BASIC INTRO: COMPUTER SECURITY MODELS

HERMANN HÄRTIG, MARCUS VÖLP, 2017

CLOSETLY FOLLOWING PRESENTATION IN MB: MATT BISHOP “COMPUTER SECURITY ART AND SCIENCE” (2003)
Q3: Is there an algorithm to determine for a system with a given setting of access control permissions, whether or not a Subject A can obtain a right on Object B?

Given a System of Entities ("Objects") acting as Subjects and/or Objects

- with clearly-defined limited access rights among themselves
- can we achieve clearly-defined Security Objectives?
Definition and Example of “higher-Level” Security Policies (Security Policy Models) (Bell La Padula, Chinese Wall)

Mechanisms to express/set clearly-defined access rights: Access Control Matrix, ACL, and Capabilities

Q3 “formalized” in 2 Models: “ACM-based” & “Take Grant”

Decidable?

No proofs (in 2017)
"Reasoning":
- Common sense
- Formal Verification
- Careful Inspection
- Mathematics

"Refinement":
- Abstraction
- Implementation
- Formal Refinement

THE GENERAL APPROACH

Modeling Distributed Systems
“Reasoning”:
- Common sense
- Formal Verification
- Careful Inspection
- Mathematics
- “Common Criteria Assurance”

“Refinement”:
- Abstraction
- Implementation
- Formal Refinement

SECURITY MODELS
Definition: Policy

Examples:
Higher-Level Policies
(very short):
- Bell La Padula
- Chinese Wall
Operating Sys. Mechanisms:
- Access Control List
- Capabilities

Explain Q3 and formalize per model!

Models:
- based on Access Control Matrix
- “take grant” model
Security Policy
A security policy $P$ is a statement that partitions the states $S$ of a system into a set of authorized (or secure) states (e.g., $\Sigma_{\text{sec}} := \{ \sigma \in \Sigma \mid P(\sigma) \}$) and a set of unauthorized (or non-secure) states.

Secure System
A secure system is a system that starts in an authorized state and that cannot enter an unauthorized state (i.e., $\Sigma_{\text{reachable}} \subseteq \Sigma_{\text{sec}}$)

Reference: Matt Bishop: Computer Security Art and Science
Definitions:

Information or data \( I \) is **confidential**

- with respect to a set of entities \( X \) if no member of \( X \) can obtain information about \( I \).

Information \( I \) or data is **integer** if (2 definitions in text books)

- (1) it is current, correct and complete
- (2) it is either is current, correct, and complete or it is possible to detect that these properties do not hold.
Model for Confidentiality

Secrecy Levels:
- Classification (documents)
- Clearance (persons)
- The higher the level the more sensitive the data
- totally ordered

Categories
EXAMPLES BLP(TANENBAUM)

- categories: NATO, Nuclear
- levels/clearance: top secret, secret, confidential, unclassified
- document: Nato, secret
- person clearance: read
  secret, Nato -> allowed
  secret, Nuclear -> not allowed
  confidential, Nato -> not allowed
CHINESE WALL POLICY

Confidentiality & Integrity

- Subjects
- Objects: pieces of information of a company
- CD: Company Data Sets
  objects related to single company
- COI: Conflict of Interest class
  data sets of competing companies
- Sanitized Objects
  version of object that does contain critical information
### Chinese Wall, Example

- **COI**: Objects
- **CD**: Objects
- **Objects**: VW, BMW, D, intel, ARM
- **Sanitized O**
- **Subject**
CHINESE WALL, RULES

PR(S): set of Objects previously read by S

S can read O, if any of the following holds

- first-time read
- $\forall O', O' \in PR(S) \Rightarrow COI(O) \neq COI(O')$
- O is a sanitized Object
CHINESE WALL, EXAMPLE

Subject

PR

write request
read request

Sanitized O

Objects

COI

CD

VW

BMW

D

intel

ARM

read request
write request
PR(S): set of Objects read by S

S can write O, if

- "S can read O"
- ∀ unsanitized O’, “S can read O’” => CD(O) = CD(O’)

CHINESE WALL, RULES
CHINESE WALL, EXAMPLE

COI

CD

Objects

Sanitized O

Subject

write request
read request

VW
BMW
D
S
intel
ARM

COI

CD

Objects

Sanitized O

Subject

PR
Operating Sys. Mechanisms:
- Access Control List
- Capabilities

Explain Q3 and formalize per model!

Models:
- based on Access Control Matrix
- “take grant” model
Subjects: S
Objects: O
Entities: E = S ∪ O
Rights: {read, write, own,…}
Matrix: S x E x R

Simple ACM Operations:
create subject / object
destroy subject / object
test / delete R into cell (s,o)
### ACM

- **Access Control List (ACL)**

<table>
<thead>
<tr>
<th></th>
<th>O1</th>
<th>O2</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>r,w,own</td>
<td>r,w</td>
<td>r,w,own</td>
<td>--</td>
<td>r,w</td>
</tr>
<tr>
<td>S2</td>
<td>r,w</td>
<td>r,w,own</td>
<td>--</td>
<td>r,w,own</td>
<td>r</td>
</tr>
<tr>
<td>S3</td>
<td>r,w</td>
<td>r</td>
<td>w</td>
<td>--</td>
<td>r,w,own</td>
</tr>
</tbody>
</table>

- **Capabilities**

<table>
<thead>
<tr>
<th></th>
<th>O1</th>
<th>O2</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>r,w,own</td>
<td>r,w</td>
<td>r,w,own</td>
<td>--</td>
<td>r,w</td>
</tr>
<tr>
<td>S2</td>
<td>r,w</td>
<td>r,w,own</td>
<td>--</td>
<td>r,w,own</td>
<td>r</td>
</tr>
<tr>
<td>S3</td>
<td>r,w</td>
<td>r</td>
<td>w</td>
<td>--</td>
<td>r,w,own</td>
</tr>
</tbody>
</table>

Ref MB: chapter 2.2
Define Protection Mechanisms of an Operating System in terms of primitive ACM operations

only the defined mechanism provided by the OS can used
Q3/MODEL 1: ACM & “LEAKAGE”

- “Leakage”: an access right is placed into S/O that has not been there before. It does not matter whether or not that is allowed.
- Is leakage decidable?
Examples for OS-Mechanisms defined by ACM-Operations:

UNIX create file (S1,F)
create object
enter own into A(S1,F)
enter read into A(S1,F)
enter write into A(S1,F)

ref MB: chapter 2.2
### Q3/MODEL 1: ACM & “LEAKAGE”

Examples for OS-Mechanisms defined by ACM-Operations:

UNIX chmod -w (S2,F)
- if own $\in$ A(caller,F)
- then delete w in A(S2,F)

<table>
<thead>
<tr>
<th></th>
<th>O1</th>
<th>O2</th>
<th>S1</th>
<th>S2</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>r,w,own</td>
<td>r,w</td>
<td>r,w,own</td>
<td>--</td>
<td>r,w, own</td>
</tr>
<tr>
<td>S2</td>
<td>r,w</td>
<td>r,w,own</td>
<td>--</td>
<td>r,w,own</td>
<td>r,-</td>
</tr>
<tr>
<td>S3</td>
<td>r,w</td>
<td>r</td>
<td>w</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

Q3:
Given an OS with a ACM-based description of protection mechanisms is “Leakage” decidable for any R in A(x,y) ?

ref MB: chapter 2.2
Q3/Model 1: Decidability of Leakage

Decidable
- no subjects/objects can be created
- only one primitive ACM operation per OS-Mechanism by exhaustive search!

Q3 in general:
- undecidable (proof: reduction to Turing machine)

ref MB: chapter 3
Q3/MODEL 2: “TAKE GRANT”

Directed Graph:
Subjects: ●
Objects: ○
Either S or O: ❌

x has capability with set of rights \( \alpha \) on y:

\[ \begin{align*}
\bullet & \rightarrow \alpha & \circ \\
x & \rightarrow y
\end{align*} \]

\( t \) take right
x has cap with set of rights \( \tau \) that includes t

\[ \begin{align*}
\bullet & \rightarrow t & \circ \\
x & \rightarrow y
\end{align*} \]

\( g \) grant right
x has cap with set of rights \( \gamma \) that includes g

\[ \begin{align*}
\bullet & \rightarrow g & \circ \\
x & \rightarrow y
\end{align*} \]
Rules:

take rule \((\alpha \subseteq \beta)\)

a takes \((\alpha \rightarrow y)\) from z

grant rule \((\alpha \subseteq \beta)\)

z grants \((\alpha \rightarrow y)\) to x

ref MB: chapter 3.3
Rules:

create rule

$x$ create \((\alpha \text{ to new vertex}) \ y\)

remove rule

$x$ removes \((\alpha \text{ to}) \ y\)

Application of rules \(\vdash^*\) creates sequences of Graphs $G_i$
CanShare(α, x, y, G₀):
there exists a sequence of G₀ … Gₙ with G₀ ⊢* Gₙ
and there is an edge in Gₙ:

\[ \xrightarrow{\alpha} \]
take rule \((\alpha \subseteq \beta)\)  

a takes \((\alpha \to y)\) from z

grant rule \((\alpha \subseteq \beta)\)  

z grants \((\alpha \to y)\) to x

Question:
Create rule

$\alpha \subseteq \beta$

$z$ takes $(g \rightarrow v)$ from $x$

$z$ grants $(\alpha \rightarrow y)$ to $v$

$x$ takes $(\alpha \rightarrow y)$ from $v$

ref MB: chapter 3.3
CanShare(α, x, y, G₀):

there exists a sequence of G₀ ... Gₙ with G₀ ⊢* Gₙ

and there is an edge:

CanShare decidable in linear time!
- three questions, 2 models per question, different answers !!!
- modeling is powerful
- need to look extremely carefully into understanding models !!!
Q1/M1:

Q1/M2:
John v. Neuman, PROBABILISTIC LOGICS AND THE SYNTHESIS OF RELIABLE. ORGANISMS FROM UNRELIABLE COMPONENTS.

Q2: most textbooks on distributed systems