Security Policy Modeling

고려대학교 (Korea Univ.)

사이버국방학과 · 정보보호대학원 (CIST)

보안성분석평가연구실 (Security Analysis aNd Evaluation Lab.)

김 승 주 (Seungjoo Kim)

www.kimlab.net

보안성분석평가연구실

Security Analysis aNd Evaluation Lab

www.KimLab.net / www.SecEng.net

Seungjoo Kim
PROFESSOR, KOREA UNIVERSITY
CNN
North Korean government website hacked
Source: CNN

김승주 교수 (skim71@korea.ac.kr)

로봇융합관 306호

연구분야

- Security Eng. for High-Assurance Trustworthy Systems
- High-Assurance Cryptography
- Security Testing (including End-to-End Provable Security, Formal Verification) and Security Evaluation (e.g. CMVP, CC, C&A, SSE-CMM)
- Usable Security

주요 경력 :

1990.3~1999.2) 성균관대학교 공학 학사·석사·박사
1998.12~2004.2) KISA 암호기술팀장 및 CC평가1팀장
2004.3~2011.2) 성균관대학교 정보통신공학부 부교수
2011.3~현재) 고려대학교 사이버국방학과·정보보호대학원 정교수
Founder of (사)HARU & SECUINSIDE

前) 육군사관학교 초빙교수
前) 선관위 DDoS 특별검사팀 자문위원
前) SBS 드라마'유령'및 영화'베를린'자문 / KBS '명견만리' 강연

現) 한국정보보호학회 이사
現) 대검찰청 디지털수사 자문위원
現) 개인정보분쟁조정위원회 위원

- '96: Convertible group signatures (AsiaCrypt)
- '97: Proxy signatures, revisited (ICICS): 670회 이상 인용
- '06: 국가정보원 암호학술논문공모전 우수상
- '07: 국가정보원장 국가사이버안전업무 유공자 표창
- '12, '16: 고려대학교 석탑강의상
- '13: Smart TV Security (Black Hat USA): 스마트TV 해킹(도청·도촬) 및 해적방송 송출 시연

주요 R&D 성과

SAMSUNG

Common Criteria Certification for Samsung Multifunction Printers

WHITE PAPER

Common Criteria Certification for Samsung Multifunction Printers

Introduction

[illegible]

Overview

[illegible]

SAMSUNG

삼성전자와 공동으로
국내 최초 프린터복합기 보안 인증 획득 (2008년)

LG전자와 공동으로
국내 최초 스마트TV 보안 인증 획득 (2015년)

Introduction to Formalization

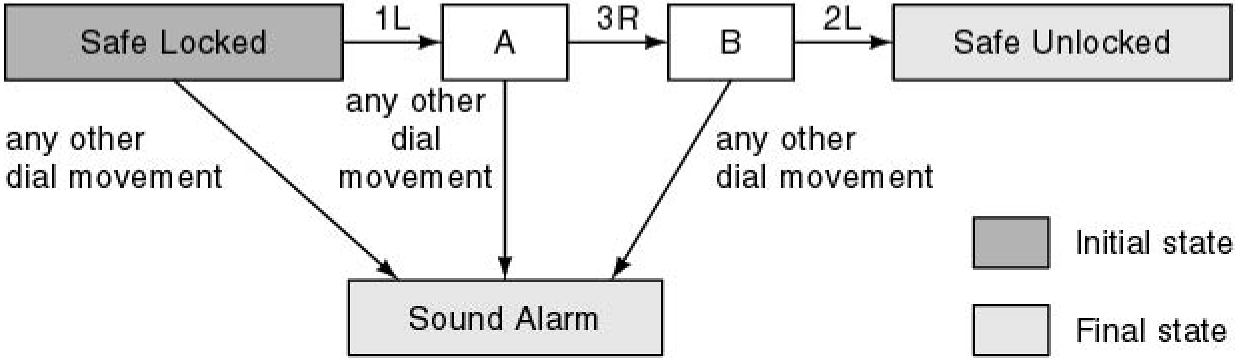
Formalization

- **Informal method**
  - English (or other natural language)
- **Semiformal methods**
  - Gane & Sarsen/DeMarco/Yourdon
  - Entity-Relationship Diagrams
  - Jackson/Orr/Warnier
  - SADT, PSL/PSA, SREM, etc.
- **Formal methods**
  - Finite State Machines
  - Petri Nets
  - Z
  - ANNA, VDM, CSP, etc.

FSM Example

- **(M202, Open University, UK)** A safe has a combination lock that can be in one of three positions, labeled 1, 2, and 3. The dial can be turned left or right (L or R). Thus there are six possible dial movements, namely 1L, 1R, 2L, 2R, 3L, and 3R. The combination to the safe is 1L, 3R, 2L; any other dial movement will cause the alarm to go off.

FSM Example



[State Transition Diagram]

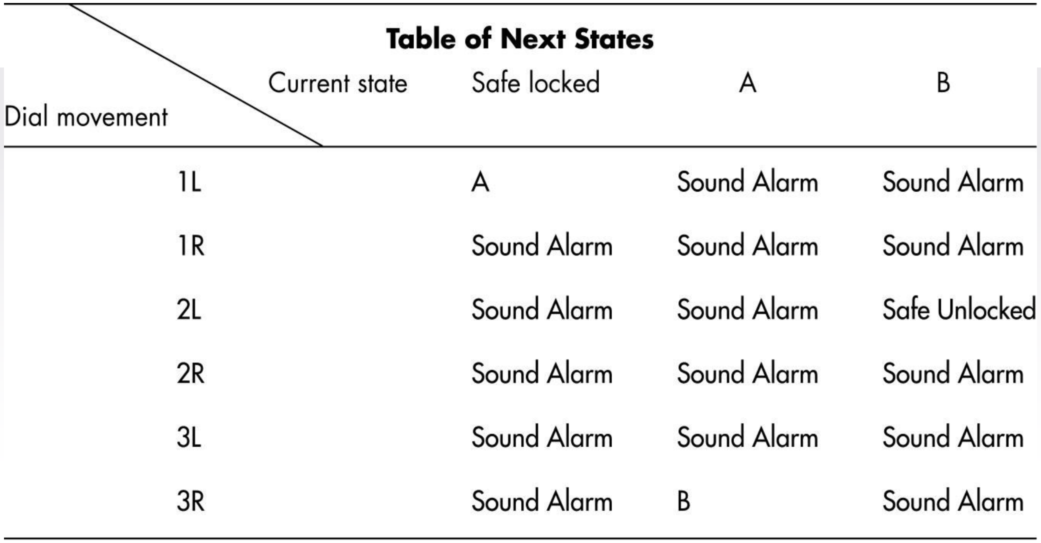


Table of Next States

Dial movement \ Current state	Safe locked	A	B
1L	A	Sound Alarm	Sound Alarm
1R	Sound Alarm	Sound Alarm	Sound Alarm
2L	Sound Alarm	Sound Alarm	Safe Unlocked
2R	Sound Alarm	Sound Alarm	Sound Alarm
3L	Sound Alarm	Sound Alarm	Sound Alarm
3R	Sound Alarm	B	Sound Alarm

[Transition Table]

What Formal Methods Can Do

- Delimit the system's boundary : the system and its environment.
- Characterize a system's behavior more precisely.
- Define the system's desired properties precisely.
- Prove a system meets its specification.
  - Tell under what circumstances a system cannot meet its specification

How They Can Help

- These capabilities of formal methods help practitioner in two ways.
  - Through (   ), focusing on designer's attention
    - What is interface
    - What are the assumptions about the system
    - What is the system supposed to do under this condition and that condition
    - What are the system's invariant properties
  - Through (   )
    - Prove a system meet its security goals
    - Find out the weakness of the system

KOREA UNIVERSITY

Two Different Styles of Formal Methods

- The UK and European style
  - Focus was on (   ), on the system’s high level design and on the paper-and-pencil analysis.

- The US and Canadian style
  - Focus was on (   ), from the system’s high level design through its code-level implementation down to its bit-level representation in hardware, and on machine-assisted analysis.

History - Past

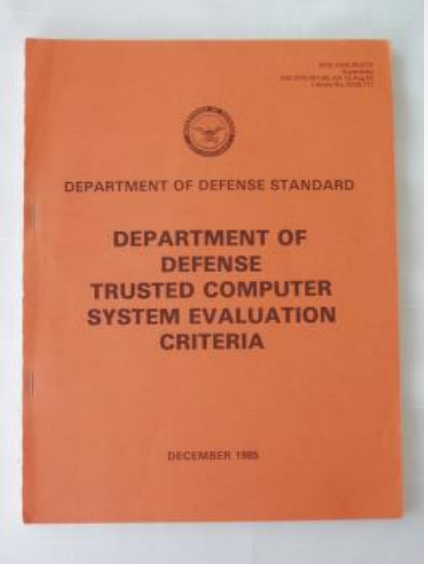
- **NSA**(National Security Agency) was the major source of funding formal methods research and development in the 70s and early 80s.
- Early formal method centered on proving systems secure.
  -
  -
  -
- The systems of interest to prove secure were operating systems, more specifically, (   ).

The Orange Book

- US Trusted Computer System Evaluation Criteria
  - Produced by NCSC (National Computer Security Center) in 1985.
  - Provide a standard metric for NCSC to compare the security of different computer systems.
  - Guide computer system vendors in the design and development of secure systems.

The Orange Book

- Provide a means for specifying security requirements in Government contracts.
  - Levels : D, C1, C2, B1, B2, B3, A1, (A2)
  - Certified (   ) means that one formally specify the system’s security requirements, formally model the system and formally prove that the model meets its specification.



DOD 5200.28-STD
Supersedes
CSC-STD-001-83, dtd 15 Aug 83
Library No. S225,711

DEPARTMENT OF DEFENSE STANDARD

DEPARTMENT OF DEFENSE TRUSTED COMPUTER SYSTEM EVALUATION CRITERIA

DECEMBER 1985

Major Results of Early 80s

- Major results of early 80s were theorem proving tools with general purpose.
  - Affirm (USC)
  - Formal Development Methodology (FDM) System (System Development Corporation)
  - Gypsy (U. Texas Austin)
  - (Enhanced) Hierarchy Development Methodology (HDM) (Stanford)

History - Present

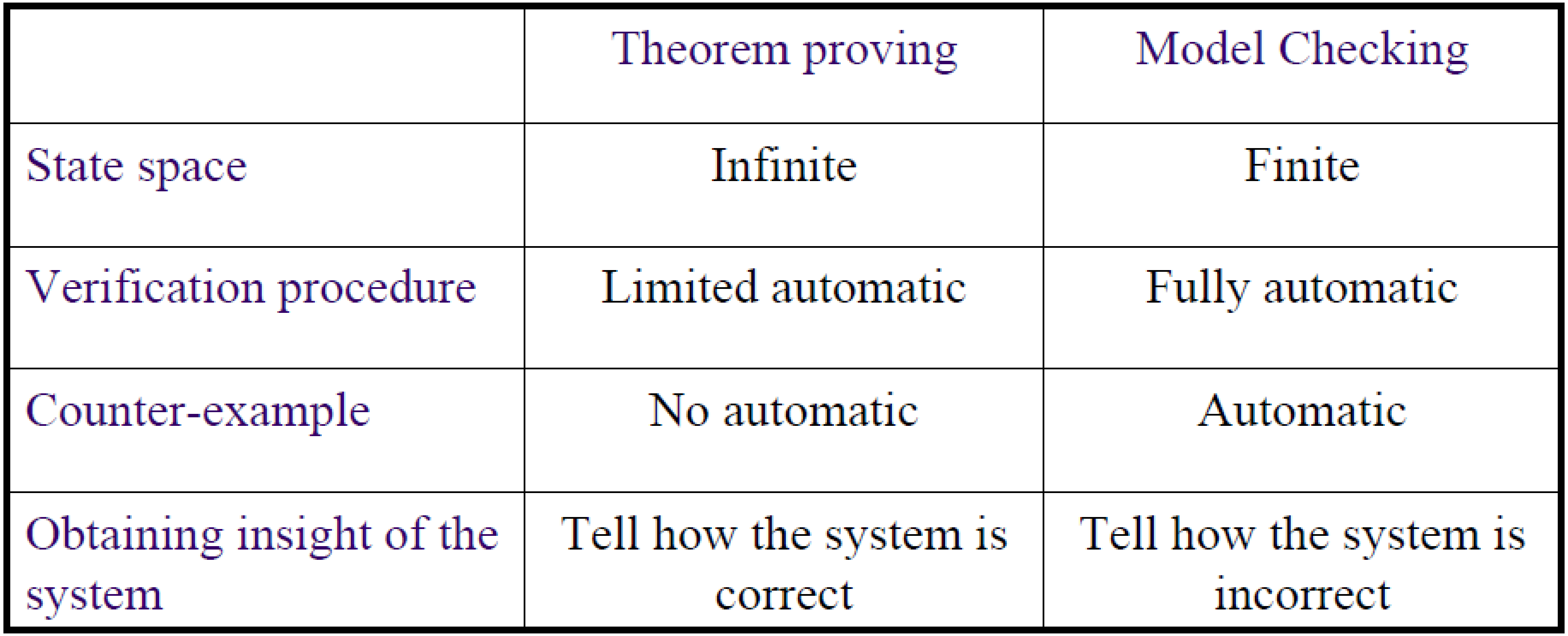
- Since early 90s, there was an explosion of new developments.
- Three threads in the development of formal methods

Categorization of Formal Methods

	Model checking		Theorem proving
Symbolic	NRL FDR AVISPA	SCYTHER ProVerif AVISPA (TA4SP)	Isabelle/HOL
Cryptographic		CryptoVerif	BPW(in Isabelle/HOL) Game-based Security Proof (in Coq)

Unbounded

Theorem Proving vs. Model Checking



	Theorem proving	Model Checking
State space	Infinite	Finite
Verification procedure	Limited automatic	Fully automatic
Counter-example	No automatic	Automatic
Obtaining insight of the system	Tell how the system is correct	Tell how the system is incorrect

Model Checking

- In 1996, model checker FDR was used to exhibit a flaw in the Needham-Schroeder protocol.
- Since then many other model check tools or other proving tools showed the same thing.
- Other protocols were or are being checked.
  - SSL by Stanford
  - IKE, SET by Meadows
  - Netbill electronic payment protocol

Theorem Proving

- General purpose theorem prover Isabelle was used to reason 5 classic authentication protocols and their variation.
- Coq was used to analyze the Needham-Schroeder protocol and SET.
- PVS was used to verify authentication protocol.

Hybrid Approaches

- Embodies both model checking and theorem proving functionalities.
- Improved NRL protocol analyzer analyze IKE and SET protocol.
- Improved NRL embodies both model checking (e.g., brute force search)and theorem proving (e.g., lemma generation).

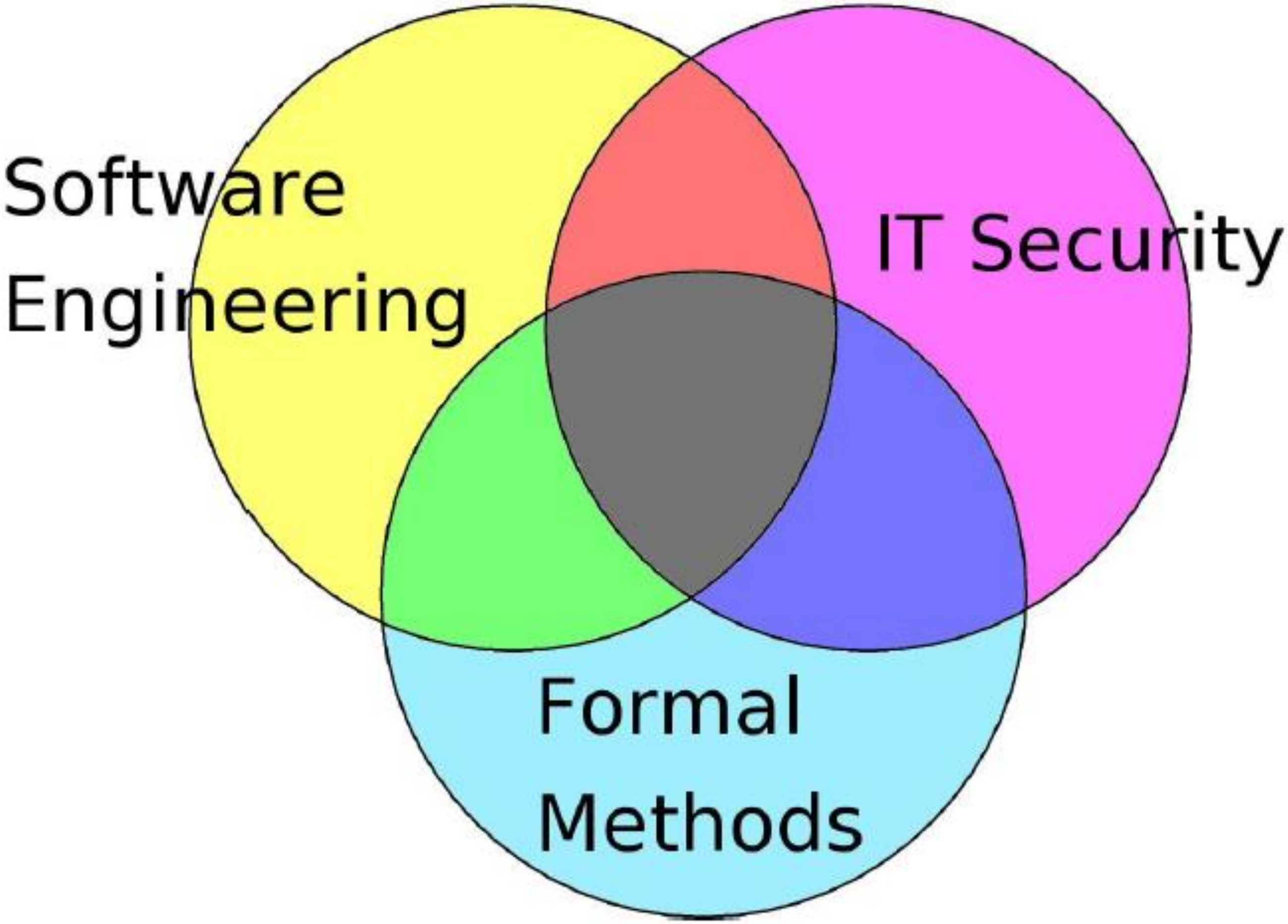
The Limits of Formal Methods

- Systems will never made 100% secure.
  - Formal methods will **not** break this axiom.

- Assumptions about the system's environment
  - Hard to state them explicitly.
  - The system could be deployed
    ( ).
    - For convenience or lack of an alternative
  - Clever intruders find out
    ( ).

Security Policy Modeling

Security (Policy) Modeling



Security Policy

- State (   ) should be protected.
- A security policy is a statement of what is, and what is not, allowed.
  - **Confidentiality** : Who is allowed to learn what?
  - **Integrity** : What changes are allowed by system.
    - … includes resource utilization, input/output to environment.
  - **Availability** : When must service be rendered.
- And (   ) this should be achieved.

Security Policy Model (SPM)

- (   ) Specification of Security Policy
  - (e.g.) DAC Model, MAC Model, Bell-LaPadula Model, Biba Model, Clark-Wilson Model, Harrison-Ruzzo-Ullman Model, Chinese Wall Model, RBAC Model, etc.

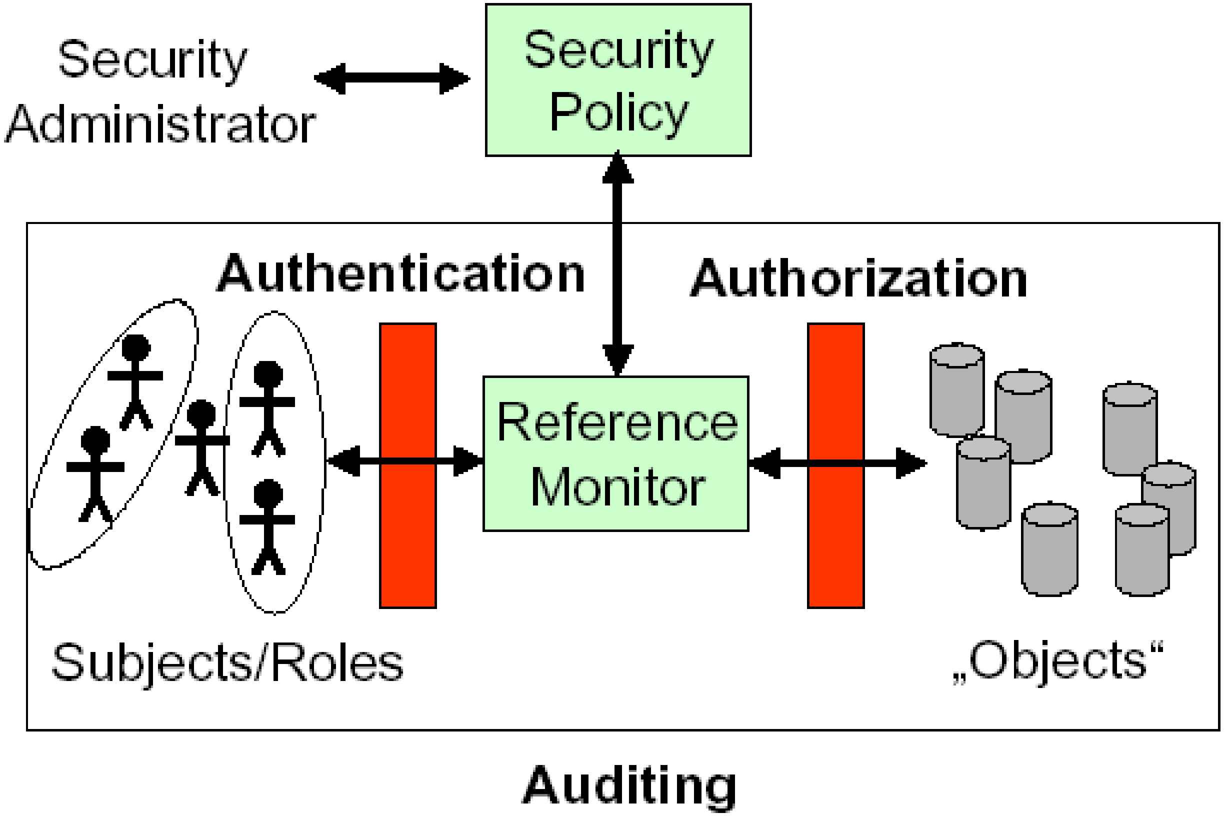
Security Policy Assurance

- Evidence that the set of security requirements is complete, consistent and technically sound.
  - **Completeness** :
    - If the completeness of rules can completely be verified, the (   ) rate will be zero.
  - **Soundness** :
    - If the soundness of rules can be completely verified, the (   ) rate will theoretically be proved to be zero.

In

- Realized that (   ) system posed security issues that went beyond the traditional concerns for secure communications. (   ) (@ NSA) talked 3 important issues :

Reference Monitor



Reference Monitor

- Reference Monitor

- Security Kernel

- TCB

Authentication & Authorization

- **Authentication :** Method of (   ) the identity.
  - **Identification :** Method of (   ) the subject's (user, program, process) identity.

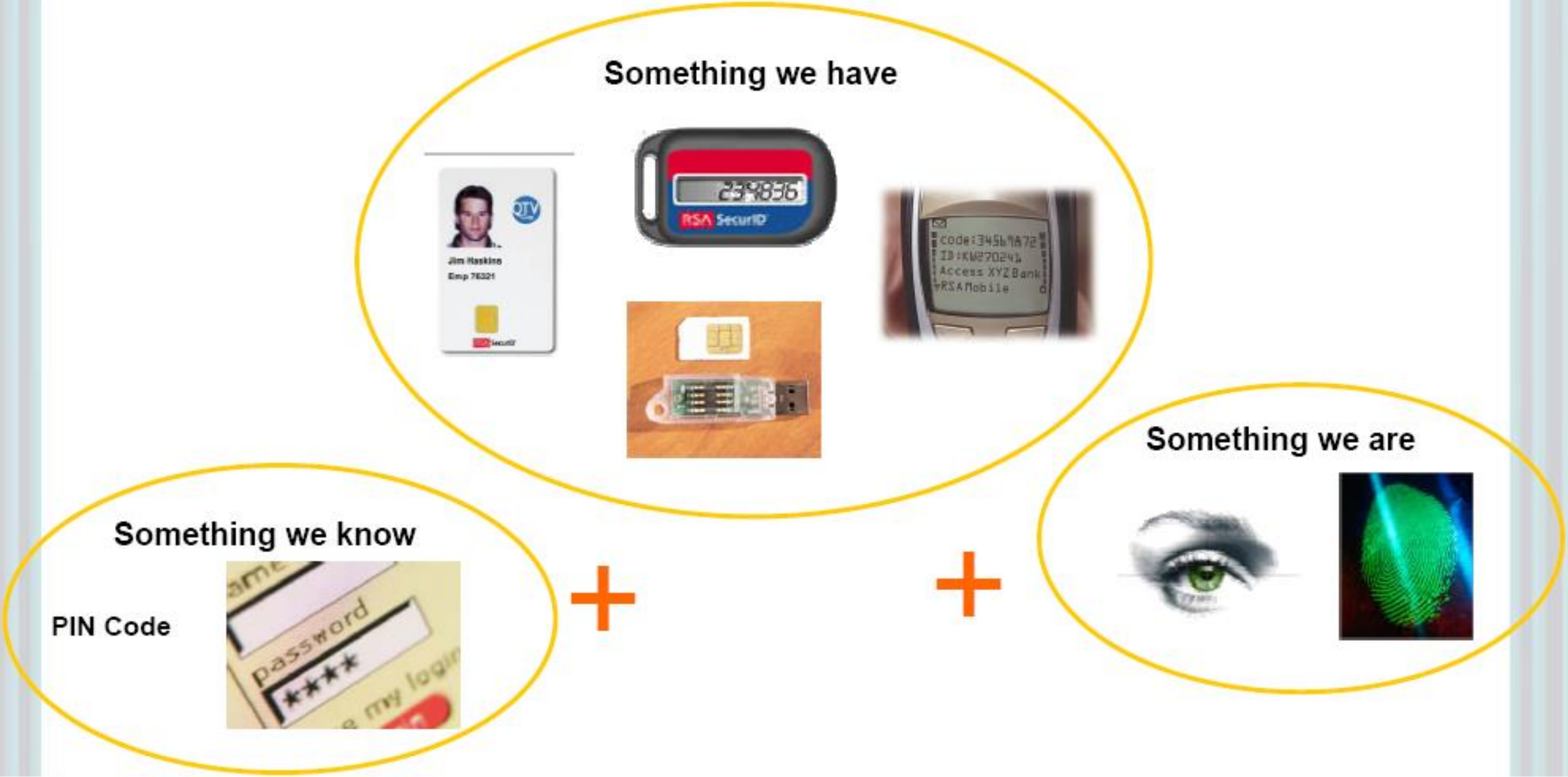
- **Authorization :** The mechanism by which a system determines what level of access a particular authenticated user should have to secured resources controlled by the system.

Authentication

Terms

- **Authentication (사용자인증)**
  - = "Verification"
  - Authentication mean verifying the user is actually who he says he is (or who she says she is).
  - (   )
  - (   ) matching
- **Identification (개인식별)**
  - Identification means you don't know anything about the person and you are trying to identify them.
  - (   )
  - (   ) searching

Methods



Something we have

Something we know

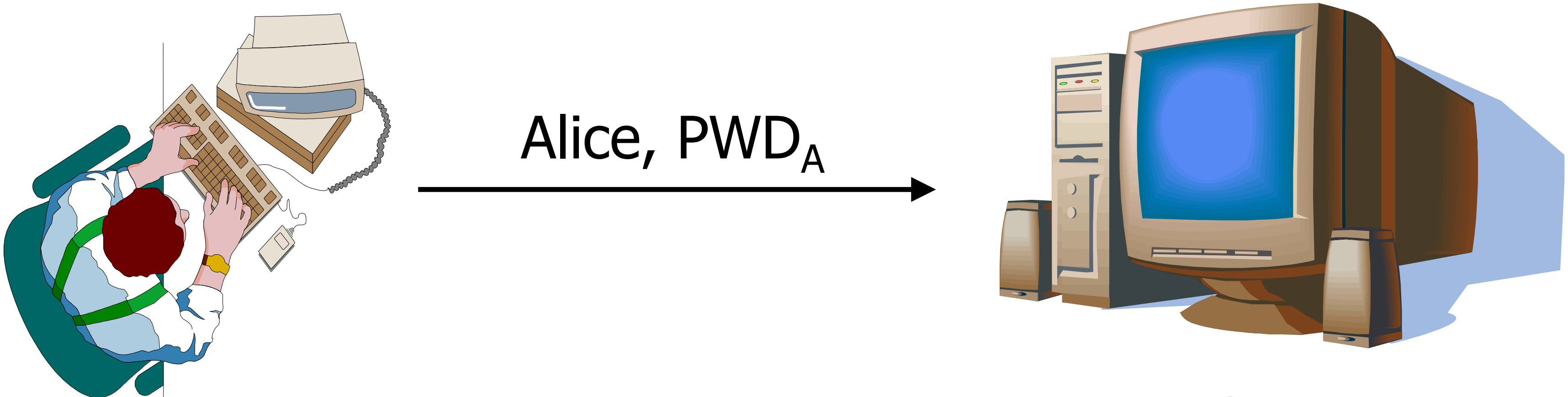
PIN Code

+

+

Something we are

Password



Alice, PWD_A

Server

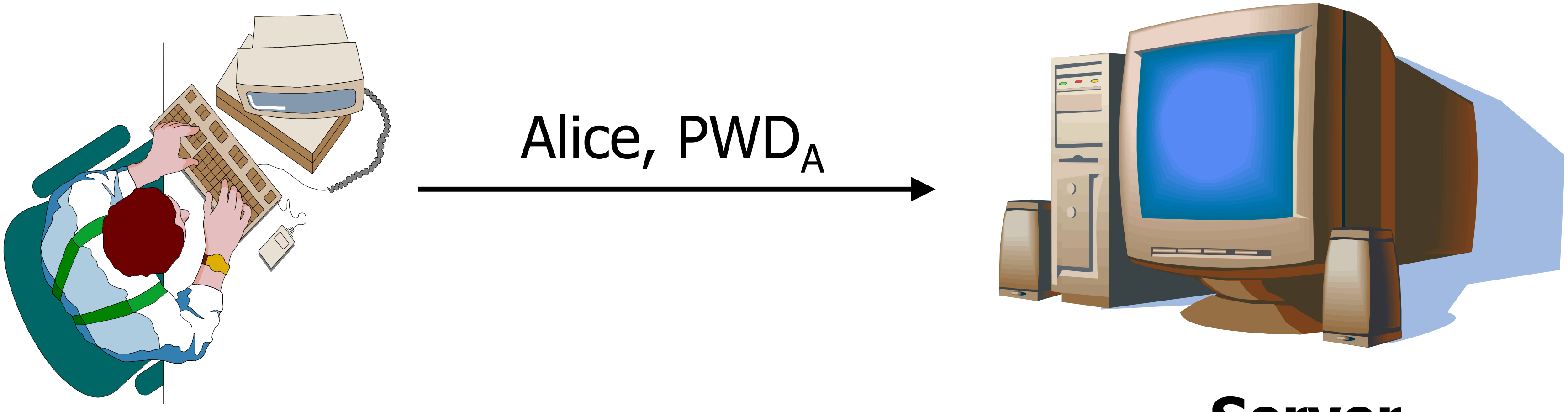
Alice (PWD_A)

ID	Password
Alice	PWD_A
Bob	PWD_B
…	…
…	…

Replay Attack

- "Replay" attacks are "Man in the middle" attacks that involve intercepting data packets and replaying them, that is, (   ) (with no decryption) to the receiving server.

Replay Attack



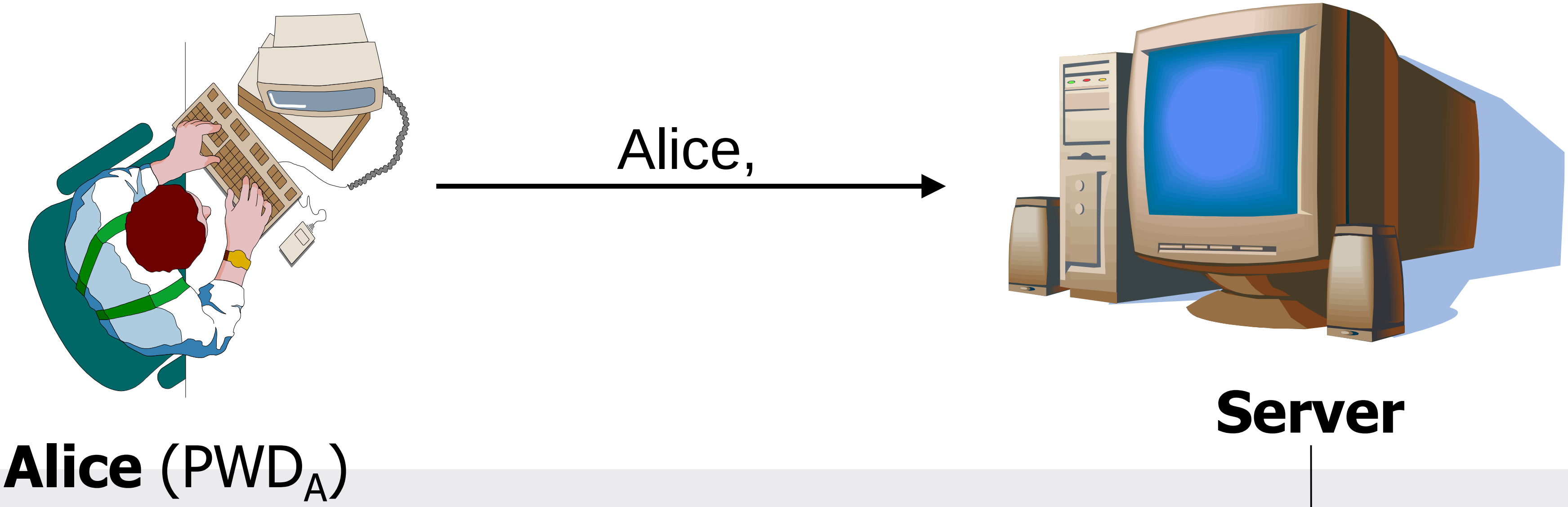
Alice, PWD_A

Alice (PWD_A)

Server

ID	Password
Alice	PWD_A
Bob	PWD_B
…	…
…	…

Countermeasures



ID	Password
Alice	PWD$_A$
Bob	PWD$_B$
…	…
…	…

Password Guessing Attacks

- Password guessing attacks can be classified into two.
  - **Brute Force Attack :** A Brute Force attack is a type of password guessing attack and it consists of trying every possible code, combination, or password until you find the correct one. This type of attack may take long time to complete. A complex password can make the time for identifying the password by brute force long.
  - **Dictionary Attack :** A dictionary attack is another type of password guessing attack which uses a dictionary of common words to identify the user's password.

Brute Force Attack

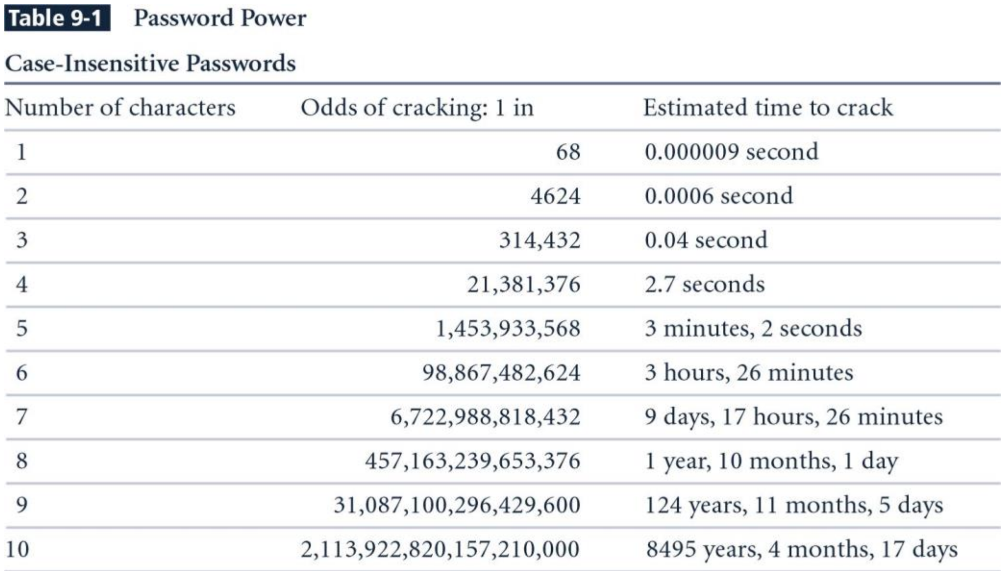


Table 9-1 Password Power

Case-Insensitive Passwords

Number of characters	Odds of cracking: 1 in	Estimated time to crack
1	68	0.000009 second
2	4624	0.0006 second
3	314,432	0.04 second
4	21,381,376	2.7 seconds
5	1,453,933,568	3 minutes, 2 seconds
6	98,867,482,624	3 hours, 26 minutes
7	6,722,988,818,432	9 days, 17 hours, 26 minutes
8	457,163,239,653,376	1 year, 10 months, 1 day
9	31,087,100,296,429,600	124 years, 11 months, 5 days
10	2,113,922,820,157,210,000	8495 years, 4 months, 17 days

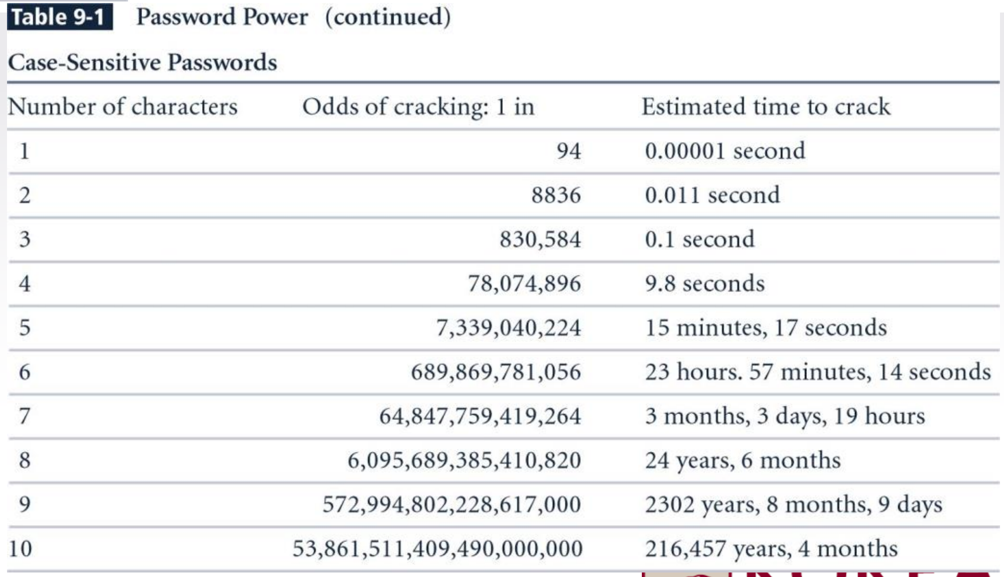


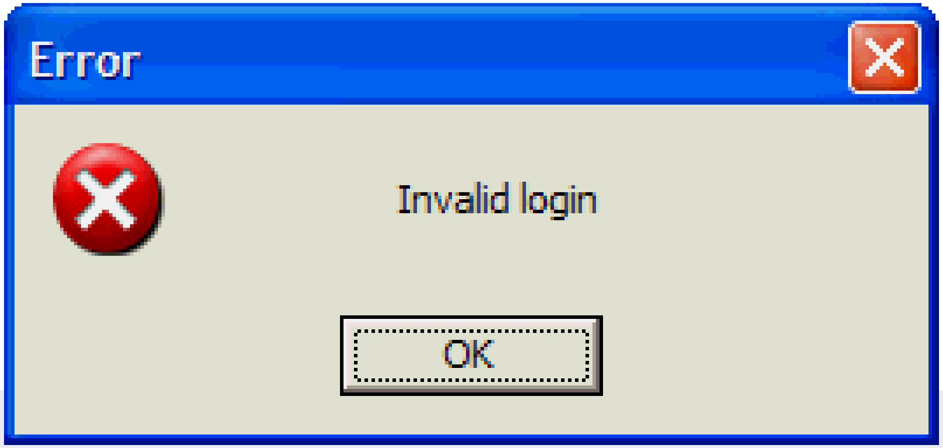
Table 9-1 Password Power (continued)

Case-Sensitive Passwords

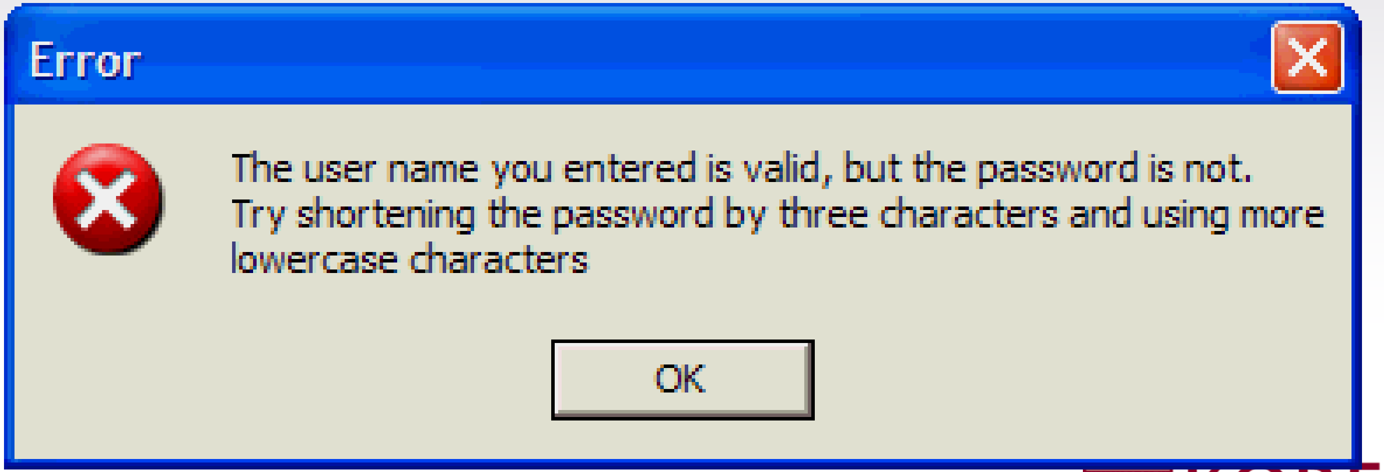
Number of characters	Odds of cracking: 1 in	Estimated time to crack
1	94	0.00001 second
2	8836	0.011 second
3	830,584	0.1 second
4	78,074,896	9.8 seconds
5	7,339,040,224	15 minutes, 17 seconds
6	689,869,781,056	23 hours. 57 minutes, 14 seconds
7	64,847,759,419,264	3 months, 3 days, 19 hours
8	6,095,689,385,410,820	24 years, 6 months
9	572,994,802,228,617,000	2302 years, 8 months, 9 days
10	53,861,511,409,490,000,000	216,457 years, 4 months

Countermeasures

- Which is the better error message?

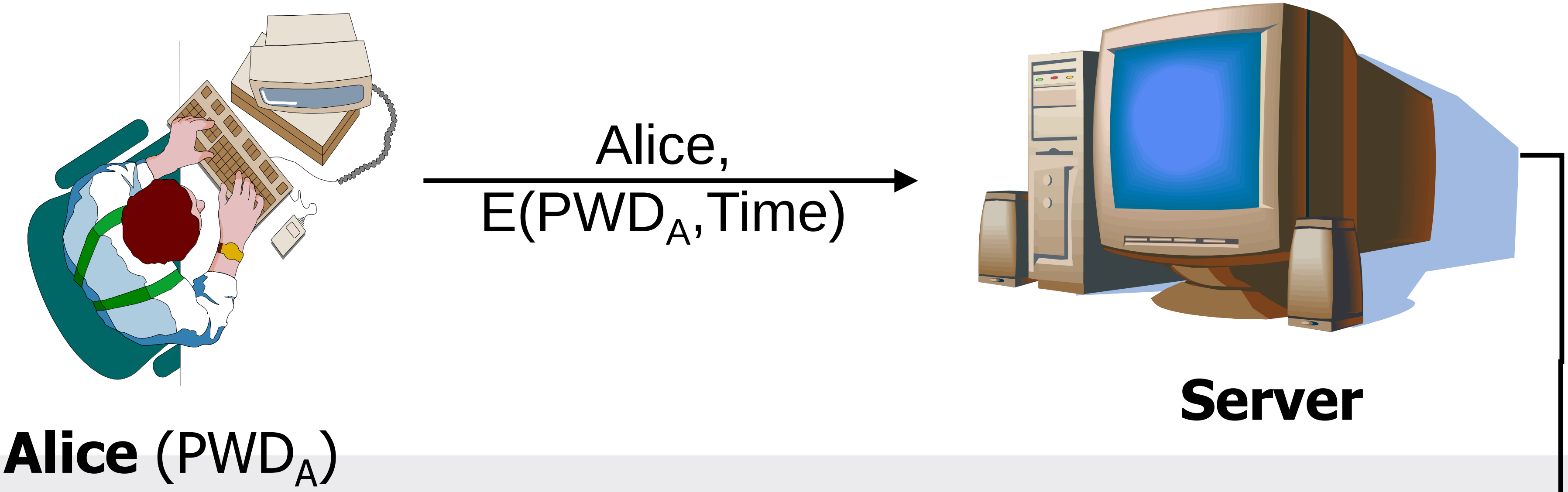


Invalid login



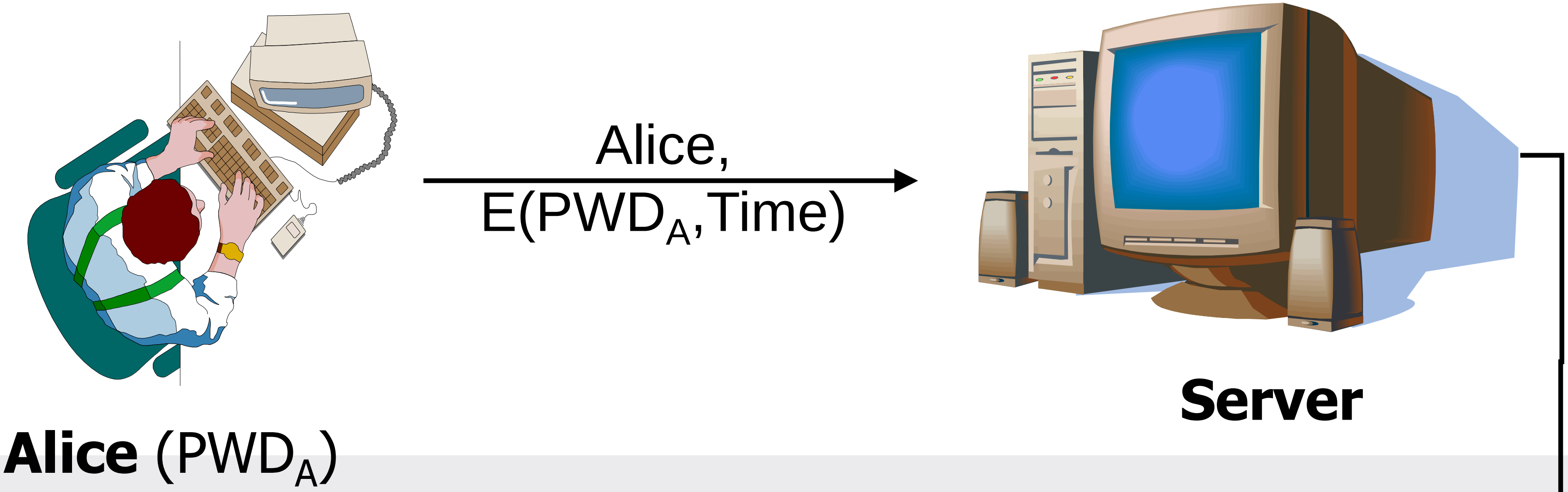
The user name you entered is valid, but the password is not.
Try shortening the password by three characters and using more lowercase characters

Dictionary Attack



ID	Password
Alice	PWD_A
Bob	PWD_B
…	…

Dictionary Attack



Alice,

$E(PWD_A, Time)$

Server

Alice (PWD_A)

ID	Password
Alice	
Bob	
…	…

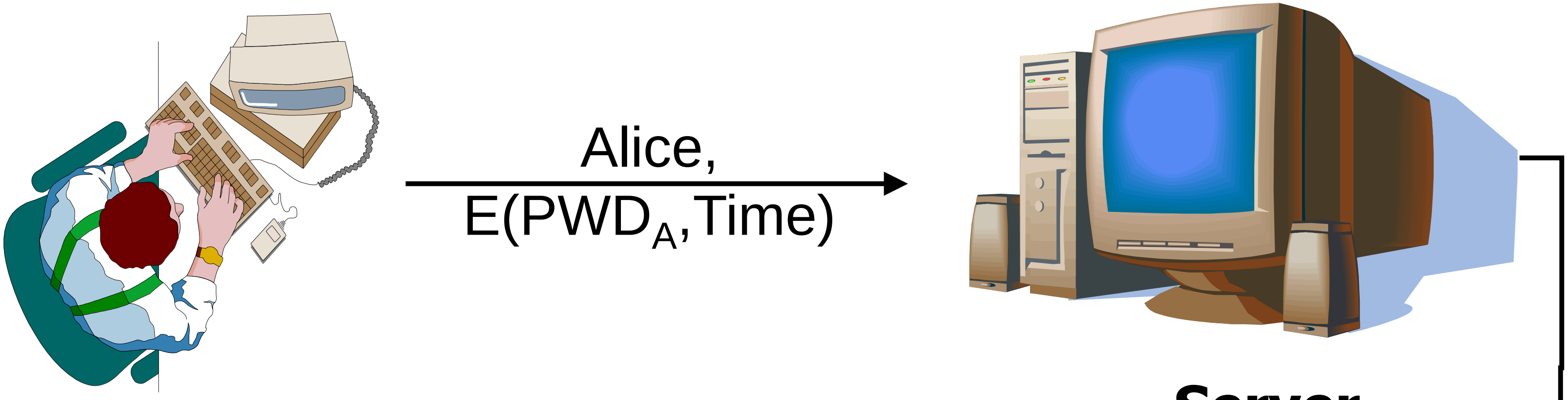
Dictionary Attack

- An attacker with unprivileged access to the system can (   ) of every user's password. Those values can be used to mount a brute force attack or dictionary attack offline, testing possible passwords against the hashed passwords relatively quickly (   ) system security arrangements designed to detect an abnormal number of failed login attempts.

Countermeasures

- The best method of preventing password cracking is to ensure that attackers cannot get access even to the hashed password.
- Using (   ) prevents attackers from efficiently mounting offline attacks against (   ) user accounts simultaneously.

Countermeasures



Alice,
E(PWD$_A$,Time)

Server

Alice (PWD$_A$)

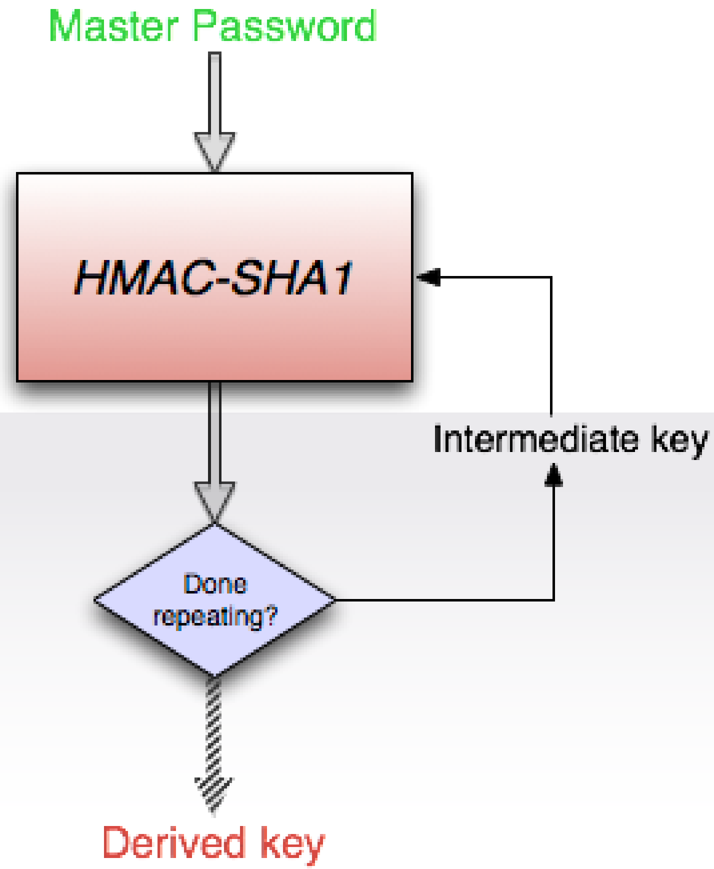
ID	**Salt**	**Password**
Alice		
Bob		
…	…	…

Countermeasures

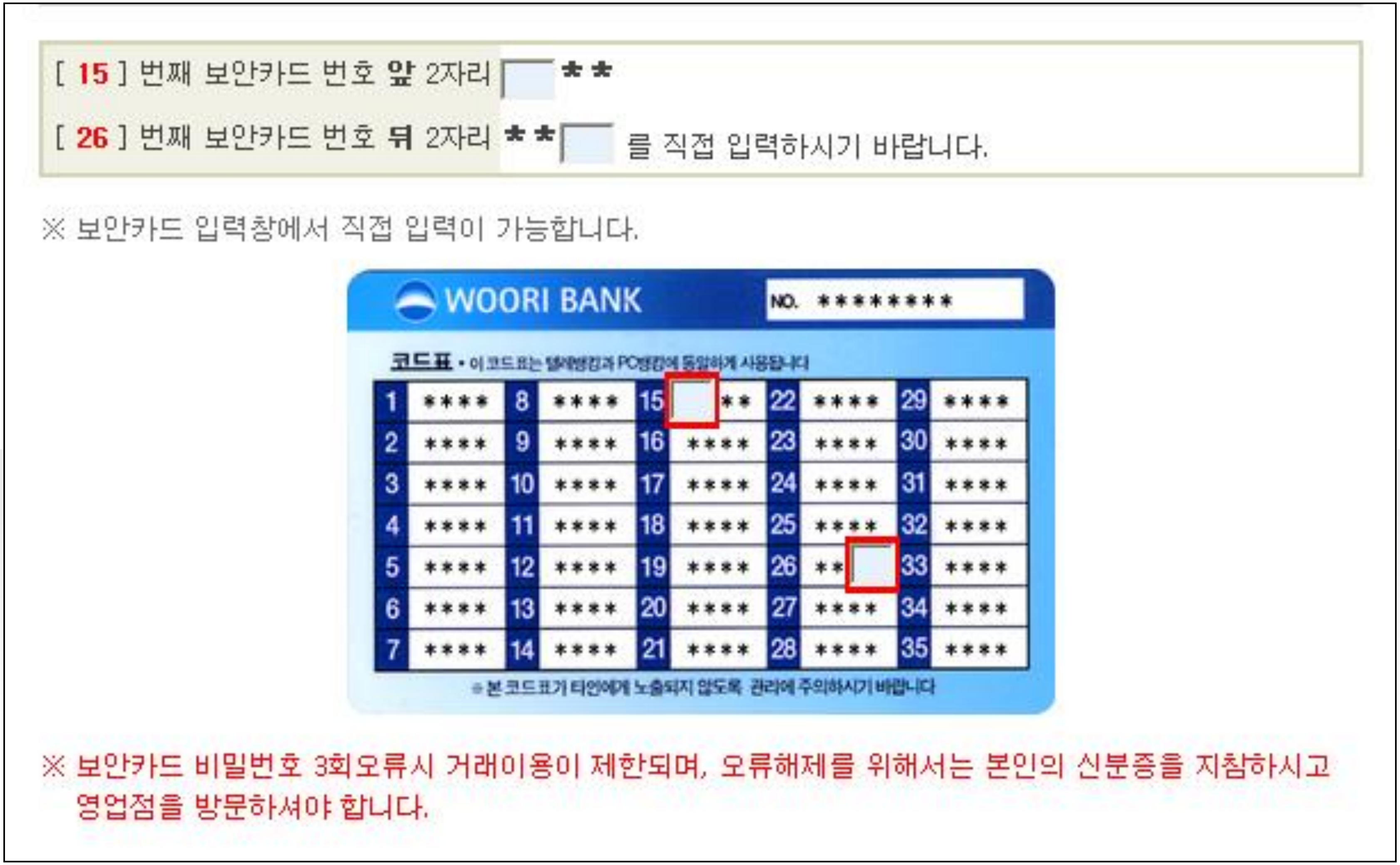
- The best method of preventing password cracking is to ensure that attackers cannot get access even to the hashed password.
- Using ( ) prevents attackers from efficiently mounting offline attacks against ( ) user accounts simultaneously.
- Using ( ), such as PBKDF2, to form password hashes can significantly reduce the rate at which a ( ) user's account can be tested (a.k.a.).

Countermeasures

- PBKDF2



Grid OTP



[15] 번째 보안카드 번호 앞 2자리 □ * *

[26] 번째 보안카드 번호 뒤 2자리 * * □ 를 직접 입력하시기 바랍니다.

※ 보안카드 입력창에서 직접 입력이 가능합니다.

WOORI BANK NO. *********

코드표 • 이 코드표는 텔레뱅킹과 PC뱅킹에 동일하게 사용됩니다

1	****	8	****	15	□ **	22	****	29	****
2	****	9	****	16	****	23	****	30	****
3	****	10	****	17	****	24	****	31	****
4	****	11	****	18	****	25	****	32	****
5	****	12	****	19	****	26	** □	33	****
6	****	13	****	20	****	27	****	34	****
7	****	14	****	21	****	28	****	35	****

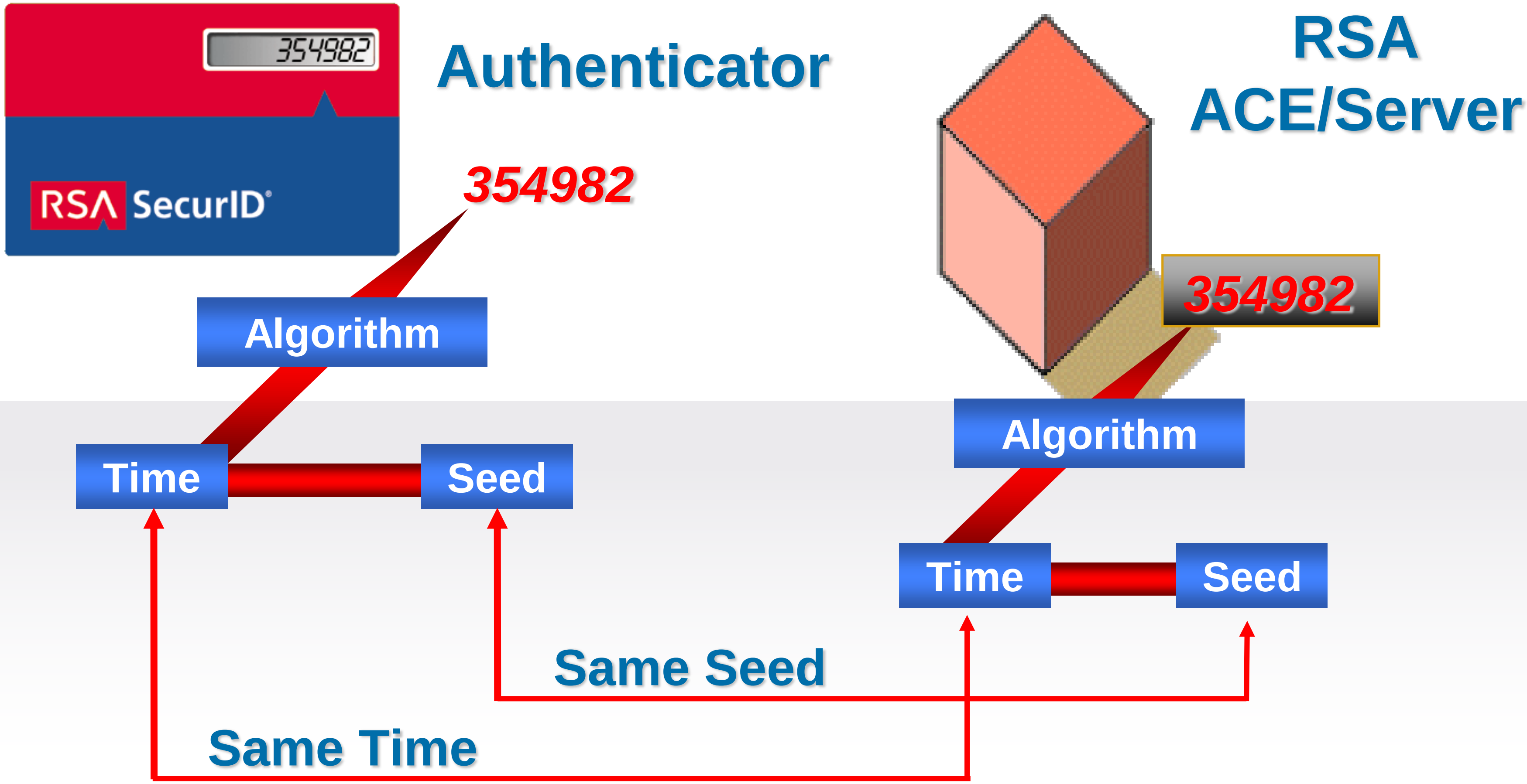
※ 본 코드표가 타인에게 노출되지 않도록 관리에 주의하시기 바랍니다

※ 보안카드 비밀번호 3회오류시 거래이용이 제한되며, 오류해제를 위해서는 본인의 신분증을 지참하시고 영업점을 방문하셔야 합니다.

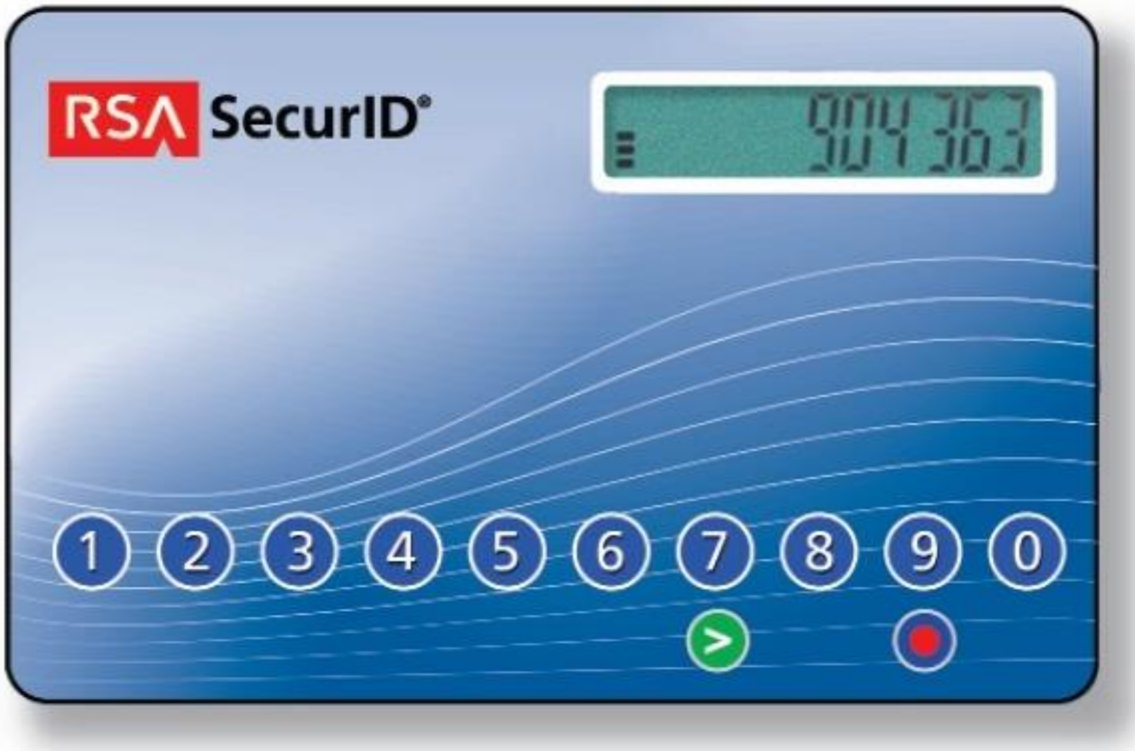
RSA SecurID

RSA SecurID®
for Microsoft® Windows®
611431
Introducing RSA SecurID®
for Microsoft® Windows®

RSA SecurID


354982
RSA SecurID
Authenticator
354982
Algorithm
Time
Seed
RSA
ACE/Server
354982
Algorithm
Time
Seed
Same Seed
Same Time

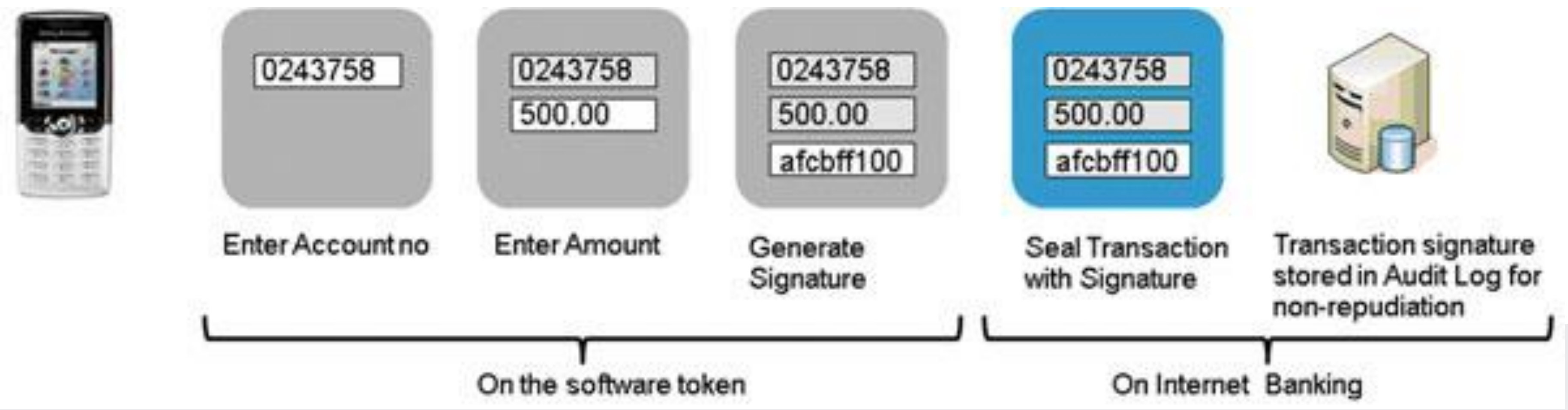
RSA SecurID with Transaction Signing



RSA SecurID SID900 Transaction Signing Authenticator.

The PIN pad enables implementation of the digital signing function.

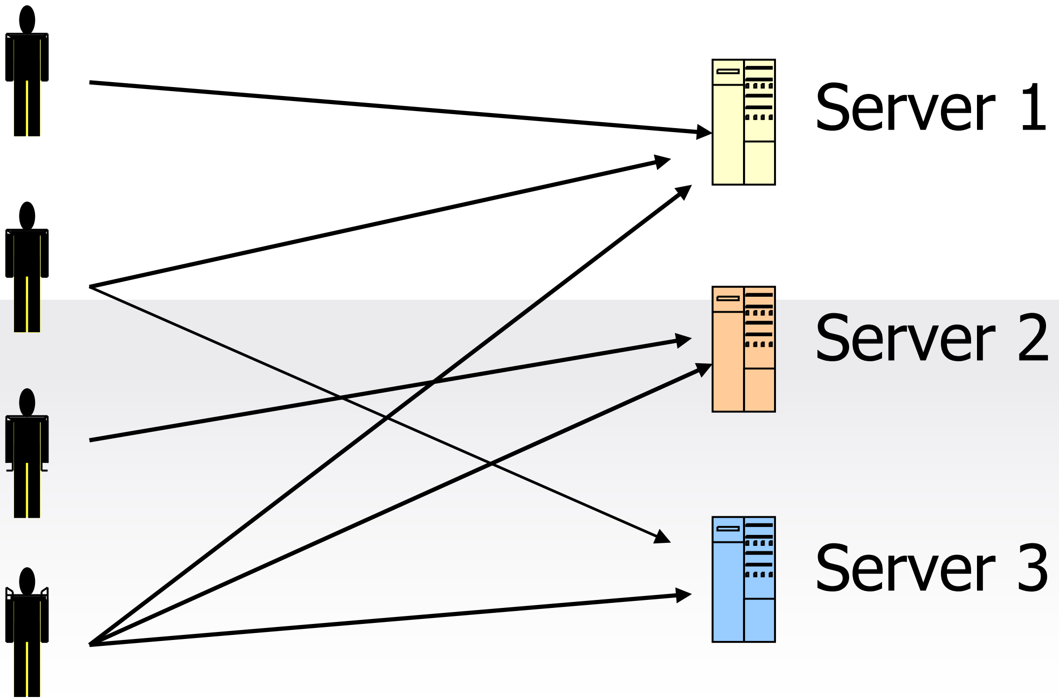
RSA SecurID with Transaction Signing



Authorization (Access Control)

DAC Model

Individuals **Resources**



Application Access List

Name	Access	
Tom	Yes	🟢
John	No	🔴
Cindy	Yes	🟢

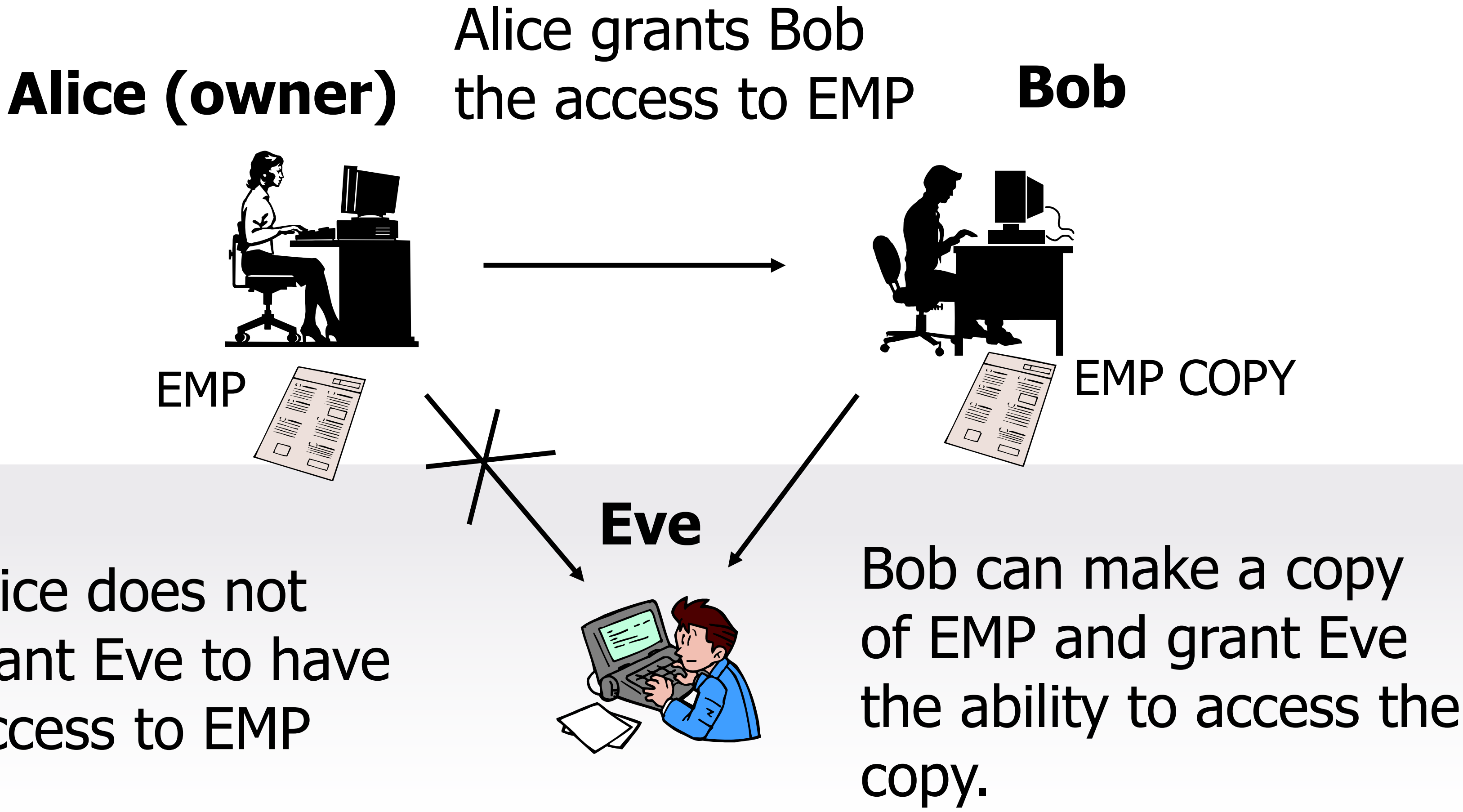
DAC Model

- DAC protects information on a (   ) basis.
- Data owners decide who has access to resources.
  - This model is called (   ) because the control of access is based on the discretion of the owner.
- DAC systems grant or deny access based on the identity of the subject. The identity can be a user identity or group membership.
- The most common implementation of DAC is through ACLs, which are dictated and set by the owners and enforced by the operating system.

DAC Model Weaknesses

- DAC does (   ) from being made and there is (   ).
  - Modern approaches to information sharing and trusted computing seek to maintain control over copies.

DAC Model Weaknesses



Alice (owner)

Alice grants Bob the access to EMP

Bob

EMP

EMP COPY

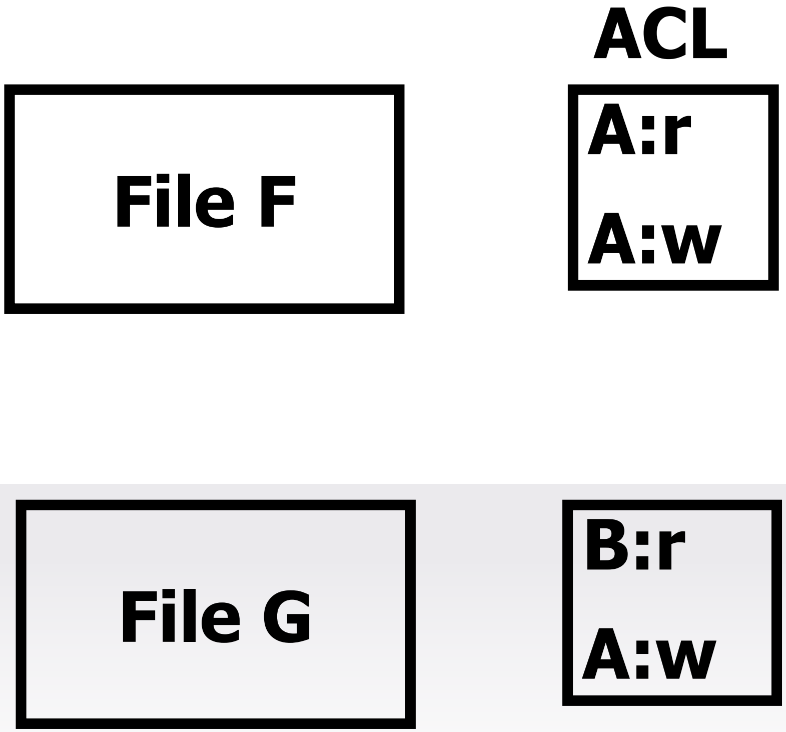
Eve

Alice does not want Eve to have access to EMP

Bob can make a copy of EMP and grant Eve the ability to access the copy.

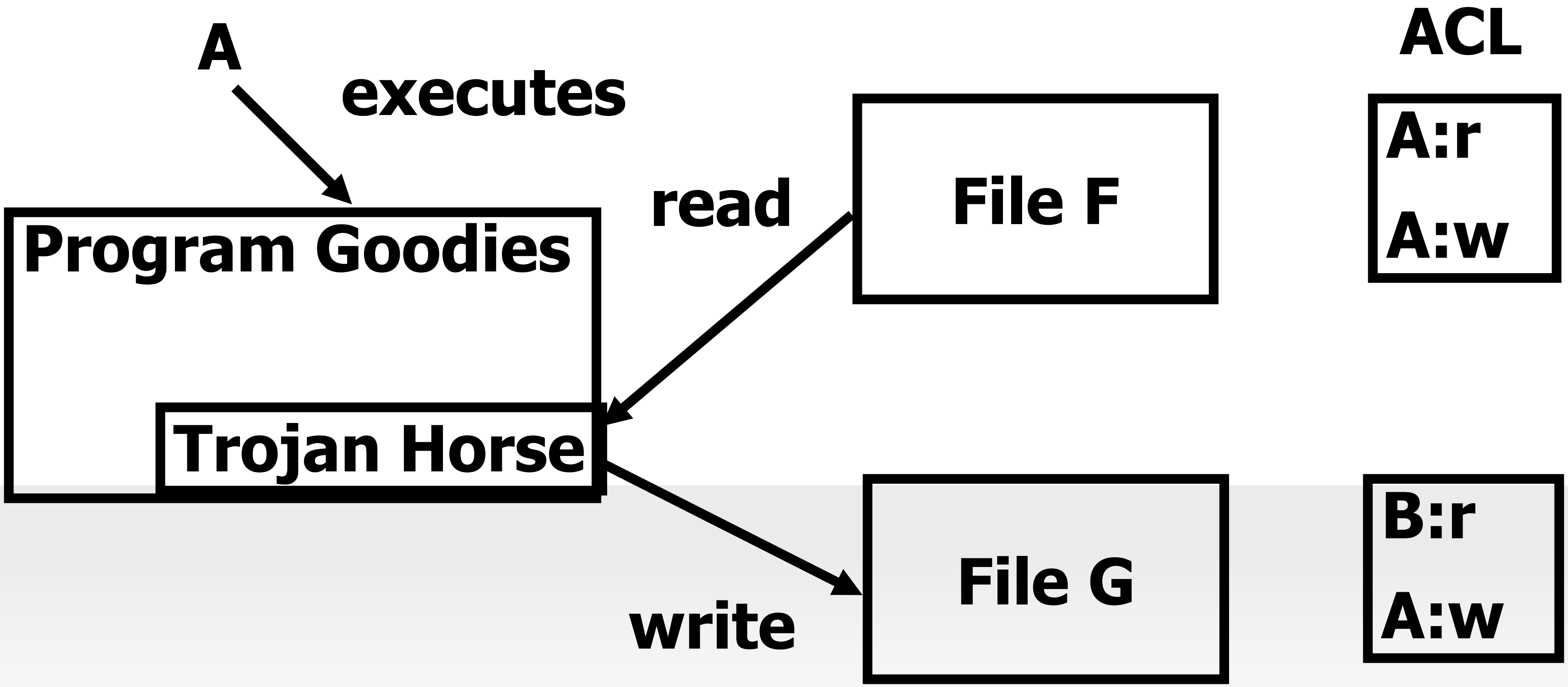
This case is a simplistic version of what can be much more pathological… The Trojan Horse…

DAC Model Weaknesses (Trojan Ver.)



Bob cannot read file F

DAC Model Weaknesses (Trojan Ver.)



A

executes

Program Goodies

Trojan Horse

read

File F

ACL

A:r

A:w

write

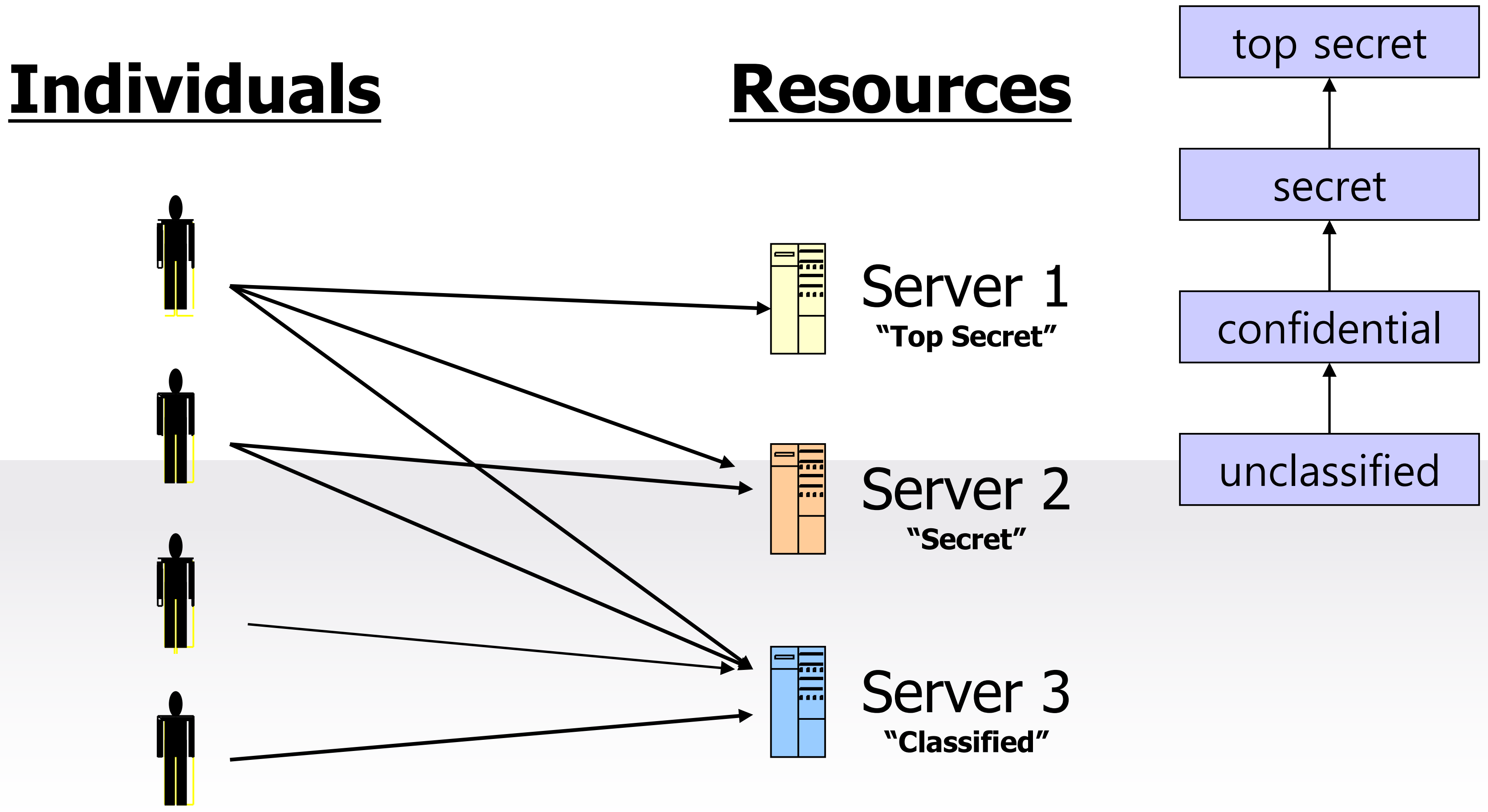
File G

B:r

A:w

B can read contents of file F copied to file G

MAC Model



Individuals

Resources

Server 1
"Top Secret"

Server 2
"Secret"

Server 3
"Classified"

top secret

secret

confidential

unclassified

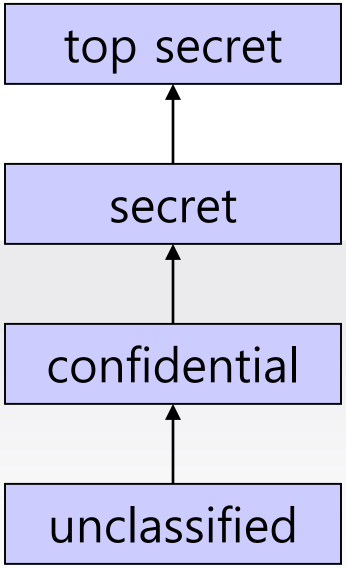
MAC Model

- Mandatory access control (MAC) model protects information by assigning clearances to users which define what the sensitivity level of information they are allowed to access.

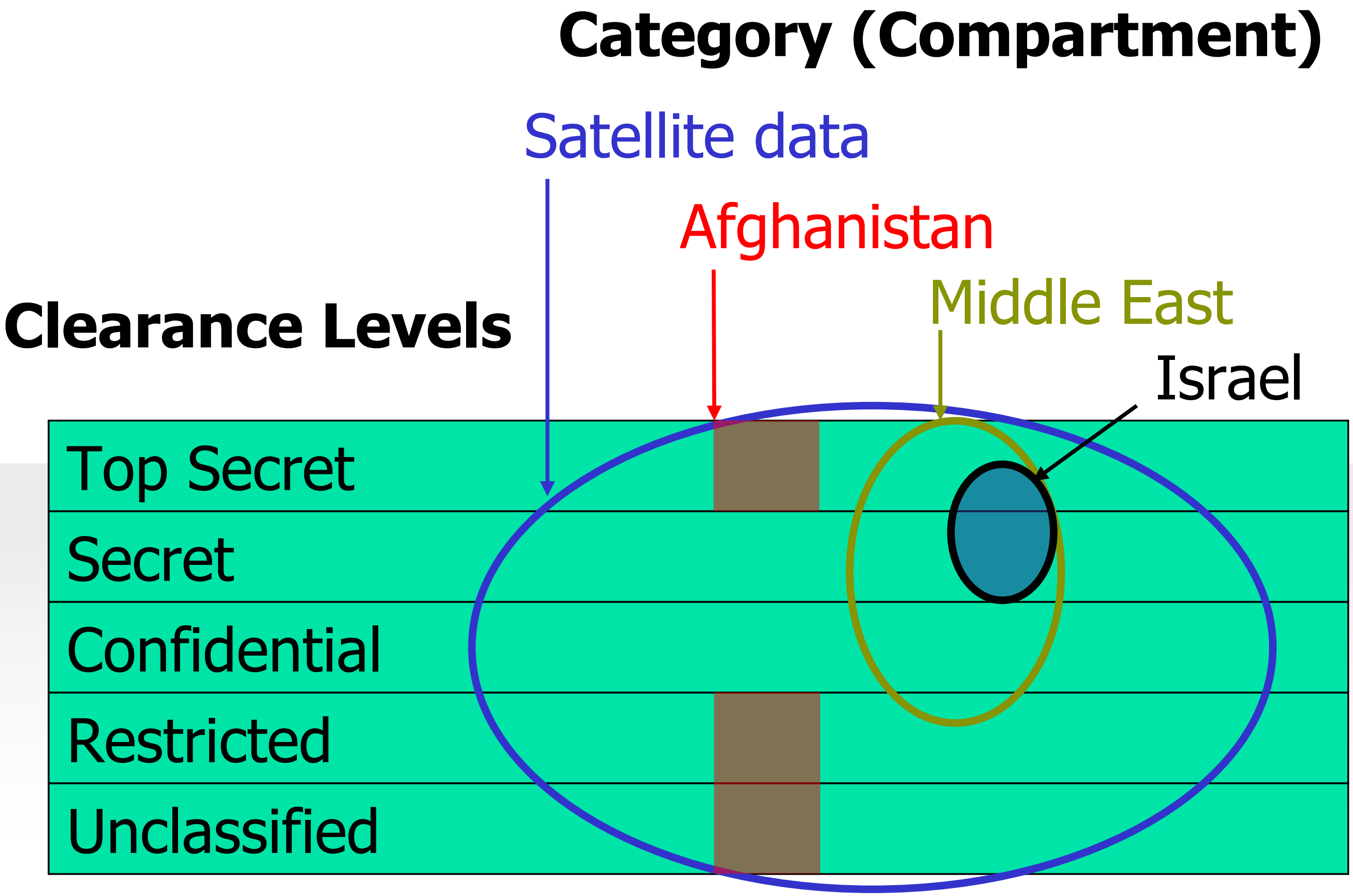
- The clearances assigned to users under MAC are strictly enforced;
  - Permissions (   ) at a user's discretion.

MAC : Traditional Model

- **Traditional MAC :** hierarchical security levels ( )



MAC : Traditional Model



Category (Compartment)

Satellite data

Afghanistan

Middle East

Israel

Clearance Levels

Top Secret

Secret

Confidential

Restricted

Unclassified

MAC : MLS Model

- **MLS(Multi-Level Security) :** By the
  (   ) policy, categories
  ((   ) by the subset relation)
  are used as well as the security levels
  ((   ) in lattices.

MAC : MLS Model

- Classification of personnel and data
  - Class = ⟨rank, compartment⟩

- Access Control
  - Document X is restricted : $\langle \text{rank}_X, \text{compartment}_X \rangle$
  - A Person has a clearance : $\langle \text{rank}_P, \text{compartment}_P \rangle$
  - If $\langle \text{rank}_X, \text{compartment}_X \rangle \leq \langle \text{rank}_P, \text{compartment}_P \rangle \rightarrow$ P may access X

- Dominance relation (≤)
  - $\langle \text{rank}_X, \text{compartment}_X \rangle \leq \langle \text{rank}_P, \text{compartment}_P \rangle \Leftrightarrow \text{rank}_X \leq \text{rank}_P$ and $\text{compartment}_X \subseteq \text{compartment}_P$
  - $\langle \text{rank}_X, \text{compartment}_X \rangle$ is dominated by $\langle \text{rank}_P, \text{compartment}_P \rangle$

BLP Model of MLS

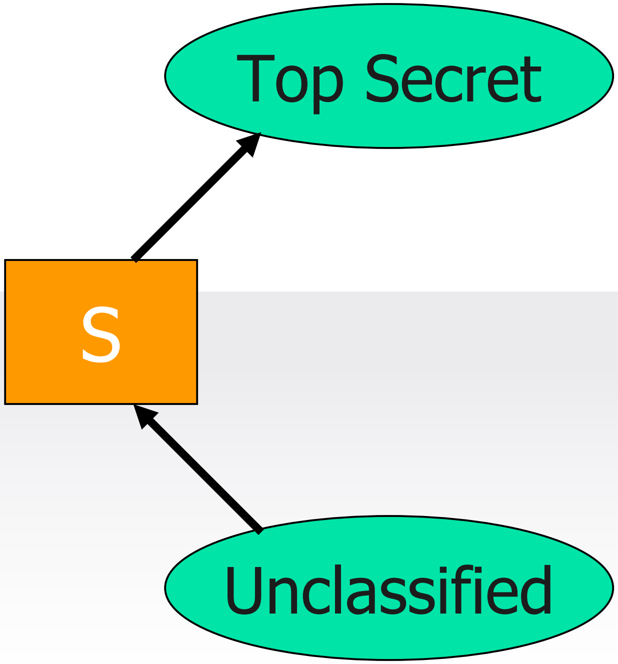
- Introduce by Mathematicians ( ) and ( ) of MITRE in 1973.
- Bell and LaPadula gave a formal, mathematical MAC model of MLS to provide ( ) of correctness.
  - Implements an ( ) using a lattice with compartments and an access control matrix. i.e., augment DAC(DS-Property) with MAC(SS-Property,) to enforce information flow policies.
- BLP information flow policy :
  - Information is allowed to flow from less trustworthy subjects to more trustworthy subjects. i.e., Information cannot leak to subjects who are not cleared for the information.

BLP Model of MLS

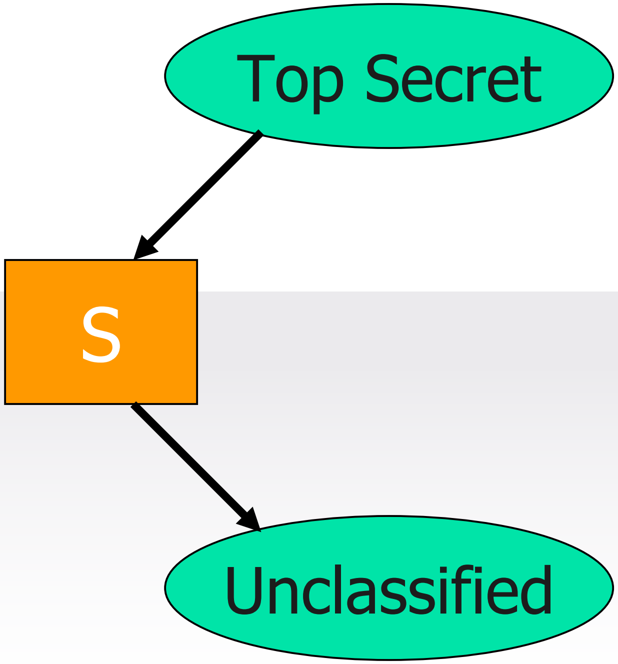
- However, there are still some problems with the BLP formulation of MLS. These include :
  - only dealing with confidentiality, **not** with **integrity**.
  - having **no** policies regulating the **modification of access rights**.
  - Does **not** deal with information flow through **covert channels**.

BLP Model in a Nutshell

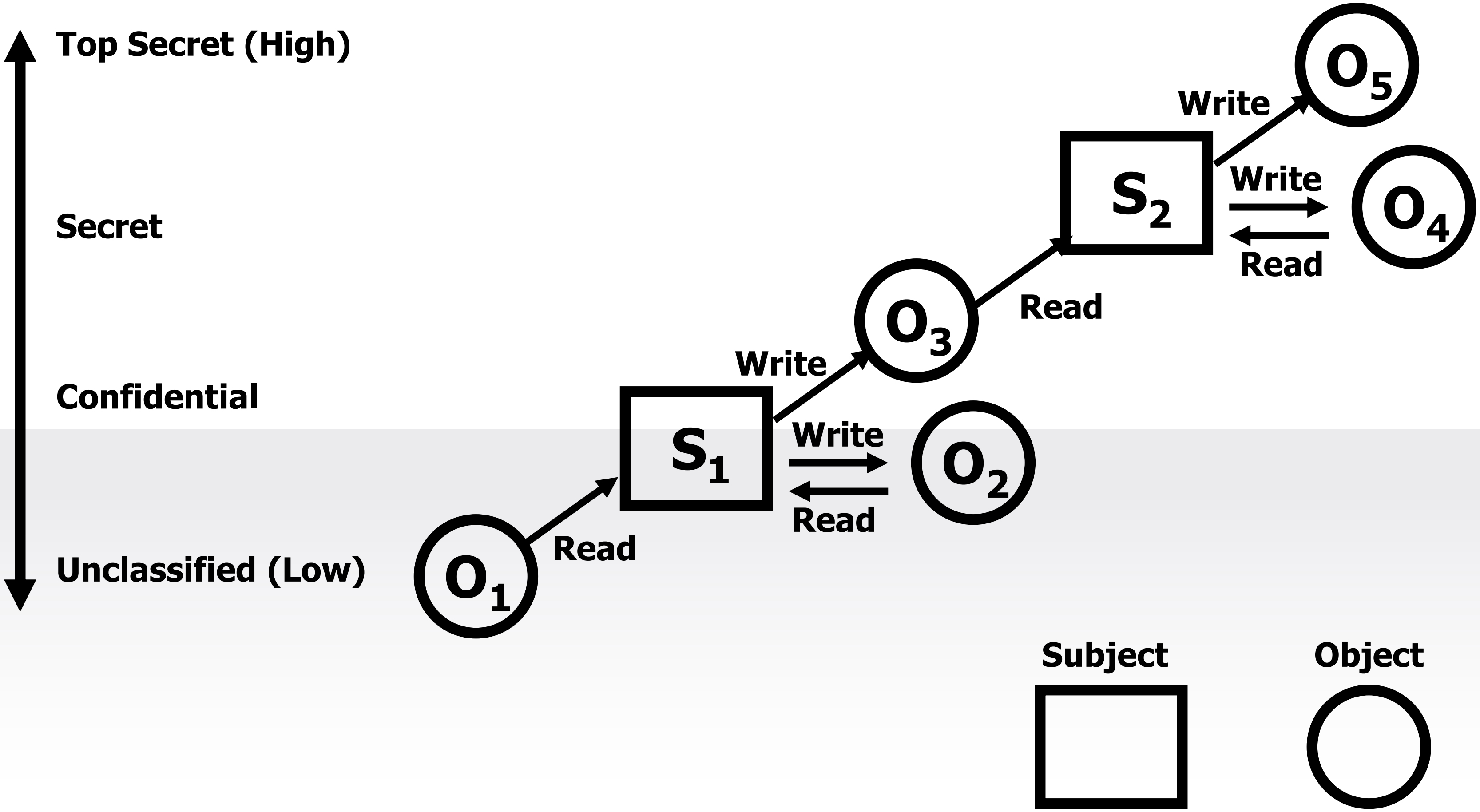
Read down, write up



Read up, write down



BLP Model in a Nutshell



BLP Model in Formal

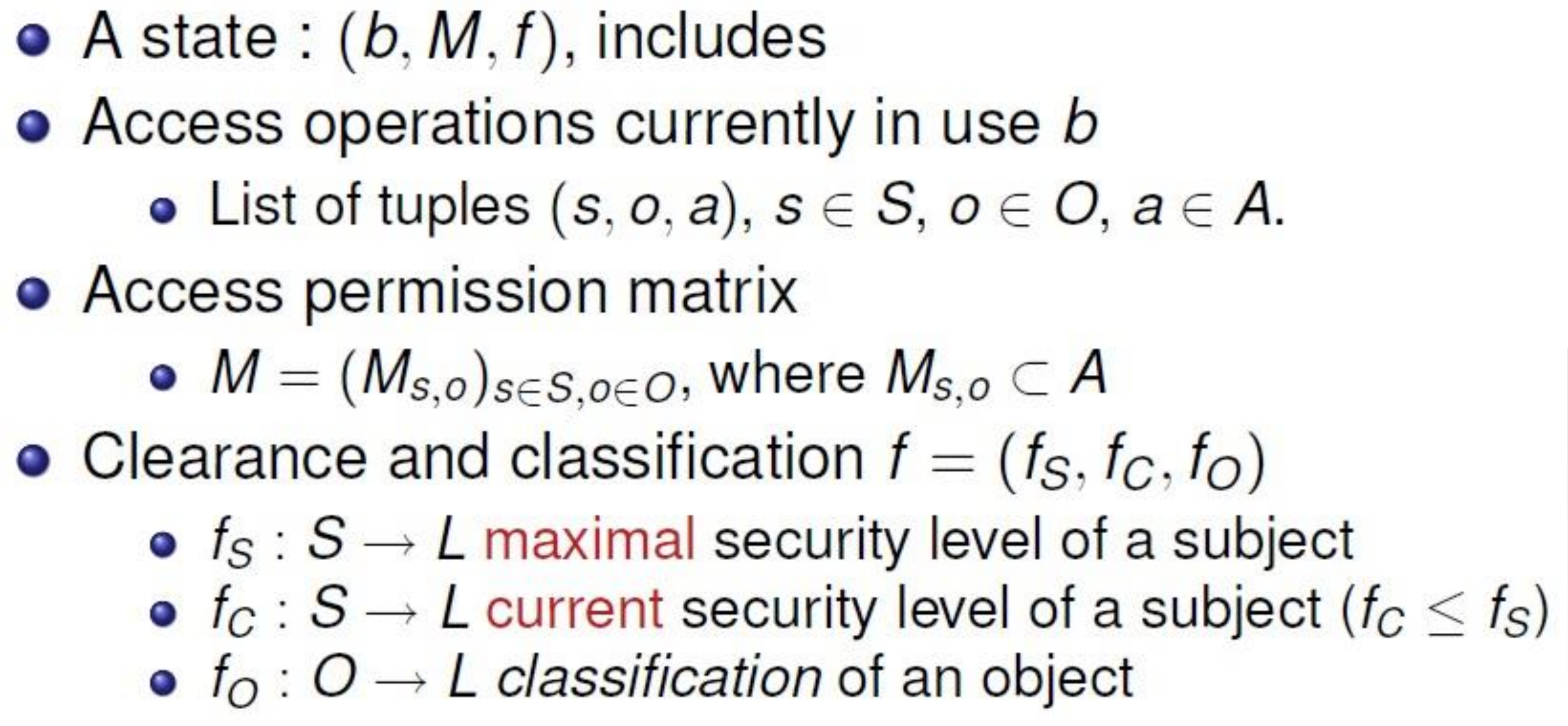
- Task was to propose a theory of multi-level security
  - supported by a mechanism implemented in a (   )
  - prevents unwanted (   )

BLP Model in Formal

- Elements of Access Control
  - a set of subjects **S**
  - a set of objects **O**
  - set of access operations **A** = {execute, read, append, write}
  - A set of security levels **L**, with a partial ordering <=

BLP Model in Formal

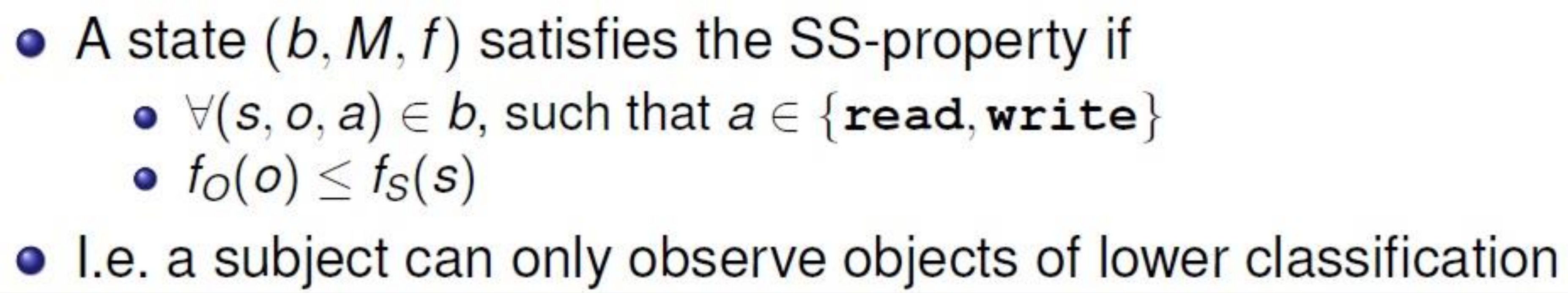
- The State Set



- A state : (b, M, f), includes
- Access operations currently in use b
  - List of tuples (s, o, a), $s \in S$, $o \in O$, $a \in A$.
- Access permission matrix
  - $M = (M_{s,o})_{s \in S, o \in O}$, where $M_{s,o} \subset A$
- Clearance and classification $f = (f_S, f_C, f_O)$
  - $f_S : S \rightarrow L$ maximal security level of a subject
  - $f_C : S \rightarrow L$ current security level of a subject $(f_C \leq f_S)$
  - $f_O : O \rightarrow L$ *classification* of an object

BLP Model in Formal

- Simple Security Property (SS-Property)

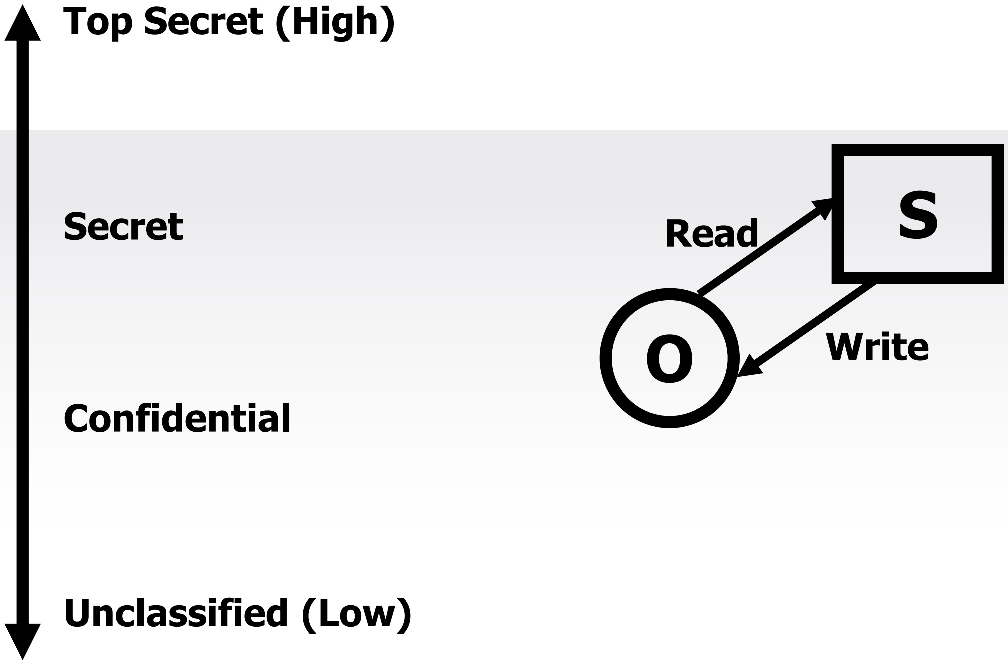


- A state (b, M, f) satisfies the SS-property if
  - $\forall (s, o, a) \in b$, such that $a \in \{\mathtt{read}, \mathtt{write}\}$
  - $f_O(o) \leq f_S(s)$
- I.e. a subject can only observe objects of lower classification

BLP Model in Formal

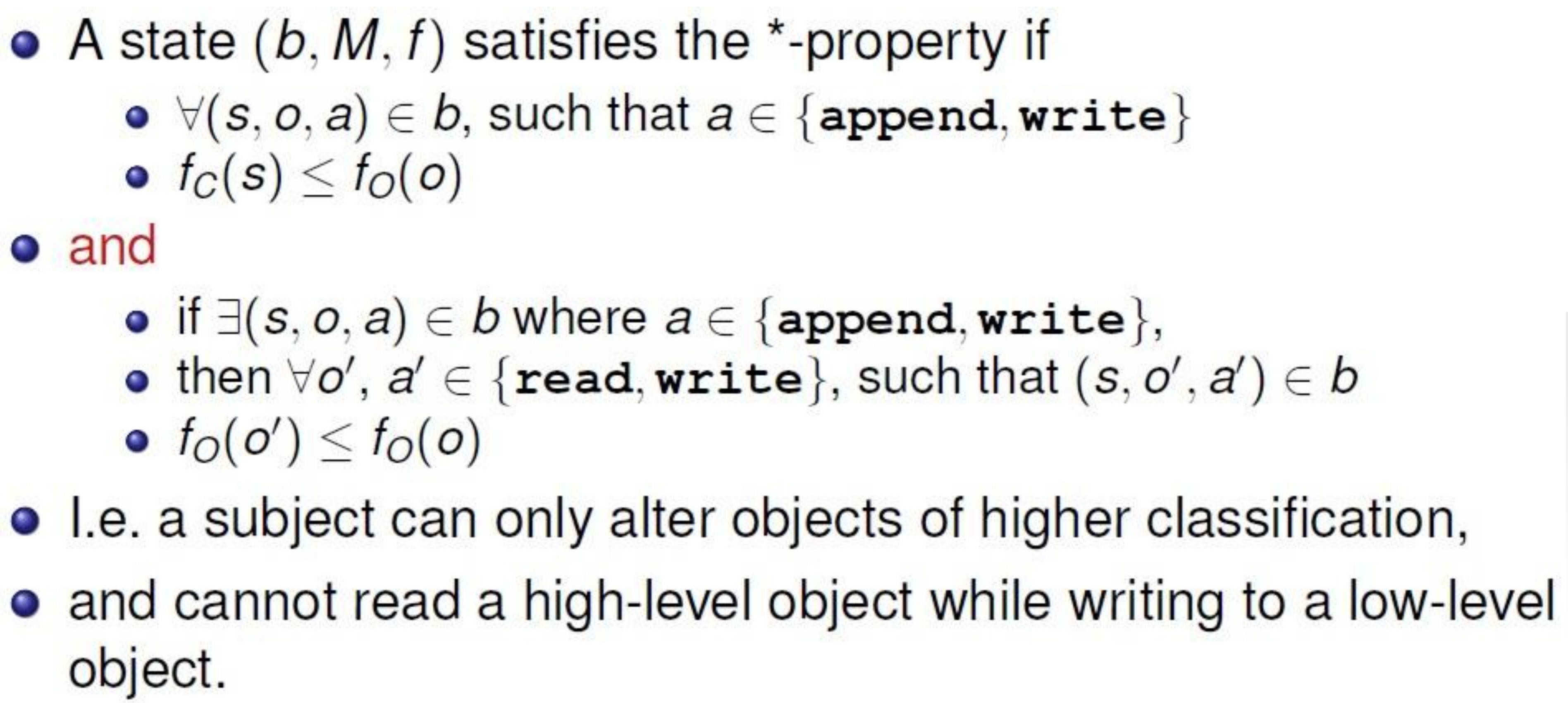
- Simple Security Property (SS-Property)

<Read Down>



BLP Model in Formal

- *-Property (Star-Property)



- A state (b, M, f) satisfies the *-property if
  - $\forall (s, o, a) \in b$, such that $a \in \{\mathtt{append}, \mathtt{write}\}$
  - $f_C(s) \leq f_O(o)$
- and
  - if $\exists (s, o, a) \in b$ where $a \in \{\mathtt{append}, \mathtt{write}\}$,
  - then $\forall o', a' \in \{\mathtt{read}, \mathtt{write}\}$, such that $(s, o', a') \in b$
  - $f_O(o') \leq f_O(o)$
- I.e. a subject can only alter objects of higher classification,
- and cannot read a high-level object while writing to a low-level object.

BLP Model in Formal

- *-Property (Star-Property)

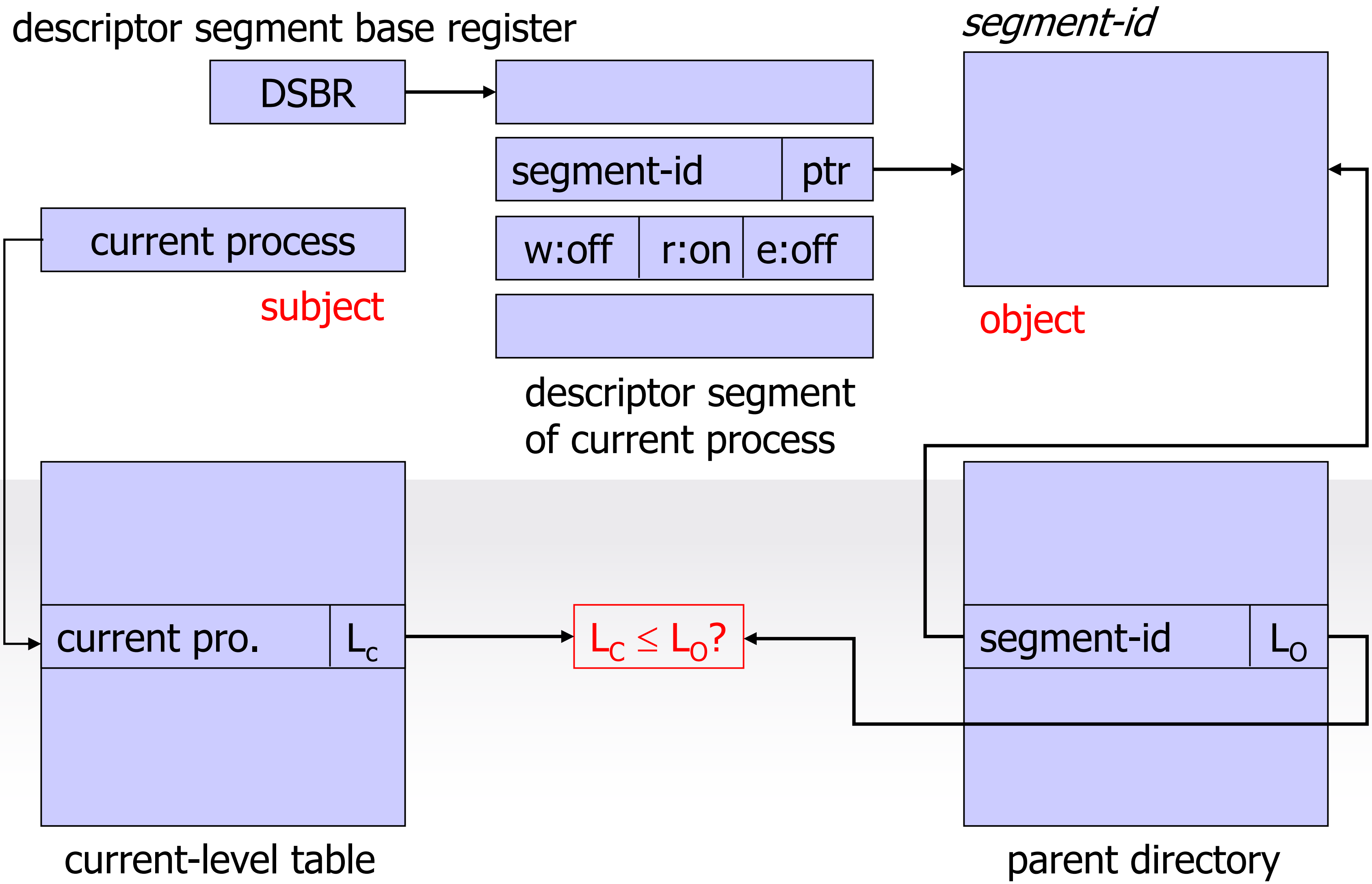
BLP Model in Formal

- Discretionary Security Property (DS-Property)
  - **Access must be permitted by the access control matrix $M_{s,o}$.**

Real World Examples Built on BLP

- MULTICS for the Air Force Data Services Centre (time-sharing OS)
- MITRE brassboard kernel
- SIGMA message system
- KSOS (Kernelized Secure Operating System)
- SCOMP (Secure Communications Processor)
- PSOS (Provably Secure Operating System)
- SELinux
- multi-level Database Management Systems

[Note] MULTICS



descriptor segment base register

DSBR

segment-id ptr

w:off r:on e:off

descriptor segment of current process

segment-id

object

current process

subject

current pro. L_c

current-level table

$L_C \le L_O$?

segment-id L_O

parent directory

www.wiley.com/go/gollmann

The Criticism of McLean

- What happens if we have …
  - Proposed System Z = BLP + (   )
- Is the system **secure**?
- It **satisfies** every security property of BLP!

The Criticism of McLean

- Tranquility
  - McLean's scenario is really (   ) for BLP.
  - BLP considered **tranquil** systems,
    -
  - Either a system or an operation may be tranquil
    - A tranquil operation does not change access rights.
    - A tranquil system has no non-tranquil operations.

The Criticism of McLean

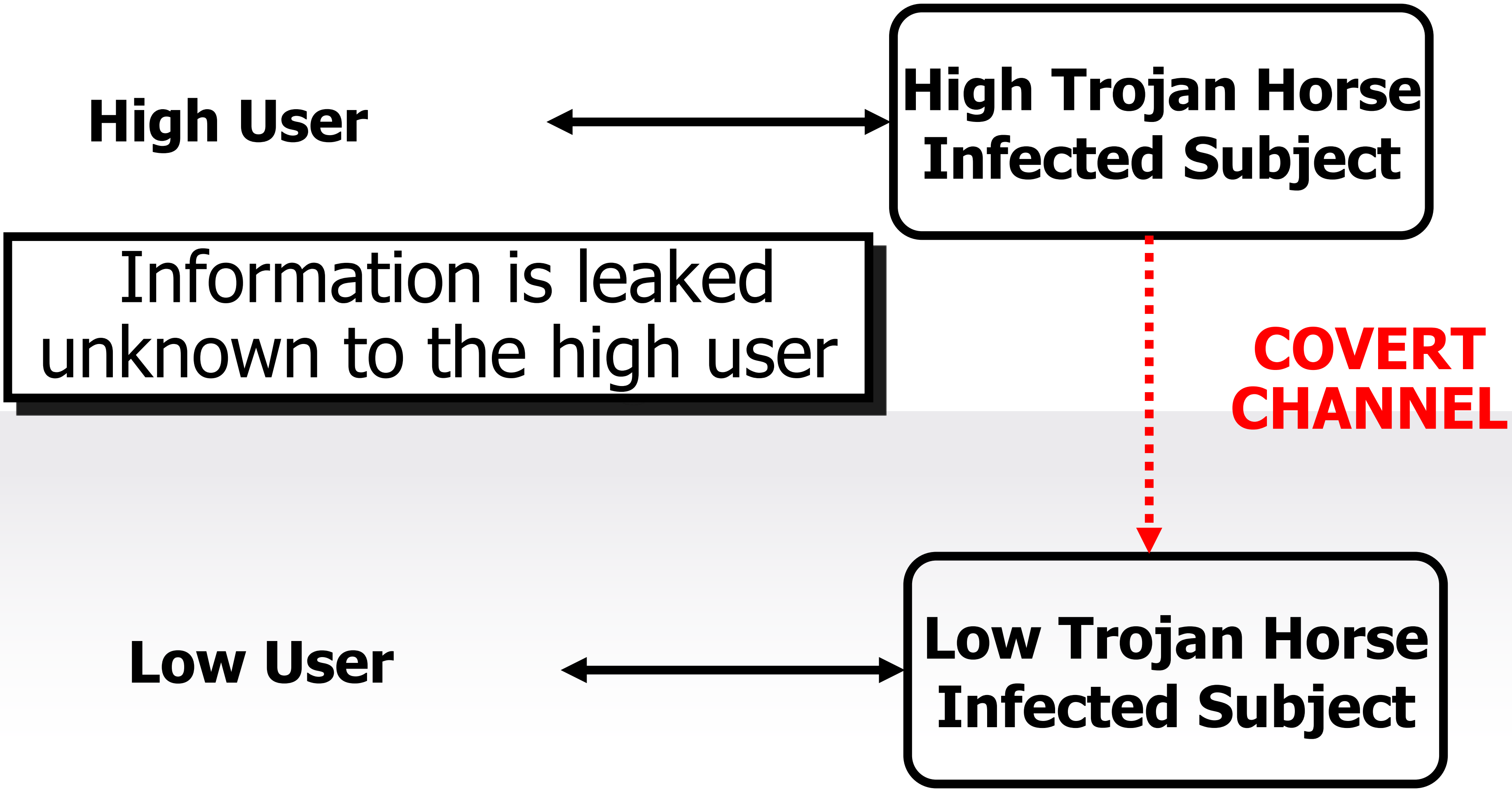
- Tranquility
  - Tranquility is a particular concern when
    - operation tries to **remove** an access right currently in use.
  - How should this be resolved?

Main Contributions of BLP

Covert Channels

- Covert channels (   ) be blocked by *-property.
  - **Convert channel :** A communication channel is covert if it is neither designed nor intended to transfer information at all.
- It is generally very difficult, if not impossible, to block all covert channels.
- One can try to (   ) of covert channels.
- Military requires cryptographic components be (   ).
  - to avoid trojan horse leaking keys through covert channels.

Covert Channels

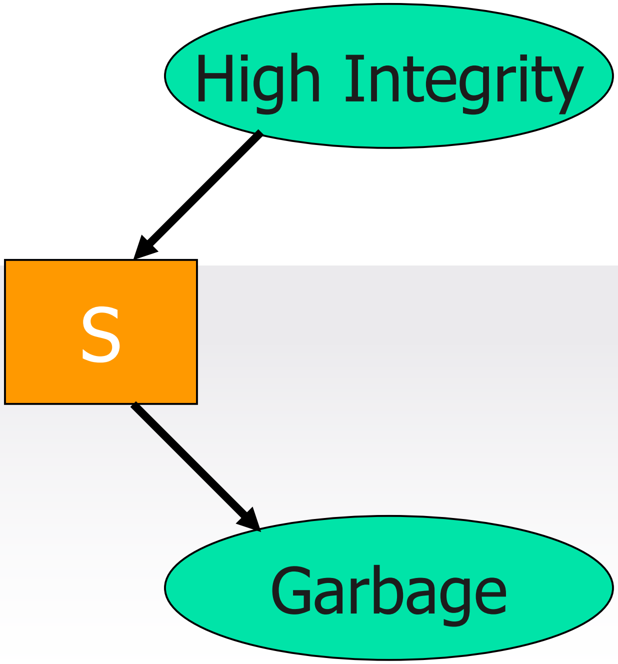

High User
High Trojan Horse Infected Subject
Information is leaked unknown to the high user
COVERT CHANNEL
Low User
Low Trojan Horse Infected Subject

Biba Model

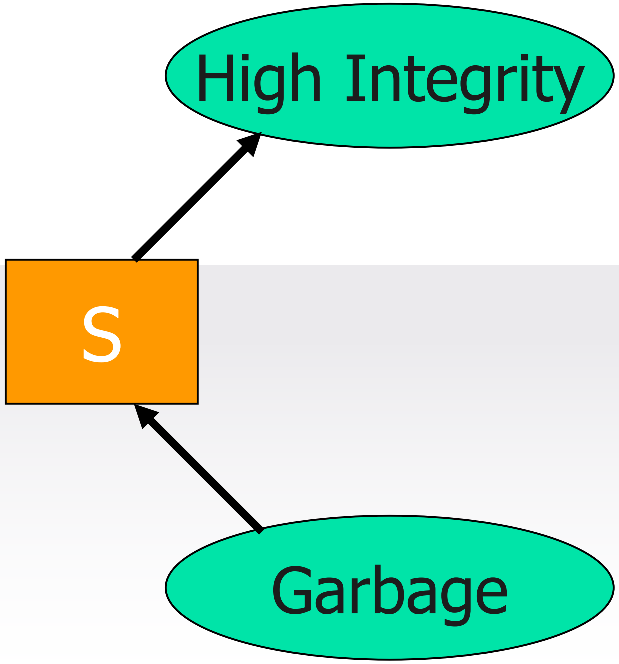
- ( ) to BLP (1977).
- State machine model similar to BLP.
- Defines ( ) levels, analogous to sensitivity/security levels of BLP.
    - No read-down (SS-Property)
        - Subjects can only view content at or above their own integrity level.
    - No write-up (*-property)
        - Subjects can only create content at or below their own integrity level.

Biba Model in a Nutshell

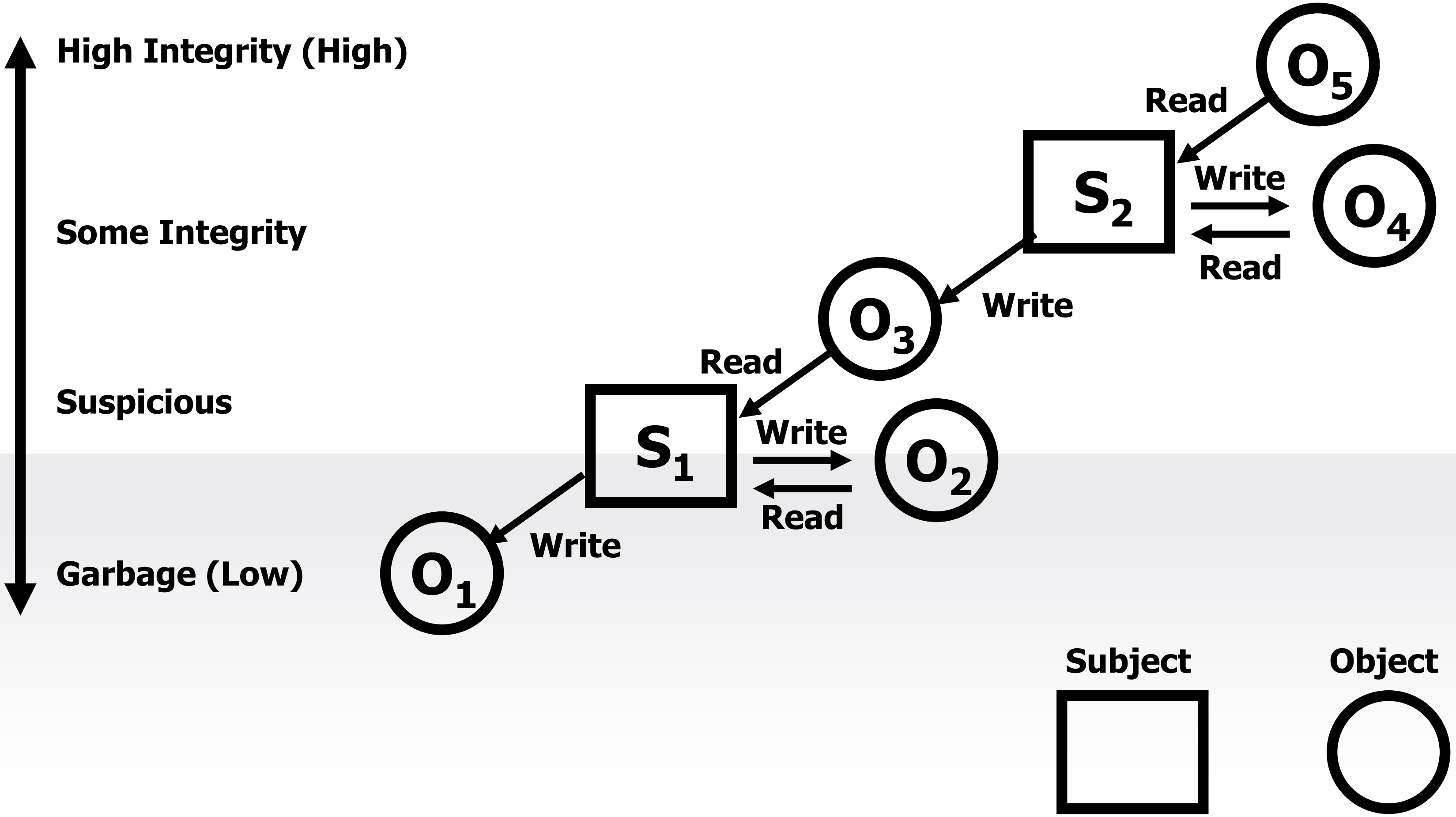
Read up, write down



Read down, write up



Biba Model in a Nutshell

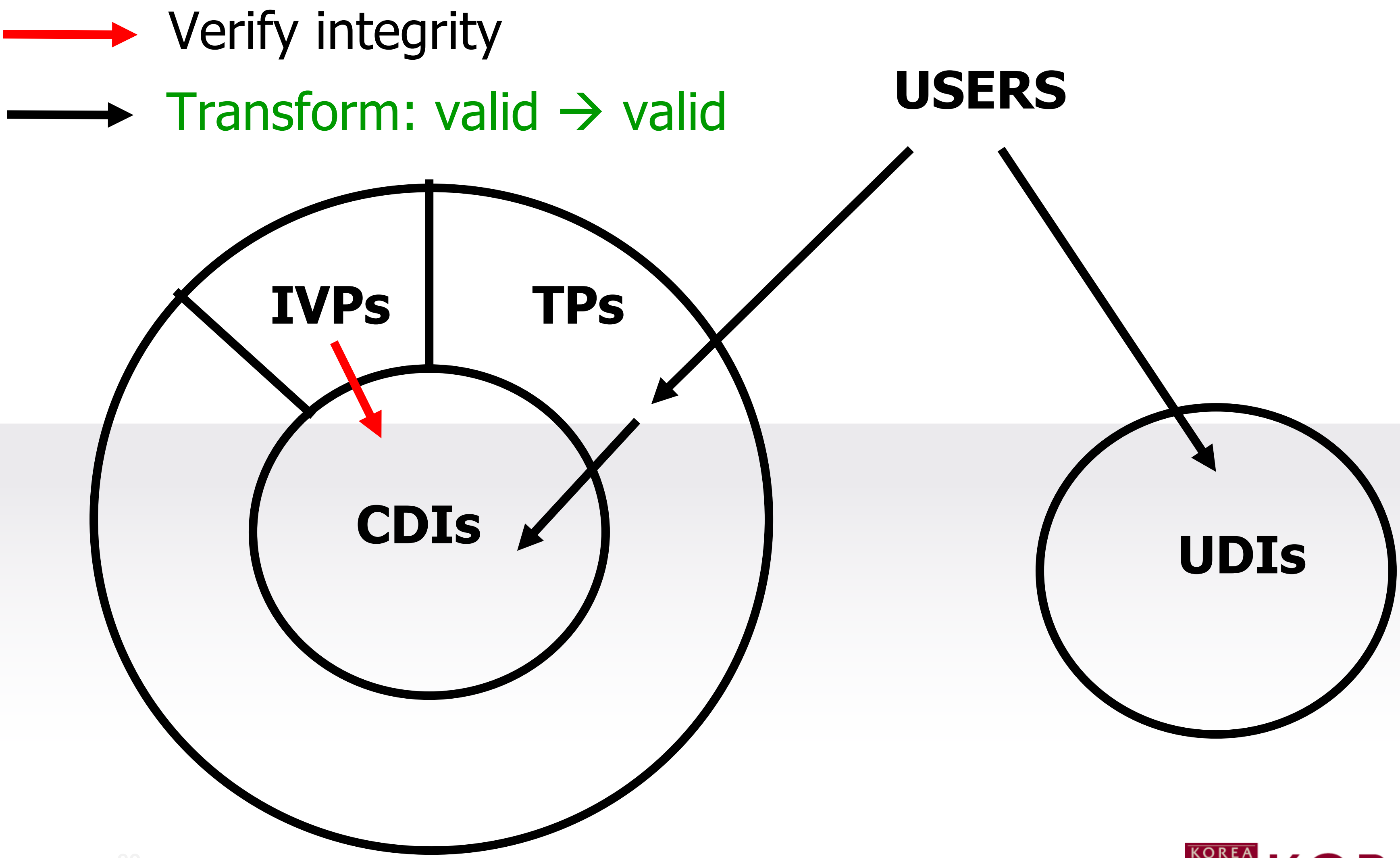


Clark-Wilson Model

- Authors address security requirements of ( ) applications (D. Clark, D. Wilson, A Comparison of Commercial and Military Computer Security Policies, IEEE Symposium on Security and Privacy, 1987).
  - In opposition to **military** requirements.

- Enforcing Integrity via ( ).
  - Well-formed transactions move a system from one consistent state to another consistent state.
  - Users have access to ( ) rather than ( ) itself.

Clark-Wilson Model



Verify integrity

Transform: valid → valid

USERS

IVPs

TPs

CDIs

UDIs

Clark-Wilson Model

- **Constrained Data Items (CDI) :** data subject to Integrity Control
  - Eg. Account balances
  - Integrity constraints constrain the values of the CDIs

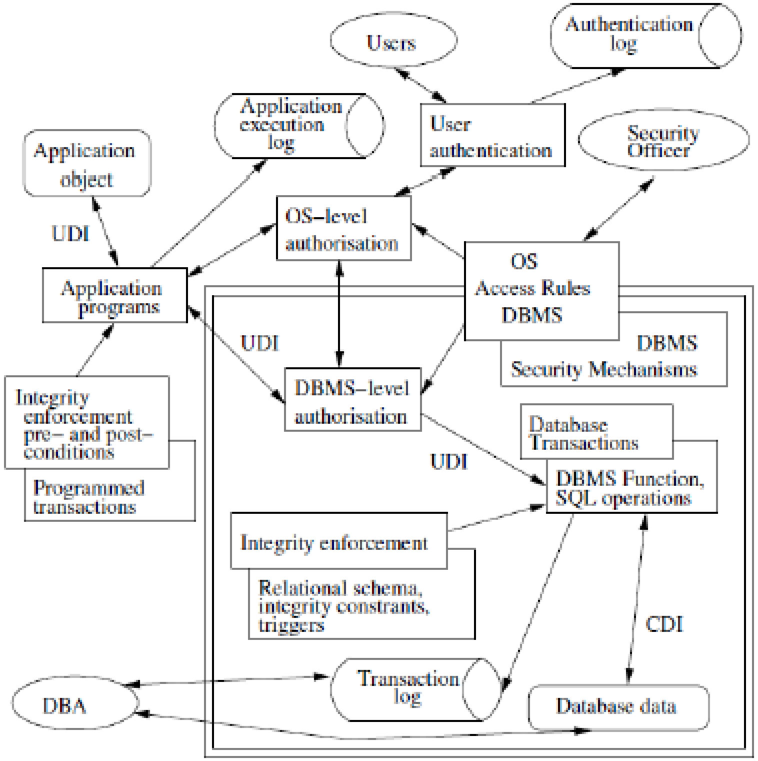
- **Unconstrained Data Items (UDI) :** data not subject to IC
  - Eg. Gifts given to the account holders

- **Integrity Verification Procedures (IVP)**
  - Test CDIs' conformance to integrity constraints at the time IVPs are run (checking that accounts balance)

- **Transformation Procedures (TP) :** E.g.,
  - Depositing money
  - Withdrawing money
  - Money transfer etc.

Real World Examples Built on CW

- Clark-Wilson security model is often implemented in modern database management systems (DBMS) such as: Oracle, DB2, MS SQL, and MySQL.



Reference: Secure Database Development and the Clark-Wilson Security Model, X.Ge, F.Polack, R.Laleau, University of York, UK.

Harrison-Ruzo-Ullman Model

- BLP has no policies for (   ) access rights or for the (   ) and (   ) of subjects and objects.
- The Harrison-Ruzzo-Ullman (HRU) model defines authorization systems that address these issues (Michael A. Harrison, Walter L. Ruzzo, Jeffrey D. Ullman: Protection in Operating Systems. Commun. ACM 19(8): 461-471 (1976)).
- Study whether any properties about reachable sets can be stated.
  - These are "(   )"
  - i.e. can a sequence of transitions reach a state of the matrix with (x, o, r)?
- Why? This would be used to build a "security argument" that the access control policy realizes some properties of the security policy!

Harrison-Ruzo-Ullman Model

- The components of the HRU model:
  - set of subjects ***S***
  - set of objects ***O***
  - set of access rights ***R***
  - access matrix $\boldsymbol{M} = (M_{s,o})_{s \in S, o \in O}$: entry $M_{s,o}$ is a subset of R giving the rights subject s has on object o

- Six primitive operations for manipulating subjects, objects, and the access matrix :
  - enter r into $M_{s,o}$
  - delete r from $M_{s,o}$
  - create subject s
  - delete subject s
  - create object o
  - delete object o

Lessons Learned in HRU Model

- The access matrix describes the state of the system;
  - Commands change the access matrix $M_{s,o} \rightarrow M_{s,o'}$.
- Checks need to be done in order to avoid undesirable access rights to be granted.
- HRU model has some definitions and theorems about the decidability of the safety of the system.
  - Saying that HRU model **does not help to verify safety in its ( ), but verification is possible with ( )**.

Lessons Learned in HRU Model

- The moral of those theorems is :

Chinese Wall Model in a Nutshell

Introduced by Brewer and Nash in 1989

- The motivation for this work was to avoid that sensitive information concerning a company be disclosed to (   ) through the work of financial consultants.
  - Provides access controls that change dynamically!

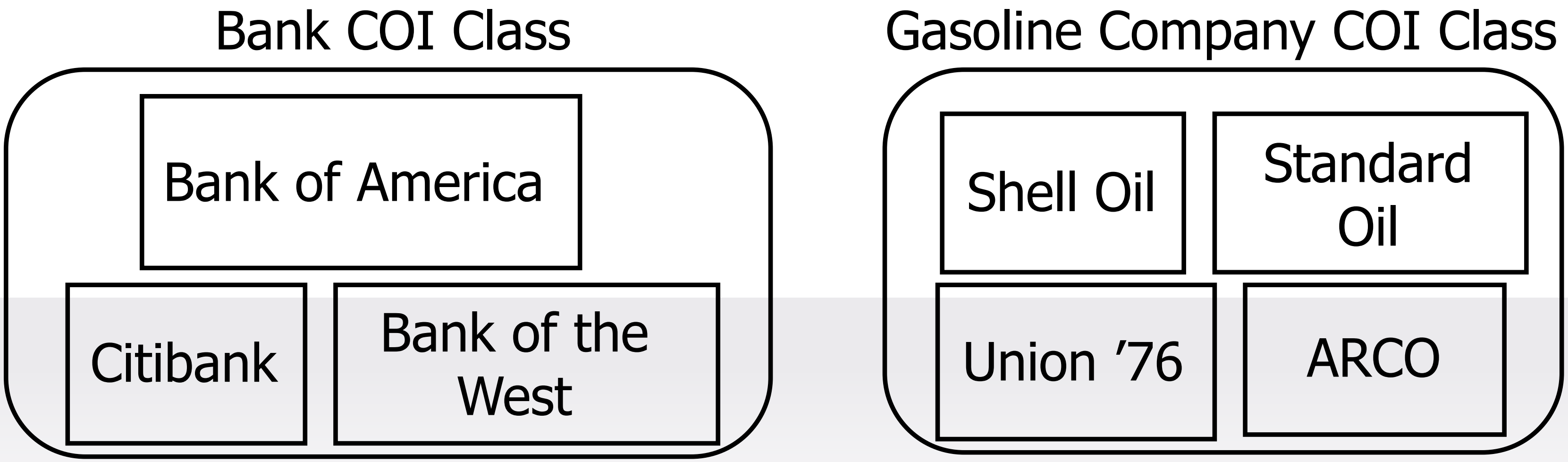
Chinese Wall Model in a Nutshell

- How it works :
  - Users have no “wall” initially.
  - Once any given file is accessed, files with competitor information become inaccessible.
  - Unlike other models, access control rules change (   ).

Chinese Wall Model

- We have :
    - Set of Companies
    - Analysts – Subjects
    - Items of Information – Objects
    - Objects concerning the same company – Company Dataset
    - Companies in competition – Conflict of Interest class. A set of companies that should not learn about the contents of the object
    - Object security label – pair (CDataset, CoI Class)
    - Sanitised Information – purged of sensitive information, thus has no Conflict of Interest
    - Object access history

Chinese Wall Model



Bank COI Class

Bank of America

Citibank

Bank of the West

Gasoline Company COI Class

Shell Oil

Standard Oil

Union '76

ARCO

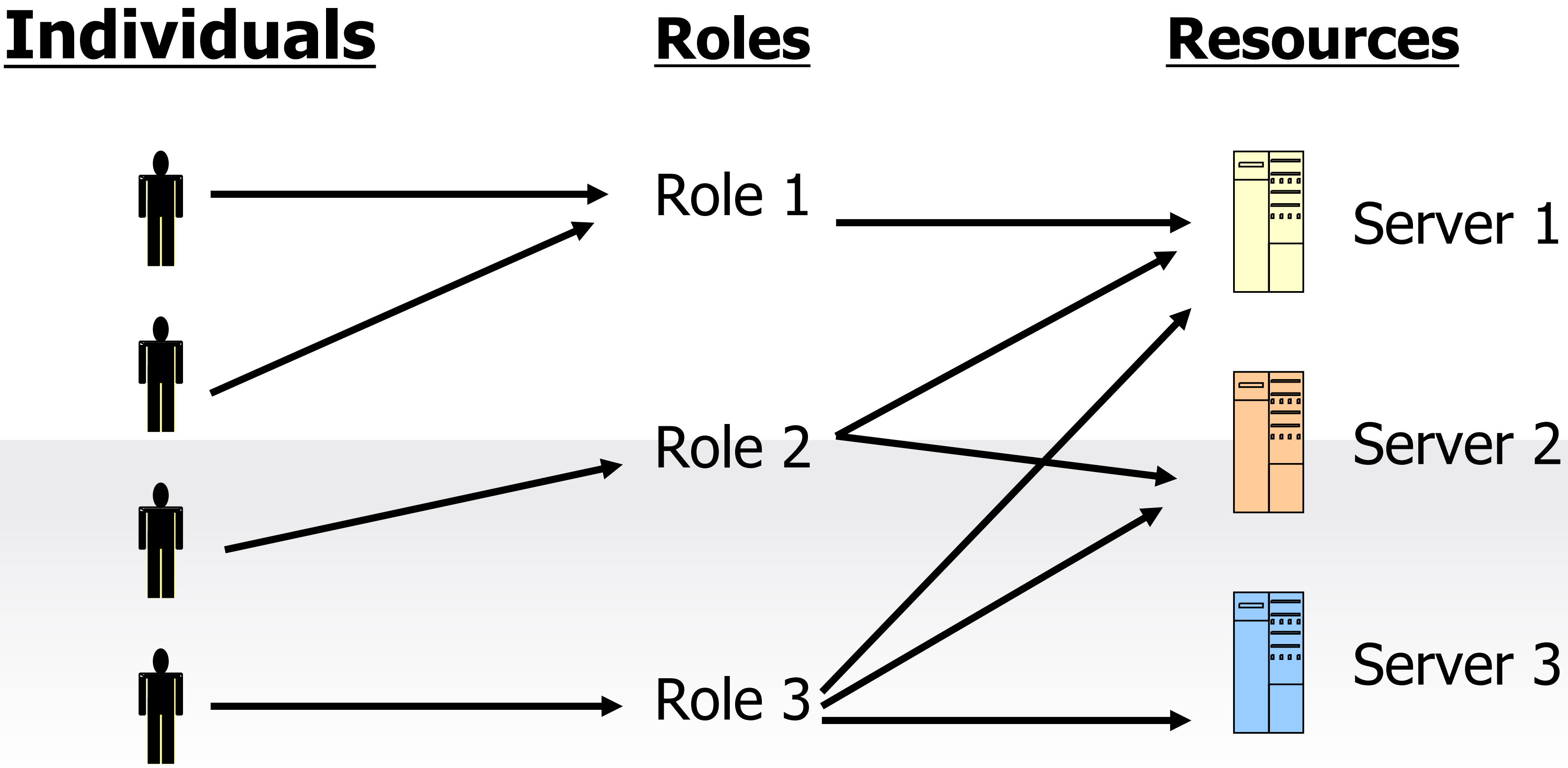
Chinese Wall Model

- **SS-Property :**
  - "Prevents a subject from being exposed to a conflict of interest."
  - Access granted if object belongs to :
    - Company dataset already held by user or
    - An entirely different conflict of interest class.

Chinese Wall Model

- ***-Property :**
  - "Prevents un-sanitised information flowing out of a company dataset."
  - Write access granted if no other object can be read that :
    - Belongs to a different company dataset.
    - Contains un-sanitised information.

RBAC Model



Individuals

Roles

Resources

Role 1

Role 2

Role 3

Server 1

Server 2

Server 3

User’s change frequently, Roles don’t

RBAC Model

- Originally developed by David Ferraiolo and Rick Kuhn (1992), and by Ravi Sandhu and colleagues (1996).

- A Role-Based Access Control (RBAC) model, also called ‘nondiscretionary access control’, allows access to resources to be based on the (   ) the user holds within the company.
  - It is referred to as (   ) because assigning a user to a role is unavoidably imposed.

- An RBAC is the best system for a company that has high employee turnover.

RBAC Model

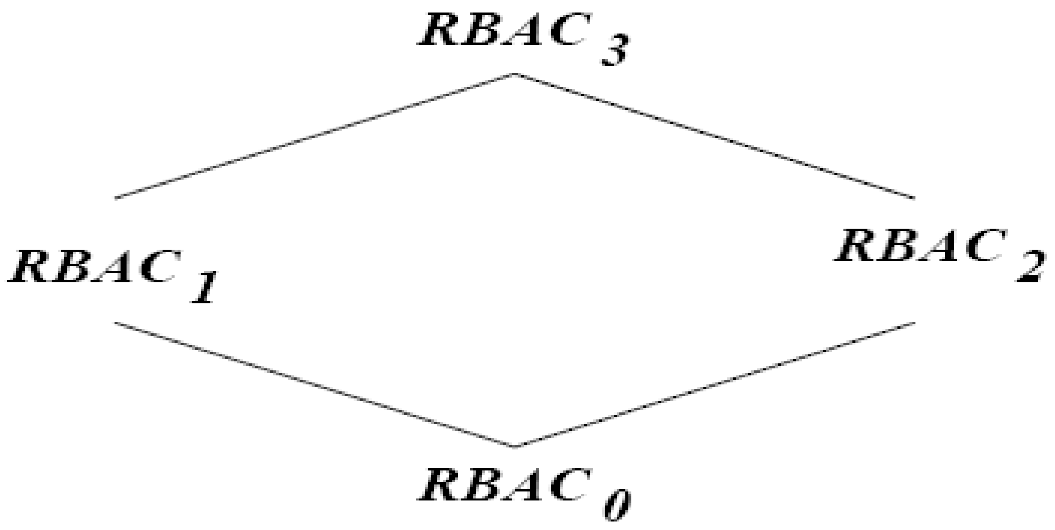
- Roles implemented in
  - Window NT (as global and local groups)
  - IBM’s OS/400
  - Oracle 8 onwards
  - .NET framework

- Now official **standard** – approved on Feb. 19 2004

Categorization of RBAC Model

- Core RBAC (also called Flat RBAC) ($RBAC_0$)
- Hierarchical RBAC ($RBAC_1$)
  - General role hierarchies
  - Limited role hierarchies
- Constrained RBAC ($RBAC_2$)
  - Static Separation of Duty Relations
  - Dynamic Separation of Duty Relations
- Consolidated RBAC ($RBAC_3$)
  - Combines Constraints and Role Hierarchies

Categorization of RBAC Model

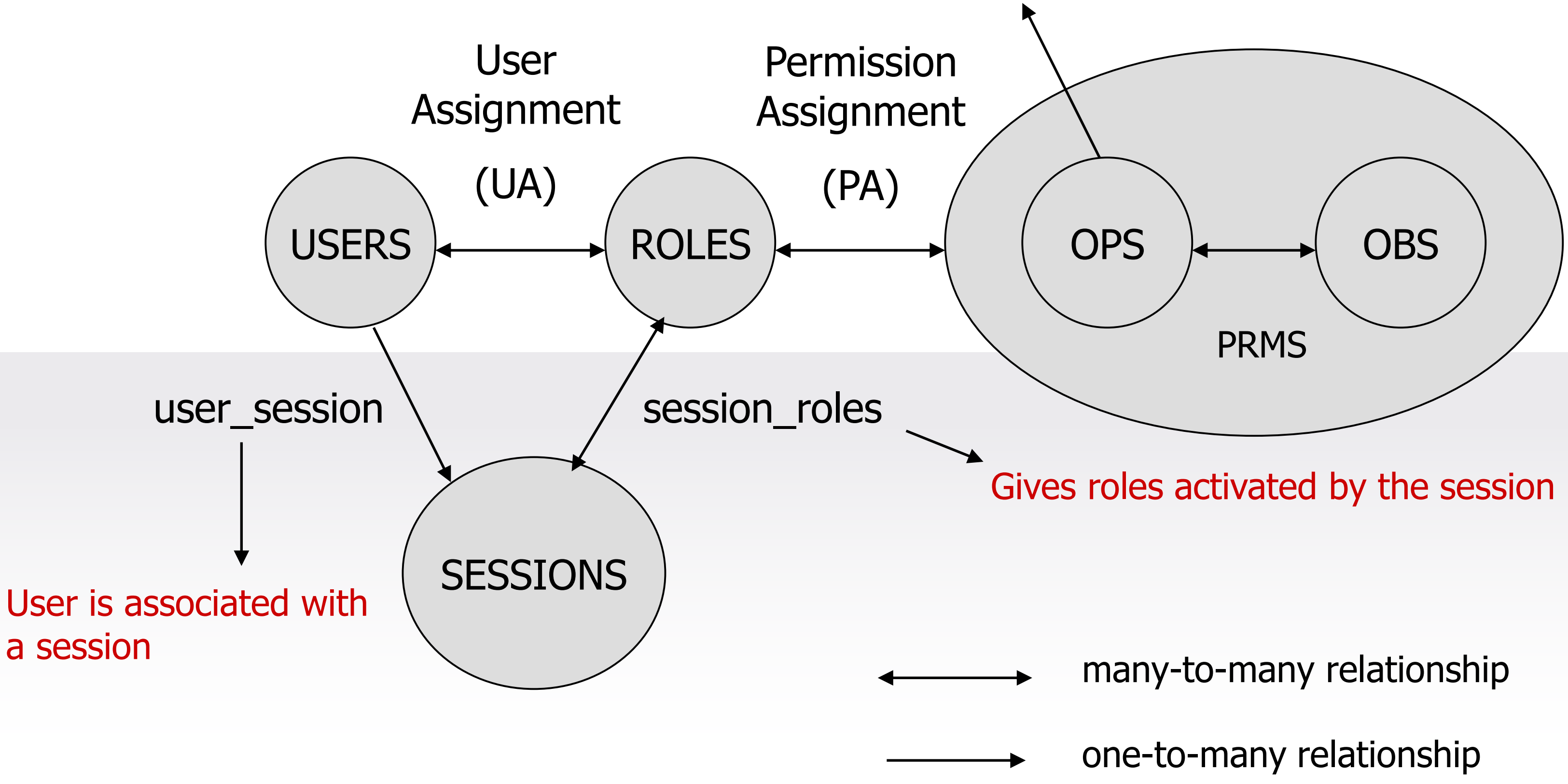


(a) Relationship among RBAC models

Core RBAC

File system operations: read, write and execute

DBMS operations: Insert, delete, append and update



Gives roles activated by the session

User is associated with a session

Core RBAC

- **UA** : user assignments
  - Many-to-many
- **PA** : Permission assignment
  - Many-to-many
- **Session** : mapping of a user to possibly many roles
  - **Permissions** : union of permissions from all roles
  - Each session is associated with a single user
  - User may have multiple sessions at the same time (one to many relationship)
  - In each session, multiple roles can be activated simultaneously (many to many relationship)

Core RBAC

- **U**sers, **R**oles, **P**ermissions, **S**essions

- $PA \subseteq P \times R$ (many-to-many)

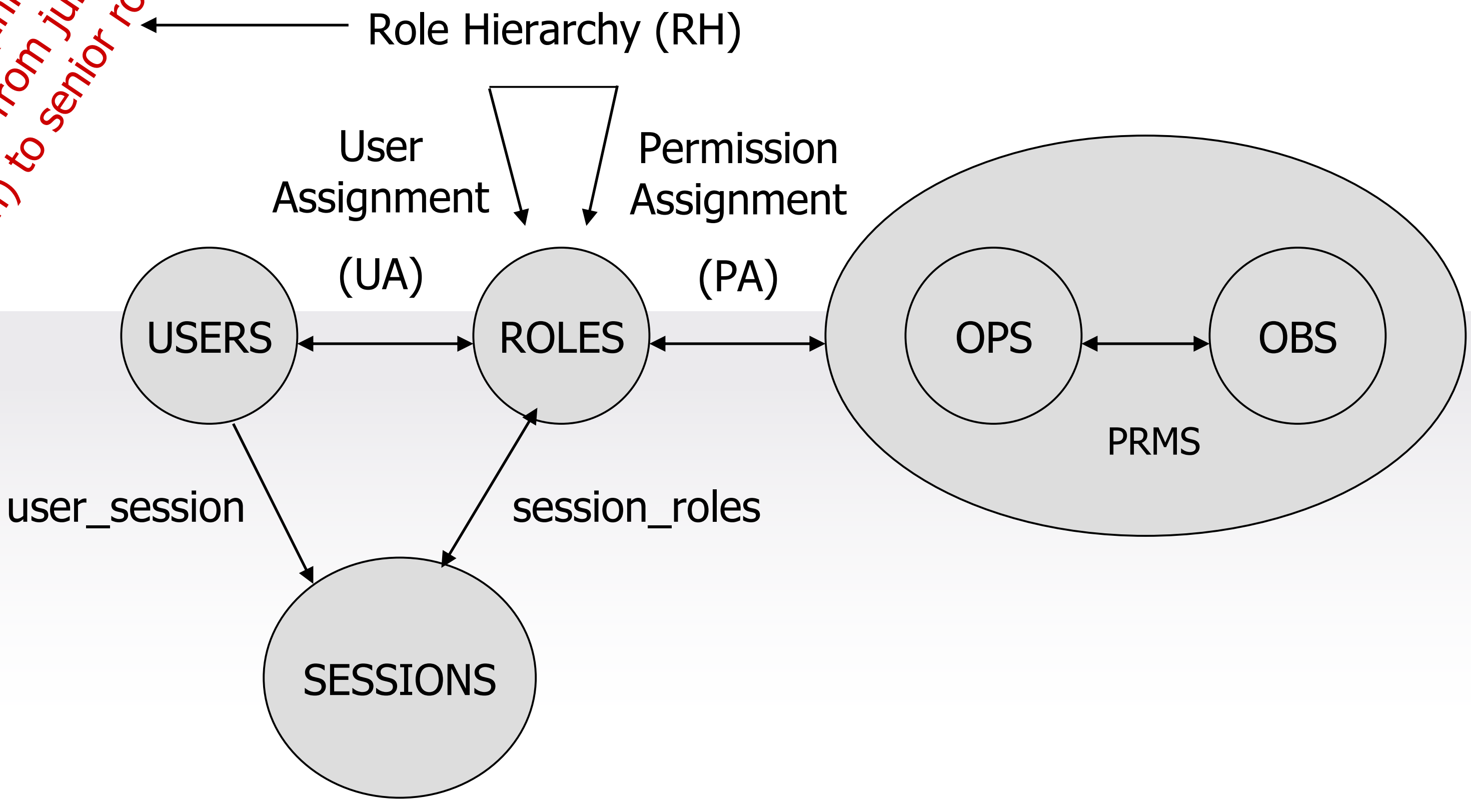
- $UA \subseteq U \times R$ (many-to-many)

- user : S → U, mapping each session s_i to a single user user(s_i)

- roles : $S \rightarrow 2^R$, mapping each session s_i to a set of roles roles(s_i) $\subseteq \{r \mid (user(s_i),r) \in UA\}$ and s_i has permissions $\cup_{r \in roles(si)} \{p \mid (p,r) \in PA\}$

Hierarchical RBAC

Role Hierarchy defines the Inheritance relationship among roles. (Inheritance of permission from junior role (bottom) to senior role (top))



Hierarchical RBAC

- Role hierarchies are a natural means for structuring roles to reflect an organization's line of authority and responsibility.

- General role hierarchies
  - Include the concept of multiple inheritance of permissions and user membership among roles.

- Limited role hierarchies
  - Impose restrictions
  - Role may have one or more immediate ascendants, but is restricted to a single immediate descendent.

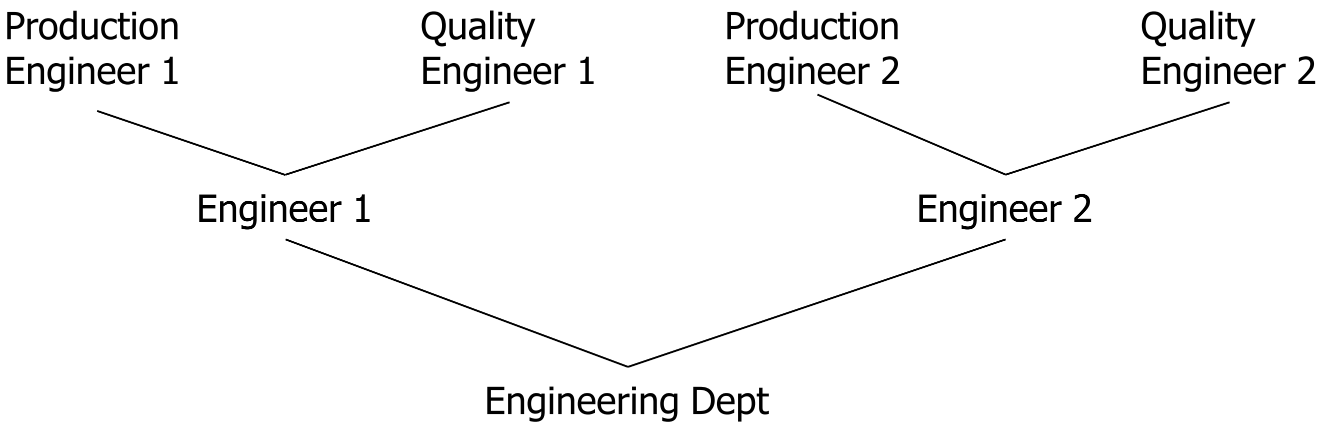
Hierarchical RBAC

- **Q)** In the absence of role hierarchy, what happens?

- **A)** In the absence of role hierarchy, there are two options for assigning many-to-many permissions to users.
  - The first is to duplicate permission assignments among roles.
  - The second option is to assign several roles to users.

Role Hierarchy Examples

- Tree



Production Engineer 1

Quality Engineer 1

Production Engineer 2

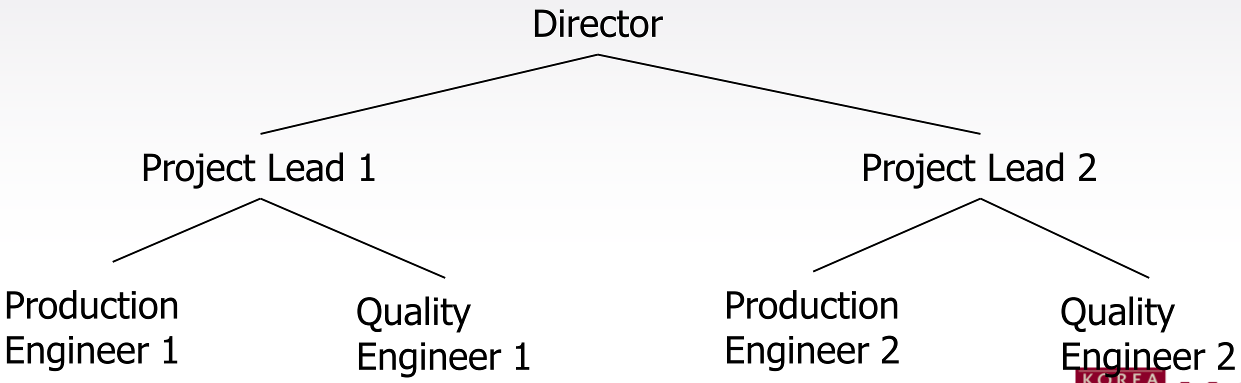
Quality Engineer 2

Engineer 1

Engineer 2

Engineering Dept

- Inverted Tree



Director

Project Lead 1

Project Lead 2

Production Engineer 1

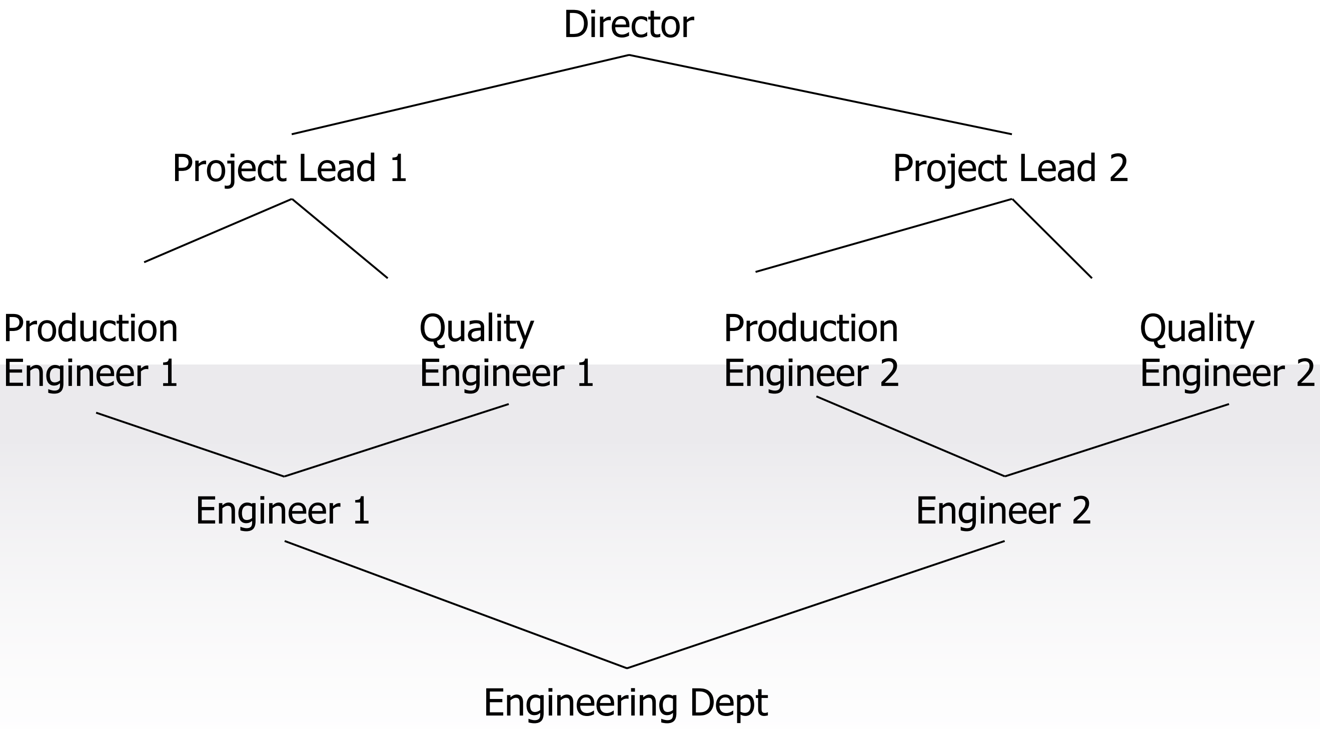
Quality Engineer 1

Production Engineer 2

Quality Engineer 2

Role Hierarchy Examples

- Lattice



Director

Project Lead 1

Project Lead 2

Production Engineer 1

Quality Engineer 1

Production Engineer 2

Quality Engineer 2

Engineer 1

Engineer 2

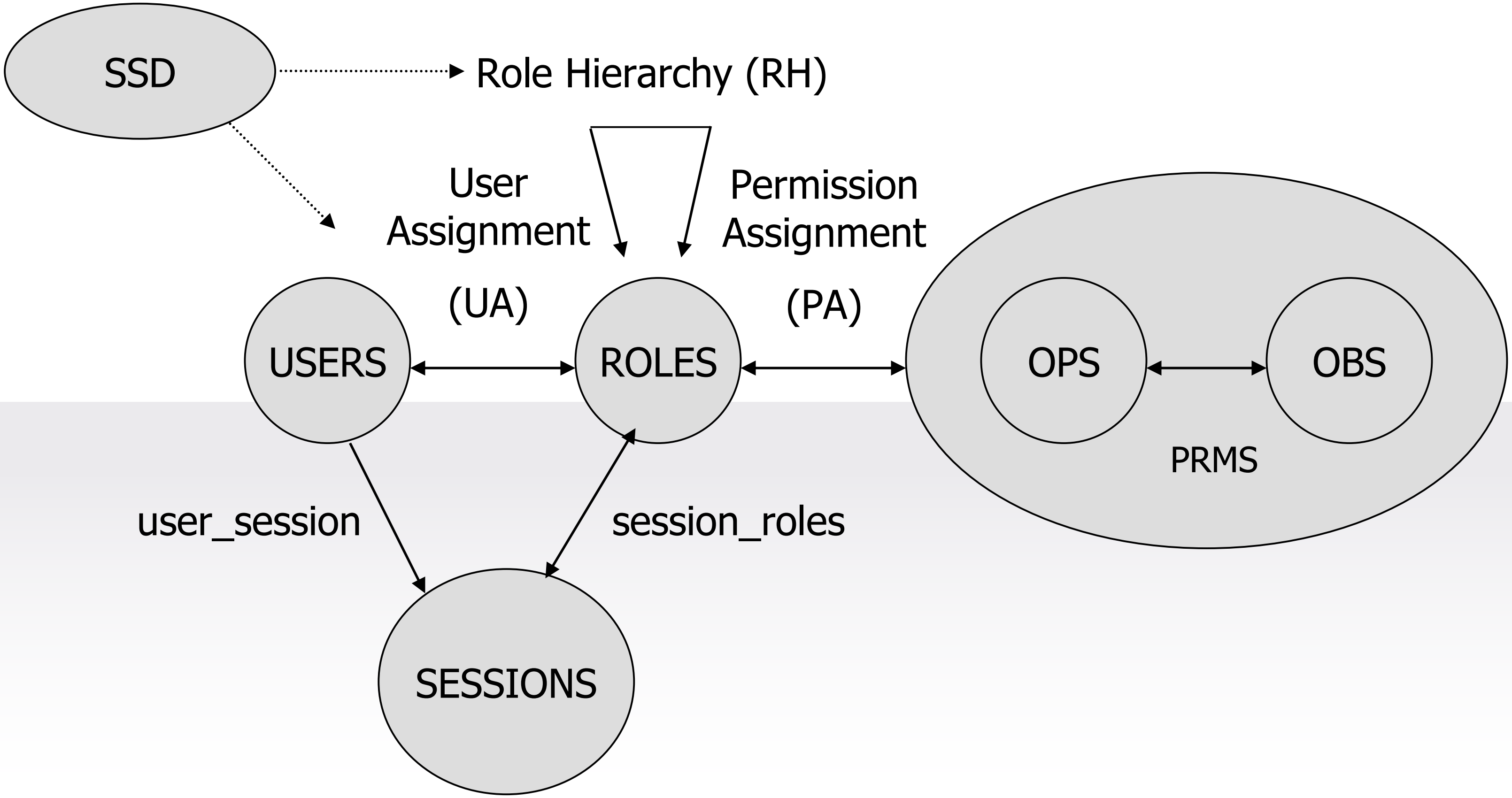
Engineering Dept

Constrained RBAC

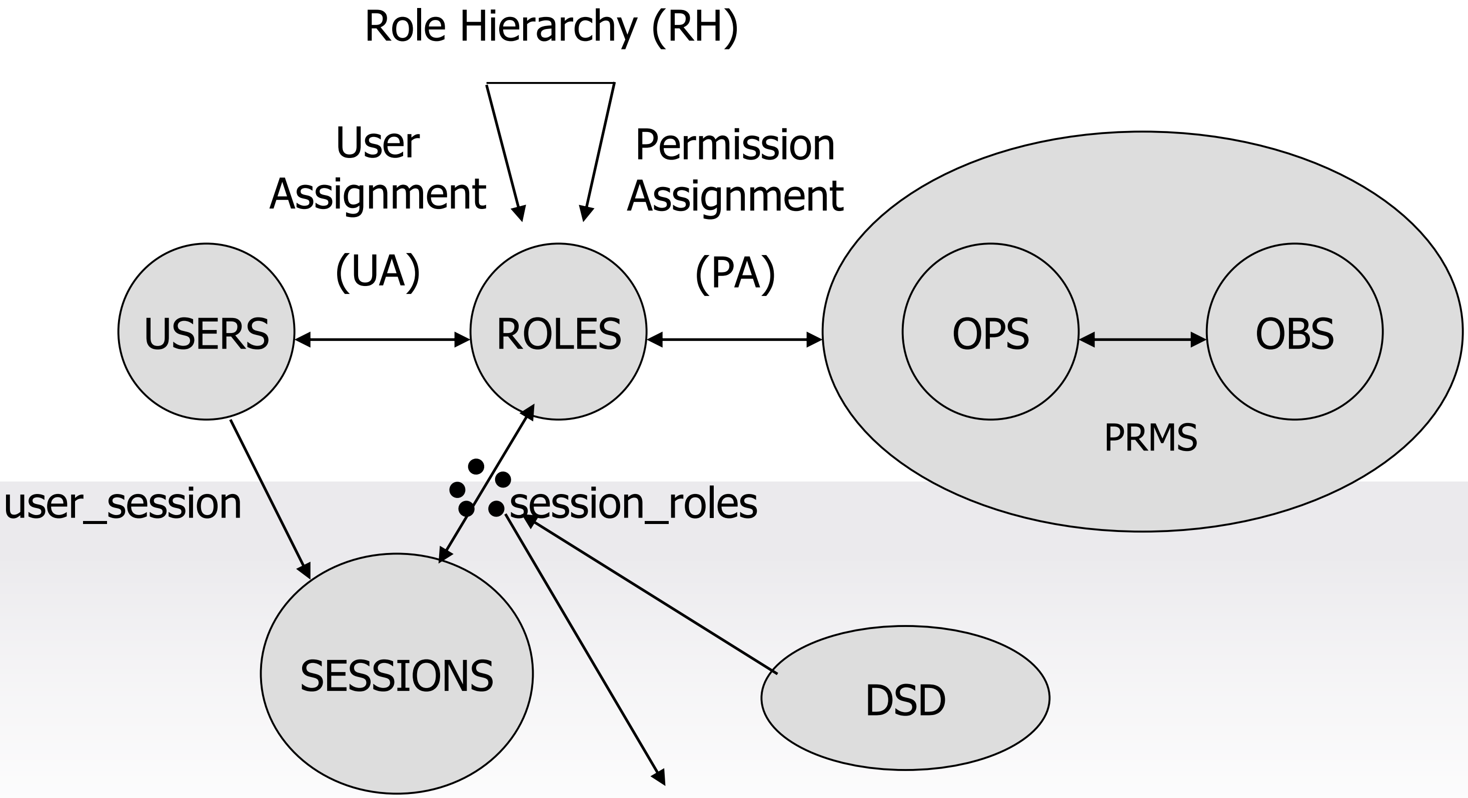
- ( ) : user cannot be authorized for both conflicting roles – mutually exclusive, e.g., teller and auditor.
  - SSoD policies deter fraud by placing constrains on administrative actions and thereby restricting combinations of privileges that are made available to users.
- ( ) : user cannot act **simultaneously** in both roles, e.g., teller and account holder.
  - DSoD policies deter fraud by placing constrains on the roles that can be activated in any given session thereby restricting combinations of privileges that are made available to user.

SSD with Hierarchical RBAC

SSD on the roles r_1 and r_2 implies that they should not be assigned to the same user.



Dynamic Separation of Duty



DSD on the conflicting roles r_1 and r_2 implies that they should not be invoked in the same session by the same user. But the same user may invoke roles r_1 and r_2 in different sessions!

Information Flow Security Model

- **Information Flow** : Transmission of information from one "place" to another.

- Two categories of information flows
  - ( ) : assignment operation
    - `y = x`
  - ( ) : conditional assignment
    - `if x then y = z`

Information Flow Security Model

- **Information Flow Security Model :** Based upon **flow of information** rather than on access controls.
  - Flow of objects are controlled by security policy that specifies where objects of various levels are permitted to flow.
  - (   ) analysis is simplified.

Information Flow Security Model

- How do we measure/capture flow?
  - By ( ) (Information Theory)
    - **Entropy :** amount of information that can be derived from an observation.
    - Change in entropy -> flow
  - By ( )

- A precise quantitative definition for information flow can be given in terms of Information Theory.

- The information flow from x to y is measured by **the equivocation (conditional entropy)**
  - H(x | y) of x, given y.

- Defined by Denning (1976)
- **Purpose :** Guarantee secure information flow as an ordering relation.
- Use mathematical framework to formulate requirements.
- Instead of a list of axioms governing users' accesses, simply require that **information transfers obey the ordering relation**.

- Lead to automatic certification programs.
- Denning uses a set of axioms to limit program code that will violate security classes.
- Sandhu uses the axioms to control information flow at the model level.

- The components of the ( ) information flow model are :
  - **SC** is a set of security classes
  - ***may_flow*** is a relation on SC x SC
  - '**+**' is a function from SC x SC to SC
  - An information flow policy is a triple **<SC,may_flow,+>**
  - Example:

- **Denning’s axioms**
  - Under the following assumptions, an information flow policy forms a finite lattice :

    1.

    2.

    3.

    4.

  - Any information flow that violates this is illegal (**covert**).

[Note] Lattice

- Partially ordered set (S, ≤) and two operations :
  - greatest lower bound (glb X)
  - least upper bound (lub X)
- **Lattice :** A finite set together with a partial ordering on its elements such that for ( ) of elements there is a least upper bound and a greatest lower bound.

[Note] Lattice

- Why We Need Lattice?
  - We wish to have unique answers to the following two questions :
    - **Given any two objects** at different security levels, what is the (   ) to be allowed to read both objects?
    - **Given any two subjects** at different security levels, what is the (   ) so that it can still be read by both subjects?
  - The mathematical structure that allows us to answer these two questions exist. It is called a **lattice**.

- A system is ‘secure’ if there is no illegal information flow.
  - **Advantages** : it (   ) kinds of information flow.
  - **Disadvantages** : far more (   ) such systems.
    - (e.g.) checking whether a given system is secure in the information flow model is an (   ).

- One must also distinguish between
  - **static enforcement :** considers the system (program) as a **static** object.
  - **dynamic enforcement :** considers the system under **execution**.

of the information flow policies.

Non-Interference Security Model

- Non-interference model (a.k.a. Goguen-Meseguer security model) is loosely based on the information flow model; however, it focuses on:
  - How the actions of a subject at a higher sensitivity level affect the system state or actions of a subject at a lower sensitivity level. (i.e.,)
    - Users (subjects) are in their own ( ) so information does not flow or contaminate other ( )
    - With assertion of non-interference security policy, the non-interference model can express multi-level security (MLS), capability passing, confinement, compartmentation, discretionary access, multi-user/multi key access, automatic distribution and authorization chains, and downgrading.

Non-Interference Security Model

- Information flow is controlled by the security policy, where security policy is a set of non-interference assertions (i.e.,
  "   ".)

- Non-interference is to address
  (   ) and (   ).

Rule Based Access Control Model

- Uses specific rules that indicate what can and cannot happen between a subject and an object.

- **Not** necessarily (   ).

- Traditionally, rule based access control has been used in MAC systems as an enforcement mechanism.
  - Today, rule based access is used in other types of systems and applications as well (e.g., routers and firewalls).

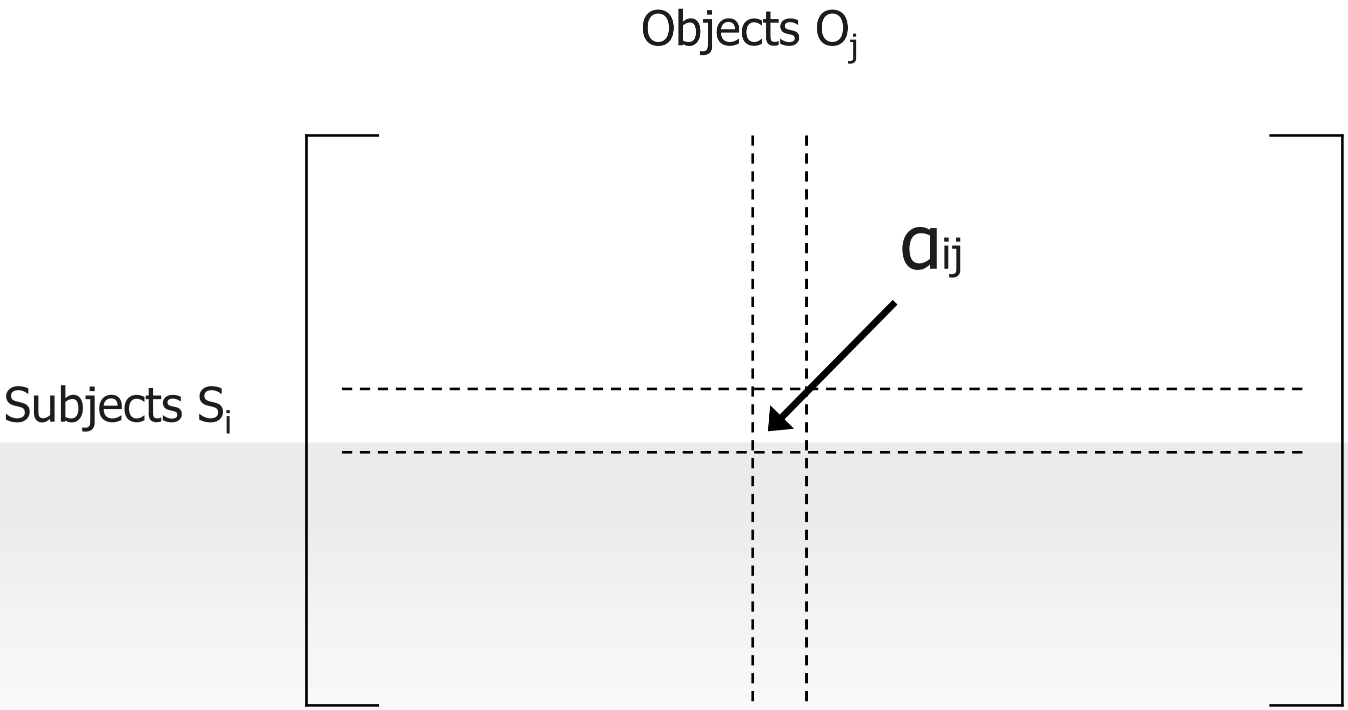
Access Control Structures

- Structures (= Implementation Technique)

Access Control Matrix

- Introduced by B. Lampson (Proc. 5th Princeton Conf. on Information Sciences and Systems, Princeton, 1971. Reprinted in ACM Operating Systems Rev. 8, 1 (Jan. 1974), pp 18-24.) and extended by Harrison, Ruzzo and Ullman (1976~8)
    - Columns indexed by **objects**
    - Rows indexed by **subjects**
    - Matrix entries are (sets of) **access operations**
    - Matrices are data structures that programmers implement as table lookups that will be used and enforced by the operating system.
    - Foundation of many theoretical security models

Access Control Matrix



Objects O_j

Subjects S_i

a_{ij}

Capabilities vs. ACL

- **Capabilities**
  - aka. Directory-based Access Control
  - (   ) of access control matrix
  - Store with (   ) in the form of a token, ticket, or key.
  - e.g., Kerberos, Android

- **ACL (Access Control List)**
  - (   ) of access control matrix
  - Store with (   )
  - e.g., Windows NT

Capabilities vs. ACL

- Example

	R1	R2	R3	R4
S1	*rw*	*rwx*		
S2		*x*	*rwx*	*rwx*
S3	*rwx*	*r*		*r*

Capabilities:

S1: {(R1, *rw*), (R2, *rwx*)}

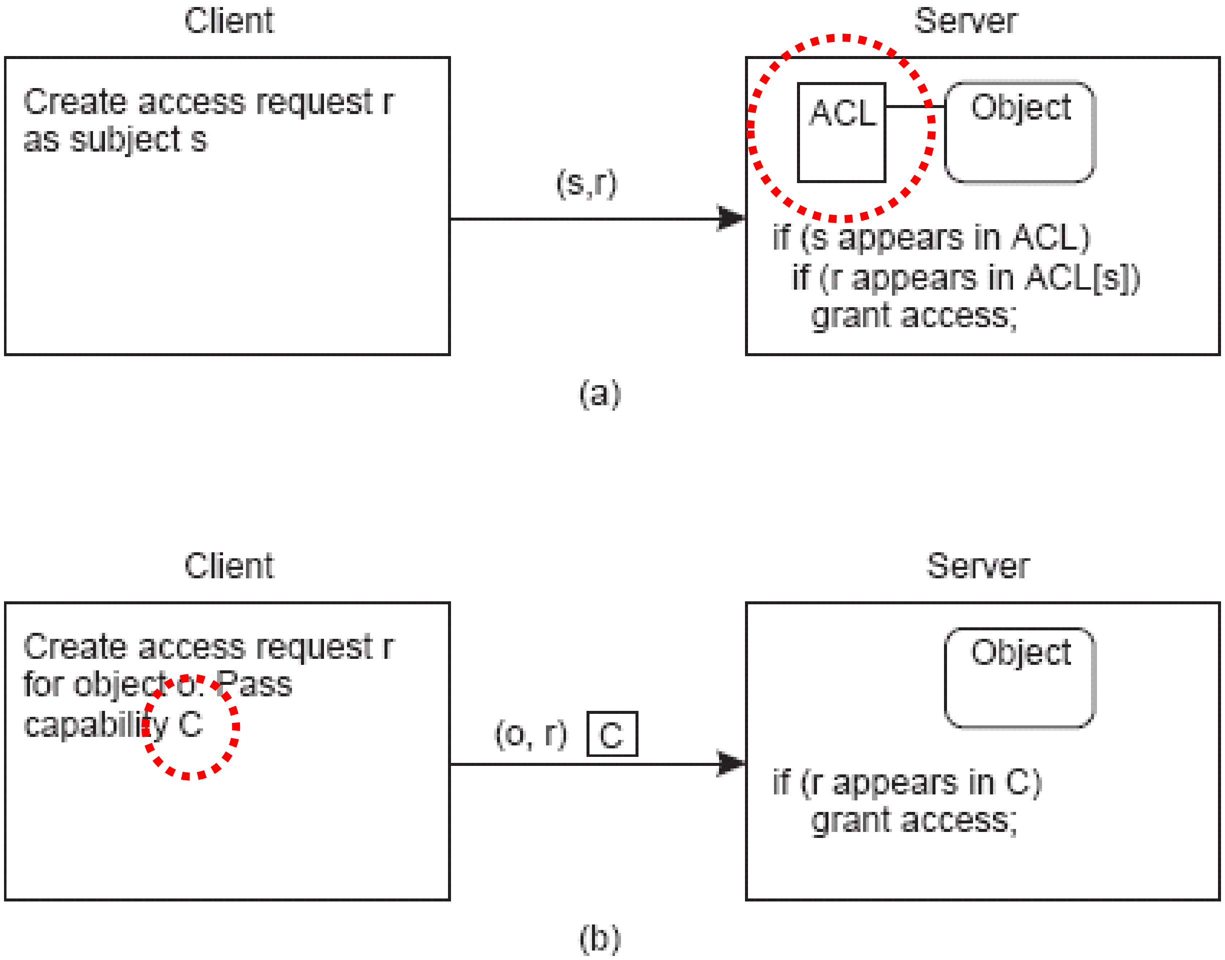
S2: …

Access Control lists:

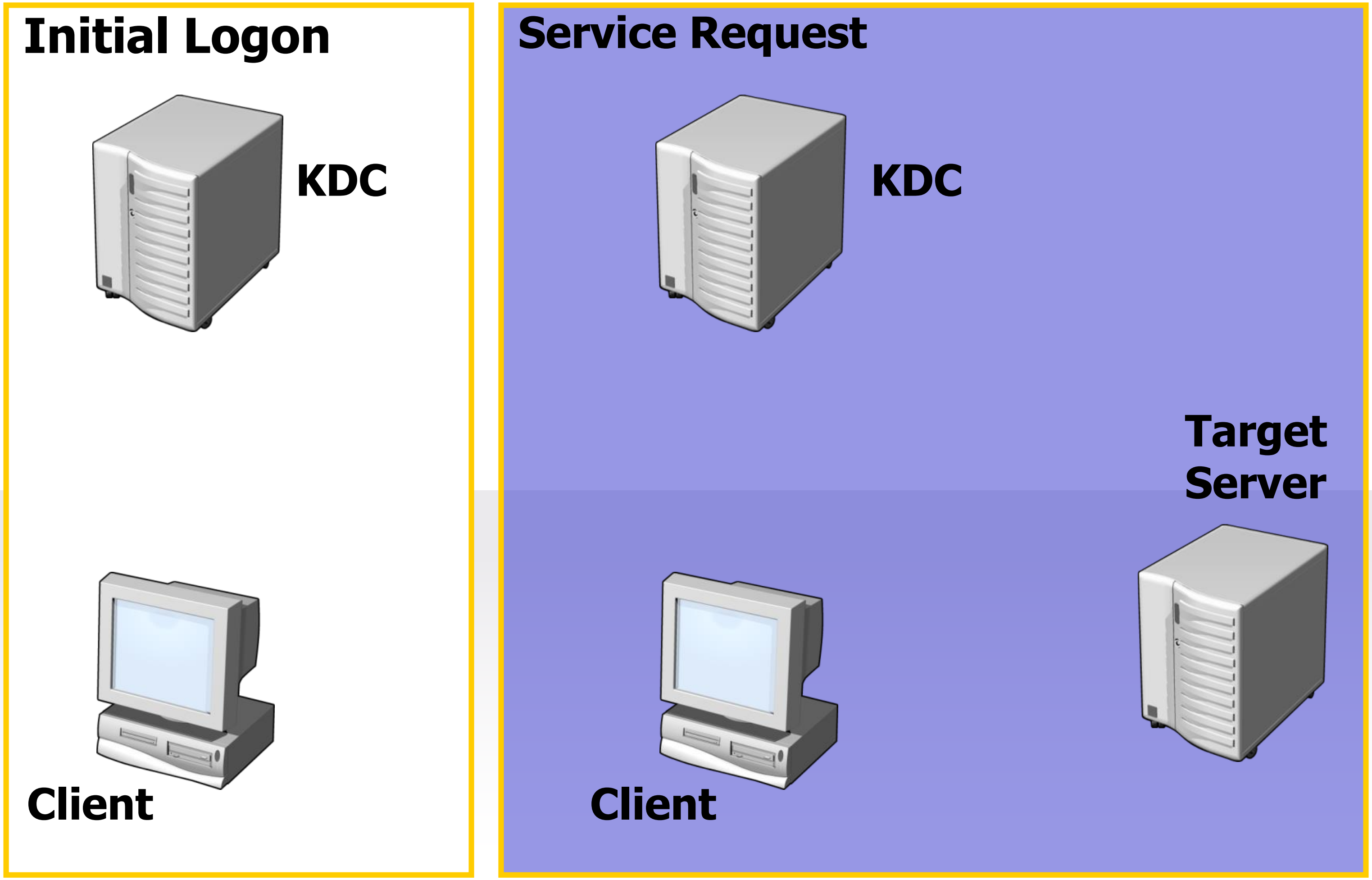
R1: {(S1, *rw*), (S3, *rwx*)}

R2: …

Capabilities vs. ACL



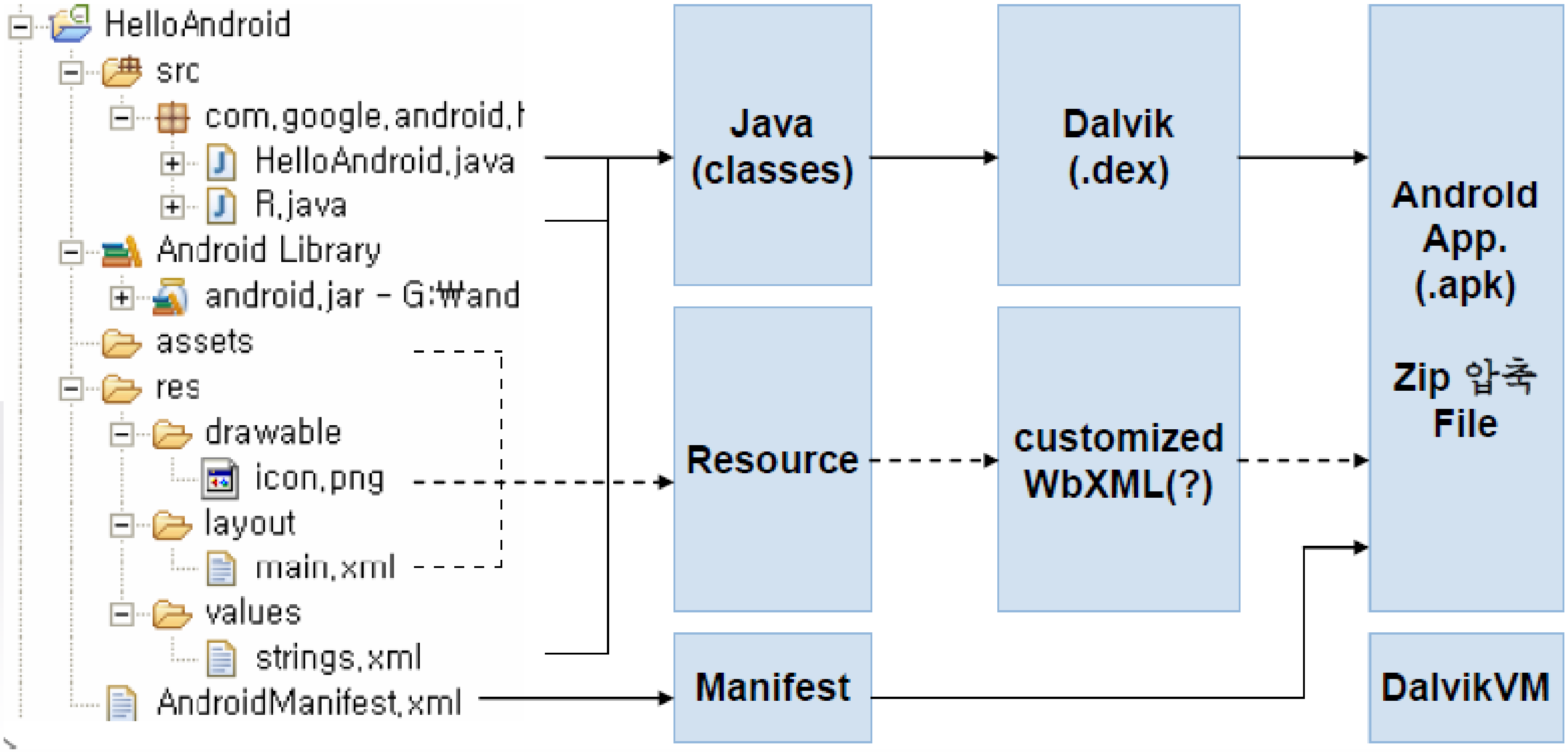
(Capabilities e.g.) Kerberos v5



TGT Ticket-Granting Ticket

ST Service Ticket

(Capabilities e.g.) Android



HelloAndroid
src
com.google.android.h
HelloAndroid.java
R.java
Android Library
android.jar - G:₩and
assets
res
drawable
icon.png
layout
main.xml
values
strings.xml
AndroidManifest.xml

Java (classes)

Dalvik (.dex)

Android App. (.apk)

Zip 압축 File

Resource

customized WbXML(?)

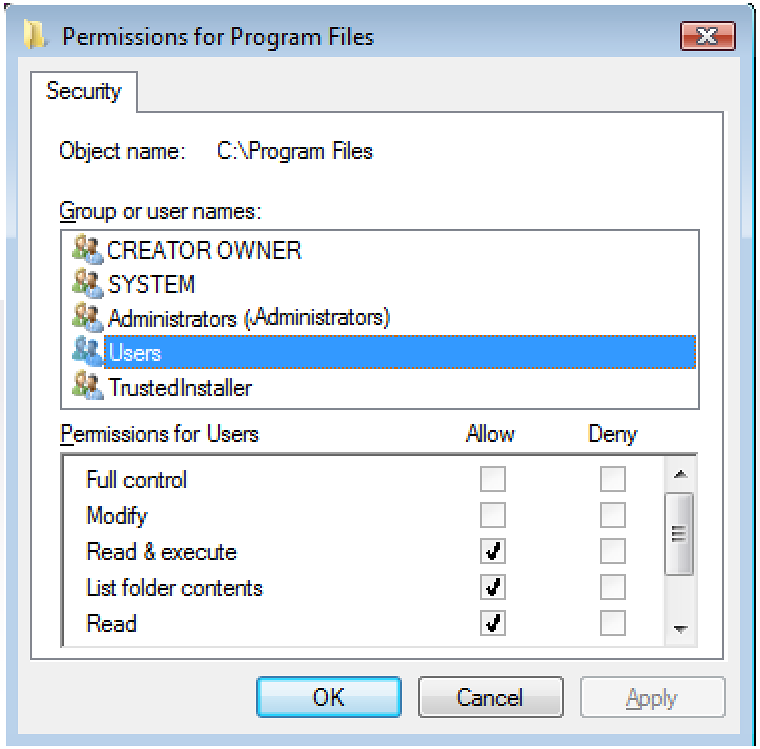
Manifest

DalvikVM

(ACLs e.g.) Windows Access Control

- Access Control Lists (ACLs) :
  - NTFS permissions
  - Share permissions

	Object_A	Object_B
USER_1	READ	WRITE
USER_2	EXECUTE	NONE
USER_N	READ WRITE	READ



Object name: C:\Program Files

Group or user names:

- CREATOR OWNER
- SYSTEM
- Administrators (Administrators)
- Users
- TrustedInstaller

Permissions for Users	Allow	Deny
Full control	□	□
Modify	□	□
Read & execute	☑	□
List folder contents	☑	□
Read	☑	□

Controversies with Capabilities

- The Revocation :

- Delegation :

Ambient Authority

- **Definition :** The “principal” (authority) is (   ) from some global property of process.
  - “Authority that is exercised, but not selected, by its user” (Shapiro et al.)
  - **Example :** open(“file1”, “rw”)
    - **Note :** the subject is missing, but inferred from the process owner

- **Upside :**
  -

- **Downside :**
  -
  -

Access Control Administration

- First an organization must choose the access control model (DAC, MAC, RBAC).
- Then the organization must select and implement different access control technologies.
- Finally, Access Control Administration comes in two basic forms :

Centralized Administration

- One entity is responsible for overseeing access to all corporate resources.
- Provides a (   ) and (   ) method of controlling access rights.
- Types of Centralized Access Control

[Note] Radius

- Is a client/server authentication protocol and authenticates and authorizes remote users.
- Most ISPs today use Radius to authenticate customers before they are allowed to access the Internet.
- Radius is an open protocol and can be used in different types of implementations.
- Uses **UDP** as a transport protocol
- Only encrypts **the user's password** as it is being transmitted from Radius client to the Radius server.
- Is appropriate protocol when simplistic username/password authentication can take place and users only need an "accept" or "deny" for obtaining access.

[Note] TACAS

- Uses **TCP** as a transport protocol.
- Encrypts **all user data** and does not have the vulnerabilities that are inherent in the radius protocol.
- Presents true AAA (Authentication, authorization, and accounting) architecture.

[Note] Diameter

- A protocol that has been developed to build upon the functionality of Radius and overcome many of its limitations.

- AAA protocol.
  - **Authentication :** Provides PAP, CHAP, EAP, end to end protection of authentication information, replay attack protection.
  - **Authorization :** redirects, secure proxies, relays and brokers, state reconciliation, unsolicited disconnect, reauthorization on demand.
  - **Auditing :** reporting, ROAMOPS accounting, event monitoring.

- Diameter provides the common AAA and security framework that **different services** can work within.

Decentralized Administration

- Gives control of access to the people who are closer to the resources

- Has **no** methods for consistent control, **lacks** proper consistency.

Reference Monitor

Execution Monitors (EM)

- Observe program execution.
- Look at a program's execution on a given input as a sequence of runtime events.
- Terminating program's execution if a violation of security is about to occur.
- Introduction to **IRM (Inlined Reference Monitors)**.

What is EM Good for?

- Debugging, tracing, breakpoints, etc.
- Auditing and Logging
- Software testing : memory leaks, out-of-bounds array accesses, race conditions, atomicity, etc.
- Security (aka. sandboxing, babysitting) like buffer overflow prevention etc.
- …

Execution Monitors (EM) in Detail

- **EM** : Focusing on one Execution Trace.
- Easy to do (just observe and constrain).
- EM can often approximate desired policy.
- Schneider shows that reference monitors can (in theory) implement any ( ).

[Note] Safety and Liveness

[Lamport 77]

- **Safety Properties :** Something (   ) will never happen.
  - The ‘safety’ property of access matrices in the HRU model meets indeed this description.
- **Liveness Properties :** Something (   ) will happen. :
  -
  -
  -

What EM Can & Can't Do

- EM (   ) do access control.
- EM (   ) do information flow.
- EM (   ) do Liveness/Availability(e.g. DoS).

Beyond EM

- Security via Static Analysis, Dynamic Analysis, Type-Safe Languages, etc.
  - However, (   ) to reason about

Reference Monitor

- Anderson, J. 'Computer Security Technology Planning Study', ESD-TR-73-51, US Air Force Electronic Systems Division (1973). Section 4.1.1
- **Execution monitor** that forwards events to security-policy-specific validity checks.
- Implementation requires …

Reference Monitor

Validity Checks

- Triggered by RM on each event.
- Encodes the security policy.
- Perform arbitrary computation to decide whether to allow event or halt.
- (PS : RM+Validity checks sometimes called the (   ))

Practical Issues

- In theory, a monitor could :
  - examine **the entire history and the entire machine state** to decide whether or not to allow a transition.
  - perform an **arbitrary** computation to decide whether or not to allow a transition.
- In practice, most systems:
- Otherwise, the **overheads** would be overwhelming.
  - so policies are practically limited by the vocabulary of labels, the complexity of the tests, and the state maintained by the monitor.

Commercial Security Kernels

- SCOMP, GEMSOS, Trusted Solaris 8, Linux Security Modules (LSM) for Linux 2.6, TrustedBSD, MAC OS X, Xen hypervisor, etc.
    - L. J. Fraim. SCOMP: A solution to the multilevel security problem. IEEE Computer, 16(7):26-34, 1983.
    - R. Schell, T. Tao, and M. Heckman. Designing the GEMSOS security kernel for security and performance. In Proceedings of the National Computer Security Conference, 1985.
    - Sun Microsystems. Trusted Solaris 8 Operating System. http://www.sun.com/software/solaris/trustedsolaris/, February 2006.
    - C. Wright, C. Cowan, S. Smalley, J. Morris, and G. Kroah-Hartman. Linux Security Modules: General security support for the Linux kernel. In Proceedings of the 11th USENIX Security Symposium, pages 17-31, August 2002.

Security Policy Modeling

고려대학교 (Korea Univ.)

사이버국방학과 · 정보보호대학원 (CIST)

보안성분석평가연구실 (Security Analysis aNd Evaluation Lab.)

김 승 주 (Seungjoo Kim)

(FB) www.fb.com/skim71 (Twitter) @skim71