CIA: Confidentiality, Integrity, Availability

Yannick Chevalier
Université de Toulouse
CSA M1, Security

Plan

Base Concepts

Security Properties

Challenges in Information Security

Confidentiality

Integrity

Model for Security

Availability and Proof

Outline

Base Concepts

Security Properties

DEFINITION (NIST)

Information Security : The protection of information and information systems from unauthorized access, use, disclosure, disruption, modification, or destruction in order to provide **confidentiality**, **integrity**, and **availability**.

CONFIDENTIALITY (1/2)

DATA CONFIDENTIALITY

Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information

EXEMPLES

- Secret informations
- Know-how in a company
- Strategy

Confidentiality (2/2)

Privacy

Assurance that the confidentiality of, and access to, certain information about an entity is protected

Examples

- Individual Preferences
- Right-to-be-forgotten
- Entails some organisational and cross-organisational access control

Integrity (1/2)

Data Integrity

Guarding against improper information modification or destruction, and includes ensuring information **non-repudiation** and **authenticity**.

Examples

- Trust in data (fake polls, fake news)
- Trust in the process producing data (steal the vote)

INTEGRITY (2/2)

SYSTEM INTEGRITY

the quality of an IT system that reflects the logical correctness and reliability of the operating system ; the logical completeness of the hardware and software that implements the protection mechanisms ; and the consistency of the data structures and occurrence of the stored data.

EXAMPLES

- No malwares nor viruses
- No illegal softwares

AVAILABILITY

DEFINITION

Ensuring timely and reliable access to and use of information.

EXAMPLES

- A slow web site
- No ransomware

Authenticity

Definition

- For an asset, integrity and proof of origins and integrity of the process that created that asset
- For a person, a proof of the identity of that person

Examples

- Sotware traceability, including during development
- Trust in a web site

Auditability

Definition

the proper association of each operation of a system with a person responsible for that operation

Plan

UNDERSTANDING THE PROPERTIES

SIMPLE SOUNDING PROPERTIES

- A security goal is usually expressible in a few words
- The exact meaning however is often elusive

EXAMPLE : A VOTE MUST BE KEPT SECRET

Understanding the Properties

Simple sounding properties

- A security goal is usually expressible in a few words
- The exact meaning however is often elusive

Example : a vote must be kept secret

- If everyone votes for the same outcome, everyone knows what everyone has voted for (it's independent of the system employed to vote)

Understanding the Properties

Simple sounding properties

- A security goal is usually expressible in a few words
- The exact meaning however is often elusive

Example : a vote must be kept secret

- If everyone votes for the same outcome, everyone knows what everyone has voted for (it's independent of the system employed to vote)
- In real elections, there is only a few outcomes possibles, but there is a huge number of passwords, and they can also in principle all be written. So what does it mean when you say you know something ? You can write it always/with a positive probability of success/sometimes ?

UNDERSTANDING THE PROPERTIES

SIMPLE SOUNDING PROPERTIES

- A security goal is usually expressible in a few words
- The exact meaning however is often elusive

EXAMPLE : A VOTE MUST BE KEPT SECRET

- If everyone votes for the same outcome, everyone knows what everyone has voted for (it's independent of the system employed to vote)
- In real elections, there is only a few outcomes possibles, but there is a huge number of passwords, and they can also in principle all be written. So what does it mean when you say you know something ? You can write it always/with a positive probability of success/sometimes ?
- Process integrity : is it really the vote a person wanted to cast, or was s/he coerced into casting it ? Is it possible that you have no choice on the possible available candidates, so the vote you cast is meaningless ?

Security is not Safety

Safety

- A system is known
- The environment is modeled with probabilities of bad things happening
- We want safety wrt these probabilities

Security

- The execution environment knows what he does (is an attacker), and he wants you to fail
- Security is parameterised by the computing capabilities of the attacker
  - quantum cryptography
  - https://threatpost.com/pita-side-channel-attack-steals-gpg-key-from-laptops/113447/
- The most common security flaws come from highly improbable corner cases (is there someone with a name of more than 200 characters ?)

Hard Problems

Need to be resilient against unforeseen threats

- Security building blocks are hard
- Often very abstract *wrt* the simple property you want

Example

Cryptography : Mathematics on very abstract objects.

A sobering read : Weil Pairing (or lattices) for asymmetric cryptography

Intricate Mechanisms

Note

- hard : need full-time study or a PhD in the field to understand what really happens
- intricate : need to spend a lot of times to gather lot of different but simple pieces

Partial Information

- Always need to reason by alternating different point of view, with different information
- Information diffusion and its consequences is hard to understand
- Example : social engineering, exploitation of publicly available data by hackers, information escalation

Distributed Systems

- Security protocols are employed to diffuse information in a system
- Harder than regular communication protocols since participants may not want to collaborate

SECURITY ARMS RACE

CONSEQUENCE OF THE PRECEDING SLIDES

- Impossible (in practice) to predict all the consequences of a security mechanism
- By necessity, the defender has to protect assets against known attacks
- The attacker tries to innovate and implement new attacks (e.g. by discovering new vulnerabilities)

Handicaps for a Secure Architecture (1/2)

Financial Handicap

- Hard to quantify RoI : cost of analysis, cost of problem resolution, cost of vulnerability exploitation, probability of exploitation
- Lot of memes : security engineer getting a hard treatment for not implementing a solution he proposed earlier, but was refused

Moving Landscape

- Software have new versions, bugs are discovered and corrected regularly
- A complete infrastructure (CVE, CVM, etc.) and tools are dedicated just to get an up-to-date picture of a system

Handicaps for a Secure Architecture (1/2)

Financial Handicap

- Hard to quantify RoI : cost of analysis, cost of problem resolution, cost of vulnerability exploitation, probability of exploitation
- Lot of memes : security engineer getting a hard treatment for not implementing a solution he proposed earlier, but was refused

Moving Landscape

- Software have new versions, bugs are discovered and corrected regularly
- A complete infrastructure (CVE, CVM, etc.) and tools are dedicated just to get an up-to-date picture of a system

Handicaps for a Secure Architecture (2/2)

Functional Constraints

- Usability trumps security for all commercial solutions
- Backward compatibility means acceptation of bugs
- Users don't like to update their machine

Security is a second thought during development

- Aspect, employed to secure a developed architecture
- No control on that architecture, can just try to adapt

Handicaps for a Secure Architecture (2/2)

Functional Constraints

- Usability trumps security for all commercial solutions
- Backward compatibility means acceptation of bugs
- Users don't like to update their machine

Security is a second thought during development

- Aspect, employed to secure a developed architecture
- No control on that architecture, can just try to adapt

OUTLINE

CHALLENGES IN INFORMATION SECURITY

Lexicon of Information Security

assets

IT assets :

HARDWARE : computers, HDD, SDD

SOFTWARE : OS, Apps

DATA : Files, DBs

NETWORK : Cables, switch, firewalls

Immaterial assets :

IMAGE : reputation, trust

FUNCTIONAL : care of patients in a hospital

MISSIONS : always ready to stop a fire

assets

Owners

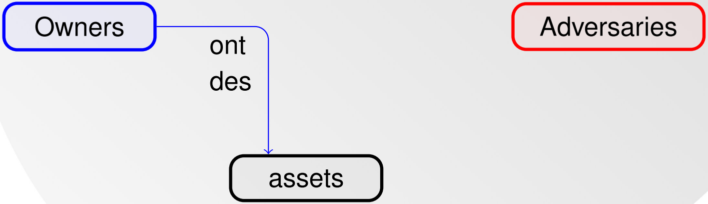
assets

Owners

ont des

Honest Agent who wants to secure his assets (you)

assets


Owners
ont
des
assets
Adversaries

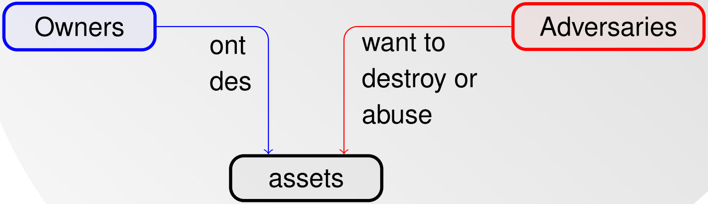
Owners

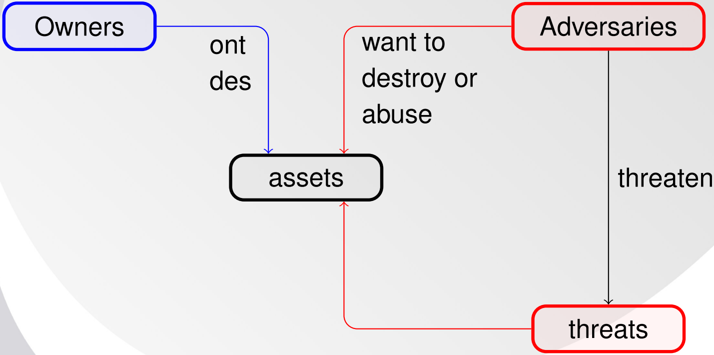
ont des

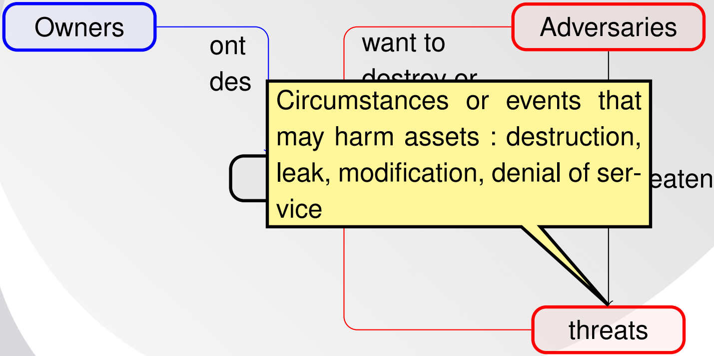
assets

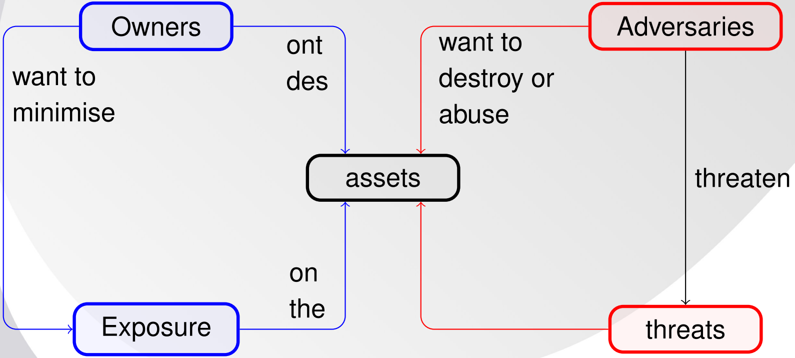
Adversaries

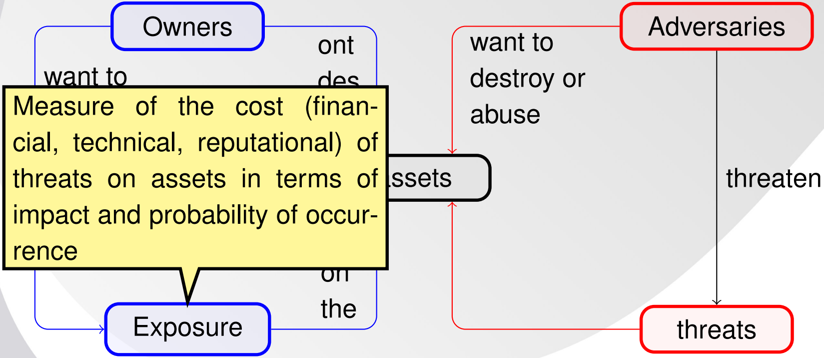
Individual, group, organisation, who does or aim to do bad things on an asset according to its owner

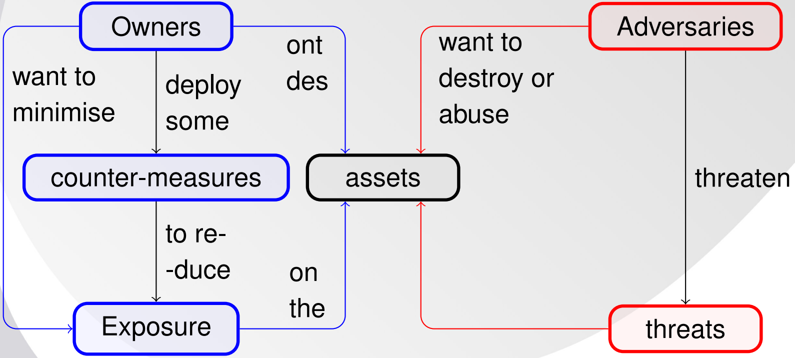

Owners
ont
des
want to
destroy or
abuse
Adversaries
assets


Owners
ont
des
want to
destroy or
abuse
Adversaries
assets
threaten
threats

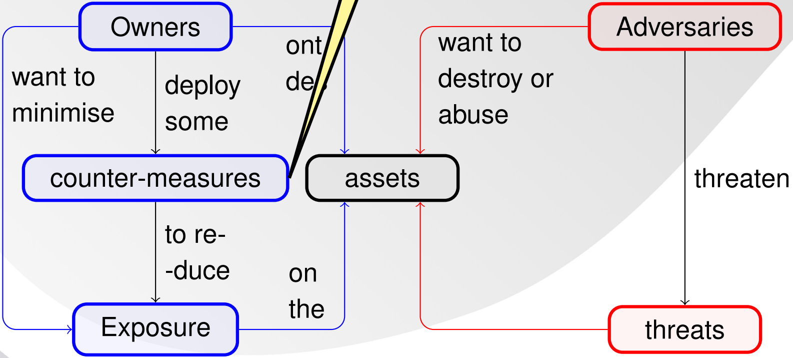

Owners
ont
des
want to
Adversaries
Circumstances or events that may harm assets : destruction, leak, modification, denial of service
eaten
threats


Owners
want to
minimise
ont
des
assets
Adversaries
want to
destroy or
abuse
threaten
on
the
Exposure
threats


Owners
want to
ont
des
Measure of the cost (financial, technical, reputational) of threats on assets in terms of impact and probability of occurrence
on
the
Exposure
ssets
Adversaries
want to destroy or abuse
threaten
threats


Owners
want to
minimise
deploy
some
ont
des
counter-measures
assets
to re-
-duce
on
the
Exposure
Adversaries
want to
destroy or
abuse
threaten
threats

device, software, or technique reducing the exposure of threats on assets



VULNERABILITIES AND THREATS

VULNERABILITY

Weakness in an information system, system security procedures, internal controls, or implementation that could be exploited or triggered by an adversary, categorised into

- CORRUPTION : change the behaviour or the result of a system
- LEAK : access to information not normally accessible
- UNAVAILABILITY : system becomes slow or unusable

RISK

Any circumstance or event that adversely impacts organizational operations (including mission, functions, image, or reputation), organizational assets, or individuals through an information system via unauthorized access, destruction, disclosure, modification of information, and/or denial of service.

THREAT

Potential exploitation of a risk by an adversary.

Vulnerability and Attacks

Attacks

Any kind of malicious activity that attempts to collect, disrupt, deny, degrade, or destroy information system resources or the information itself. Classified into 2 categories :

- **ACTIVE :** Modification of a system or of its behaviour
- **PASSIVE :** System analysed from available or public information

Adversary (or Threat-source)

- **INTERNAL :** has authorized access to the system, and use that access for nefarious purposes
- **EXTERNAL :** isn't known to the system

SUMMARY

1. A honest owner wants to ensure that his assets are secure
2. An adversary threatens some of the security properties desired by the owner of the assets
3. He (exploits) the vulnerabilities on assets in an attack
4. Counter-measures are deployed to reduce the exposure on the owner when a threat is exploited in an attack, or to reduce the benefits to the adversary

PLAN

Preserving Confidentiality

Generic Procedure

1. Inventory all confidentiality-sensitive “assets”

   GDPR : Awareness (Inventory impacted employees), list assets

2. Set to each asset a protection level

   GDPR : based on “Impact Assessment”

3. Inventory processes applied on or using sensitive assets

   GDPR : list processes applied on assets, check consent

4. Implement classification procedures (protection by design) on acquired assets,, and declassification procedures on out-going assets

Rest of this Course

- Definition of levels of protections
- Data confidentiality : Bell-LaPadula
- Data integrity : Biba

Bell-LaPadula

Context (1975)

- Beginning of the multi-task, multi-users systems (Multics, Unix)
- Developed at MITRE for high-security systems
- Basis for Trusted Computer System Evaluation Criteria (TCSEC) (DoD, Orange book, now Common Criteria)
- Based on a serie of articles defining security

Relevance today

- Still used in Defence, Aeronautics, Aerospace
- NEAT-spirit
- Failure to meet this model help understand flaws in other models

BELL-LAPADULA

CONTEXT (1975)

- Beginning of the multi-task, multi-users systems (Multics, Unix)
- Developed at MITRE for high-security systems
- Basis for Trusted Computer System Evaluation Criteria (TCSEC) (DoD, Orange book, now Common Criteria)
- Based on a serie of articles defining security

RELEVANCE TODAY

- Still used in Defence, Aeronautics, Aerospace
- NEAT-spirit
- Failure to meet this model help understand flaws in other models

Bases

Subjects and objects

- A user, a group, etc. and the processes representing them within the system are the subjects
- Files (Unix, so directories, hardware) are the objects
- In most cases : all subjects are also objects (can be acted upon by other subjects)

Static Configuration

- Access Control was decided before the start of the system, and is frozen
- Based on the context of access and on the trust in the subjects : clearance level
- Objects have a security level so that they can be accessed only by certain users in certain contexts

Example

Objects

- services (mail, calendar,. . .)
- files (HOME, but also git, svn, etc.)
- hardware & software resources (printer, DB, etc.)

Subjects

- Web server users (clients)
- Developpers, accountant (employees)
- Other employees/sub-contractors (cleaning service, etc.)

Granularité

- plus le SI est complexe, plus il faudra de catégories de sujets et d'objets pour exprimer précisément la politique de contrôle d'accès qu'on veut implémenter
- dans les exemples, souvent 2 catégories seulement, $\top$ (haut) et $\perp$ (bas)

Security Levels Lattice

Informal Definition

- In a tree, every two nodes have a common ancestors
- In a lattice (as a directed acyclic graph), every two nodes also have a common descendent
- Least common ancestor : $x \sqcup y$
- greatest common descendent : $x \sqcap y$

- Construction : from any order, add a $\top$ (most confidential) and a $\perp$ element
- The set of subsets of a set is a lattice (with $\cup$ and $\cap$)
- The product of two lattices is a lattice

  couple (l, S) where l is a trust level and S is a set of contexts in which that trust can be employed.
- Example : public (minimum), TS/Compartimented (maximum)

Security Levels in Practice

Security level

- *public*, *confidential*, etc : trust level
- set of contexts S : system separated into different contexts

Security levels for objects

- In most cases : $(l, \{c\})$ where c is the context (*e.g.* branch in a company, working groups)
- multi-contexts : user need the level in all the contexts to access the object

Habilitation levels for subjects

In most cases : (l, S) where S are the context the user is trusted to work

Read access decision

User at habilitation level (l, S) can read object with security level (l', S') if $l' \leq l$ and $S' \subseteq S$.

read : flow of information from the object to the user

Potential Leaks

Transitivity of reading

- Assume a user has a (I,S) habilitation level and reads from an object at a (I',S') security level
- The user may *at the same time* write the information at a lower security level (I'',S') with $(I'',S') \leq (I',S')$
- Consequence : a user at the (I'',S') security is able to read the information

Sub-goal : No Write-Down

- To prevent leaks, forbid a user to write below his habilitation level

No write down

- In practice : a user can temporarily downgrade his habilitation level to write to an object
- When downgrading, the system must ensure that no object is currently accessed at a level above the new one

Logical Model

Prédicats

- $\texttt{hab}(x, l)$: subject x has security level l
- $\texttt{lvl}(x, l)$: object x has the security level l
- $\texttt{read}(x, y)$: subject x can read object y
- $\texttt{wrt}(x, y)$: subject x can write object y
- $l \preceq l'$: (habilitation,security) level l is lower than (habilitation,security) level l'

Access control rules

$$\begin{array}{rcl} \texttt{read}(x, y) & \leftarrow & \texttt{hab}(x, l), \texttt{lvl}(y, l'), l' \preceq l \\ \texttt{wrt}(x, y) & \leftarrow & \texttt{hab}(x, l), \texttt{lvl}(y, l'), l \preceq l' \\ l \preceq l' & \leftarrow & l \preceq l'', l'' \preceq l' \end{array}$$

Logical Model

Prédicats

- $\mathtt{hab}(x, l)$: subject x has security level l
- $\mathtt{lvl}(x, l)$: object x has the security level l
- $\mathtt{read}(x, y)$: subject x can read object y
- $\mathtt{wrt}(x, y)$: subject x can write object y
- $l \preceq l'$: (habilitation,security) level l is lower than (habilitation,security) level l'

Access control rules

$$\begin{array}{rcl} \mathtt{read}(x, y) & \leftarrow & \mathtt{hab}(x, l), \mathtt{lvl}(y, l'), l' \preceq l \\ \mathtt{wrt}(x, y) & \leftarrow & \mathtt{hab}(x, l), \mathtt{lvl}(y, l'), l \preceq l' \\ l \preceq l' & \leftarrow & l \preceq l'', l'' \preceq l' \end{array}$$

Logical Model

Prédicats

- $\mathtt{hab}(x, l)$: subject x has security level l
- $\mathtt{lvl}(x, l)$: object x has the security level l
- $\mathtt{read}(x, y)$: subject x can read object y
- $\mathtt{wrt}(x, y)$: subject x can write object y
- $l \preceq l'$: (habilitation,security) level l is lower than (habilitation,security) level l'

Access control rules

$$
\begin{array}{rcl}
\mathtt{read}(x, y) & \leftarrow & \mathtt{hab}(x, l), \mathtt{lvl}(y, l'), l' \preceq l \\
\mathtt{wrt}(x, y) & \leftarrow & \mathtt{hab}(x, l), \mathtt{lvl}(y, l'), l \preceq l' \\
l \preceq l' & \leftarrow & l \preceq l'', l'' \preceq l'
\end{array}
$$

DECLASSIFICATION (ON OBJECTS)

DECLASSIFICATION

- Subjects can lower their level, not objects
- FOIA, etc. : need to make a document accessible
- Trusted users : users (including processes) trusted to lower the security level of objects, including by altering them

PERVASIVE CONCEPT

- Internet ($\perp$), DMZ ($+$), Local Network ($\top$)
- Web pages from developer's computer to Web site (passwords)
- Publication of data from a DB on a Web Server

  whitelisting/blacklisting by the server
- In most cases the trusted subject is a program (firewall, DB controller)

PLAN

Origin

Context

- Slightly post Bell-LaPadula
- Problem with security levels :
  - In a File System...
  - ...should the security levels be increasing or decreasing ?

Answer :

- Decreasing : the deeper one go, the less one needs security (compare / et /tmp)
- Increasing : If you don't know a directory, you cannot know its contents

Origin

Context

- Slightly post Bell-LaPadula
- Problem with security levels :
  - In a File System. . .
  - . . . should the security levels be increasing or decreasing ?

Answer :

- Decreasing : the deeper one go, the less one needs security (compare / et /tmp)
- Increasing : If you don't know a directory, you cannot know its contents

Integrity Simplified

- Bell-LaPadula : anyone can write at a higher security level
- Consequence : the content of even a high security level object cannot be trusted
- Biba Model : establish a parallel hierarchy for trust in information

Examples

- Twitter, Facebook, Instagram : low reliability information
- TV : better but not optimal
- Wikipedia : start of reliable information
- Scientific article : also reliable

Note

- In all cases the reliability is defined by the reviewers of the information
- If the need is to get more view, their emphasis is not on reliability
- Wikipedia tends to become a better quality than a lot of scientific articles (open review vs. usually anonymous reviews)

INTEGRITY IN INFORMATION SYSTEMS

GENERAL PRINCIPLE

- The largest the target audience of an information is, the most care should be taken on its integrity
- The more it has an impact, the more it should be reviewed

EXAMPLE

- The SWIFT system is the international dollar payment system between commercial and national banks
- After intruding into the Bengladesh Central Bank, attempt at stealing USD10^9, end up at successfully laundering USD10^8
- Attack spotted thanks to a clerk who found a spelling mistake

INTEGRITY LEVELS

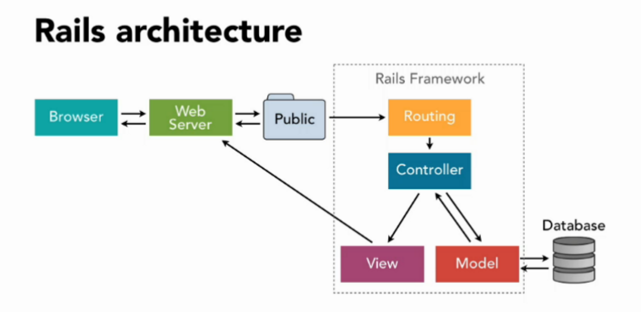
INTUITION

- Rumours can be based on verified sources
- Converse must be excluded without additional verifications

NWU, NRD

- Same level model as BLP, but with integrity levels
- Difference in the policy
  - No write up : subjects can only write at a level below their own
  - No read down : subjects can only read at a level above their own

Example : Web Server



Information flows

- User $\rightarrow$ DB
- DB, public docs $\rightarrow$ User

Annotate with BLP and Biba integrity levels. What security protection is needed?

Classification in the Biba Model

Quality Assurance Procedures

To elevate the integrity level of an information, sanitisation procedures are needed

- Softwares, Networks :
  - White box : Code review, static analysis
  - Black box : Security coating to isolate the application
- Data :
  - Authenticity&Audit
  - End-to-end monitoring of process

In practice

- Aeronautics & space : critical high-integrity systems
- Desire : integrate COTS to lower the cost of production
- Consequence : need for certification of these COTS

LOGICAL MODEL

- As in BLP, a lattice of Integrity levels
- Read/Write rules are reversed

PRÉDICATS

- $\mathtt{ass}(x, l)$: Subject x has assurance level l
- $\mathtt{int}(x, l)$: Object x has integrity level l
- $\mathtt{read}(x, y)$: Subject x can read object y
- $\mathtt{wrt}(x, y)$: Subject x can write object y
- $l \preceq l'$: Assurance/Integrity level l is lower than Assurance/Integrity level l'

ACCESS CONTROL RULES

$$
\begin{array}{rcl}
\mathtt{read}(x, y) & \leftarrow & \mathtt{ass}(x, l), \mathtt{int}(y, l'), l \preceq l' \\
\mathtt{wrt}(x, y) & \leftarrow & \mathtt{ass}(x, l), \mathtt{int}(y, l'), l' \preceq l \\
l \preceq l' & \leftarrow & l \preceq l'', l'' \preceq l'
\end{array}
$$

Logical Model

- As in BLP, a lattice of Integrity levels
- Read/Write rules are reversed

Prédicats

- $\mathtt{ass}(x, l)$: Subject x has assurance level l
- $\mathtt{int}(x, l)$: Object x has integrity level l
- $\mathtt{read}(x, y)$: Subject x can read object y
- $\mathtt{wrt}(x, y)$: Subject x can write object y
- $l \preceq l'$: Assurance/Integrity level l is lower than Assurance/Integrity level l'

Access Control Rules

$$\begin{array}{rcl} \mathtt{read}(x,y) & \leftarrow & \mathtt{ass}(x,l), \mathtt{int}(y,l'), l \preceq l' \\ \mathtt{wrt}(x,y) & \leftarrow & \mathtt{ass}(x,l), \mathtt{int}(y,l'), l' \preceq l \\ l \preceq l' & \leftarrow & l \preceq l'', l'' \preceq l' \end{array}$$

Logical Model

- As in BLP, a lattice of Integrity levels
- Read/Write rules are reversed

Prédicats

- $\mathtt{ass}(x, l)$: Subject x has assurance level l
- $\mathtt{int}(x, l)$: Object x has integrity level l
- $\mathtt{read}(x, y)$: Subject x can read object y
- $\mathtt{wrt}(x, y)$: Subject x can write object y
- $l \preceq l'$: Assurance/Integrity level l is lower than Assurance/Integrity level l'

Access Control Rules

$$\begin{array}{rcl} \mathtt{read}(x,y) & \leftarrow & \mathtt{ass}(x,l), \mathtt{int}(y,l'), l \preceq l' \\ \mathtt{wrt}(x,y) & \leftarrow & \mathtt{ass}(x,l), \mathtt{int}(y,l'), l' \preceq l \\ l \preceq l' & \leftarrow & l \preceq l'', l'' \preceq l' \end{array}$$

Plan

THE SHOW SO FAR...

ASSETS

- Information system is decomposed into segregated assets
- Exceptions to this separation are to be controlled

ANALYSIS, DEVELOPMENT

- Analysis : difficulty to find a decomposition into assets that is manageable (small assets and few communications) and relevant
- Development : Programmers must keep in mind the place of the code they write in the global infrastructure to prepare for declassification and sanitisation procedures within the code

Information system as a graph

$G = (X, E, l_X, l_E)$:

- X : set of assets (from inventory)
- E : set of edges modelling the information flow
- l_X labelling of nodes (objects) with security and integrity levels
- L_E : labelling of edges with all the subjects performing the data flow (can be network, specific softwares, humans)

Analysis

- Assign default habilitation&Assurance levels to subjects to ensure that all flows are permitted
- Compare these levels to desirable ones, and put in place classification/sanitsation procedures (coating or alteration) to elevate their levels to the desired ones

Example : Web Server

```
class UsersController < ApplicationController
  before_action :set_user, only: [:show, :destroy,:update]
  before_filter :authenticate_user!, only: [:destroy,:updat
  after_action :verify_authorized
  ...
  def show
    authorize @user
  end
  ...
    def update_params
      params.require(:user).permit(policy(@user).permitted_
    end
end
```

authorize and policy global method managing access control for the application

Good Practice : Fat Models, Skinny Controllers

- Model : definition of the data stored in the DB, define the policy there (policy attributes)
- Controller (shown) : methods handling the HTTP requests, just needs to

PLAN

Beyond Existence, the Quantity

Attack Examples

- Denial of Service : flood a server with useless queries
- RansomWare : encrypt data on disk

Problem : lack/quality of access to the data

Availability

Ensuring timely and reliable access to and use of information

AVAILABILITY

RELEVANCE

- Security mechanisms must be available
- The usage of computer resources is one of the main goal of Information Security
- Often contradicts (and supercedes) NEAT's Non-bypassable rule

DILEMNA&RESOLUTION

- Emphasis on availability : in case of unforeseen action, one must let the action be done (blacklisting rather than the more secure whitelisting)
- Safeguard : log the action for further processing *if* it turned out to be malicious

Certification&Accreditation (1/2)

Certification

- "Proof" that the system meets the security objectives
- Needs to cover all cases

Accreditation

"Social choice" :

- Given a certified system
- Approve it for certain usage

Constraints on a small drone are less stringents than on a civilian airframe

AUDIT

DEFINITION

Independent review and examination of records and activities to assess the adequacy of system controls, to ensure compliance with established policies and operational procedures

NOTES

- Based on logs describing the state of the system through time
- Rely on a sound association between processes&humans
  - authentification of the human responsible for a process
  - accountable of an action to a process

not just "write access", but what was written

AUTHENTICATION&ACCOUNTABILITY

AUTHENTICATION

- Processes & humans are subjects
- Access control can manage a bash terminal
- But to be relevant, must relate terminals to human users
- Variant for data (code) : data modifications must be related to users

ACCOUNTABILITY

- Having checked $\neq$ having a proof
  - You can check that users have read the license aggreement
  - To have a proof, you need a MCQ and to store the results signed by the human
- Relies ultimately on a proof of integrity of the security counter-measures