ACCESS CONTROL CONT.
Lecture 4

COMPSCI 702
Security for Smart-Devices

Muhammad **Rizwan** Asghar

March 9, 2021

THE UNIVERSITY OF AUCKLAND
NEW ZEALAND

SOME QUESTIONS

EXECUTABLE APP ONLY

- *Only the executable is downloaded and reverse engineering tools can produce source code*
- *Is this the reason obfuscation of source code is required?*
- There are tools that aim to recover source code from executables
- Yes, a reverse engineer should not be able to understand program data and business logic despite having source code

PIN AUTHENTICATION

- Original code:

```
if(input == “1234")
{
  //authenticate
}
```

DATA OBFUSCATION: HASH FUNCTION EXAMPLE

- A hash function is a cryptographic checksum

- Let's assume:

  hash("1234")="9876"

- The obfuscated version should be:

```
if(hash(input) == "9876")
{
  //authenticate
}
```

DATA OBFUSCATION: SPLITTING VARIABLE

- Let's assume

  v=5

- We can split **v** into two:

  a=2 and b=3 and

  replace v with a+b

- Likewise, also consider a string

  name="Ronald Rivest"

- We can split this **name** into two:

  FirstName="Ronald" and LastName="Rivest"

CONTROL FLOW EXAMPLE

- Consider the following expression:

  (a-b)^2 = a^2 + b^2 - 2ab

- The expression seems to be true always but it is not the case

- Values of a and b can be chosen to trigger integer overflow on the right side

APP EXAMPLE

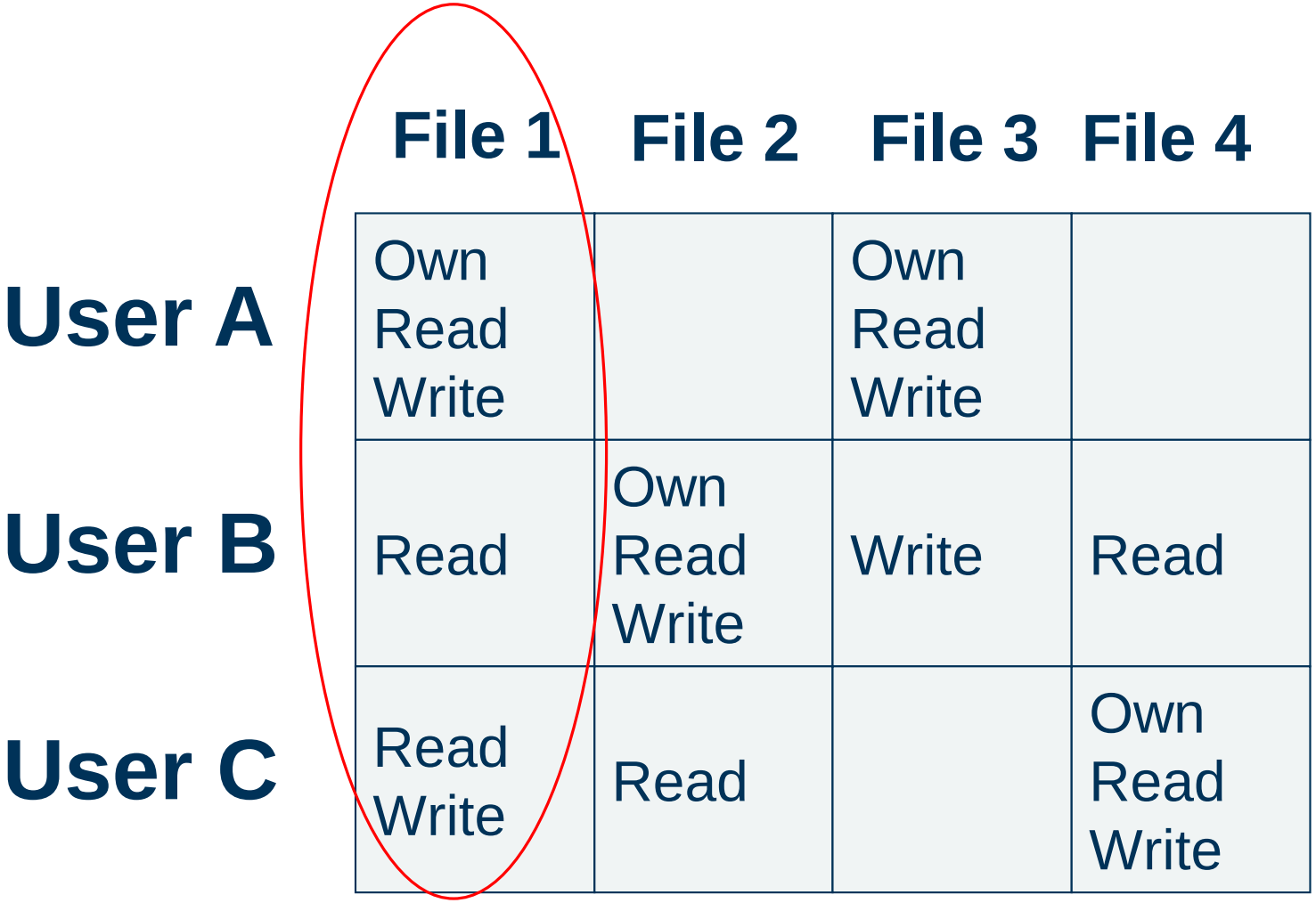
- *Can we implement an app that communicates with a micro controller like Arduino?*
- The app should be self-contained
- Other groups must be able to run it without depending on any additional hardware

ACCESS CONTROL CONT.

ACCESS CONTROL MATRIX

	File 1	File 2	File 3	File 4
User A	Own Read Write		Own Read Write	
User B	Read	Own Read Write	Write	Read
User C	Read Write	Read		Own Read Write

ACCESS CONTROL LIST



	File 1	File 2	File 3	File 4
User A	Own Read Write		Own Read Write	
User B	Read	Own Read Write	Write	Read
User C	Read Write	Read		Own Read Write

CAPABITILITY LIST

	File 1	File 2	File 3	File 4
User A	Own Read Write		Own Read Write	
User B	Read	Own Read Write	Write	Read
User C	Read Write	Read		Own Read Write

MANDATORY ACCESS CONTROL

- MAC restricts the access on the basis of security labels
- Unlike DAC, entities cannot enable other entities to access their resources
- Users have security clearance
- Resources have security labels that contain data classifications
- This model is used in environments where information classification and confidentiality are very important
  - E.g., military

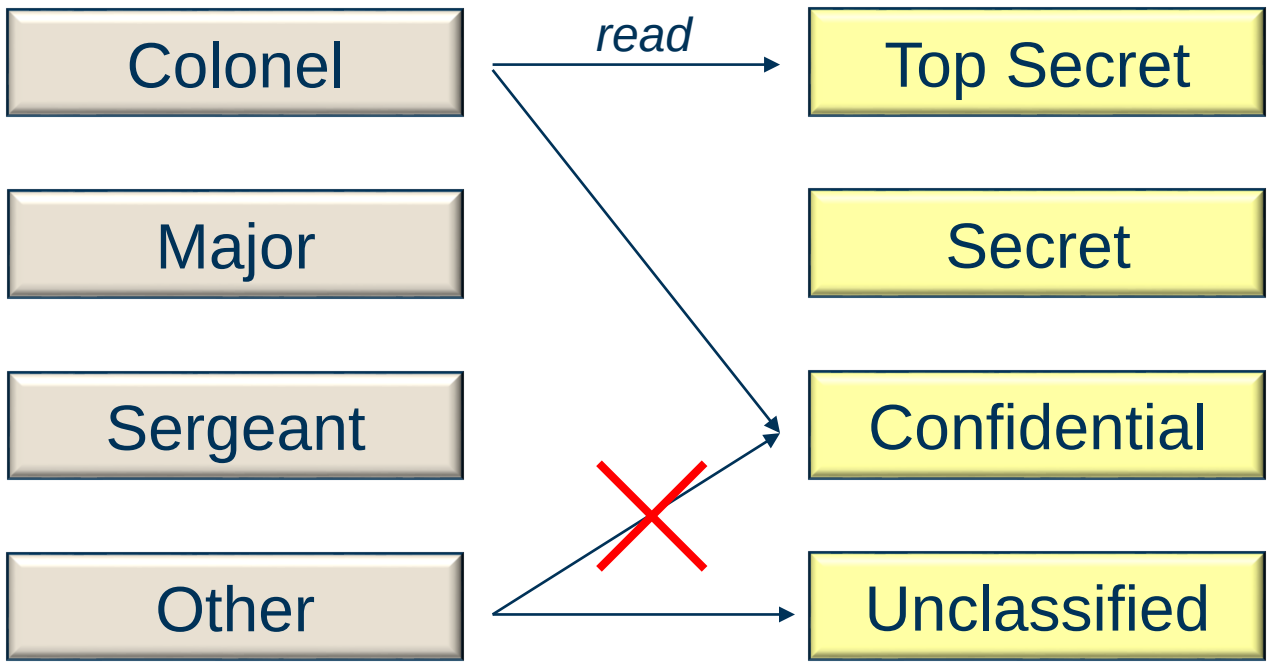
MAC: BELL-LAPADULA MODEL (‘76)

The main goal is to control **confidentiality of information**

Security Clearance	Security Labels
Colonel	Top Secret
Major	Secret
Sergeant	Confidential
Other	Unclassified

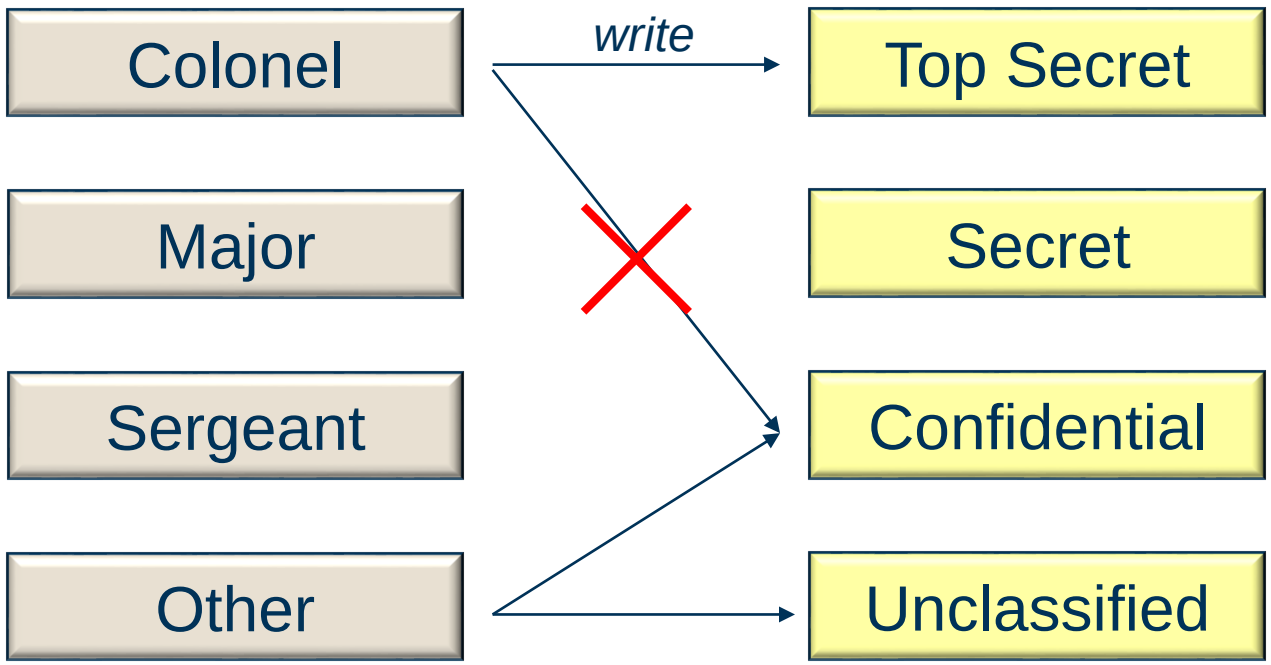
MAC: CONFIDENTIALITY RULES

Simple Security Property: No Read Up



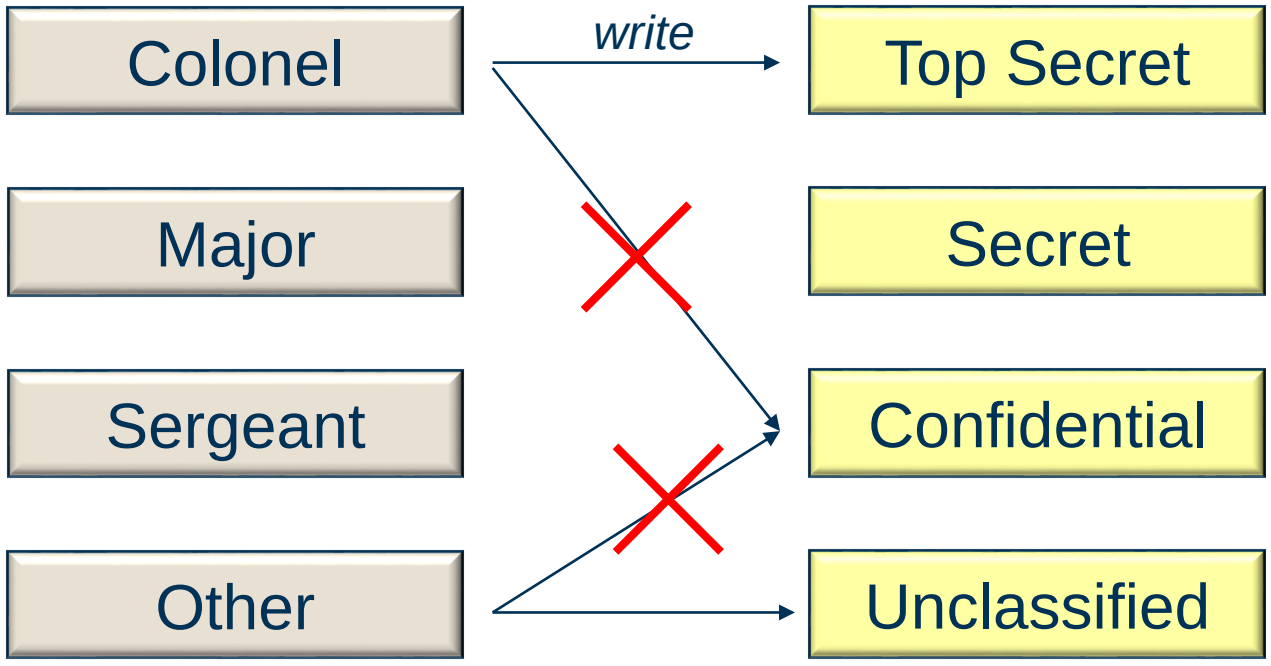
MAC: CONFIDENTIALITY RULES

**(Star)property: No Write Down*



MAC RULES

*Strong *(Star)-property: No Write Down & No Write Up*



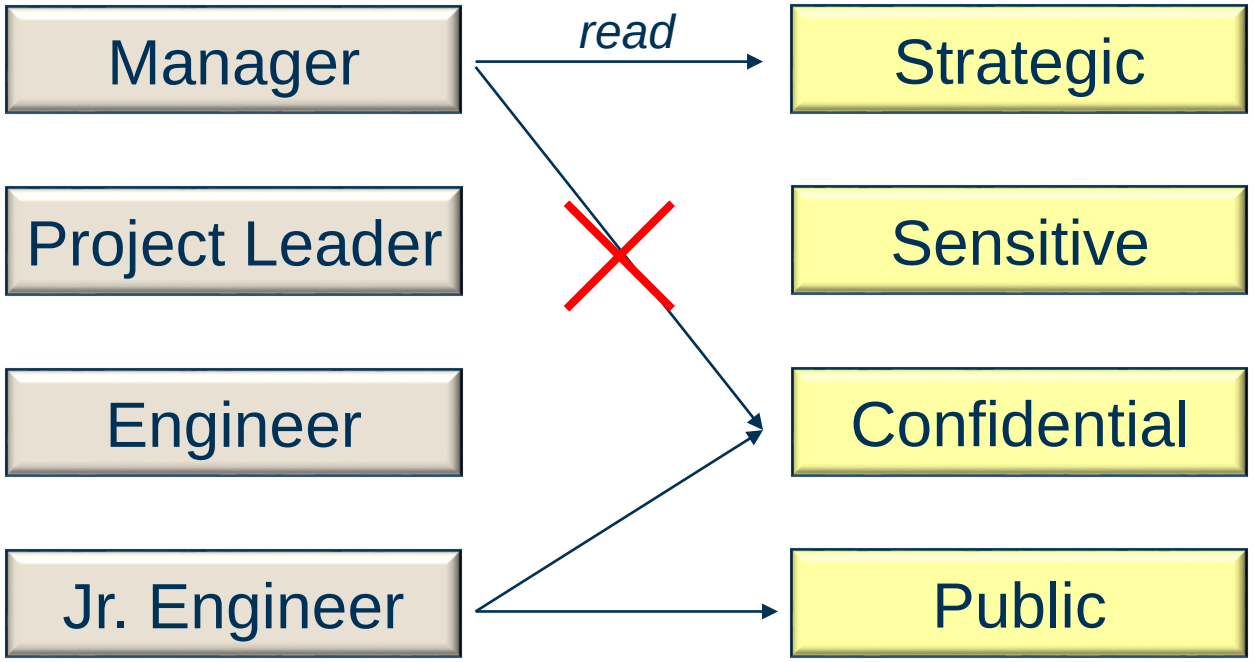
MAC: BIBA INTEGRITY MODEL (‘77)

The main goal is to control **integrity of information**

Security Clearance	Security Labels
Manager	Strategic
Project Leader	Sensitive
Engineer	Confidential
Jr. Engineer	Public

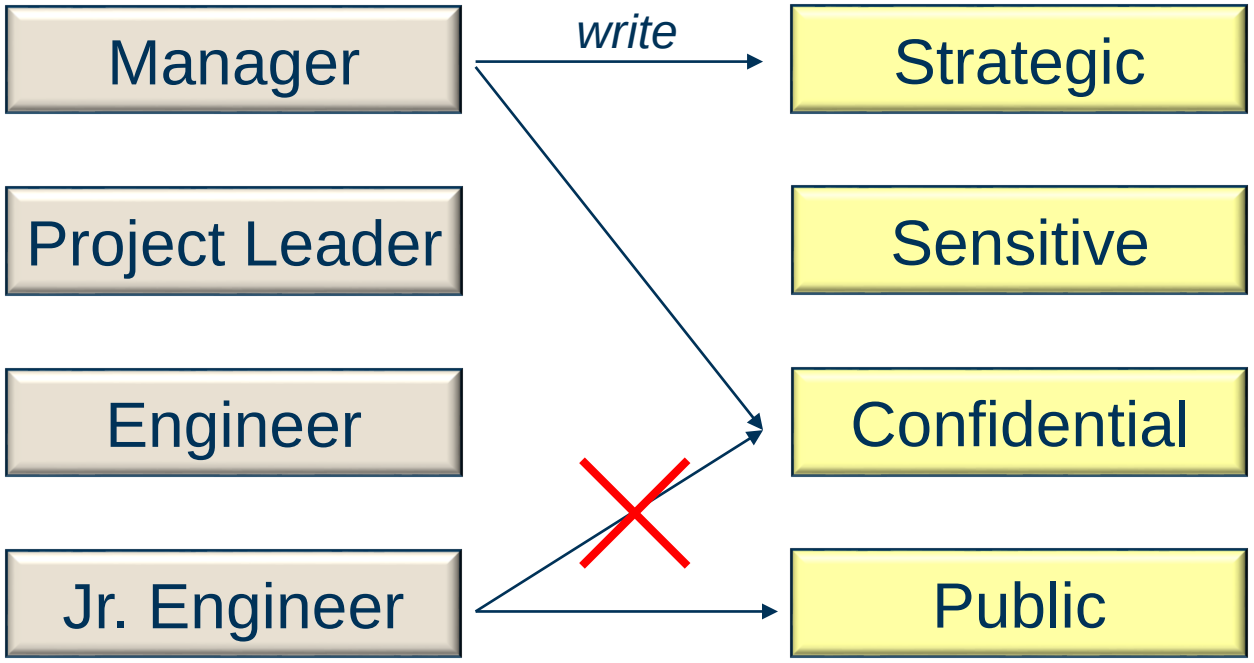
MAC: INTEGRITY RULES

Simple Integrity Axiom: No Read Down



MAC: INTEGRITY RULES

**(Star)-Integrity Axiom: No Write Up*



WHERE IS MAC USED?

- Bell–LaPadula (BLP) model
  - Implemented in the multi-level security policy for the US Department of Defense (DoD)
- Biba model
  - Implemented in the FreeBSD MAC policy
- A combined version of BLP and BIBA is used in Android!

ROLE-BASED ACCESS CONTROL

- RBAC maps roles to access rights
- Supports complex access control
- Reduces errors in administration
- Ease of administration
  - Move users in and out of roles
  - Move permissions in and out of roles
  - Very flexible
- Least privilege
  - Restricts access according to needs
  - Separation of duties through constraints

RBAC MODEL

- User
  - Typically a human being
- Permissions
  - Approval of a mode of access to some object
- Roles
  - Job title
- Assignments
  - User-role and role-perm
- Session
  - Mapping of users to roles
- Constraints
  - Sessions, assignments, and roles

TO BE CONTINUED

- See the next lecture

Questions?

Thanks for your attention!