

CS574

Computer Security

San Diego State University

Spring 2008

Lecture #14

Today's Structure

- Administrivia
- Recent News
- Questions
- Lecture
- Turnin assistance

Administrivia

- Spring Break Next Week - no classes!
- Turn in exam question rewrites today.
- Assignment #1 Due Date: Friday 11:59 PM
- Turnin script should be enabled tonight
- Assignment #2 to be assigned after spring break

Recent News

- Software faults with Facebook and Wii
- RedHat risk assessment article

Questions?

Lecture

The Confinement Problem

- The *confinement problem* is the problem with preventing a subject from bypassing MAC and leaking information to an unauthorized entity.
- *Confinement* is a mechanism for enforcing the principle of *least privilege*.
- A *covert channel* is a path of communication that wasn't designed for [that sort of] communication.

Covert Channels

- Un-mediated communications.
- Used to cross MAC levels.
- Uses a shared resource.
- Failure of separation/virtualization.
- Two categories:
 - storage channels
 - timing channels

Covert Channels

- *A covert storage channel* uses an attribute of a shared resource to send/lead information.
- *A covert timing channel* uses a temporal or ordering relationship among accesses to a shared resources to send/lead information.
- *A noiseless covert channel* is one that uses a resource available only to the sender and receiver.
- *A noisy covert channel* is a covert channel that uses resources available to subjects other than just the sender and receiver.

Covert Storage Channel Examples

- Shared directory with visible filenames.
- File/record locking (i.e., where lock error is different from a MAC error).
- Shared devices (e.g., tape drives, “cd/dvd-burners”, printers, network ports, etc.).
- Process slot exhaustion.

Covert Timing Channel Examples

- System paging rate.
- I/O or network usage rates.
- Process creation rate.
- Timeslice relinquishment.

Covert Channel Mitigation

- Provide full isolation for all resources.
- Allocate resources at program start, including runtime.
- Allocate resources in a uniform manner.
- Inject randomness into potential covert channels so that noise dominates the channel.

Covert Channel Mitigation

- Tradeoff between system performance and covert channel mitigation.
- Characterize covert channels by bandwidth; high-bandwidth covert channels = insecure.
- Audit the use of potential covert channels.
- DoD characterizes covert channels by bandwidth:
 - 100 bits/sec or more – “high” bandwidth
 - 1 bit/sec or less – “low” bandwidth
 - .1 bits/sec or more – audit needed

Virtualization

- Keep resources logically separate by emulation / simulation.
- All shared resources must be virtualized. (memory, disk, devices, etc.)
- Prevent all un-mediated communication.
- Prevent a denial of service.

Virtualization

- A *sandbox* is an environment where the access of processes are restricted according to a security policy.
- A *virtual machine* is a program that simulates the hardware of a (possibly abstract) computer system.
- A *virtual machine monitor* is a special OS that runs virtual machines.

Virtualization

- Virtualize to a user level.
 - most operating systems do this already.
- Virtualize to an OS level.
 - each OS thinks it is “in control” of the hardware (and unaware of other OSes).
 - an operating system shouldn't have to be modified to run in a virtual machine.

Virtualization Examples

- One big disk can look like several smaller disks.
- Several small disks can look like one big disk.
- RAM + slow disk can look like a fast disk.
- Noncontiguous RAM + MMU + disk can look like a huge chunk of contiguous RAM.
- One fast CPU + lots of RAM can look like several slower CPUs with less RAM.

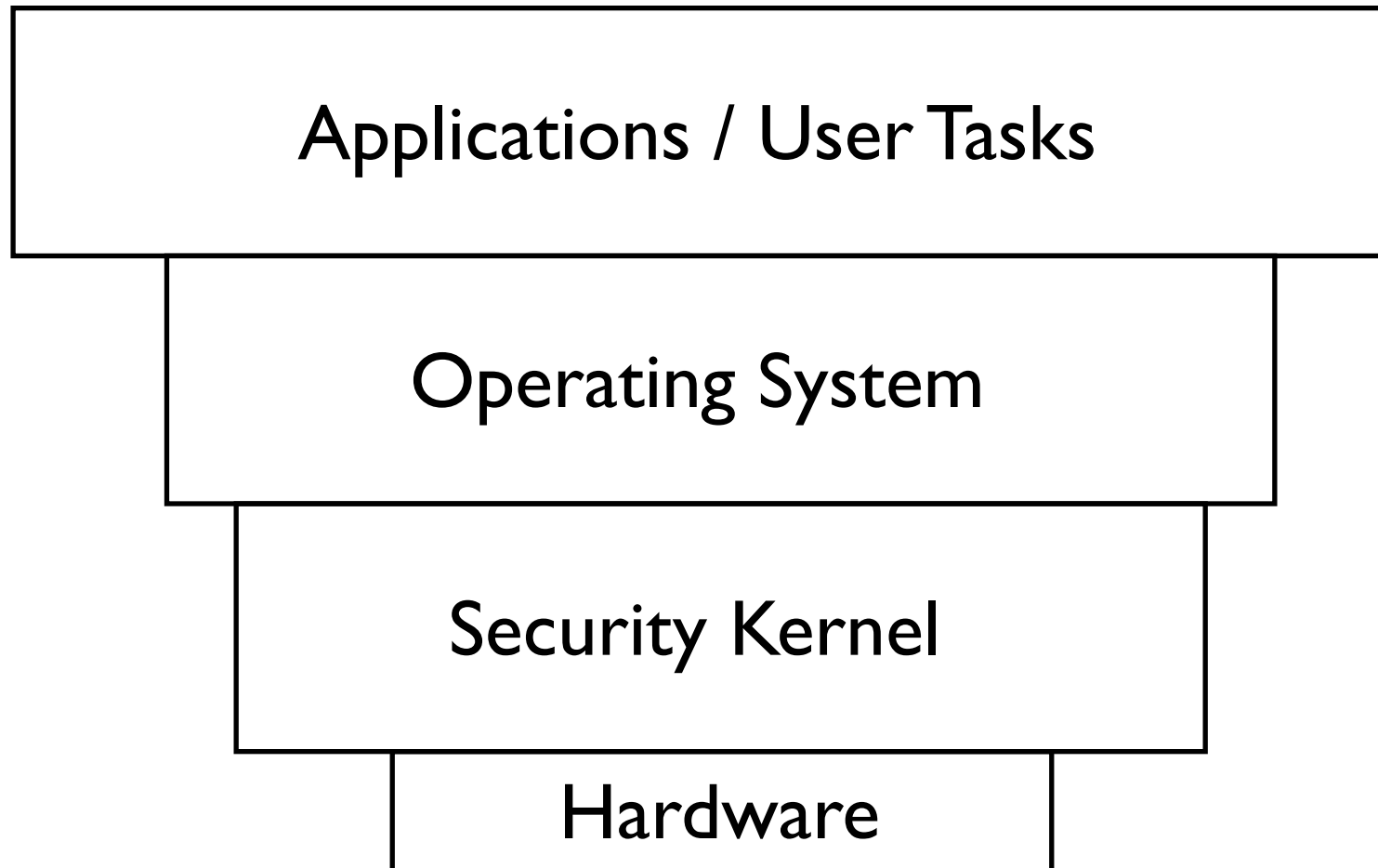
Layers

- Hardware / kernel / OS / user
- Design the OS in layers for:
 - better encapsulation
 - better damage control
- Hierarchies minimize critical parts of the system, and allow for better prioritization.

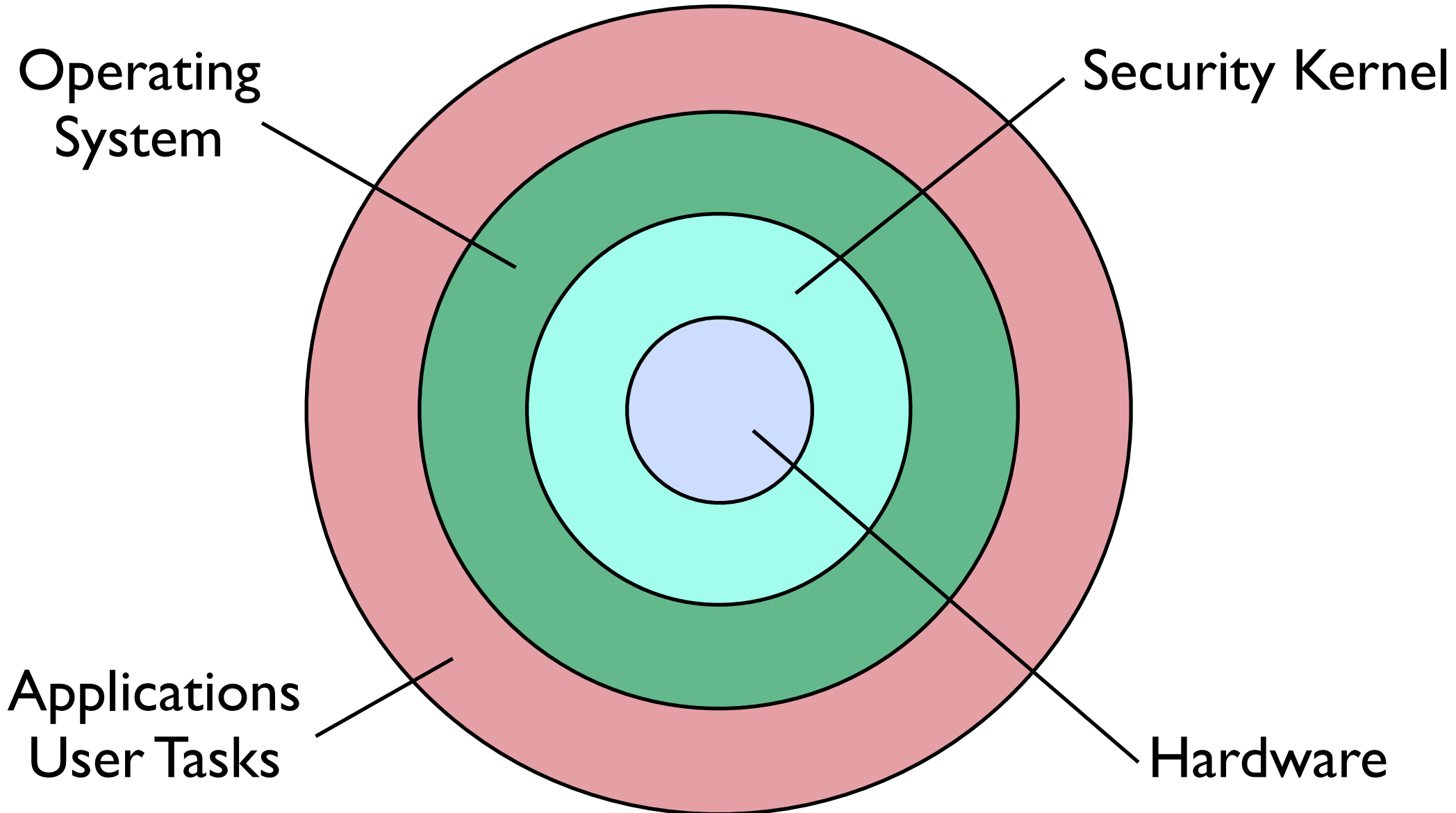
Layers

- Inner/lower layers – more trusted.
- Outer/upper layers – less trusted.
- Multiple (nested) security perimeters.
- Not everything has to have the same level of trust.

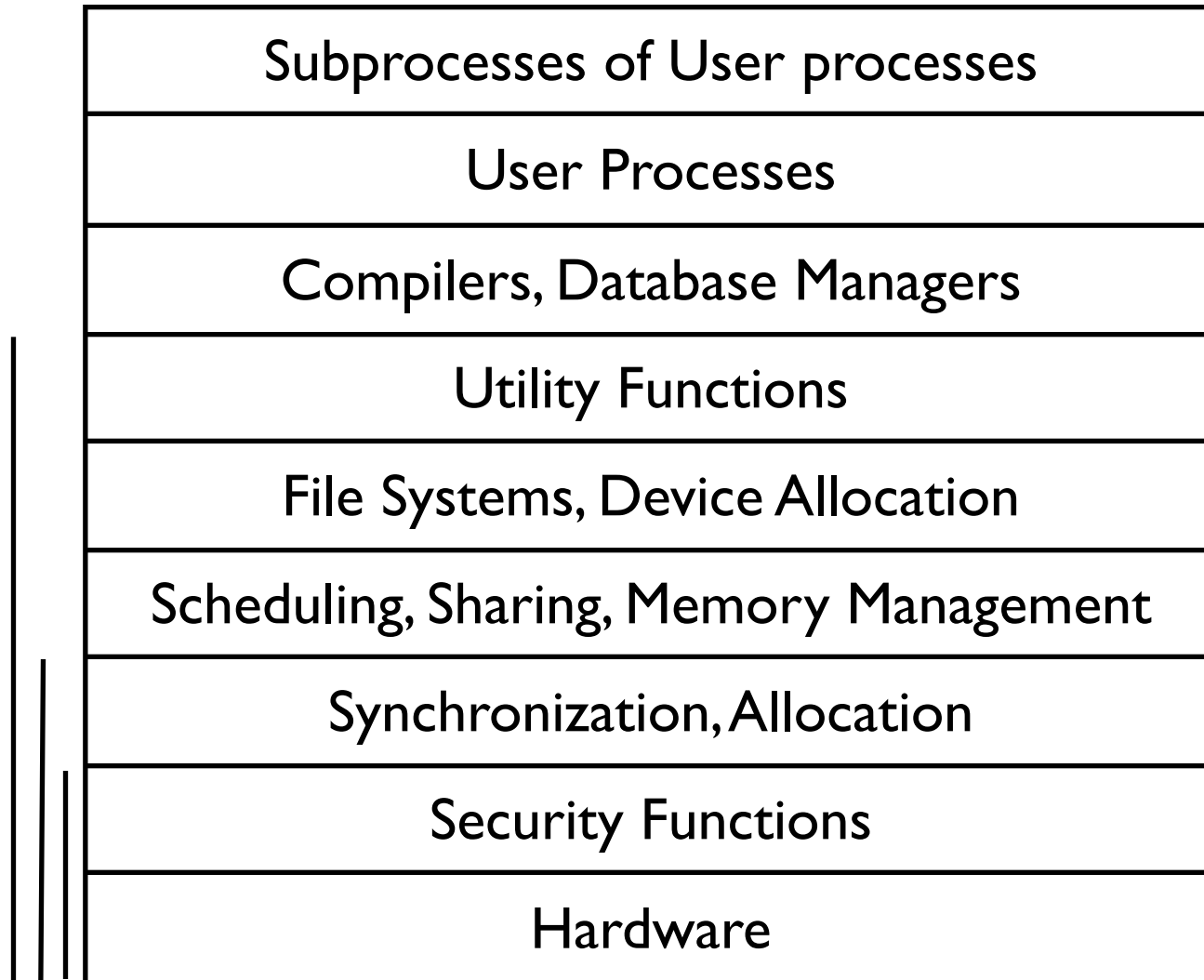
Layered Execution Domains



Layered Execution Domains



Layered Example



Layered Design

- Different sensitivities can spread a module over several execution domains.
- Layers allow for a hierarchical structure and the identification of critical components.
- Problems with a component affect only those components that depend on it.

TCB

Trusted Computing Base

- Also Known As “TCB”
- Where all the critical security features are located / the name we give to the set of all the critical security features.
- Something must be trusted – this is it.
- Combination of selected software, firmware, and hardware.

Trusted Computing Base Concepts

- Reference Monitor
- Reference Validation Mechanism
- Security Kernel
- Security Perimeter

Reference Monitor

A reference monitor is an access control concept of an abstract machine that mediates all access to protected objects.

Reference Monitor

- Recall ‘complete mediation’ principle.
- This is the piece that controls *all* access.
- Must be:
 - tamperproof
 - always invoked (unbypassable)
 - verifiable (analyzable)

Reference Validation Mechanism

- Implementation of the Reference Monitor concept.
- Often broken up into multiple pieces for each subsystem:
 - memory
 - files
 - other system objects
- Each piece enforces the same policy.

Security Kernel

- Minimal kernel that deals just with security.
- Most common implementation of RM/RVM.
- Also provides:
 - auditing
 - authentication
 - management of security
- Combination of hardware, firmware, and software (which is why TCB combines these).

Security Kernel Design Features

- Coverage - involved in every access.
- Separation - isolate security mechanisms.
- Unity - one codebase for security code.
- Modifiability - changes easy to make/test.
- Compactness - only manage security.
- Verifiability - analyzed for correctness.

Security Perimeter

- Perimeters delineate a boundary.
- Things on the inside are what you *have* to trust.
- Things on the outside you don't have to trust.
- All the security-critical parts of the OS need to be on the inside.
- We'll see this concept again when we get to network security.

TCB Components

- Typical security kernel **plus** security-critical programs.
- Most often this means the operating system + “trusted programs”.
- Comprised of all components inside the security perimeter.
- Excluded elements need not be trusted.

TCB Components (UNIX)

- All SUID (0) programs.
- The login (authentication) program(s).
- Password-changing program(s).
- Device drivers.
- Programs that open raw sockets
(ping, traceroute, tcpdump, etc.).

TCB Depends On...

- Hardware
- Processes
- Files
- Protected Memory
- Interprocess Communication

TCB Interactions

- Process Activation
- Execution Domain Switching
- Memory Protection
- I/O Operations

TCB on PCs

- TCG - Trusted Computing Group
- TCPA - Trusted Computing Platform Alliance
- Palladium - Microsoft's super-TCPA
- NGSCB - Microsoft's Next Generation Secure Computing Base
- LeGrande (now TET) - Intel's DRM scheme

TCPA/TCG

- Essentially a smartcard on the motherboard.
- Specifies a chip that provides
 - a trusted boot function
 - public key crypto functions
 - initialization/management functions
- Can prevent booting of “alternative” or modified operating systems.

TCPA/TCG

- Can “seal” data (such as symmetric keys) so that this data can be “unsealed” only if the system configuration is the same as when the data was sealed.
- IBM has been shipping the ESS (Embedded Security Subsystem) chip for years.

Palladium/NGSCB

- Does all the TCGA/TCG does, plus some.
- Adds another ring to CPU protection levels.
- Protects memory used by security system.
- Adds a trusted path to the keyboard.
- Adds a trusted path to the display.

Application

- Who is protecting what from whom?
- Can be used to implement originator-based access control.
- Designed to protect IP.
- Much corporate desire for an effective “DRM” (copy protection) system.

End Lecture

Lecture References

- Pfleeger, *Security in Computing*, 3rd & 4th edition
- Bishop, *Introduction to Computer Security*
- Safford, *Clarifying Misinformation on TCPA*
- Schoen, *Palladium Details*
- Anderson, *Security in Open versus Closed Systems - The Dance of Boltzmann, Coase and Moore*
- Department of Defense *Trusted Computer System Evaluation*, DoD 5200.28-STD
- Arbaugh, <http://www.cs.umd.edu/~waa/TCPA/TCPA-goodnbad.html> [2002] (Accessed March 2008)
- Anderson, <http://www.cl.cam.ac.uk/~rja14/tcpa-faq.html> [2003] (Accessed March 2008)

Reading

- Chapter 5

Assignment #1 Turnin

- Running the turnin script.
- Running just the unit tests.
- Using PAGER to help inspect output.
- Obtaining help via email.

Finis