

CIS 4930: Secure IoT

Lecture 7

Prof. Kaushal Kafle

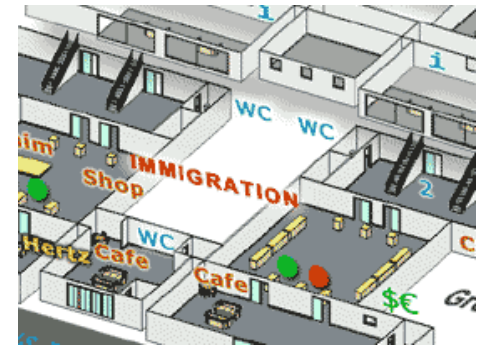
Class Notes

- Final project proposal **deadline is today!**
- Homework 2 is due 09/19.

- Buckle up! **QUIZ TIME!!**

Policy

- A policy specifies the rules of security
 - Some statement of secure procedure or configuration that parameterizes the operation of a system
 - Example: Airport Policy
 - Take off your shoes
 - No bottles that could contain > 3 ozs
 - Empty bottles are OK?
 - You need to put your things through X-ray machine
 - Laptops by themselves, coat off
 - Go through the metal detector
- **Goal:** prevent on-airplane (metal) weapon, flammable liquid, dangerous objects ... (successful?)



Computer Security Policy Goals

- **Secrecy**
 - Don't allow reading by unauthorized subjects
 - Control where data can be written by authorized subjects
 - Why is this important?
- **Integrity**
 - Don't permit dependence on lower integrity data/code
 - Why is this important?
 - What is "dependence"?
- **Availability**
 - The necessary function must run
 - Doesn't this conflict with above?

... policy <> straightforward

- Driving license test: take until you pass
 - Mrs. Miriam Hargrave of Yorkshire, UK failed her driving test **39** times between 1962 and 1970!!!!
 - ... she had 212 driving lessons
 - She finally got it on the 40th try.
 - Some years later, she was quoted as saying, “sometimes I still have trouble *turning right*”

“A policy is a set of acceptable behaviors.”

- F. Schneider



Access Control/Authorization

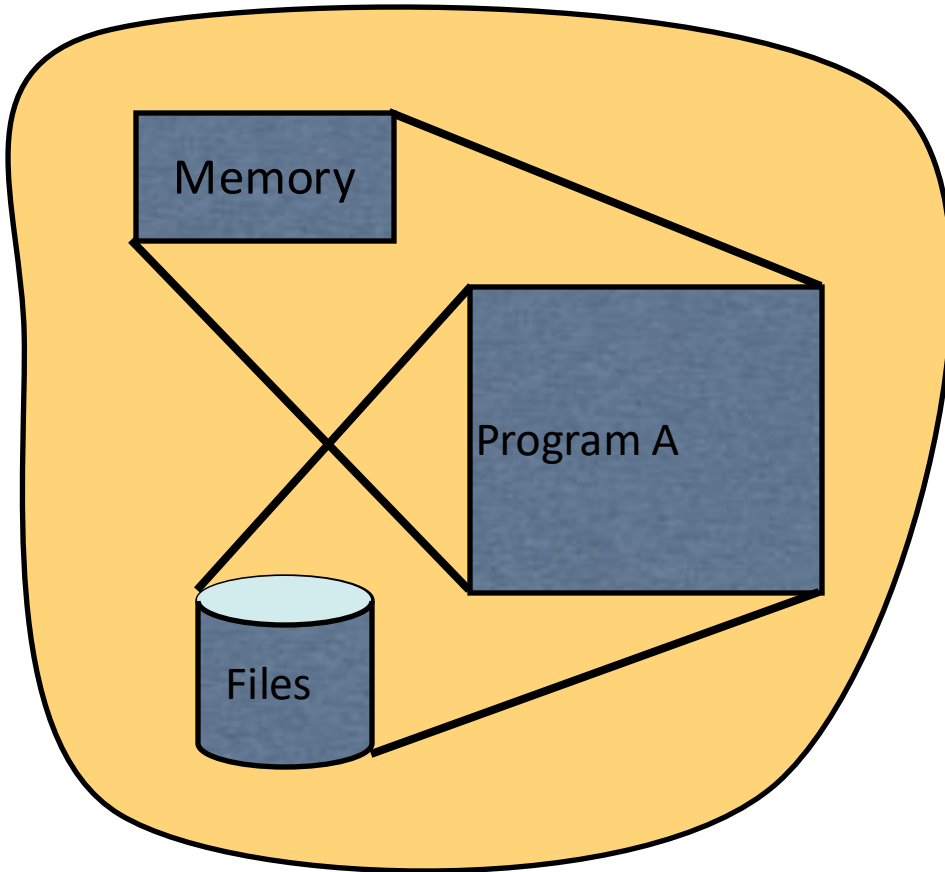
- An **access control** system determines what *rights* a particular *entity/subject* has for a set of *objects*
- It answers the question
 - E.g., do *you* have the right to *read /etc/passwd*
 - Does *Alice* have the right to *view* the *CS website*?
 - Do *students* have the right to *share project data*?
 - Does *the professor* have the right to *change* your *grades*?
- An **Access Control Policy** answers these questions

Simplified Access Control

- **Subjects** are the active entities that do things
 - E.g., *you, Alice, students, the Professor*
- **Objects** are passive things that things are done to
 - E.g., */etc/passwd, CSCI website, project data, grades*
- **Rights** are actions that are taken
 - E.g., *read, write, share*

Protection Domains

Protection domain



- A *protection domain* specifies the set of resources (objects) that a process can access and the operations that the process may use to access such resources.
- How is this done today?
 - Memory protection
 - E.g., UNIX protected memory, file-system permissions (rwx...)

Policy is defined with respect to the protection domain it governs.

Access Policy Enforcement

- A *protection state* defines what each subject can do
 - E.g., in an access bits --- the policy
- A *reference monitor* enforces the protection state
 - A service that responds to the query...
- A correct reference monitor implementation meets the following guarantees
 1. Complete Mediation
 2. Tamperproof
 3. Simple enough to verify
- A *protection system* consists of a (1) protection state, (2) operations to modify that state, and (3) a reference monitor to enforce that state

Trusted Computing Base (TCB)

- The trusted computing base is the infrastructure that you assume will behave correctly
- What do we trust?
 - Hardware (keyboard, monitor, ...)
 - Operating Systems
 - Implementations
 - Local networks
 - Administrators
 - Other users on the same system
- **Axiom:** the larger the TCB, the more assumptions you must make (and hence, the more opportunity to have your assumptions violated).



Project: Important to recognize what you are trusting to secure your system

The Access Matrix

- An access matrix is one way to represent policy.
 - Frequently used mechanism for describing policy
- Columns are objects, subjects are rows.
 - To determine if S_i has right to access object O_j , find the appropriate entry.
 - There is a matrix for each right.
- No system stores the Access Control Matrix (ACM). ----Why?

	O_1	O_2	O_3
S_1	Y	Y	N
S_2	N	Y	N
S_3	N	Y	Y

The Access Matrix

- Do systems store the entire access control matrix?
- Two ways:
 - Store with the objects (Access control lists (ACL))
 - Store with the subjects (Capability Lists (CL))

	O ₁	O ₂	O ₃
S ₁	Y	Y	N
S ₂	N	Y	N
S ₃	N	Y	Y

Access Control

- Suppose the private key file for J is object O_1
 - Only J can read
- Suppose the public key file for J is object O_2
 - All can read, only J can modify
- Suppose all can read and write from object O_3
- What's the access matrix?

	O_1	O_2	O_3
J	?	?	?
S_2	?	?	?
S_3	?	?	?

Secrecy

- Does the following protection state ensure the secrecy of J's private key in O_1 ?

	O_1	O_2	O_3
J	R	RW	RW
S_2	-	R	RW
S_3	-	R	RW

Integrity

- Does the following access matrix protect the integrity of J's public key file O_2 ?

	O_1	O_2	O_3
J	R	RW	RW
S_2	-	R	RW
S_3	-	R	RW

Trusted Processes

- *Does it matter if we do not trust some of J's processes?*
- *Trojan Horse*: Attacker controlled code run by J can violate secrecy.
- *Confused Deputy*: Attacker may trick trusted code to violate integrity

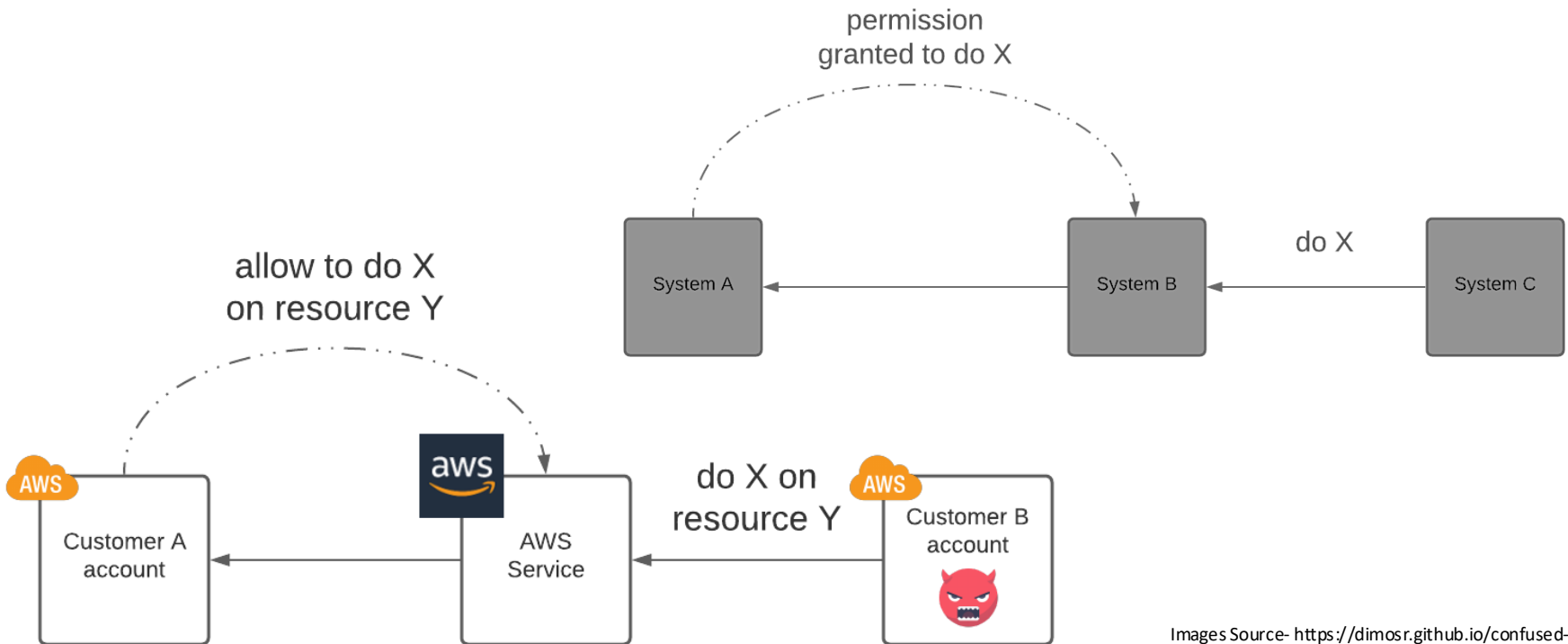
	O ₁	O ₂	O ₃
J	R	RW	RW
S ₂	-	R	RW
S ₃	-	R	RW

Trusted Processes

- *Trojan Horse*: Attacker controlled code run by J can violate secrecy.
 - Disguises itself as legitimate code
 - Once installed (or inside the user's system), can perform actions that the legitimate user can.

Trusted Processes

- *Confused Deputy*: Attacker may trick trusted code to violate integrity



Images Source- <https://dimosr.github.io/confused-deputy/>

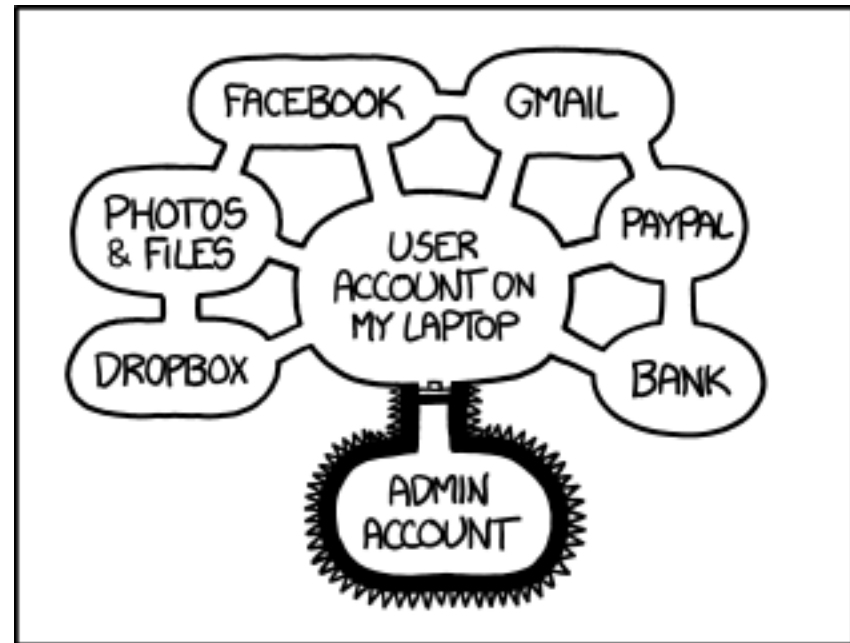
<https://docs.aws.amazon.com/IAM/latest/UserGuide/confused-deputy.html>

Protection vs. Security

- Protection
 - Security goals met under *trusted* processes
 - Protects against an error by a non-malicious entity
- Security
 - Security goals met under *potentially malicious* processes
 - Protects against any malicious entity
 - Hence, For J:
 - Non-malicious process shouldn't leak the private key by writing it to O_3
 - A potentially malicious process that may contain a Trojan horse that writes the private key to O_3 should not be able to do so

Do you own a computer?

- Linux/ Windows/ Mac?
- (DONOT) execute everything as the admin user!
 - Create a separate "standard" user. *Why?*
- *Caveat: Still need to protect the standard user account.*



IF SOMEONE STEALS MY LAPTOP WHILE I'M LOGGED IN, THEY CAN READ MY EMAIL, TAKE MY MONEY, AND IMPERSONATE ME TO MY FRIENDS, BUT AT LEAST THEY CAN'T INSTALL DRIVERS WITHOUT MY PERMISSION.

<https://xkcd.com/1200/>

Principle of Least Privilege

A system should only provide those rights needed to perform the processes function and no more.

- **Implication 1:** you want to reduce the protection domain to the smallest possible set of objects
- **Implication 2:** you want to assign the minimal set of rights to each subject
- **Caveat:** of course, you need to provide enough rights and a large enough protection domain to get the job done.

Least Privilege

- Limit permissions to those required and no more
- Restrict privilege of the process of J to prevent leaks
 - Cannot R/W O3

	O ₁	O ₂	O ₃
J	R	RW	-
S ₂	-	R	-
S ₃	-	R	RW