

# **DSCI 519: Foundations and Policy for Information Security**

Introduction to Characteristics of Policy,  
Reference Monitor abstraction

*Tatyana Ryutov*

# Lecture Outline

---

- Characteristics of good security policy
- Access Control policy types
  - Military and commercial policies
  - Discretionary and mandatory policies
- Access Control Matrix
  - ACL
  - Capabilities
- The Reference Monitor (RM)
- Interpreting Reference Monitor components

# Reminders

---

- Lab 1 is due on September 11<sup>th</sup> by 11:59pm
- HW1 is assigned, due on September 20<sup>th</sup> by 11:59pm
- Submit on D2L
- Do the assigned readings!

## L2.Q6



- 
- Think about all the material covered today
  - Indicate two **specific** topics/questions for review next lecture
    - focused questions/topics I can give you an answer to
  - If everything is clear, it's OK to say: “none”

# Relate Security Policy Breaches to Risk

---

- Risk Analysis and Risk Management
  - How important to enforce a policy
  - Legislation may play a role
- Risk reflects the need to enforce policy
- Want to manage risk when
  - Vulnerability exists
  - Realistic threat (+ reward for penetration)
  - Valuable information or liability for access
- “Simple” Solution: reduce vulnerability
  - But how much to trust?
  - Must provide appropriate **assurance (trust)**

# Risk Management Approach

---

- A number of risk management frameworks exist
  - e.g., NIST <https://csrc.nist.gov/Projects/Risk-Management/rmf-overview>



## 1. Risk Assessment

- Classify assets on the basis of risk

**Risk = Threat x Vulnerability x Cost (or Expected Loss)**

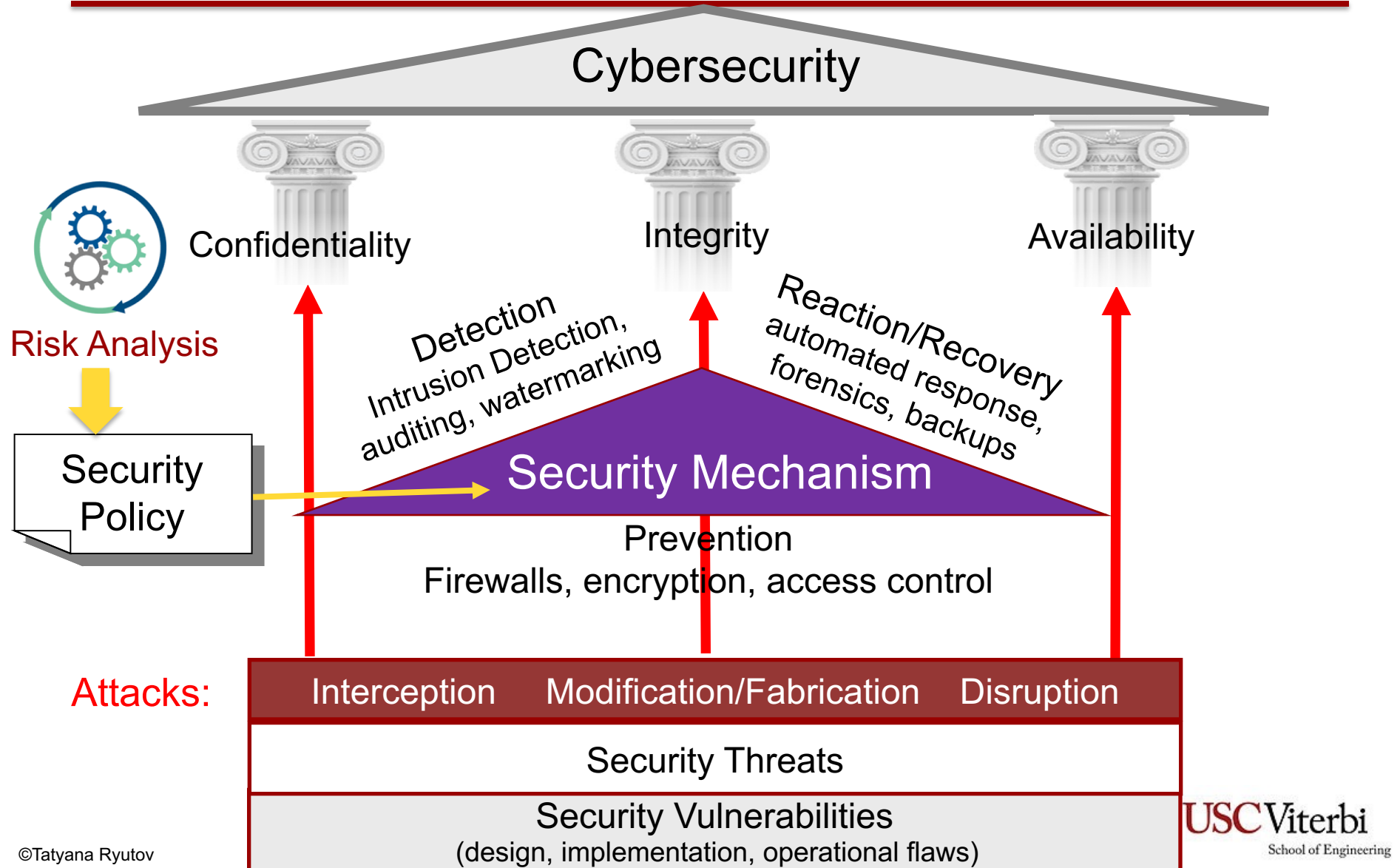
## 2. Risk Mitigation

- Choose one of the options: accept, transfer, limit, or avoid the risk

## 3. Evaluation

- Ensure that the option chosen for each asset
  - implemented
  - performs as designed in compliance with security policies

# Information Security: Summary



# Current Solutions Not Working

- Cyber infrastructure is intrinsically insecure
  - Based on fundamentally flawed architecture
- Dependence on “Industry Best Practices”
  - Focus on mitigating vulnerabilities
    - Defense in depth
    - Endless patching
    - Surveillance (e.g., IDS)
    - Trivially bypassable controls
- But what about determined adversary?
- Subversion is primary tool of choice
  - Trojan horse and trap door
    - No serious chance of being able to find or attribute





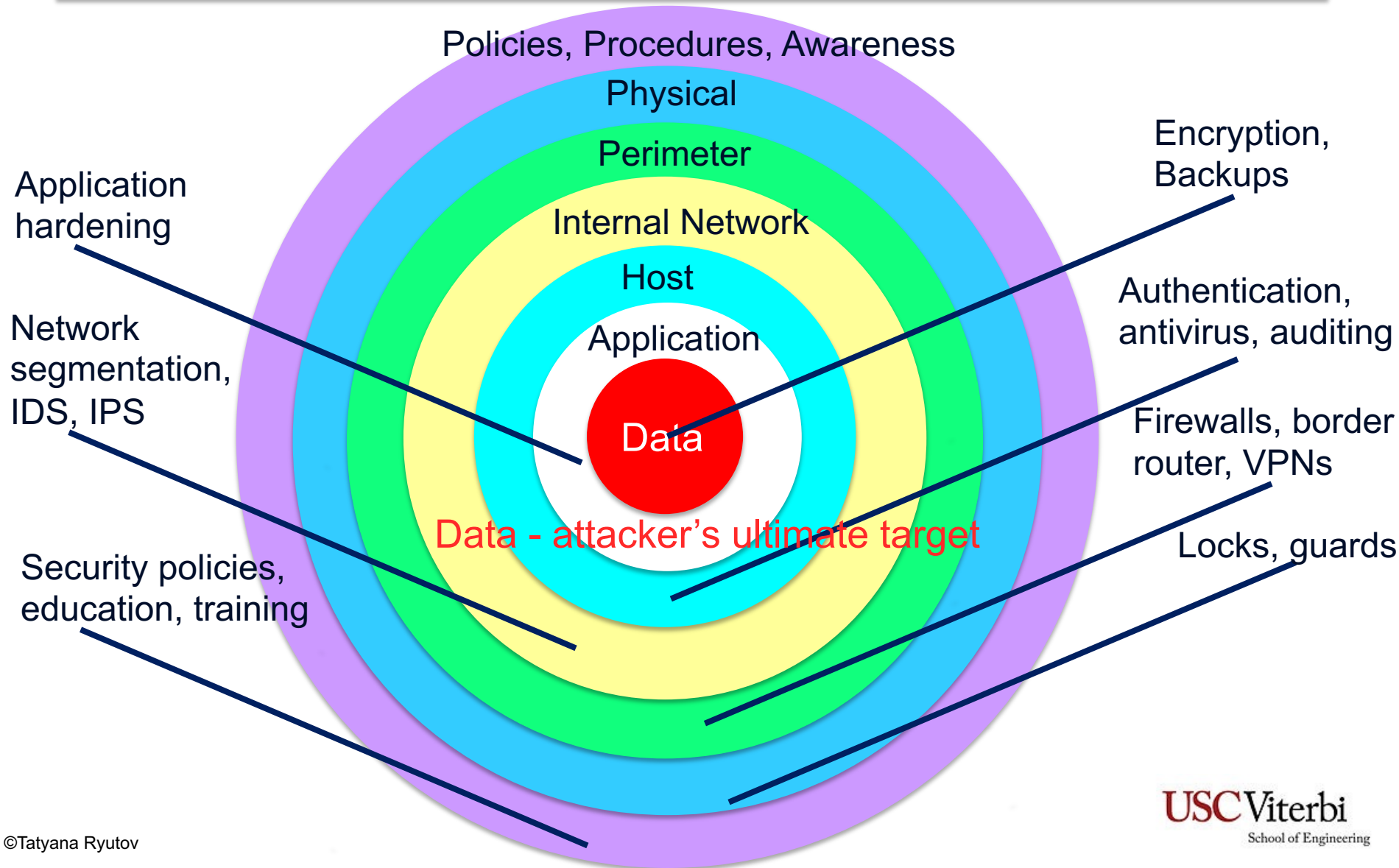
# Result of Using Current “Industry Best Practices”

---

- Uncountable system vulnerabilities
- Endless patching
- Expensive and (nearly) useless security add-ons
- Focus on fixing the wrong things
- Continual losses to individuals, business, and government
- Specific technologies are getting better, but the fundamental problems are not being solved
- **Need a paradigm shift: build trusted systems that are resistant to subversion (Trojan Horse and Back Door)!**



# Current Approach: “Defense in Depth”



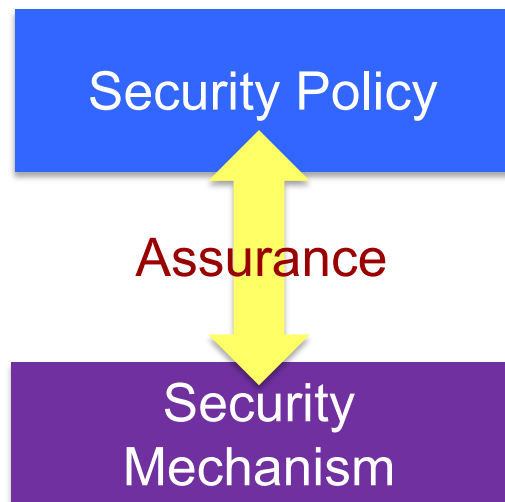
# What is a Trusted System?

- Professionals prefer to speak of **trusted** instead of secure OS
    - “Secure”: absolute
    - “Trusted”: meets the necessary security requirements
      - May not be completely secure – degrees of trust
      - Needs to justify the user’s confidence
- trust level in  $[0, 1)$

Secure	Trusted
<i>Either-or</i> : something either is or is not secure	<i>Graded</i> : There are degrees of “trustworthiness”
Property of <i>presenter</i>	Property of <i>receiver</i>
<i>Asserted</i> based on product characteristics	<i>Judged</i> based on evidence and analysis
<i>Absolute</i> : not qualified as to how, where, when, or by whom used	<i>Relative</i> : viewed in context of use
A <i>goal</i>	A <i>characteristic</i>

# How do we determine “trust”?

- **Security policy**
  - Defines what is and is not authorized
  - Overall strategy, holds everything together
    - Effectively a definition of security for a system
  - Confidentiality + Integrity = access control
- **Security Mechanism**
  - Method/tool designed to prevent, detect or recover from a security attack
  - Enforces security policy
- **Assurance**
  - Determining how much to trust a system to enforce policy, based on evidence

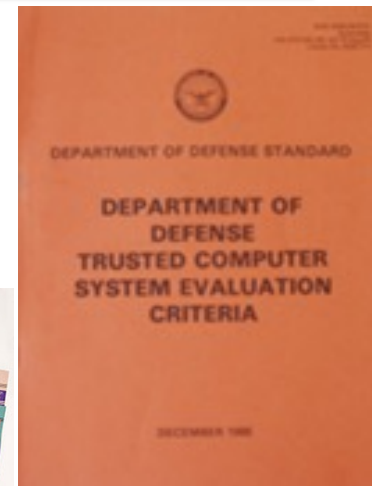


# How can we assign trust to a system?

- Trusted system evaluation criteria TCSEC (1983)
  - Known as Orange Book, DoD 5200.28-STD
    - Written by skilled computer security practitioners
    - First widely used evaluation criteria, withdrawn in 1999
    - Set high technical standards for system evaluation
- Reference monitor – unifying approach
- Trust rating classes:
  - **D**: Minimal protection
  - **C** (C1,C2): Discretionary protection
  - **B** (B1, B2, B3): Mandatory protection
  - **A** (A1): Highly-secure
- Evaluations were done by the National Security Agency
  - Intended to be objective, competent and third party
- Reports are publicly available
  - E.g., Multics meets the requirements of Class B2 <https://multicians.org/multics-fer.html>



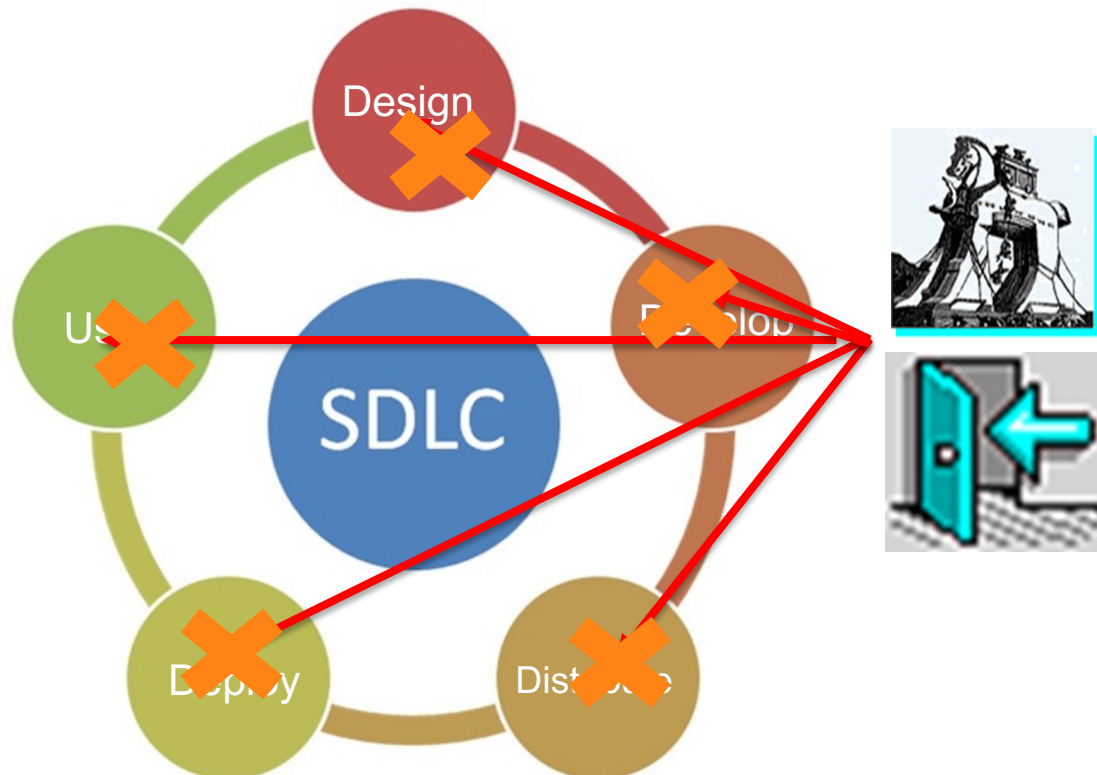
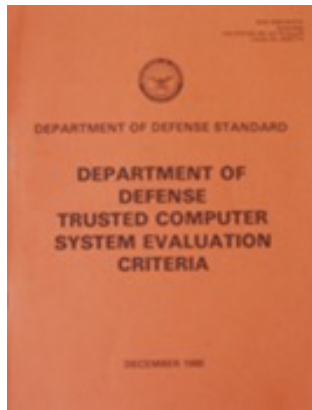
Rainbow Series



D C1 C2 B1 B2 B3 A1

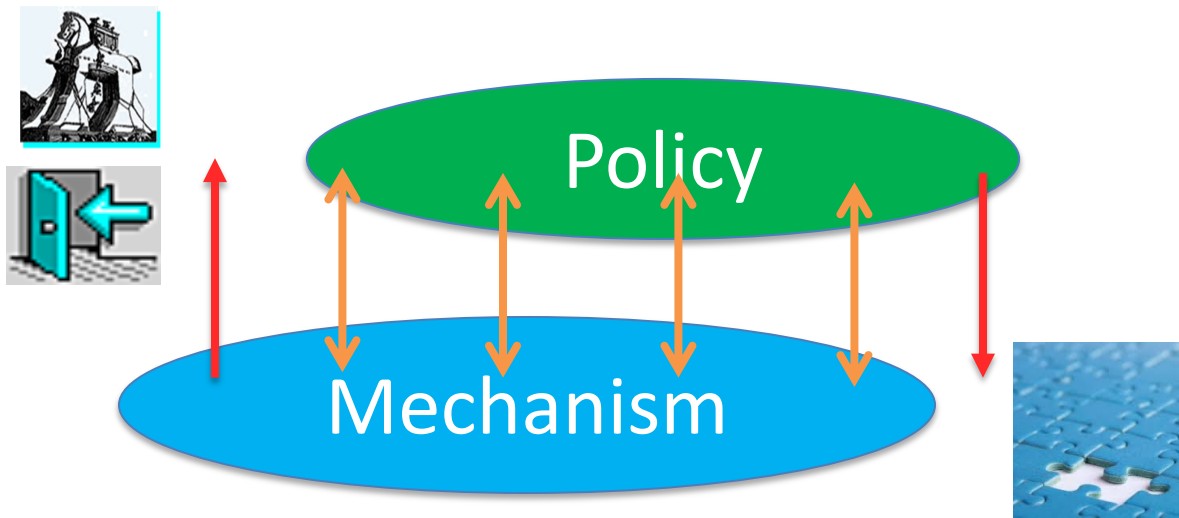
# Recall: Subversion is “Attack of Choice”

- **System subversion**—the intentional insertion of an artifice at some point during Software Development Life Cycle (SDLC)  
Can be SW, HW or firmware
- Most attacks are subversion attacks



# Mechanisms vs. Policy

- Protection measures must be traceable to policy
- Policy must be traceable to the measures
- If traceability fails, usually something breaks
  - Information is not adequately protected
  - Implementation contains superfluous components
- However, mechanism should not define policy
  - E.g., does an unlocked door mean entry is permitted?
  - The state of the lock does not define the policy



# Recall: Trap Door Example

---

- Code inserted by programmer to bypass normal check

```
while (TRUE) {  
    printf("login: ");  
    get_string(name);  
    disable_echoing( );  
    printf("password: ");  
    get_string(password);  
    enable_echoing( );  
    v = check_validity(name, password);  
    if (v) break;  
}  
execute_shell(name);
```

```
while (TRUE) {  
    printf("login: ");  
    get_string(name);  
    disable_echoing( );  
    printf("password: ");  
    get_string(password);  
    enable_echoing( );  
    v = check_validity(name, password);  
    if (v || strcmp(name, "zzzzz") == 0) break;  
}  
execute_shell(name);
```



# Lecture Outline

---

- Characteristics of good security policy
- Access Control policy types
  - Military and commercial policies
  - Discretionary and mandatory policies
- Access Control Matrix
  - ACL
  - Capabilities
- The Reference Monitor (RM)
- Interpreting Reference Monitor components

# Organizational Policy

---

- How to develop security policy?
  - Arises from business case
  - Perform risk assessment
  - Incorporate legal and regulatory concerns, relevant ethical standards, organizational characteristics
- A useful cyber security policy specifies
  - What information is to be protected
  - Why it is to be protected
  - Who may have what form of **access** to information
  - But not how to protect!

# Characteristics of Good Organizational Policy

---

- Be implementable and enforceable
- Be concise and easy to understand
- Prioritize efforts with respect to finite resources
- Balance protection with productivity
- State reasons why policy is needed
- Describe what is covered by the policies (scope)
- Define contacts and responsibilities
- Discuss how violations will be handled (sanctions)

# Lecture Outline

---

- Characteristics of good security policy
- Access Control policy types
  - Military and commercial policies
  - Discretionary and mandatory policies
- Access Control Matrix
  - ACL
  - Capabilities
- The Reference Monitor (RM)
- Interpreting Reference Monitor components

# Military Security Policy

---

- Basis of many OS security policies
- Based on protecting classified information
- Hierarchical Levels:
  - Top Secret (most sensitive), Secret, Confidential, Unclassified (least sensitive)
- Limited by the **Need-to-Know** rule: access is allowed only to subjects who need to know data to perform job
- **Compartments** - independent and non-comparable (unordered)
  - E.g., “Crypto” category for protecting keys
  - E.g., HR, Engineering



# Commercial Security Policies

---

- Worried about espionage
- Degrees of sensitivity:
  - Public
  - Proprietary
  - Internal
- No dominance function for most commercial policies since no formal clearance is needed
- Integrity and availability are just, if not more, important than confidentiality

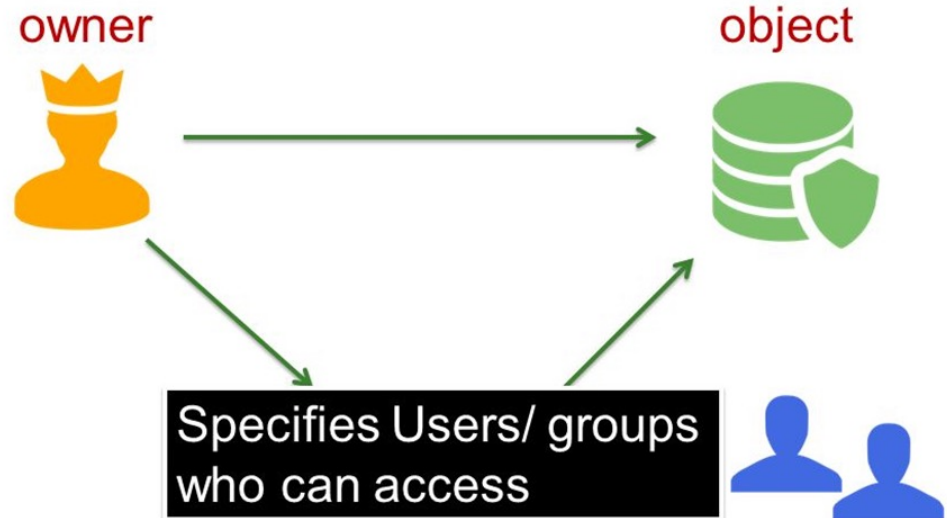
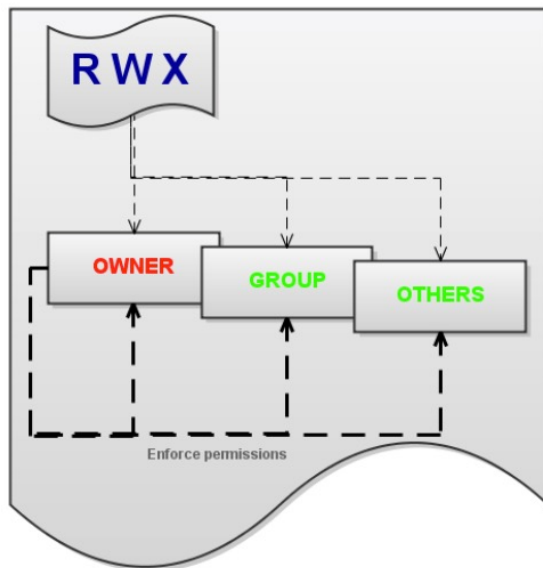
# Access Control Policy Types

---

- **Discretionary access control (DAC) policy**
  - Support for commercial security policies
  - Object access based on subject (or group) identity
  - Policy permits users (subjects) to use own discretion to grant access
- **Mandatory access control (MAC) policy**
  - Often called “non-discretionary” policy
  - Support for military security policies
  - Access based on sensitivity of information contained in objects
  - Information belongs to a security domain (has “label”)
  - Subjects need formal authorization to access domain (i.e., clearance and need to know)

# How does DAC work?

- Concept of owner is important
  - Can set permissions for others at their own discretion
  - Copied objects become owned by the copier
- Permissions are given based on identity and/or group



## DAC examples





# How does MAC work?

---

- Controlled by a central authority
- Information belongs to an organization, rather than individual members of it
- Defines levels of access
- Owner cannot change permissions/levels of access

## Low Assurance MAC examples



Pitbull

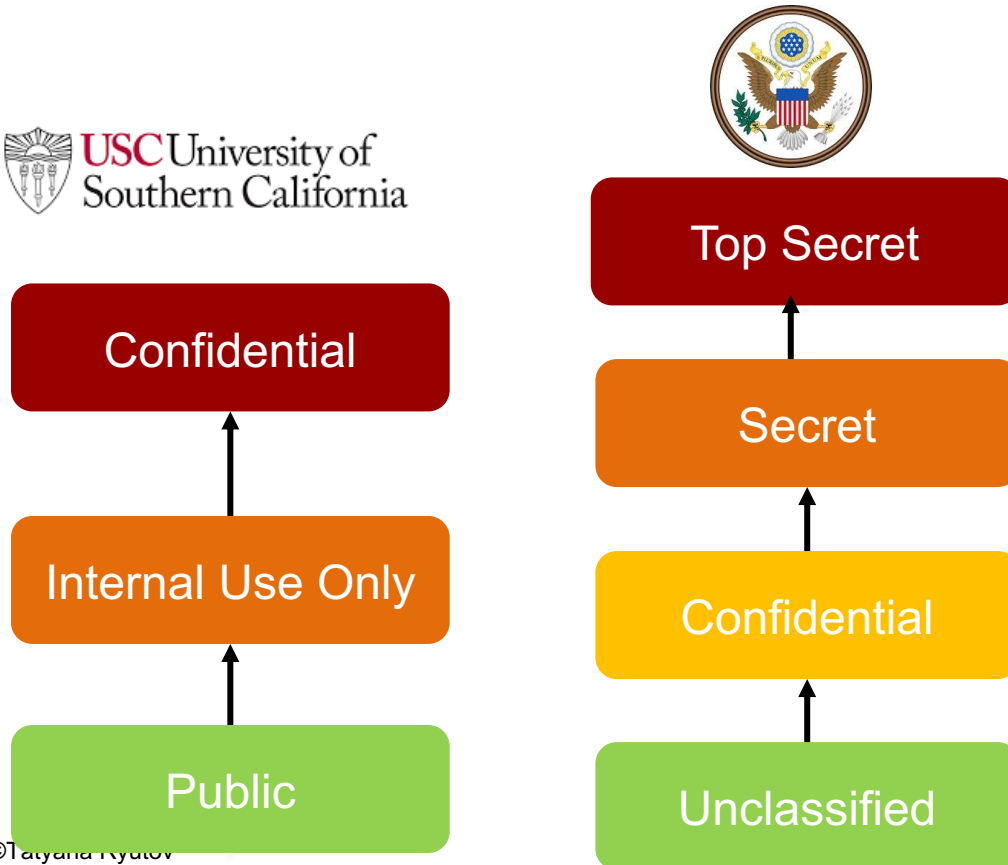
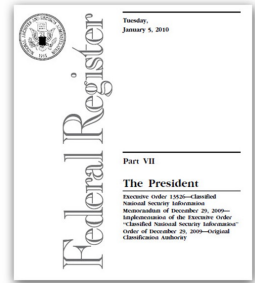


Selinux



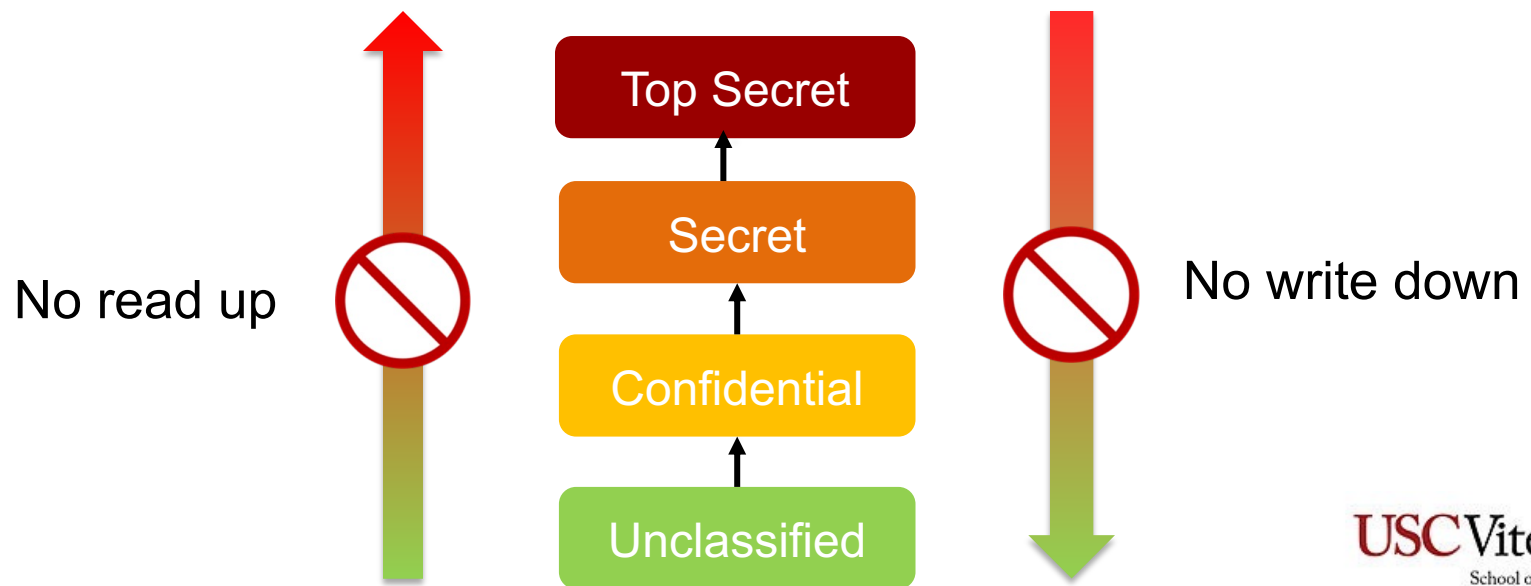
# Mandatory Access Control (MAC)

- Basis of many OS security policies
- Based on protecting classified information
- Levels: Top Secret, Secret, Confidential, Unclassified
- Example: U. S. Classified Information Executive Order 13526

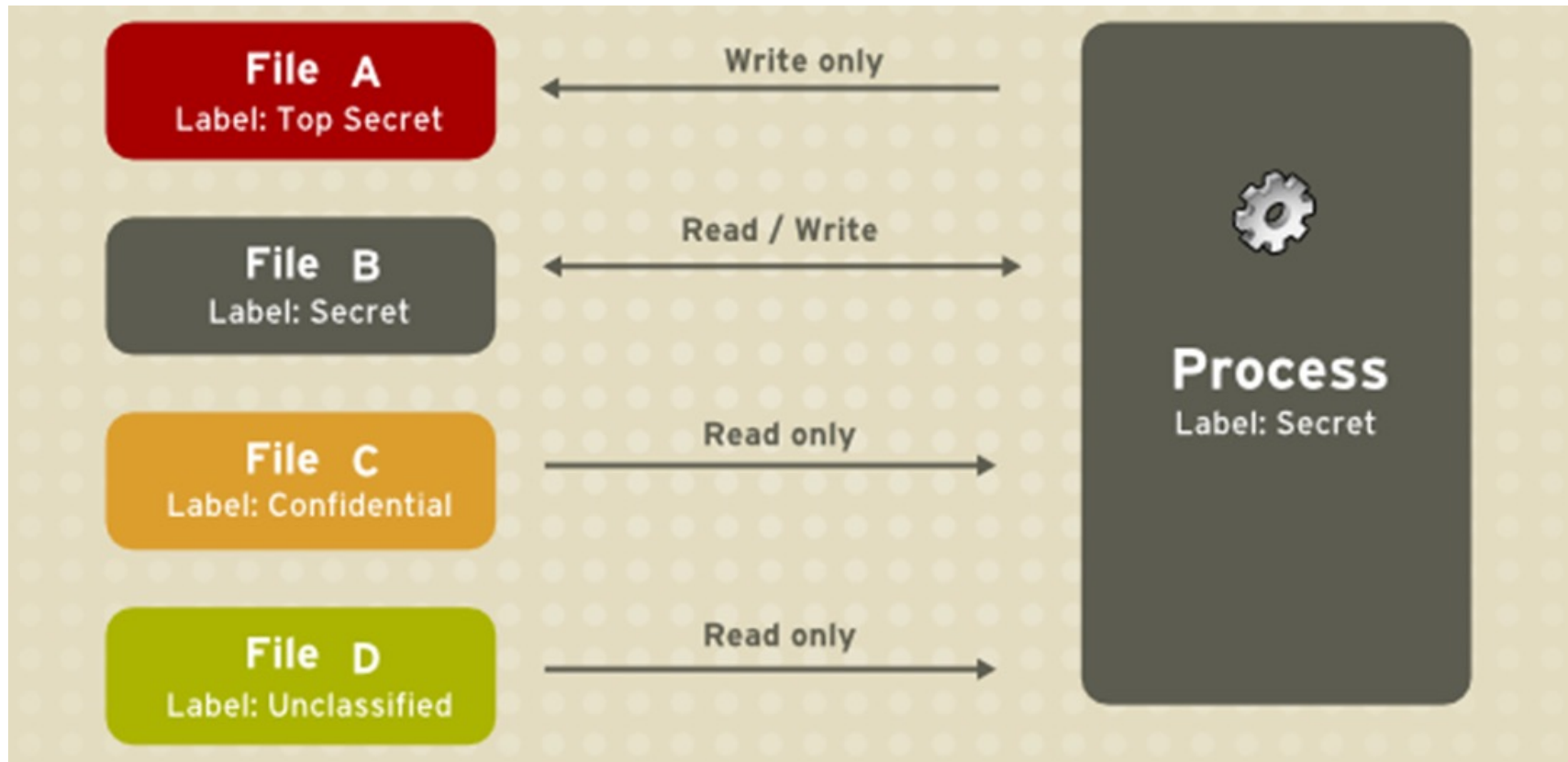


# MAC Example: Bell-LaPadula (BLP) Model

- A state machine model that enforces the confidentiality
  - Can be applied to integrity just modify the rules
- Considers **semantics** of the information
- Summarized in two axioms:
  1. No user may read information classified above his/her clearance level (“No read up”)
  2. No user may lower the classification of information (“No write down”)



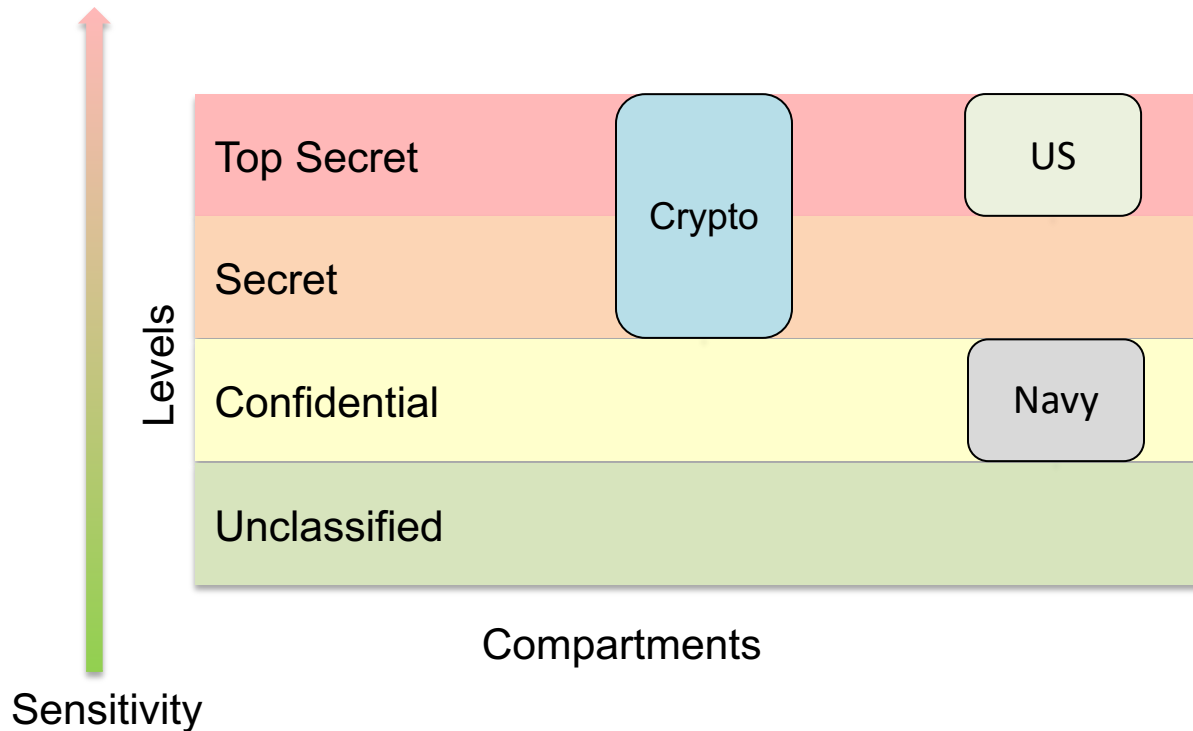
# Example: BLP Rules



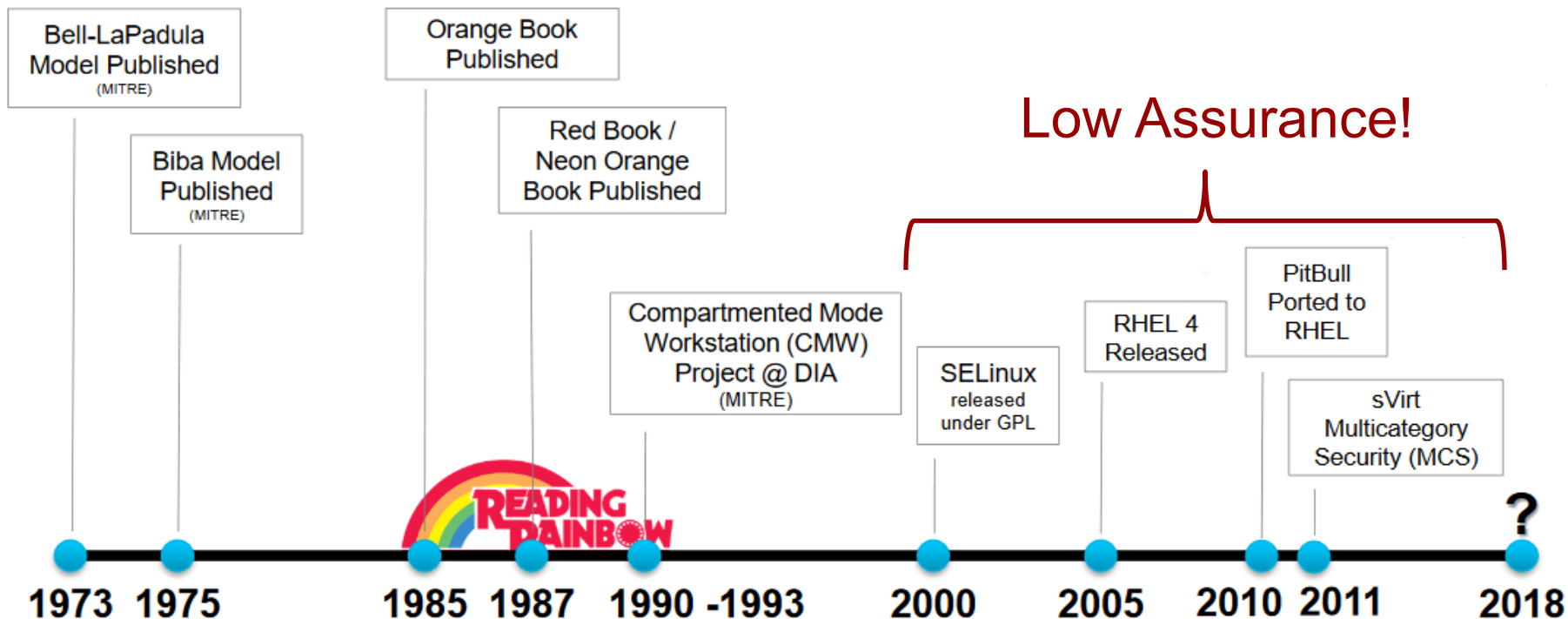
No read up, no write down

# MAC: Levels and Compartments

- Information access is limited by “need to know”
- Each piece of classified information can be associated with one or more **compartments**



# History of MAC



Currently Available  
Commercial Operating  
Systems with Mandatory  
Access Control and  
Multilevel Security  
(MLS) Support

**BAE SYSTEMS**

Honeywell SCOMP / BAE Systems STOP OS



Solaris (SunOS MLS, CMW, Trusted Solaris, Solaris with Trusted Extensions)



Red Hat (4.x + with SELinux)

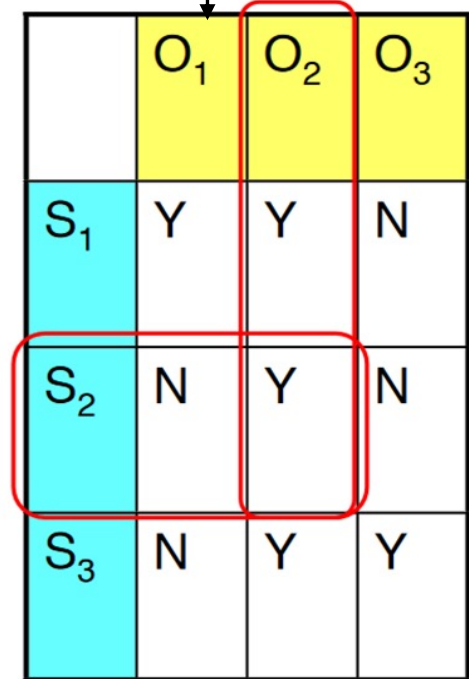


PitBull (AIX, Solaris, Linux)

# MAC Example: SELinux Type Enforcement

- MAC Policy
    - Subjects and Objects Labeled
  - Access Matrix Policy
    - Processes with subject label
    - Can access object of object label
    - If operations in matrix cell allow
  - Focus: Least Privilege
    - Just permissions necessary
- no rules that say "/bin/bash can execute /bin/ls"

Domain of objects



The diagram shows an access matrix with 3 rows and 4 columns. The first column contains subject labels S<sub>1</sub>, S<sub>2</sub>, and S<sub>3</sub>, highlighted in light blue. The next three columns contain object labels O<sub>1</sub>, O<sub>2</sub>, and O<sub>3</sub>, highlighted in yellow. The matrix cells contain 'Y' for 'yes' and 'N' for 'no'. A red rectangle highlights the intersection of subjects S<sub>1</sub> and S<sub>2</sub> with objects O<sub>2</sub> and O<sub>3</sub>. Arrows point from the text 'Domain of objects' to the top of the matrix and 'Domain of subjects' to the bottom of the matrix.

	O <sub>1</sub>	O <sub>2</sub>	O <sub>3</sub>
S <sub>1</sub>	Y	Y	N
S <sub>2</sub>	N	Y	N
S <sub>3</sub>	N	Y	Y

Domain of subjects

# MAC: Modes of Operation

---

- First came the **Dedicated System** mode
  - No separation of data within a single system
  - All users can access ALL data
  - No transfer of data between different systems
- Then came the **System High** mode
  - Logical separation of data within a single system
  - All users must be cleared to the highest level
  - Users can access SOME data, based on their need to know
  - Limited transfer of data between different systems
- Finally came the hard part – **Multilevel mode**
  - Store data at multiple classification levels on a system
  - Logical separation of multi-level data
  - Not all users are cleared to all data
  - All users can access SOME data, based on their clearance and “need to know”
  - Transfer of data between different systems



# Dedicated System Mode

---

- All system users have:
  - Clearance for all information (no MAC)
  - Need-to-know for all information (no DAC)
- Different enclaves have own dedicated system
  - Separated by “air gaps”
- No access control enforcement by system hardware & software
- What are the downsides of the dedicated mode?
- Can we use the same computer system to process different classes of information?



# Dedicated System Mode Today

---

## Air-Gapped Network



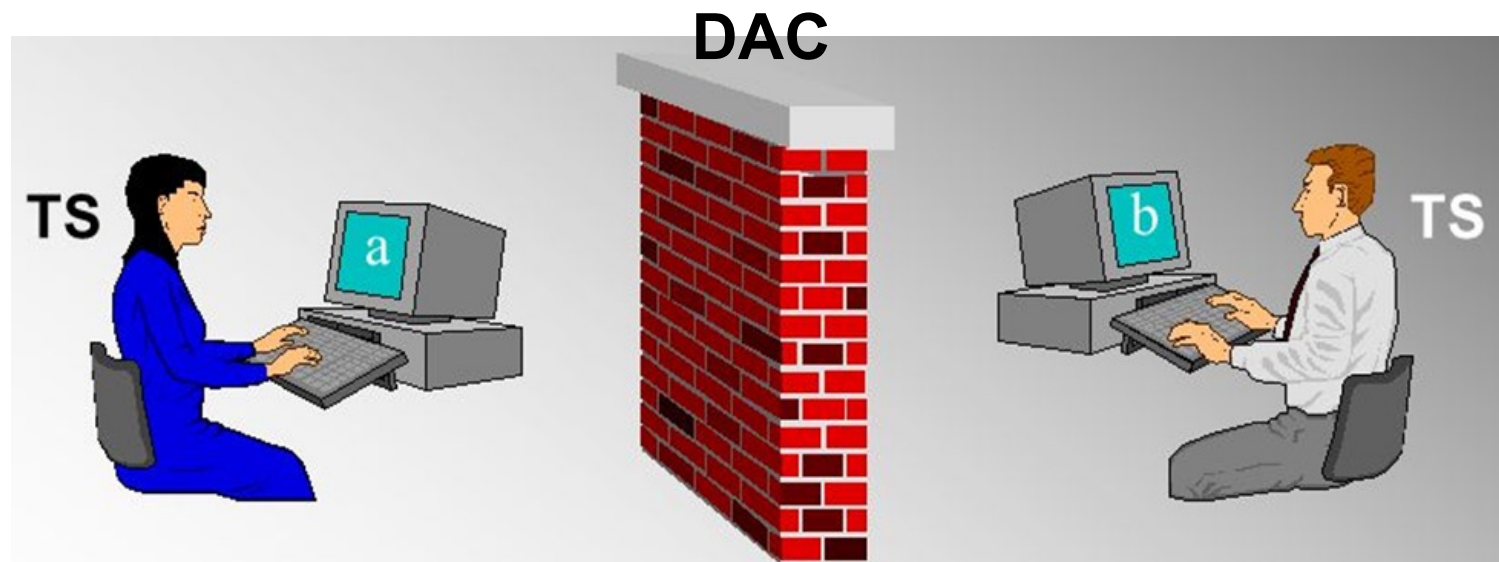
## The Internet



*Air Gap*

# System High Mode

- All system users have clearance for all information (no MAC)
  - Not all users have need-to-know (has DAC policy)
- Different enclaves have own system high system
- DAC, but not MAC, enforced by computer



# System High Mode Today

---

DAC



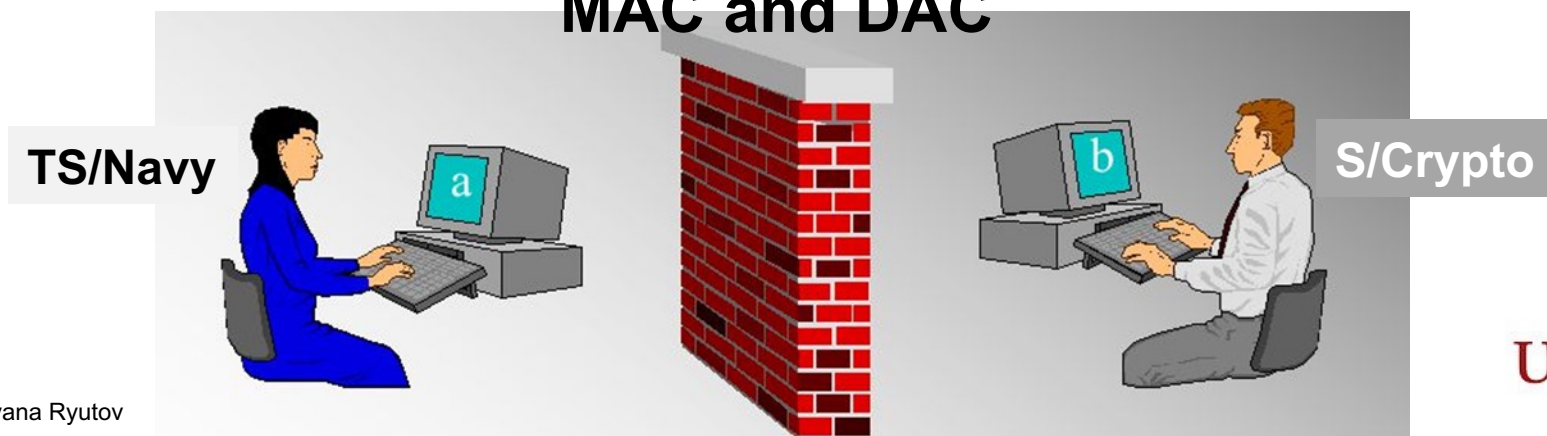
Low Assurance MAC



# Multilevel Mode

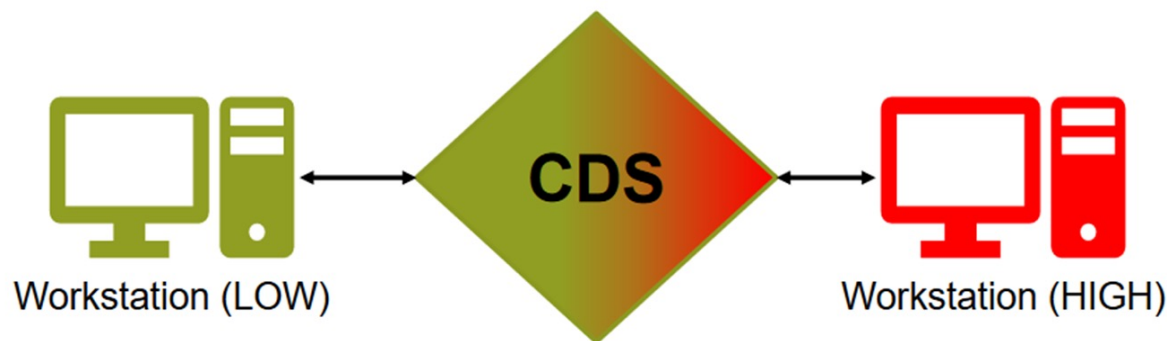
- Replace the air gap with MAC on a computer
- Users do not all have clearance for all information
- Can transfer Information between computers
  - MAC enforcement ensures digital data meets MAC policy
- DAC is as trustworthy as it is for system high
- MAC enforcement is a huge additional demand
  - Historically multilevel security (MLS) drove assurance
  - Most attempts for MLS failed due to lack of assurance
- MAC assurance was reason reference monitor was created

## MAC and DAC



# Multilevel Mode Today

- A cross-domain solution (CDS) transfer information between two or more differing security domains, e.g., System High and System Low
  - System High is the highest Sensitivity Level and Category being processed on the system
  - System Low is the lowest Sensitivity Level being processed on the system
  - Example: network pump
- Issue: **low assurance**



# Lecture Outline

---

- Characteristics of good security policy
- Access Control policy types
  - Military and commercial policies
  - Discretionary and mandatory policies
- **Access Control Matrix**
  - ACL
  - Capabilities
- The Reference Monitor (RM)
- Interpreting Reference Monitor components

# Access Matrix as Auth Database

---

- Lampson's access matrix is a model of an authorization database
  - For DAC policy
- Interpretation of reference monitor authorization database
  - Use access matrix for any expressible policy [Bishop]
  - Primary access abstraction in computer security
- Components:
  - Set of objects **O**
  - Set of subjects **S**
  - Set of rights **R** for each entry in matrix
    - Authorized DAC access right can be arbitrary function
  - Access matrix **A**
    - Entry **a[s,o]** has authorization **r** of subject **s** to object **o**
  - Triple **[S,O,A]** represent system **protection states**
    - A **secure state** is one enforcing organizational policy



# DAC Access Matrix Illustration

---

**Access Matrix A**

Subjects \ Objects	Object 1	Object 2	Object 3	Object 4
Bob Process	read	read, write		write
Flo Process	read, write, <b>own</b>	write		
Alice Process	read	read	read	read, write
Dan Process	read		read, write, <b>own</b>	read

$a[\text{Alice process, Object 1}] = \text{read}$

Organization policy: Alice authorized to read information stored in Object 1

# Access Control List (ACL)

---

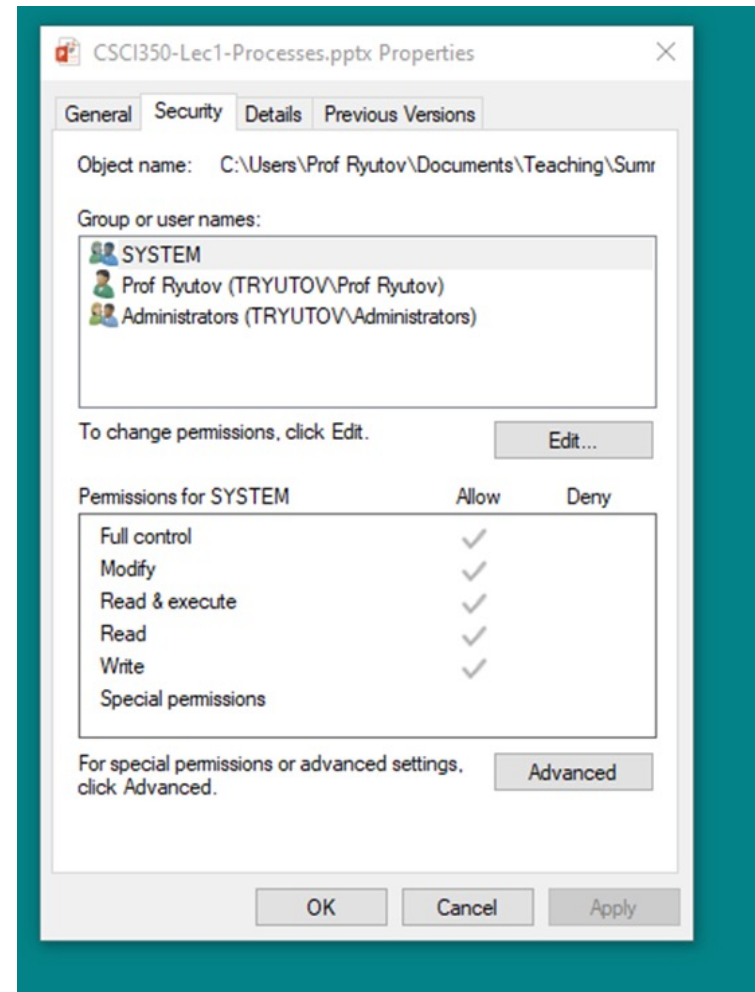
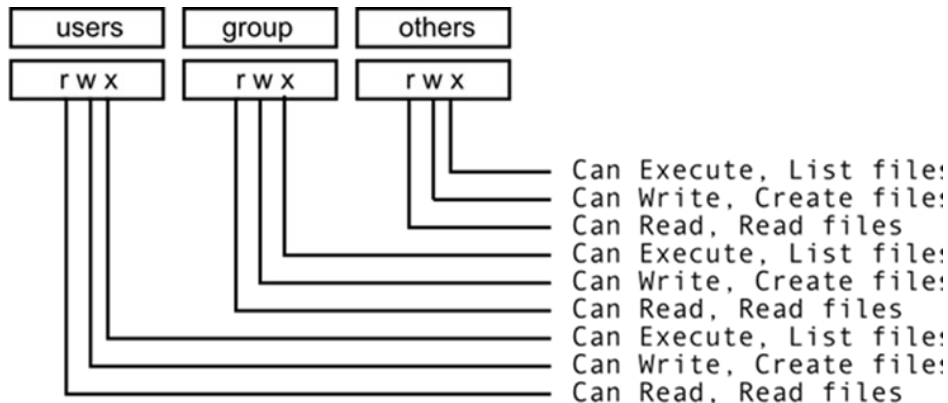
## Access Control List for Object 1

Subjects	Object 1
Bob Process	read
Flo Process	read, write, <b>own</b>
Alice Process	read
Dan Process	read



# Example: Linux and Windows ACL

```
dave@howtogeek:~/work$ ls -l
total 80
drwxr-xr-x 2 dave dave 4096 Aug 23 08:02 archive
-rw-rw-r-- 1 dave dave 780 Aug 20 11:11 command_cls.page
-rw-rw-r-- 1 dave dave 828 Aug 20 11:11 command_exit.page
-rw-rw-r-- 1 dave dave 819 Aug 20 11:11 command_gc.page
-rw-rw-r-- 1 dave dave 799 Aug 20 11:11 command_osm.page
-rw-rw-r-- 1 dave dave 829 Aug 20 11:11 command_quit.page
-rw-rw-r-- 1 dave dave 832 Aug 20 11:11 command_satellite.page
-rw-rw-r-- 1 dave dave 811 Aug 20 11:11 command_street.page
-rw-rw-r-- 1 dave dave 28127 Aug 20 11:11 GC Help.mm
-rwxrwxr-x 1 dave dave 46 Aug 20 11:11 mh.sh
-rw-rw-r-- 1 dave dave 16149 Aug 20 11:11 window_tool.page
dave@howtogeek:~/work$
```



# Capability

---

## Access Capability for Bob process

Objects	Object1	Object 2	Object 4
Bob Process	read	read, write	write



# Example: IoT Capability Token

- JSON format is used to represent the capability token

Identifier  
Issued-time  
Issuer  
Subject  
Device  
Signature  
Access rights

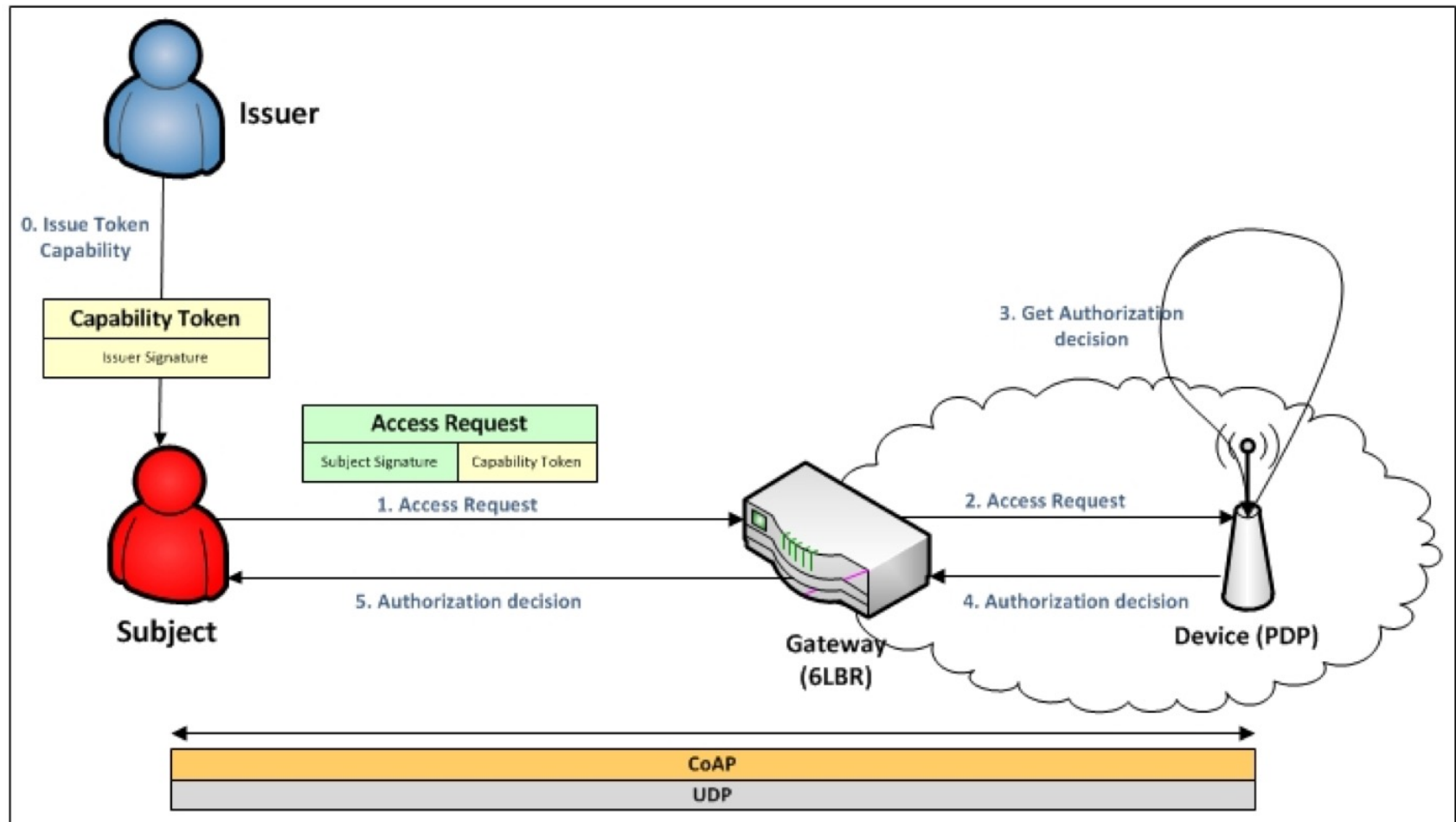
```
{
  "id": "0h7be34m_0q2cx-7",
  "ii": 1369300359,
  "is": "jara@um.es",
  "su": "lnnH3/IYZz/pqBbSud+JOyMtNCM=g2o3XRd/3r7iZSjpIIX9BRRULtc=",
  "de": "coap://aaaa::2/",
  "si": "uPTor1jxykFQUGxXnRVRm0l+uZM=kebPXS9VERYSyX3VFeHH9gW/yQI=",
  "ar": [
    {
      "ac": "GET",
      "re": "temperature",
      "f": 1,
      "co": [
        {
          "t": 5,
          "v": 25,
          "u": "Cel"
        },
        {
          "t": 6,
          "v": 21,
          "u": "Cel"
        }
      ]
    }
  ],
  "nb": 1369300359,
  "na": 1369300500
}
```

Action  
Resource  
Condition flag  
Conditions

Type  
Value  
Unit

Not before  
Not after

# Example: Distributed Capability-based Access Control



# ACL vs. Capability Lists

---

- ACL
  - Require authentication of subjects
    - Require unforgeability of authentication
  - Better access review on a per-object basis
  - Better revocation facilities on a per-object basis
- Capability lists
  - Do not require authentication of subjects
    - Require unforgeability and control of propagation of capabilities
  - Better review on a per-subject basis
  - Better revocation facilities on a per-subject basis

# Capability List (C-List)

---

**Access Capability List for Bob process**

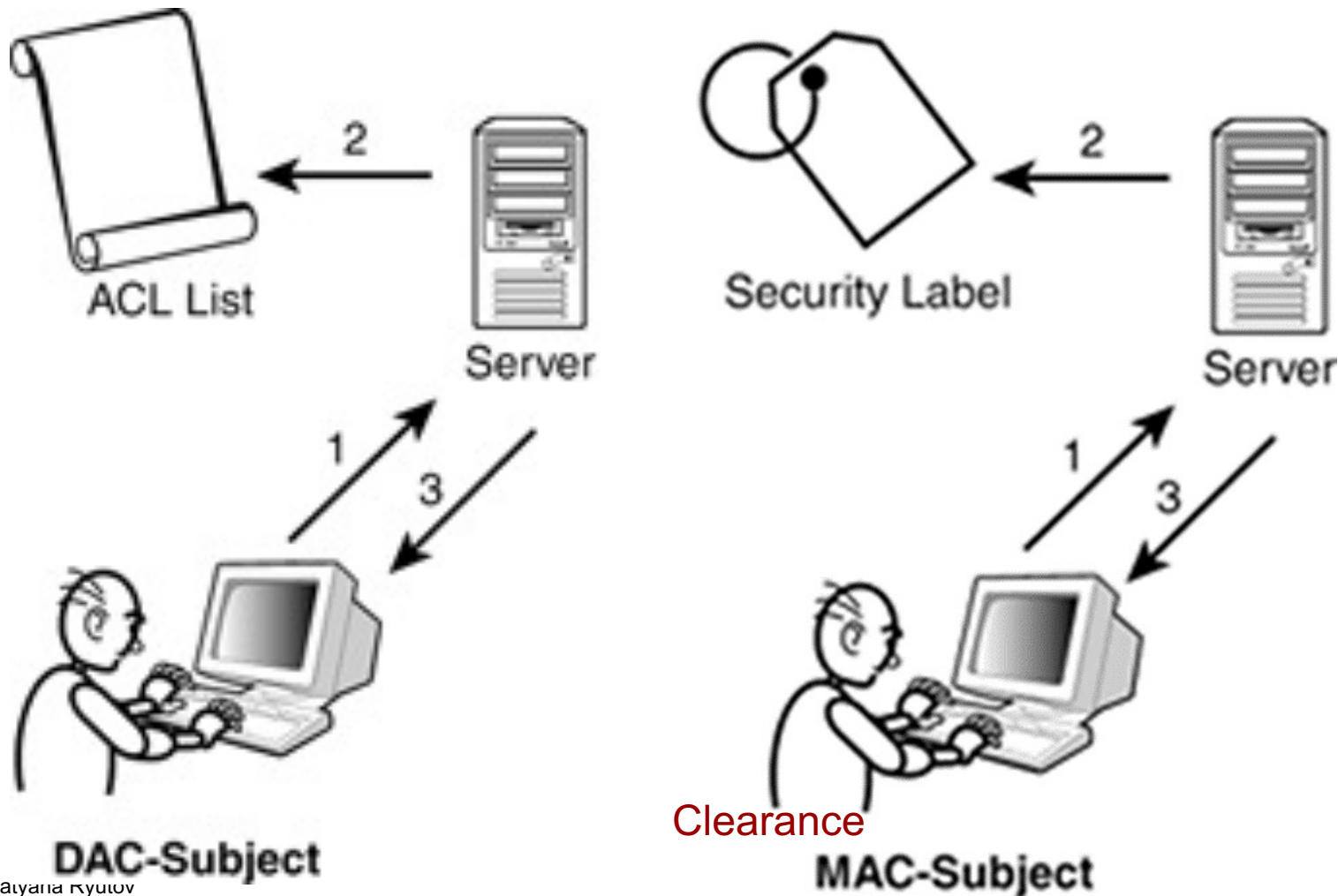
Objects	Object1	Object 2	Object 4
	read	read, write	write





# MAC and DAC in action

Request to read file X on behalf of Bob



# Reference Monitor (RM)

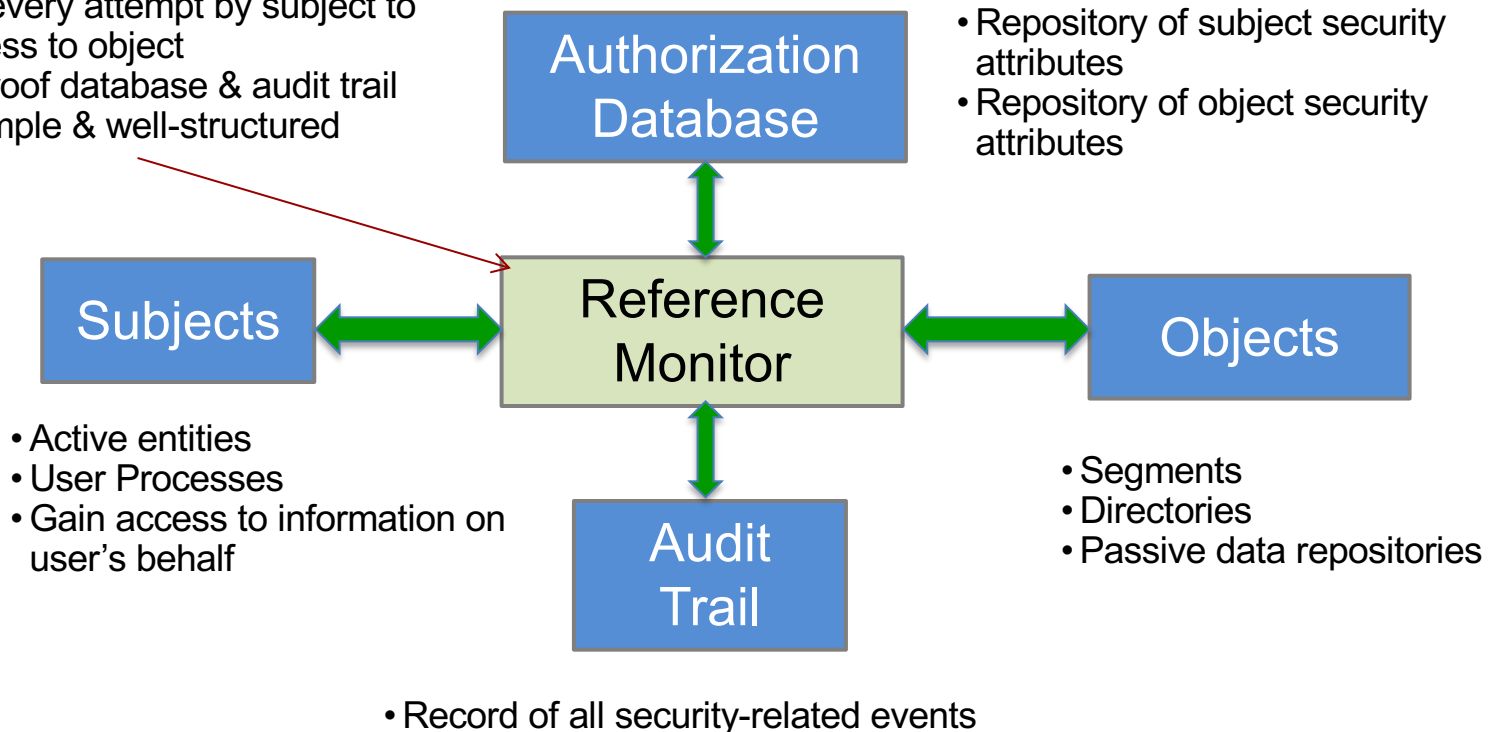
---

- The fundamental concept that makes cybersecurity a field of engineering rather than just an ad hoc set of “best practices”, e.g., “defense in depth”
- **Reference Monitor** - **abstract** access control policy enforcement mechanism
  - Not a real machine
- Key components:
  - **Subjects**: active entities, e.g., people or programs
  - **Objects**: passive entities, e.g., docs or segments
  - **Authorization database**: repository of subject and object security attributes
  - **Audit database**: for individual accountability
- RM mediates access attempts by subjects to objects based on authorization DB
- **A reference validation mechanism (RVM)** is an implementation of the reference monitor concept

# Basic Reference Monitor Abstraction

- Conceptual access control abstraction
- RM does not specify a particular access control policy

- Enforces security policy
- Mediate every attempt by subject to gain access to object
- Tamperproof database & audit trail
- Small, simple & well-structured



# Reference Monitor (RM) Functions

---

- Two classes
  - Reference and Authorization
- Reference functions
  - Subject wants to access an object – is it OK?
- Authorization functions
  - Change in authorization data base
  - E.g., add a new user

# RM Audit File

---

- Primary tool for enforcing user *accountability*
- Can help debug security policy-related problems
- Needs to include identity of user initiating event
- Often includes date and time of user action
- Identifies:
  - Type of event
  - Success or failure of event
- Audit service may include information not from RM

# Reference Monitor (RM) Principles

---

1. Tamper proof
    - **Isolation**: protected from unauthorized alteration
    - Resists subversion and malicious software
  2. Always invoked (non-bypassable)
    - **Completeness**: every access is mediated
  3. Small and simple enough to allow assurance of its correctness
    - **Verifiability**: verified to perform its functions properly
    - Ability to implement policy with high assurance
- Defines what it means for a system to be “secure”
  - **Most system vulnerabilities can be traced to violations of one or more of the RM principles**
  - A reference validation mechanism (RVM) is implementation of RM
    - It must be tamperproof, must always be invoked and can never be bypassed, and must be small enough to be subject to analysis and testing, the completeness of which can be assured

# RM Use

---

- What do we use the RM for?
  - Organizational goal: control access of **individuals** to **information** (policy)
  - Provide evidence system is “secure” (meets policy)
  - Verify system **enforces** access control policy
- What choices must we make in order to verify that a system satisfies a policy?
  - Choose *interpretation* of an abstract policy
  - Choose *interpretation* of each RM abstraction
    - Subjects, objects, authorization database, audit trail

# Reference Monitor as Ultimate Goal

---

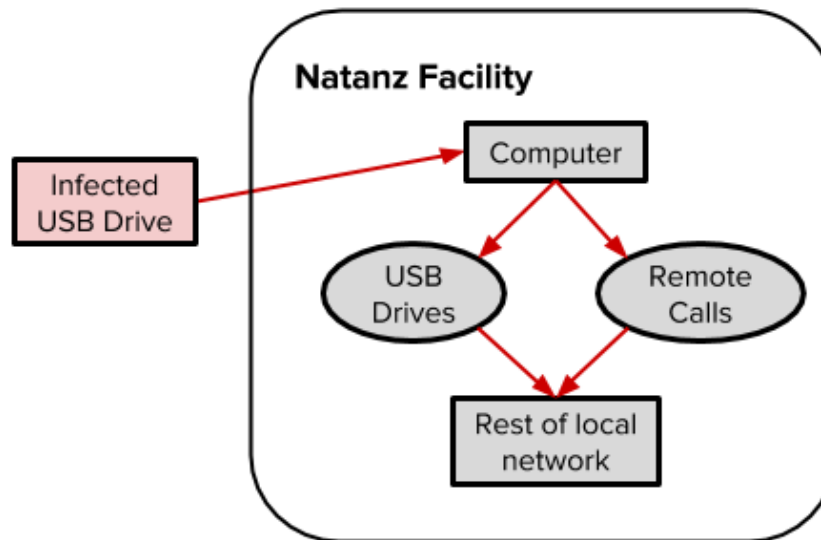
- A reference monitor is what every secure system wants to be
  - But not everyone building “secure” systems knows it
- Conscious application of the RM concept during requirements specification and system design can help ensure the security of systems





# Example: Application of RM Approach

- Consider Stuxnet attack:
  - Stuxnet spread from USB flash drives to Windows computers by exploiting a zero-day vulnerability in Windows **autorun**
- Analysis: use the RM concept as a framework to identify problems that made this attack successful
  - Violation of the verifiability
    - The reputed 40+ million lines of code in Windows is too large and complex to thoroughly analyze and test for security
  - Isolation was violated
  - It isn't clear if the access control policy was bypassed (violating the completeness) or was simply inadequate



# Reference Monitor in Computer System

---

- Most primitive portion relied on to control access
- Reference Validation Mechanism (RVM)
  - Combination of hardware & software that implements RM
  - Defined as **security kernel** (TCSEC – Orange Book)
- Use Reference Monitor to show system “secure”
  - I.e., verify that computer system meets policy
  - Access control implementation corresponds to policy
- Verifying policy based on interpretation choices
  - Choices for the “subjects” of reference monitor (RM)
  - Choices for “objects” for the RM
  - Choices for formulation of “access control policy”

# Lecture Outline

---

- Characteristics of good security policy
- Access Control policy types
  - Military and commercial policies
  - Discretionary and mandatory policies
- Access Control Matrix
  - ACL
  - Capabilities
- The Reference Monitor (RM)
- Interpreting Reference Monitor components

# Interpret Modes of Operation

---

- Dedicated
  - User has clearance for all information (no MAC)
  - User has need-to-know for all information (no DAC)
- System High
  - User has clearance for all information (no MAC)
  - User has need-to-know for some information
    - System enforced DAC need-to-know
- Multilevel
  - System enforces MAC and DAC
- **Goal:** interpretation for each mode of operation

# Interpret “Dedicated” Mode for RM

---

- Consider a computer system
- Interpretations for “dedicated” mode of operation
  - Subjects – every user has total access
    - All processes in computer (it is the active entities)
  - Objects
    - Collection of all the information in computer (passive)
  - Authorization database
    - “Policy” is all subjects authorized access to all objects
  - Audit file
    - External record of **physical** access, for identified user

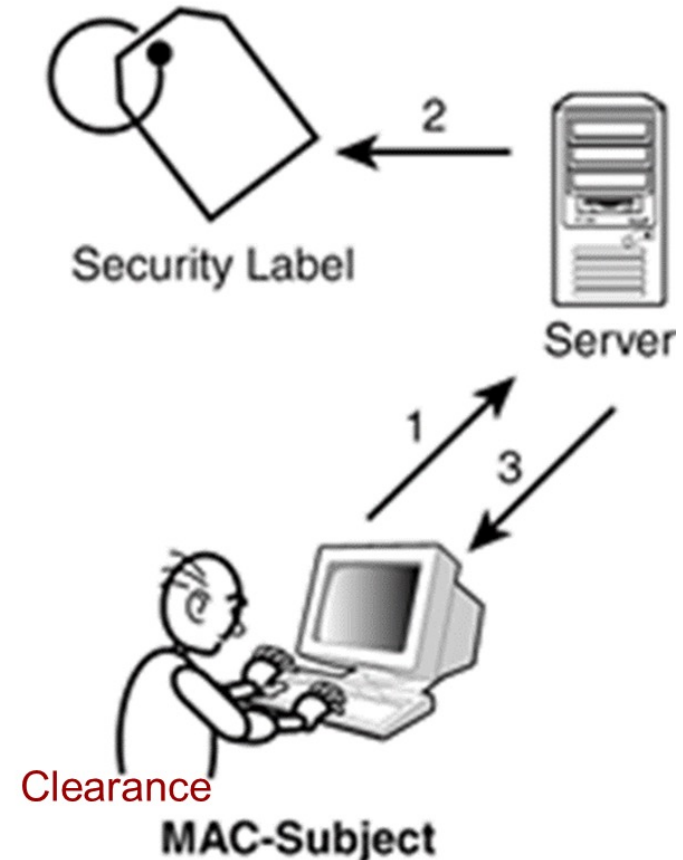
# Interpret “System High” Mode for RM

- Interpret “system high” mode of operation
  - Subjects
    - Each process for identified user
    - Surrogate for user who is subject to DAC policy
  - Objects
    - Distinct repositories or information containers, e.g., files
  - Authorization database
    - DAC authorizations for users and groups of users
  - Audit file
    - Can record object access requests, with identity of user

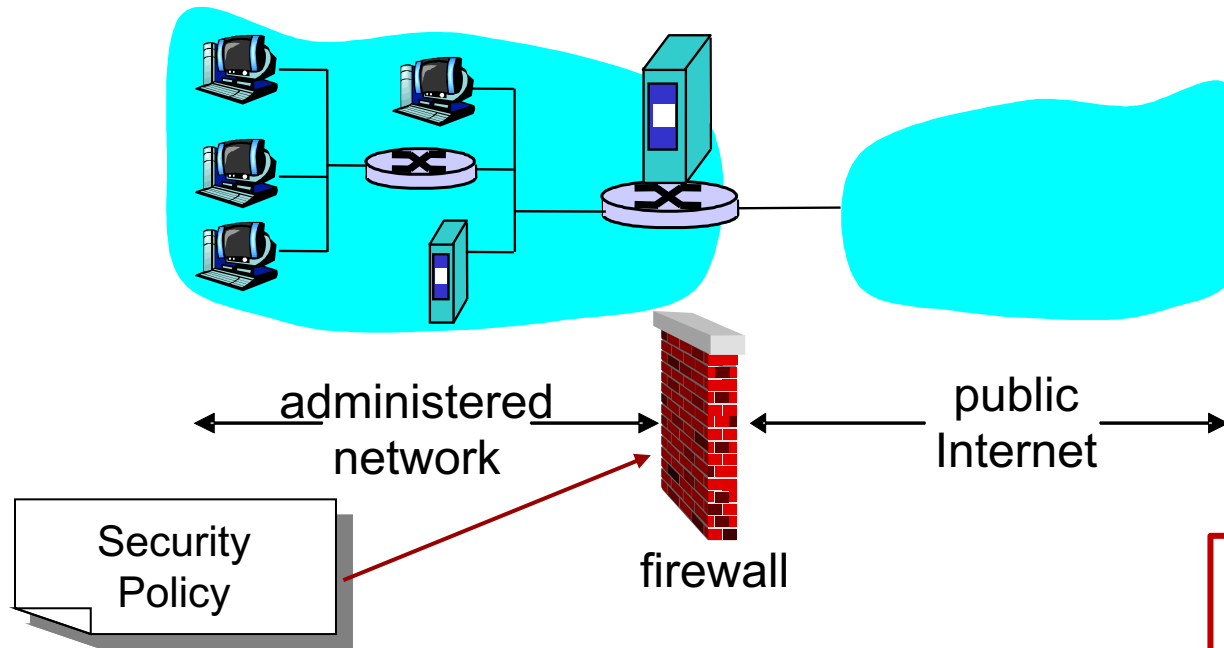


# Interpret “Multilevel” Mode for RM

- Interpretations for “multilevel” mode of operation
  - Subjects
    - Processes for users, each with a label for clearance
  - Objects
    - Information repository, with MAC security label, e.g., segment
  - Authorization database
    - DAC authorizations for identified users and groups
    - MAC authorization - clearance needed for classification
  - Audit file
    - Record object access request, with user identity & labels



# Example: Firewall as RM



## Properties:

1. Tamperproof
2. Non-bypassable
3. Verifiable

- Check Point Security vulnerabilities:
  - [https://www.cvedetails.com/vulnerability-list/vendor\\_id-136/Checkpoint.html](https://www.cvedetails.com/vulnerability-list/vendor_id-136/Checkpoint.html)
- CISCO security vulnerabilities
  - <https://us-cert.cisa.gov/ncas/current-activity/2020/03/05/cisco-releases-security-updates>



# RM Open Issues

---

- Complete mediation requires that **all** security-sensitive operations are identified
  - Often these operations are not precisely defined
- Tamperproof
  - Most systems have a trusted computing base that is too large to determine whether tampering is prevented
- Verification is the most difficult to satisfy
  - Requires designing a general algorithm to prove that an arbitrary program behaves correctly
    - Equivalent to solving the Halting problem