Outline

Authorization and access control
Multi level security models
Availability and Denial-of-Service
Summary

Recall from Lecture 1

Resource security (access control)
- **authorization**: "bad resource calls don’t happen"
- **availability**: "good resource calls do happen"

In an operating or a computer system
- all resource constraints are security properties

What is a resource?

A resource is whatever we (humans, animals, organisms) compete for.

Examples
- territory, food, storage, CPU...
- axe, printer, program...
- money, information, reputation...
What is a resource?

A resource is anything that can be secured.

Simplest resource security requirements

> privately owned: requires authorization
  - den, shelter, home, account...
> publicly shared: requires availability
  - well, path, printer, Internet...

Resource use in social and computational systems is based on complex combinations of owning and sharing.

What is a resource?

A resource is anything that can be secured.

Simplest resource security requirements

> privately owned: requires authorization
  - den, shelter, home, account...
> publicly shared: requires availability
  - well, path, printer, Internet...

Resource use in social and computational systems is based on complex combinations of owning and sharing.
Access control

Privately owned resources

...can be traded, jointly owned, partially shared etc.

<table>
<thead>
<tr>
<th>q0</th>
<th>sheep</th>
<th>oil</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>use</td>
<td>Ø</td>
</tr>
<tr>
<td>Bob</td>
<td>Ø</td>
<td>use</td>
</tr>
</tbody>
</table>

Table: Permission matrix

<table>
<thead>
<tr>
<th>q1</th>
<th>sheep</th>
<th>oil</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>(milk, wool)</td>
<td>cup oil</td>
</tr>
<tr>
<td>Bob</td>
<td>cup milk</td>
<td>use</td>
</tr>
</tbody>
</table>

Table: Permission matrix

Permission matrix

For the given sets

\[ S \text{ of subjects} \]
\[ O \text{ of objects} \]
\[ A \text{ of actions} \]

a permission matrix at a state \( q \) is an assignment

\[ S \times O \xrightarrow{M^q_{ui}} \varnothing A \]

of the pairs \((u, i) \in S \times O\) to

to the sets (possibly empty) of actions \( M^q_{ui} \subseteq A \)

which the subject \( u \) is permitted to execute on the object \( i \).

Access matrix

For the given sets

\[ S \text{ of subjects} \]
\[ O \text{ of objects} \]
\[ A \text{ of actions} \]

an access matrix at a state \( q \) is an assignment

\[ S \times O \xrightarrow{B^q_{ui}} \varnothing A \]

of the pairs \((u, i) \in S \times O\) to

to the sets (possibly empty) of actions \( B^q_{ui} \subseteq A \)

which the subject \( u \) attempts to execute on the object \( i \).

Authorization

Access control is thus enforced by

\[ \text{preventing the accesses in } B^q_{ui} \]
\[ \text{that are not permitted in } M^q_{ui}. \]

The operating system makes sure at every state \( q \) that

\[ B^q_{ui} \subseteq M^q_{ui} \]

holds for every subject \( u \) and every object \( i \).
In UNIX-like operating systems,

- $S =$ users
- $O =$ files
- $A = \{r, w, x\}$, i.e., read, write and execute

**Capabilities**

Symbian does not maintain large global matrices

$$S \times O \xrightarrow{MB} \wp(A)$$

but smaller subject-based Capabilities

$$S \rightarrow \wp(O \times A)$$

where each subject stores cryptographically protected capability tags $(i, a)$. 

**Access Control Lists (ACL)**

UNIX does not maintain large global matrices

$$S \times O \xrightarrow{MB} \wp(A)$$

but smaller object-based Access Control Lists

$$O \rightarrow \wp(A)^U$$

where $U = \{u, g, o\}$, with $u \in S$, $g \subseteq S$ and $o = S$. 

**Homework**

Read the about UNIX permission matrices (ACLs) in your favorite UNIX reference. What do the commands `chmod`, `setacl` and `getacl` do?

Compare the UNIX access control with the Windows access control. The paper "Windows access control demystified" by Govindavjaha and Appel may help.
Multi level security

In the meantime, at the dawn of Neolithic, Bob builds protected vaults $f_2$ and $f_3$, with a secure chamber $f_5$.

Security levels

For the given
- set $S$ of subjects
- set $O$ of objects
- partially ordered set $L$ of security levels

a clearance structure at a state $q$ consists of the maps

- $c^q : S \rightarrow L$ of clearances
- $f_2^q : S \rightarrow L$ of subject locations
- $f_3^q : O \rightarrow L$ of object locations (or classifications)

Maintaining multi level security

In the meantime, Alice and Bob agree to store Alice's sheep in Bob's protected vault $f_2$.

Clearance structure

Maintaining multi level security: state $q_0$
Maintaining multi level security: state $q_1$

In the meantime, Alice and Bob agree to store Alice’s sheep in Bob’s protected vault $\ell_2$.

As a receipt for the deposit of her sheep into Bob’s vault, Alice gets a secure token in a clay envelope.

- To take the sheep, Alice must give the token.
- To give the sheep, Bob must take the token.
- Anyone who gives the token can take the sheep.

As a receipt for the deposit of her sheep into Bob’s vault, Alice gets a secure token in a clay envelope.

- To take the sheep, Alice must give the token.
- To give the sheep, Bob must take the token.
- Anyone who gives the token can take the sheep.

No-read-up: state $q_1$

Alice cannot take ("read") the sheep out of the vault, because she cannot enter there.
No-read-up: state $q_1$

Only a subject cleared to enter the vault can take ("read") an object from there

$$\tau \in B_u \implies c(u) \geq t(i)$$

No-write-down: state $q_1$

Bob cannot give ("write") the sheep out of the vault while he is in there.

$$w \in B_u \implies t(u) \leq t(i)$$

Maintaining multi level security: state $q_1$

When Alice wants to take ("read") her sheep,

Maintaining multi level security: state $q_2$

When Alice wants to take ("read") her sheep, Bob comes out, breaks the token, and gives ("writes") the sheep.
History of multi level security

- This security protocol goes back to Uruk (Iraq), 4000 B.C.
- More robust security tokens and promisory notes were made not only of clay, but also of horn, ivory, copper, silver, gold.
- Security annotations on clay tokens evolved into cuneiform pictograms, the earliest writing and numeral system.
- Writing and arithmetic have evolved from resource security protocols.
- In computers, banks, companies and governments Access Control and Multi Level Security are still organized around the same security model.

Outline

- Authorization and access control
- Multi level security models
- Availability and Denial-of-Service
- Summary
Security model
Bell-LaPadula, Biba, Clark-Wilson

Given a state machine $Q$, describing the computation with

- a set $S$ of subjects
- a set $O$ of objects
- a set $A$ of actions
- a poset $\mathbb{L}$ of security levels

a security model consists of the following data for each state $q \in Q$

- a permission matrix $M^q: S \times O \rightarrow A$
- an access matrix $B^q: S \times O \rightarrow A$
- a clearance map $c^q: S \rightarrow \mathbb{L}$
- a location map $\ell^q: S + O \rightarrow \mathbb{L}$

Secure states

Homework
Formalize the details of the described sheep bank protocol with in terms of the multi-level security model. Do not forget to include the clay token in the model, or else Bob may release the sheep to Eve.

Can Alice sell the sheep while in the vault?
Describe a similar protocol for digital content instead of the sheep.

Secure states

Warning
The terminology of “security models” and “secure states” can be misleading.

The modeling methodology itself does not guarantee security. There are models where the formally secure states are intuitively insecure.

Example: McLean’s System Z
Every security model can be extended by the transitions to the state $z$ with

$c^z(u) = \top$

$\ell^z(u) = \ell^z(i) = \bot$

where $\bot$ is the lowest and $\top$ the highest security level.

Secure states

Warning
The terminology of “security models” and “secure states” can be misleading.

The modeling methodology itself does not guarantee security. There are models where the formally secure states are intuitively insecure.

Comment
The state $z$ corresponds to a situation where all security constraints are eliminated. Such situations do happen, and sometimes need to be described.

A good language does not disallow false statements, but allows recognizing them.
In order to control
  ▶ downgrading of objects, and
  ▶ authorization of subjects
the state transitions must be constrained.

This leads to the distinction of
  ▶ discretionary access control,
    ▶ where the authorizations can be delegated
  ▶ mandatory access control
    ▶ where the authorizations are centrally managed

Many practical access control systems combine the two.
DoS attack on TCP: SYN flooding

![Normal 3-way handshake in TCP](image)

**Figure:** Normal 3-way handshake in TCP

DoS attack on TCP: SYN flooding

![SYN flood: half open connections lock the server](image)

**Figure:** SYN flood: half open connections lock the server

Commons: publicly shared resources

For centuries, Alice, Bob and Charlie have been sharing an **open field system**.

In England, such open fields were called **Commons**. Alice, Bob and Charlie alternated different crops with grazing, and maintained the land together.

Two remarkable social processes ensued:
- Tragedy of the Commons, and
- Enclosure Movement
Tragedy of the Commons

Charlie realized that it was in his rational interest to invest
- all effort into exploiting the public resource, and
- no effort into maintaining it.

Charlie became a free rider.

Alice and Bob realized that it was in their rational interest
- to stop maintaining the resource for Charlie, and
- to hurry to exploit the resource too.

A race to the bottom ensued. The resource got depleted.

Fair sharing of public resources is a security problem.

The Internet is a common resource. Spam is a symptom of the Tragedy of the Commons.
Security policies

Security policies are both technical and political tools. They regulate computation and social life, as processes of sharing and distributing resources.

Regulation

Charlie the free-rider drew more value out of the land, and enclosed it, away from Alice and Bob.

In England, this happened in XV–XVII centuries. (The Colleges were among the notable beneficiaries.)

Enclosure

The law locks up the man or woman
Who steals the goose from off the common
But leaves the greater villain loose
Who steals the common from off the goose.

The law demands that we alone
When we take things we do not own
But leaves the lords and ladies fine
Who take things that are yours and mine.

The poor and wretched don't escape
If they conspire the law to break;
This must be so but they endure
Those who conspire to make the law.

The law locks up the man or woman
Who steals the goose from off the common
And goose will still a common lack
Till they go and steal it back.

Anonymous, England, XVII century

Homework

Read the article “The Second Enclosure Movement and the Construction of the Public Domain” by James Boyle.

Discuss and contrast the possible technical and political solutions of the security problems arising around modern Commons.
Resource security is among the oldest and the deepest layers of social structure.

- Already microorganisms compete to secure resources.
- The first security protocols date back to 4000 B.C. They led to the invention of money and writing.
- Our banks, our governments and our operating systems use similar security models.

The problems of resource security are both technical and political:
- public availability vs private ownership,
- the Commons vs the Enclosure.

Security policies are engineering problems.

Security engineering is a political tool.

(For better or for worse.)
Summary

- The problems of resource security are both technical and political:
  - public availability vs private ownership,
  - the Commons vs the Enclosure.
- Security policies are engineering problems.
- Security engineering is a political tool.
  (For better or for worse.)
- Cryptography (the next part of the course) is much simpler :)

Security 2: Resource Security
Dusko Pavlovic
Authorization
Security models
Availability
Summary