Week 8 - Friday

COMP 4290

Last time

- E-mail attacks
- OS security
- Mandatory access control
  - Bell-La Padula

Questions?

Project 2

Assignment 3

Call for Mentors

- Hanby Elementary in Westerville is looking for mentors for two programs
- FIRST LEGO League (Tuesdays 2:45–4:45 PM)
  - Students research real-world problems and build and program LEGO EV3 Mindstorm robots to complete themed missions.
- Girls Who Code (Mondays 2:45–3:45 PM)
  - Girls Who Code students work on projects such as app or game design, website creation, and 3D printing prototypes that solve real-world problems.
- Both are great opportunities to give back to the community and build your resume
- If interested, send me an e-mail

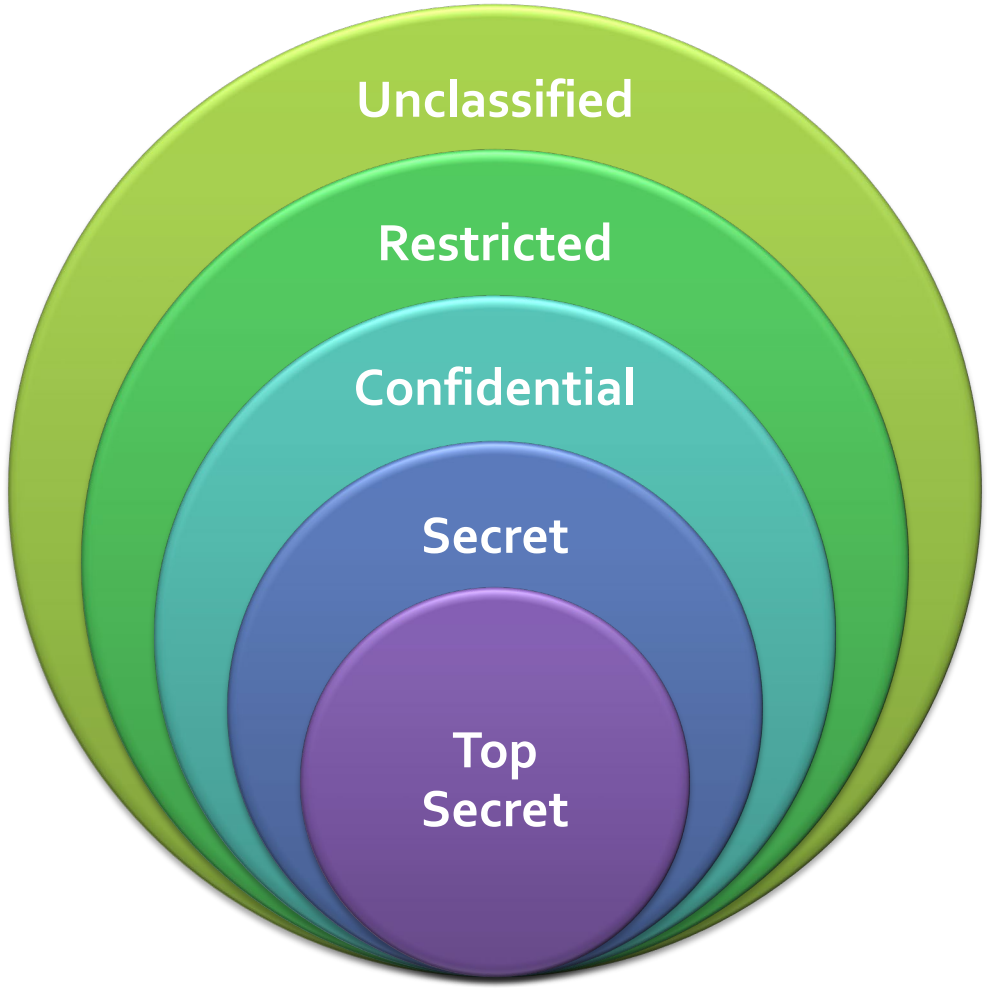
Adam Garantche Presents

Mandatory Access Control

Bell-La Padula Model

Bell-La Padula overview

- Confidentiality access control system
- Military-style classifications
- Uses a linear clearance hierarchy
- All information is on a **need-to-know** basis
- It uses **clearance** (or **sensitivity**) levels as well as project-specific **compartments**



Security clearances

- Both subjects (users) and objects (files) have security clearances
- Below are the clearances arranged in a hierarchy

Clearance Levels	Sample Subjects	Sample Objects
Top Secret (TS)	Tamara, Thomas	Personnel Files
Secret (S)	Sally, Samuel	E-mail Files
Confidential (C)	Claire, Clarence	Activity Log Files
Restricted (R)	Rachel, Riley	Telephone List Files
Unclassified (UC)	Ulaley, Ursula	Address of Headquarters

Simple security condition

- Let $level_O$ be the clearance level of object ***O***
- Let $level_S$ be the clearance level of subject ***S***
- The simple security condition states that ***S*** can read ***O*** if and only if the $level_O \leq level_S$ and ***S*** has discretionary read access to ***O***
- In short, you can only read down
- In a few slides, we will expand the simple security condition to make the concept of level

*-Property

- The *-property states that ***S*** can write ***O*** if and only if the $level_S \leq level_O$ and ***S*** has discretionary write access to ***O***
- In short, you can only write up

Basic security theorem

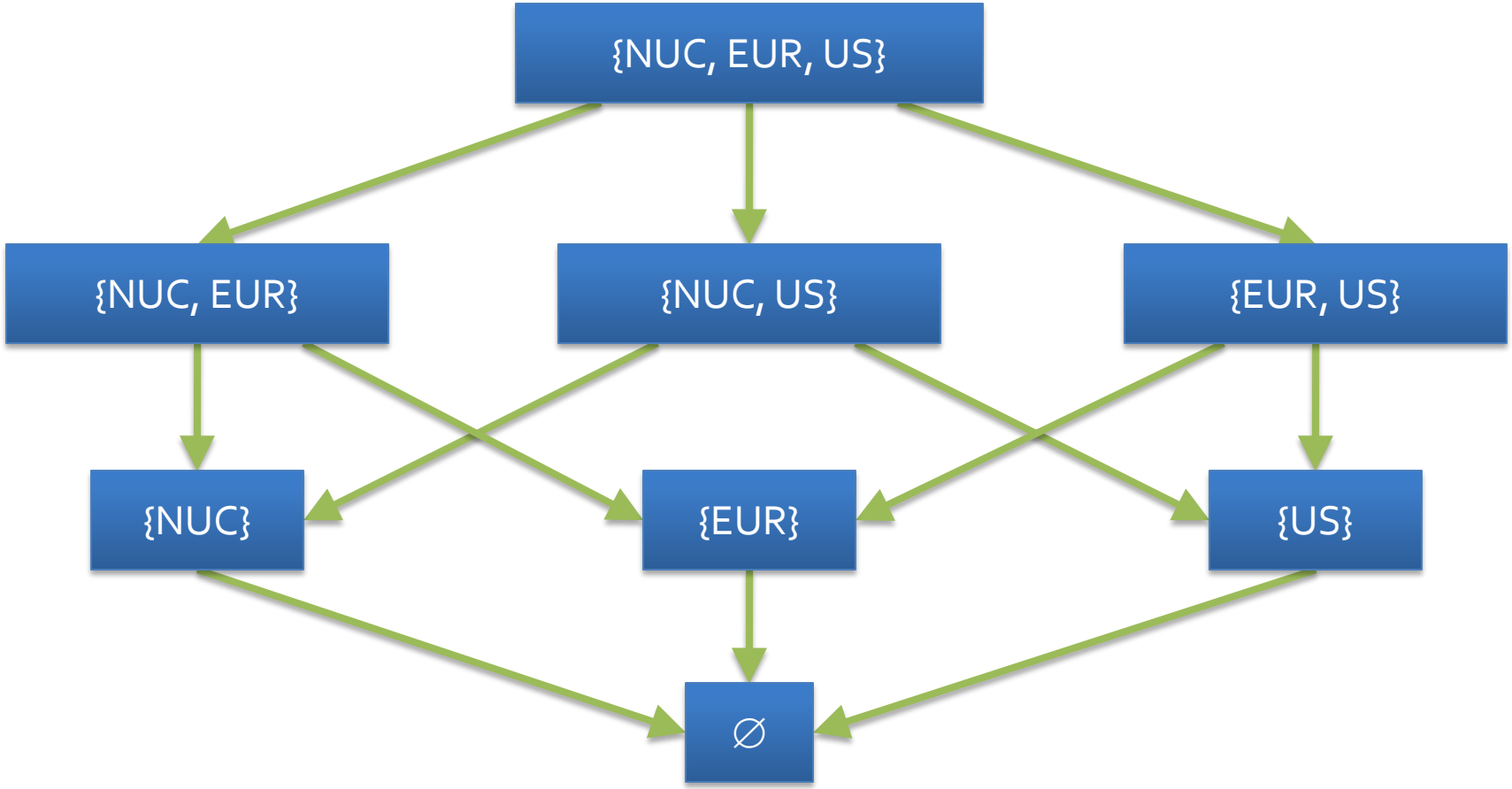
- Assume your system starts in a secure initial state
- Let ***T*** be all the possible state transformations
- If every element in ***T*** preserves the simple security condition and the *-property, every reachable state is secure
- This is sort of a stupid theorem, because we define "secure" to mean a system that preserves the security condition and the *-property

Adding compartments

- We add compartments such as NUC = Non-Union Countries, EUR = Europe, and US = United States
- The possible sets of compartments are:
  - ∅
  - {NUC}
  - {EUR}
  - {US}
  - {NUC, EUR}
  - {NUC, US}
  - {EUR, US}
  - {NUC, EUR, US}
- Put a clearance level with a compartment set and you get a **security level**
- The literature does not always agree on terminology

Romaine lattice

- The subset relationship induces a **lattice**



Updated properties

- Let L be a clearance level and C be a category
- Instead of talking about $level_O \leq level_S$, we say that security level (L, C) **dominates** security level (L', C') if and only if $L' \leq L$ and $C' \subseteq C$
- Simple security now requires (L_S, C_S) to dominate (L_O, C_O) and S to have read access
- *-property now requires (L_O, C_O) to dominate (L_S, C_S) and S to have write access
- Problems?

Chinese Wall Model

Chinese Wall overview

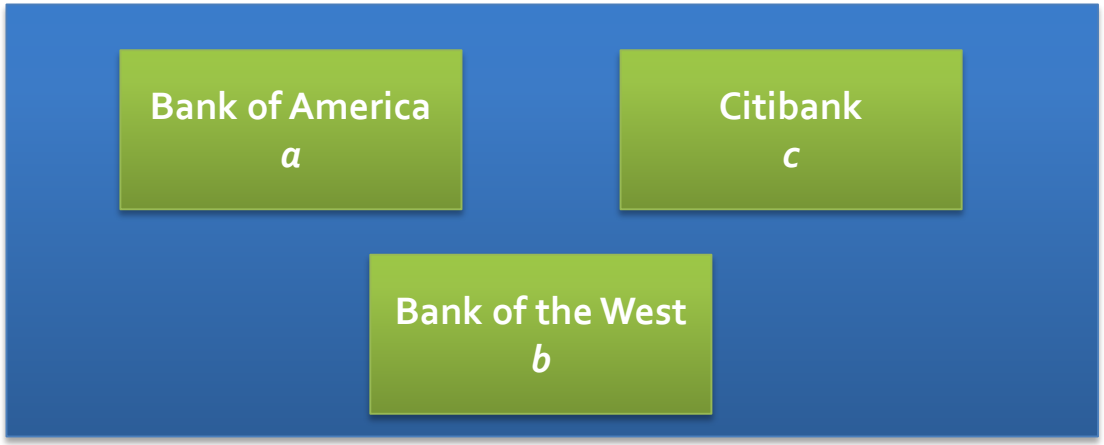
- The Chinese Wall model respects both confidentiality and integrity
- It's important in business situations where there are **conflict of interest** issues
- Real systems, including British law, have policies similar to the Chinese Wall model
- Most discussions around the Chinese Wall model are couched in business terms

Chinese Wall definitions

- We can imagine the Chinese Wall model as a policy controlling access in a database
- The **objects** of the database are items of information relating to a company
- A **company dataset** (CD) contains objects related to a single company
- A **conflict of interest** (COI) class contains the datasets of companies in competition
- Let COI(***O***) be the COI class containing object ***O***
- Let CD(***O***) be the CD that contains object ***O***
- We assume that each object belongs to exactly one COI

COI Examples

Bank COI Class



Gasoline Company COI Class

CW-Simple Security Condition

- Let PR(***S***) be the set of objects that ***S*** has read
- Subject ***S*** can read ***O*** if and only if any of the following is true
  1. There is an object ***O′*** such that ***S*** has accessed ***O′*** and CD(***O′***) = CD(***O***)
  2. For all objects ***O′***, ***O′*** $\in$ PR(***S***) $\Rightarrow$ COI(***O′***) $\neq$ COI(***O***)
  3. ***O*** is a sanitized object
- Give examples of objects that can and cannot be read

CW-*-Property

- Subject ***S*** may write to an object ***O*** if and only if both of the following conditions hold
  1. The CW-simple security condition permits ***S*** to read ***O***
  2. For all unsanitized objects ***O′***, ***S*** can read ***O′*** $\Rightarrow$ CD(***O′***) = CD(***O***)

Biba Model

Biba overview

- Integrity based access control system
- Uses integrity levels, similar to the clearance levels of Bell-LaPadula
- Precisely the **dual** of the Bell-LaPadula Model
- That is, we can only read up and write down
- Note that integrity levels are intended only to indicate integrity, **not** confidentiality
- Actually a measure of accuracy or reliability

Formal rules

- $\boldsymbol{S}$ is the set of subjects and $\boldsymbol{O}$ is the set of objects
- Integrity levels are ordered
- $\boldsymbol{i}(\boldsymbol{s})$ and $\boldsymbol{i}(\boldsymbol{o})$ gives the integrity level of $\boldsymbol{s}$ or $\boldsymbol{o}$, respectively
- Rules:
  1. $\boldsymbol{s} \in \boldsymbol{S}$ can read $\boldsymbol{o} \in \boldsymbol{O}$ if and only if $\boldsymbol{i}(\boldsymbol{s}) \leq \boldsymbol{i}(\boldsymbol{o})$
  2. $\boldsymbol{s} \in \boldsymbol{S}$ can write to $\boldsymbol{o} \in \boldsymbol{O}$ if and only if $\boldsymbol{i}(\boldsymbol{o}) \leq \boldsymbol{i}(\boldsymbol{s})$
  3. $\boldsymbol{s}_1 \in \boldsymbol{S}$ can execute $\boldsymbol{s}_2 \in \boldsymbol{S}$ if and only if $\boldsymbol{i}(\boldsymbol{s}_2) \leq \boldsymbol{i}(\boldsymbol{s}_1)$

Extensions

- Rules 1 and 2 imply that, if both read and write are allowed, ***i***(***s***) = ***i***(***o***)
- By adding the idea of integrity compartments and domination, we can get the full dual of the Bell-La Padula lattice framework
- Real systems (for example the LOCUS operating system) usually have a command like ***run-untrusted***
- That way, users have to recognize the fact that a risk is being made
- What if you used the same levels for integrity **and** security, could you implement both Biba and Bell-La Padula on the same system?

Rootkits

Rootkits

- As you probably know, root is the highest level of privilege on a Unix-like system
- A **rootkit** is a program that gives you high-level access to an OS
- By downloading a rootkit, people who are not sophisticated hackers might be able to take over an OS

Phone rootkits

- Rootkits on phones used to be impossible (and pointless)
- Now, phones are incredibly important to our lives
- Researchers have created rootkits for phones that can:
  - Turn on your phone's microphone
  - Drain the batteries
  - Find your location with GPS
- Snowden's revelations showed that the NSA has listened to people's phones without them knowing
- There are no virus scanners for phones! (yet)

Rootkits evading detection

- Rootkits often work by evading detection
- The problem is that, if the rootkit has control of your OS, you can't trust what your OS shows you
- The rootkit could normally list files in your file explorer as long as they are not rootkit files
- Researchers have discovered rootkits by
  - Doing low level file operations to read the files present
  - Comparing the size of files reported by the OS with disk usage

Sony XCP rootkit

- Sony put a program on music CDs called XCP (extended copy protection) which allowed users to listen to the CD on Windows but not rip its contents
- It installed itself without the user's knowledge
- It had to have control over Windows and be hard to remove
- It would hide the presence of any program starting with the name **`$sys$`**

Patching Sony XCP

- Once people heard about XCP and became upset, Sony provided an uninstaller for it
- The uninstaller ran by connecting to a webpage, but changing to a different webpage allowed malicious code to run on your computer with full privileges
- Some researchers claim that the power that DRM needs on your computer is similar to what malicious programs want

TDSS rootkits

- TDSS is a family of rootkits, TDL-1 through TDL-4
- TDL-1 hides any files from the user that start with **`tdl`**
- It does so with special drivers that are loaded on startup by editing the registry
  - The regular drivers are **hooked** so that they jump to rootkit drivers before going back to regular code
- Later TDL versions obfuscate their code, send encrypted messages to their creators, and circumvent Windows protections for drivers
- In 2009 NetworkWorld estimated that 3 million computers were control by TDSS

Good rootkits?

- Some employers may put rootkits on their office computers, giving them complete control
  - If an employee is fired, leaves, or dies
  - If an employee is up to no good
- Parents can put rootkits on their children's computers
- Tools like antivirus software operates a lot like a rootkit
  - High privileges
  - Hard to disable or detect
  - What happens if your antivirus software is corrupted?

Ticket out the Door

Upcoming

Next time…

- Network security
- Network attacks
- Austin Rheyne presents
- **No class on Monday!**

Reminders

- Keep reading Sections 6.2 through 6.5
- Keep working on Project 2
  - Due next Friday
- Finish Assignment 3
  - **Due tonight by midnight!**