Class	Type	Suppression Agents
A	Common combustibles	Water, foam, dry chemicals
B	Liquid	Gas, CO ₂ , foam, dry chemicals
C	Electrical	Gas, CO ₂ , dry chemicals
D	Combustible metals	Dry powders
K	Commercial kitchens	Wet chemicals



Fire Suppression Agents

- **Carbon Dioxide (CO₂) extinguishers provide a colorless, odorless chemical that displaces oxygen in the air.**
- **Halon gas - contains a white bromine powder produced in chlorofluorocarbon compounds (CFC)**
 - **factor in the depletion of the ozone layer.**
 - **Use limited by 1987 Montreal Protocol on Substances that Deplete the Ozone Layer**
- **FM200 most effective alternative - requires 7% concentration (Halon requires 5%)**



Sprinkler Systems

- **Water Sprinkler Systems**
 - Water could be a conductor of electricity - it may compound the problems in computer rooms.
 - Water can cause damage to electrical equipment.
 - “Pre-action” or “dry-pipe” system - water is held back by a valve and is released when the **sensor activates.**



Gas & Water Threats

- **Gas Leakage**
 - Identify Location and Test the main Shut-Off valve
 - Secure the Natural Gas Line (using layered defenses)
 - Communicate Natural Gas Line Design to Fire Department
 - Clearly mark Shut-off Valves
- **Water Detection Sensors**
 - Raised Floors
 - Emergency Shut-off Valves
 - Server room above ground level
- **Water pipes not located above server rooms**



Electrical Power

- **Disruptions in electrical power can have a serious business impact.**
- **Goal is to have “clean and steady power.”**
 - **Dedicated feeders**
 - **Alternate power source**
 - **Access Controls**
 - **Secure breaker and transformer rooms.**



Electrical Power Countermeasures

- **Power Loss**
 - Surge Suppressors
 - UPS and UPS Testing
 - Electrical Facilities separated from Data Center
 - Generators
- **Electric Power Controls – ‘clean power’**
 - Have an Emergency Power Off (EPO) switch that allows someone to shut down the power.
 - Install a power line monitor that detects and records fluctuations in frequency and voltage.
 - Ensure there is enough backup power to conduct an orderly shutdown to avoid data loss or device damage.



Power Outages

- **Complete loss of power**
 - **Blackout**
 - **Prolonged loss of commercial power**
 - **Fault**
 - **Momentary loss of power**



Degraded Power

- **Brownout**
 - Intentional reduction of voltage by the utility company for a prolonged period of time
- **Sag/Dip**
 - A short period of low voltage
- **Surge**
 - Sudden rise in voltage in the power supply
- **Transients**
 - Line noise that is superimposed on the supply circuit can cause a fluctuation in power.
- **Inrush Current**
 - The initial surge of current required when there is an increase in power demand.
- **Electrostatic Discharge**
 - A power surge generated by a person or device contacting another device and transferring a high voltage shock.

Interference

- **Noise** – A natural occurrence that happens when unwanted signals are generated in circuits that are in close proximity.
 - Typically, this disrupts the affected circuit.
- **Electromagnetic Interference (EMI)**
 - Caused by motors, lightning, etc.
- **Radio Frequency Interference (RFI)**
 - Created by components of electrical system
 - Caused by electric cables, fluorescent lighting, truck ignition



Heating, Ventilation and Air Conditioning Issues

- **HVAC computerized controls**
 - **Location**
 - **Access controls**
- **Appropriate maintenance of**
 - **Temperature**
 - **Humidity levels**
 - **Air quality**
- **Independence of the data center air conditioning system from the rest of the building.**
- **Documented maintenance procedures**



Layered Physical Security

- **Approaching security through ‘layers’ of controls**
- **Multi-layered**
- **Starts with the perimeter, then building grounds, then building entry points, etc.**



Perimeter and Building Grounds Boundary Protection

- **Perimeter security controls are the first line of defense.**
- **Protective barriers can be either natural or structural.**
 - **Natural protective barriers offer terrains that are difficult to cross, such as mountains, bodies of water, deserts, etc.**
 - **Structural barriers are devices such as fences, gates, bollards, and facility walls.**



Bollards

- A rising post designed for use in traffic control and protecting property premises.
- Provides security against vehicles ramming into, or stopping near buildings.
- Lighted bollards can be used for lighting controls along parks, paths, sidewalks, etc.



Perimeter and Building Grounds Boundary Protection

- **Lighting** – is the illumination of a locale, typically by artificial means such as light fixtures or lamps.
 - consistent level of light supplying reasonably good visibility needs to be available.
- **Features:**
 - Good lighting is one of the most successful crime preventive measures.
 - When used properly, light discourages unlawful activity, improves natural observation, and decreases fear.
 - Typically used with other controls, such as fences, patrols, alarm systems.



Locks

- **Most accepted and used physical security device**
- **Considered delay devices and not foolproof bars to entry - they are easily defeated**
- **All lock types are subject to force and special tools that can be used to gain entry**
- **Should be just one aspect of many physical security controls**
 - **Lock Components**
 - **Lock Body (Cylinder)**
 - **Bolt**
 - **Strike**
 - **Key**



Physical Access Control

- **Card Access Controls or Biometric Systems**
- **Smart cards, Magnetic Stripe cards, Proximity Cards, etc.**
- **Fingerprint, retina scans, signature dynamics, voice recognition, hand geometry, etc.**



Physical security

Dr. Drew Hamilton



Mississippi State University Center for Cyber Innovation

Domain 3 Security Engineering



278

Definitions of Physical Security

DoDI 5200.8-R

- **Physical Security** - That part of security concerned with physical measures designed to safeguard personnel; to prevent unauthorized access to equipment, installations, material, and documents; and to safeguard them against espionage, sabotage, damage, and theft.
- **Identity Protection and Management** is a key element that enables the physical security specialist to execute the DoD physical security program.



Physical Security Requirements

- **Physical Security Requirements can be segmented into 11 different sections**

1. Documentation	7. Mobile Computing Devices
2. Safety	8. Sensitive Data
3. Physical Access	9. Hard Copy Output
4. Facilities	10. Marking
5. Environmental	11. Incident Response
6. Human Threat	



Physical Security Measures

- alarms
- building construction
- cabling
- communications centers
- environmental controls (humidity and air conditioning)
- filtered power
- fire safety controls
- information systems centers
- physical access control systems (key cards, locks and alarms)
- power controls (regulator, uninterruptible power service (UPS), and emergency power-off switch)
- protected distributed systems
- shielding
- stand-alone systems and peripherals
- storage area controls



Summary

- **Engineering processes using secure design principles**
- **Security models fundamental concepts**
- **Security evaluation models**
- **Security capabilities of information systems**
- **Security architectures, designs, and solution elements vulnerabilities**
- **Web-based systems vulnerabilities**
- **Mobile systems vulnerabilities**
- **Embedded devices and cyber-physical systems vulnerabilities**
- **Cryptography**
- **Site and facility design secure principles**
- **Physical security**

