# Last time

- Security Policies and Models
  - Bell La-Padula and Biba Security Models
  - Information Flow Control
- Trusted Operating System Design
  - Design Elements
  - Security Features

# This time

- Trusted Operating System Design
  - Security Features
  - Trusted Computing Base
  - Least Privilege in Popular OSs
  - Assurance
- Security in Networks
  - Network Concepts
  - Threats in Networks

# Accountability and Audit

- Keep an audit log of all security-related events
- Provides accountability if something goes bad
  - Who deleted the sensitive records in the database?
  - How did the intruder get into the system?
- An audit log does not give accountability if attacker can modify the log
- At what granularity should events be logged?
  - For fine-grained logs, we might run into space/efficiency problems or finding actual attack can be difficult
  - For coarse-grained logs, we might miss attack entirely or don’t have enough details about it

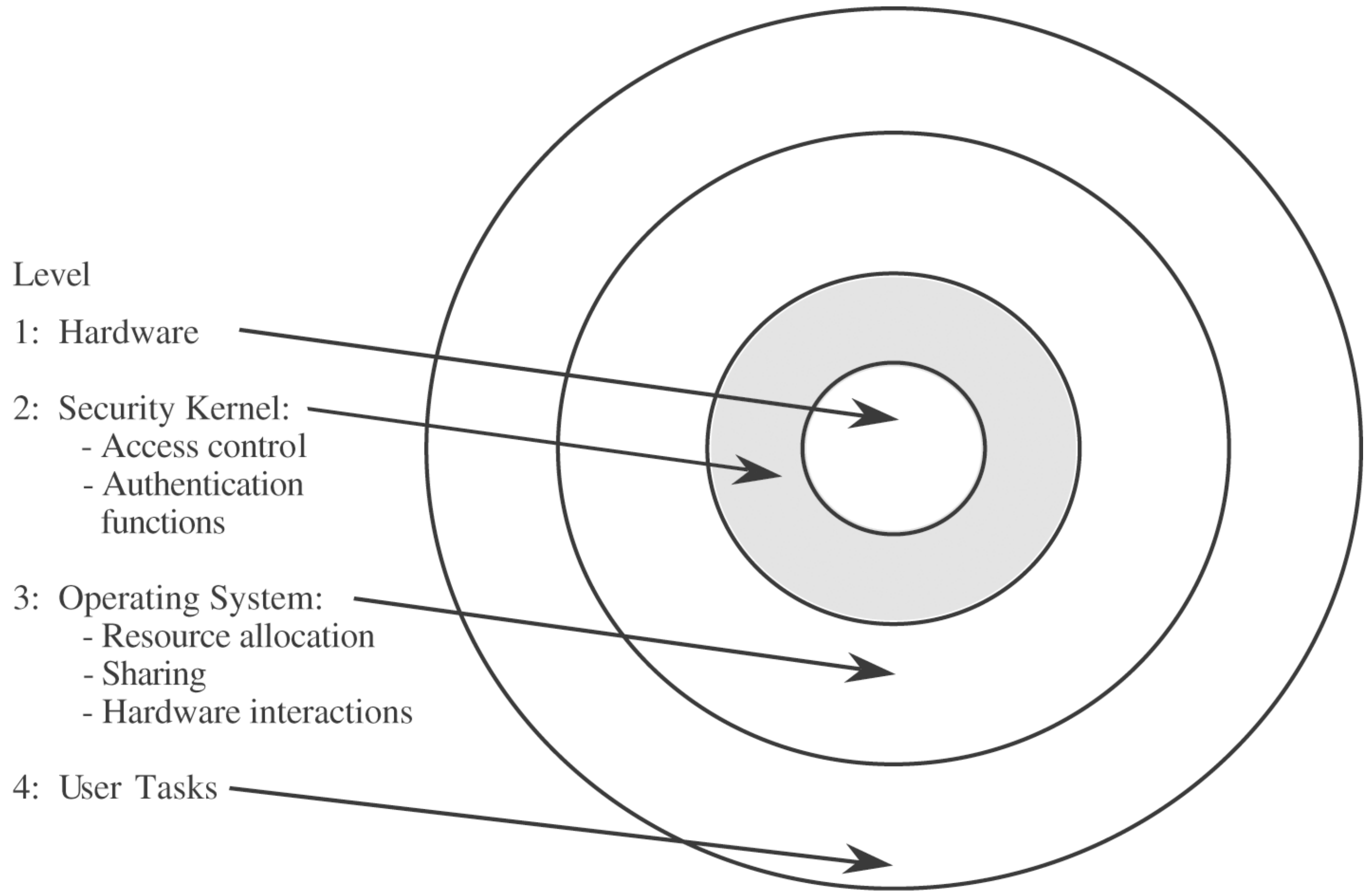
# Intrusion Detection

- There shouldn't be any intrusions in a trusted OS
- However, writing bug-free software is hard, people make configuration errors,…
- Audit logs might give us some information about an intrusion
- Ideally, OS detects an intrusion as it occurs
- Typically, by correlating actual behaviour with normal behaviour
- Alarm if behaviour looks abnormal
- See later in Network Security unit

# Trusted Computing Base (TCB)

- Part of a trusted OS that is necessary to enforce OS security policy
  - Changing non-TCB part of OS won't affect OS security, changing its TCB-part will
  - TCB better be complete and correct
- TCB can be implemented either in different parts of the OS or in a separate security kernel
- Separate security kernel makes it easier to validate and maintain security functionality
- Security kernel runs below the OS kernel, which makes it more difficult for an attacker to subvert it

# Security Kernel



# Rings

- Some processors support this kind of layering based on “rings”
- If processor is operating in ring n, code can access only memory and instructions in rings ≥ n
- Accesses to rings < n trigger interrupt/exception and inner ring will grant or deny access
- x86 architecture supports four rings, but Linux and Windows use only two of them
  - user and supervisor mode
  - i.e., don’t have security kernel
- Some research OSs (Multics, SCOMP) use more

# Reference Monitor

- Crucial part of the TCB
- Collection of access controls for devices, files, memory, IPC,…,
- Not necessarily a single piece of code
- Must be tamperproof, unbypassable and analyzable
- Interacts with other security mechanism, e.g., user authentication

# Virtualization

- Virtualization is a way to provide logical separation (isolation)
- Different degrees of virtualization
- Virtual memory
    - Page mapping gives each process the impression of having a separate memory space
- Virtual machines
    - Also virtualize I/O devices, files, printers,…
    - Currently very popular (VMware, Xen, Parallels,...)
    - If Web browser runs in a virtual machine, browser-based attacks are limited to the virtual environment
    - On the other hand, a rootkit could make your OS run in a virtual environment and be very difficult to detect (“Blue Pill”)

# Least Privilege in Popular OSs

- Pretty poor
- Windows pre-NT: any user process can do anything
- Windows pre-Vista: fine-grained access control. However, in practice, many users just ran as administrators, which can do anything
  - Some applications even required it
- Windows Vista
  - Easier for users to temporarily acquire additional access rights ("User Account Control")
  - Integrity levels, e.g., Internet Explorer is running at lowest integrity level, which prevents it from writing up and overwriting all a user's files

# Least Privilege in Popular OSs (cont.)

- Traditional UNIX: a root process has access to anything, a user process has full access to user's data
- SELinux and AppArmor provide Mandatory Access Control (MAC) for Linux, which allows the implementation of least privilege
  - No more root user
  - Support both confidentiality and integrity
  - Difficult to set up
- Other, less invasive approaches for UNIX
  - Chroot, privilege separation, SUID (see next slides)
- What about the iPhone?

# Chroot

- Sandbox/jail a command by changing its root directory
  - `chroot /new/root command`
- Command cannot access files outside of its jail
- Some commands/programs are difficult to run in a jail
- But there are ways to break out of the jail

# Privilege Separation

- Run as much of a program in an unprivileged way as possible
- Example: Privilege separation in OpenSSH
- Split SSH daemon into a privileged monitor and an unprivileged, jailed child
- Child processes (maybe malicious) network data from a client
  - Child might get corrupted
- Child needs to talk to monitor when it needs access to privileged information (e.g., password file)
  - Small, well-defined interface
  - Makes it much more difficult to also corrupt monitor
- Monitor shuts down client if it detects suspicious behavior

# setuid/suid Bit

- In addition to bits denoting read, write and execute access rights, UNIX ACLs also contain an suid bit
- If suid bit is set for an executable, the executable will execute under the identity of its owner, not under the identity of the caller
  - /usr/bin/passwd belongs to root and has suid bit set
  - If a user calls /usr/bin/passwd, the program will assume the root identity and can thus update the password file
- Make sure to avoid “confused deputy” attack
  - Eve executes /usr/bin/passwd and manages to convince the program that it is Alice who is executing the program. Eve can thus change Alice’s password

# Assurance

- How can we convince others to trust our OS?
- Testing
  - Can demonstrate existence of problems, but not their absence
  - Might be infeasible to test all possible inputs
  - Penetration testing: Ask outside experts to break into your OS
- Formal verification
  - Use mathematical logic to prove correctness of OS
  - Has made lots of progress recently
  - Unfortunately, OSs are probably growing faster in size than research advances

# Assurance (cont.)

- Validation
  - Traditional software engineering methods
  - Requirements checking, design and code reviews, system testing

# Evaluation

- Have trusted entity evaluate OS and certify that OS satisfies some criteria
- Two well-known sets of criteria are the “Orange Book” of the U.S. Department of Defense and the Common Criteria
- Orange Book lists several ratings, ranging from “D” (failed evaluation, no security) to “A1” (requires formal model of protection system and proof of its correctness, formal analysis of covert channels)
  - See text for others
  - Windows NT has C2 rating, but only when it is not networked and with default security settings changed
  - Most UNIXes are roughly C1

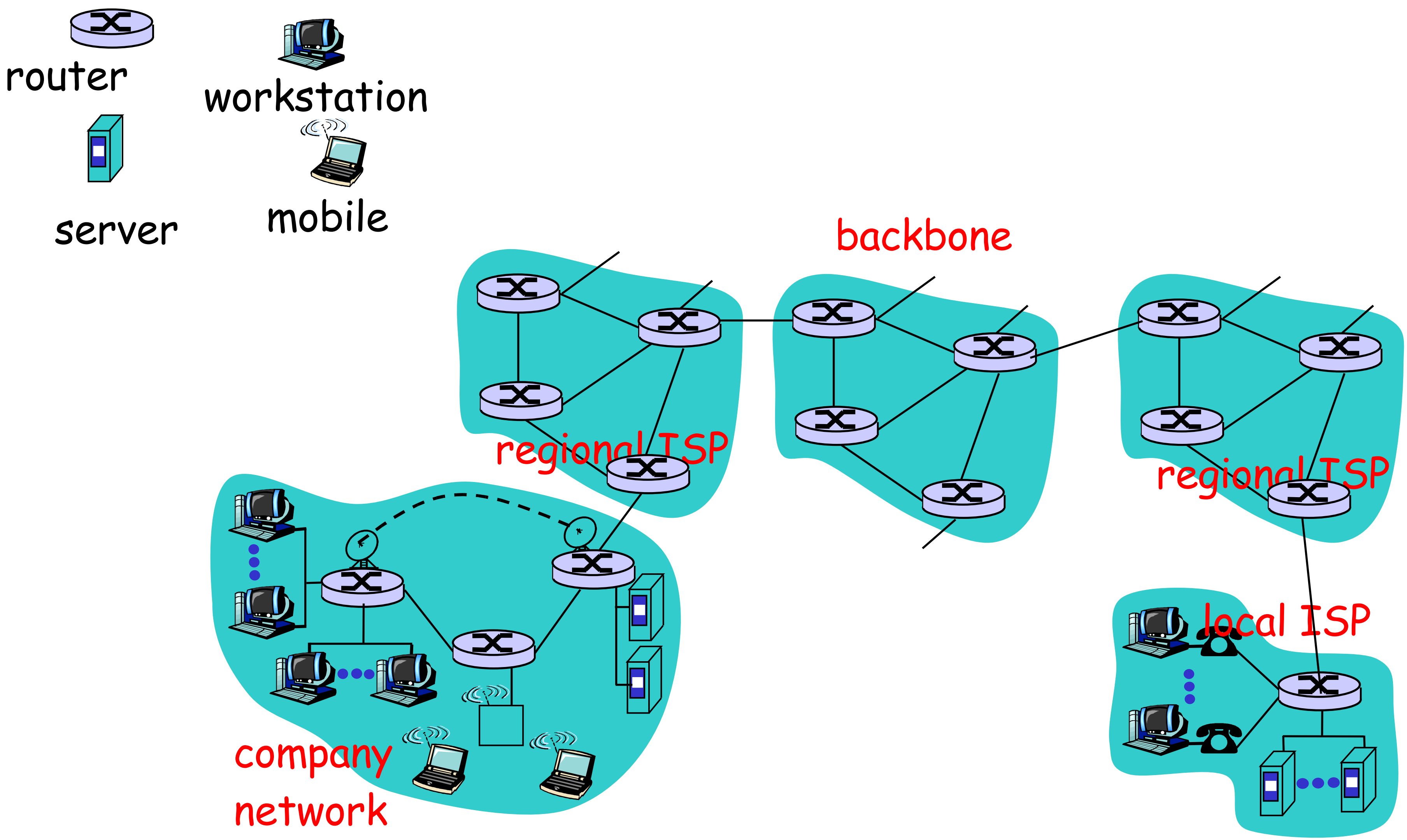
# Common Criteria

- Replace Orange Book, more international effort
- Have Protection Profiles, which list security threats and objectives
- Products are rated against these profiles
- Ratings range from EAL 1 (worst) to EAL 7 (best)
- Windows XP has been rated EAL 4+ for the Controlled Access Protection Profile (CAPP), which is derived from Orange Book's C2
  - Interestingly, the continuous release of security patches for Windows XP does not affect its rating

# Security in Networks

- Security in Networks
  - Network Concepts
  - Threats in Networks
  - Network Security Controls
  - Firewalls
  - Intrusion Detection Systems

# Architecture of the Internet

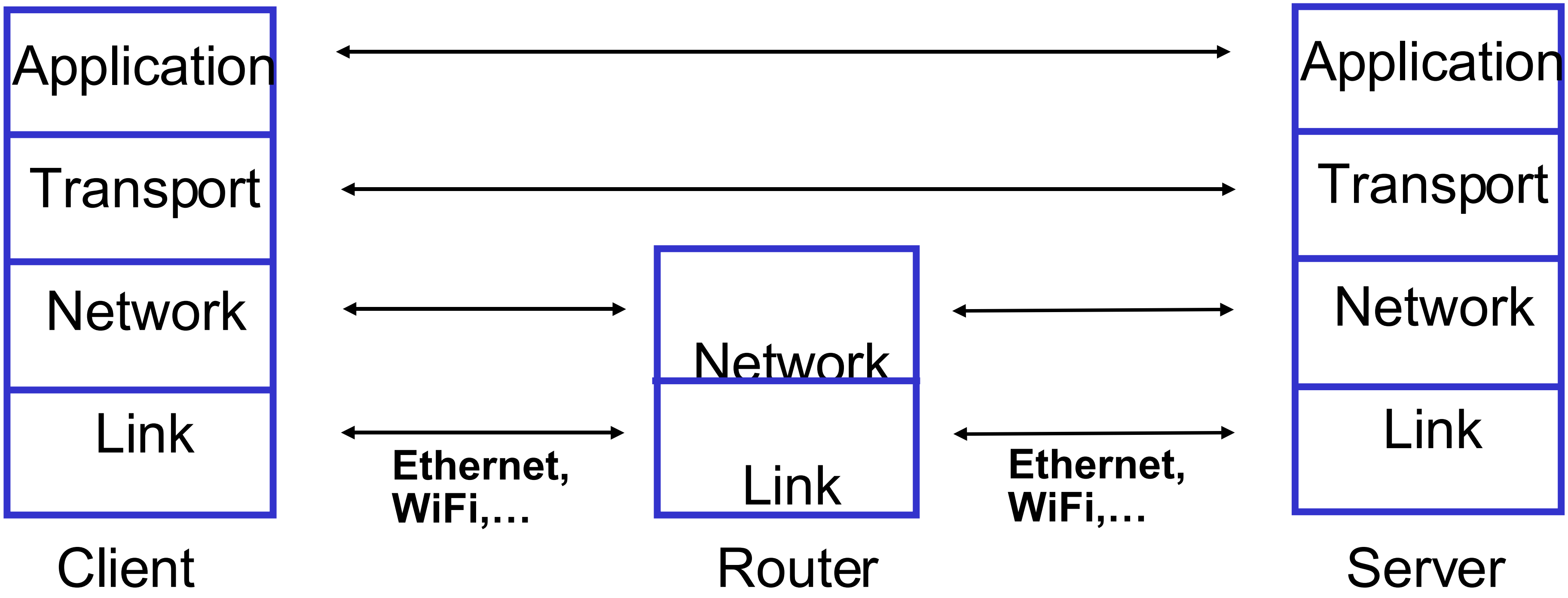



Slide adapted from “Computer Networking” by Kurose & Ross

# Characteristics of the Internet

- No single entity that controls the Internet
- Traffic from a source to a destination likely flows through nodes controlled by different, unrelated entities
- End nodes cannot control through which nodes traffic flows
  - Worse, all traffic is split up into individuals packets, and each packet could be routed along a different path
- Different types of nodes
  - Server, laptop, router, UNIX, Windows,…
- Different types of communication links
  - Wireless vs. wired
- TCP/IP suite of protocols
  - Packet format, routing of packets, dealing with packet loss,…

# TCP/IP Protocol Suite




- Transport and network layer designed in the 1970s to connect local networks at different universities and research labs
- Participants knew and trusted each other
- Design addressed non-malicious errors (e.g., packet drops), but not malicious errors

# Threats in Networks

- Reconnaissance
- Attacks on confidentiality
- Impersonation and spoofing
- Attacks on integrity
- Protocol failures
- Web site vulnerabilities
- Denial of service
- Threats in active/mobile code
- Script kiddies

# Port Scan

- To distinguish between multiple applications running on the same server, each application runs on a “port”
  - E.g., a Web server typically runs on port 80
- Attacker sends queries to ports on target machine and tries to identify whether and what kind of application is running on a port
  - Identification based on loose-lipped applications or based on how exactly application implements protocol
- Goal of attacker is to find application with remotely exploitable flaw
  - E.g., Apache web server prior to version 1.3.26 is known to be vulnerable to buffer overflow
  - Exploits for these flaws can be found on the Internet

# Recap

- Trusted Operating System Design
  - Security Features
  - Trusted Computing Base
  - Least Privilege in Popular OSs
  - Assurance
- Security in Networks
  - Network Concepts
  - Threats in Networks

# Next time

- Security in Networks
  - Threats in Networks
  - Network Security Controls
  - Firewalls