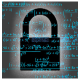
# Formal Methods and Security

Pascal Lafourcade

November 2016, ARC6

# What is cryptography based security?

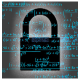
## Cryptography:



- Primitives: RSA, Elgamal, AES, DES, SHA-3 ...
- Protocols: Distributed Algorithms

# What is cryptography based security?

**Cryptography:**



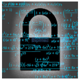
- Primitives: RSA, Elgamal, AES, DES, SHA-3 ...
- Protocols: Distributed Algorithms

**Properties:**


- Secrecy,
- Authentication,
- Privacy
- Non Repudiation ...

# What is cryptography based security?

## Cryptography:



- Primitives: RSA, Elgamal, AES, DES, SHA-3 ...
- Protocols: Distributed Algorithms

## Properties:

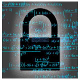

- Secrecy,
- Authentication,
- Privacy
- Non Repudiation ...

## Intruders:

- Passive, active
- CPA, CCA ...

# What is cryptography based security?

**Cryptography:**



- Primitives: RSA, Elgamal, AES, DES, SHA-3 ...
- Protocols: Distributed Algorithms

**Properties:**


- Secrecy,
- Authentication,
- Privacy
- Non Repudiation ...

**Intruders:**

- Passive, active
- CPA, CCA ...

Designing **secure** cryptographic protocols is **difficult**

# Shamir 3-Pass Protocol

## Shamir 3-Pass Protocol

$$
\begin{array}{lllll}
1 & A & \rightarrow & B: & \{m\}_{K_A} \\
2 & B & \rightarrow & A: & \{\{m\}_{K_A}\}_{K_B} \\
3 & A & \rightarrow & B: & \{m\}_{K_B}
\end{array}
$$

# Logical Attack on Shamir 3-Pass Protocol (I)

**Perfect encryption one-time pad (Vernam Encryption)**

$\{m\}_k = m \oplus k$

**XOR Properties (ACUN)**

- $(x \oplus y) \oplus z = x \oplus (y \oplus z)$ — **A**ssociativity
- $x \oplus y = y \oplus x$ — **C**ommutativity
- $x \oplus 0 = x$ — **U**nity
- $x \oplus x = 0$ — **N**ilpotency

# Logical Attack on Shamir 3-Pass Protocol (I)

**Perfect encryption one-time pad (Vernam Encryption)**

$\{m\}_k = m \oplus k$

**XOR Properties (ACUN)**

- $(x \oplus y) \oplus z = x \oplus (y \oplus z)$ **A**ssociativity
- $x \oplus y = y \oplus x$ **C**ommutativity
- $x \oplus 0 = x$ **U**nity
- $x \oplus x = 0$ **N**ilpotency

Vernam encryption is a commutative encryption :

$$\{\{m\}_{K_A}\}_{K_I} = (m \oplus K_A) \oplus K_I = (m \oplus K_I) \oplus K_A = \{\{m\}_{K_I}\}_{K_A}$$

# Logical Attack on Shamir 3-Pass Protocol (II)

## Perfect encryption one-time pad (Vernam Encryption)

## Shamir 3-Pass Protocol

$$\begin{array}{llcll} 1 & A & \rightarrow & B: & m \oplus K_A \\ 2 & B & \rightarrow & A: & (m \oplus K_A) \oplus K_B \\ 3 & A & \rightarrow & B: & m \oplus K_B \end{array}$$

# Logical Attack on Shamir 3-Pass Protocol (II)

## Perfect encryption one-time pad (Vernam Encryption)

## Shamir 3-Pass Protocol

$$
\begin{array}{lllll}
1 & A & \rightarrow & B: & m \oplus K_A \\
2 & B & \rightarrow & A: & (m \oplus K_A) \oplus K_B \\
3 & A & \rightarrow & B: & m \oplus K_B
\end{array}
$$

# Second Example

## Needham Schroeder Key Echange 1976

$$A \rightarrow B : \{A, N_A\}_{Pub(B)}$$

$$B \rightarrow A : \{N_A, N_B\}_{Pub(A)}$$

$$A \rightarrow B : \{N_B\}_{Pub(B)}$$

- Use cryptography
- Small programs
- Distributed

# Cryptography is not sufficient !

## Example : Needham Schroeder Key Echange

$$A \rightarrow B : \{A, N_A\}_{Pub(B)}$$

$$B \rightarrow A : \{N_A, N_B\}_{Pub(A)}$$

$$A \rightarrow B : \{N_B\}_{Pub(B)}$$

# Cryptography is not sufficient !

**Example : Needham Schroeder Key Echange**

$$A \rightarrow B : \{A, N_A\}_{Pub(B)}$$

$$B \rightarrow A : \{N_A, N_B\}_{Pub(A)}$$

$$A \rightarrow B : \{N_B\}_{Pub(B)}$$

Broken 17 years after, by G. Lowe

$$A \rightarrow I : \{A, N_A\}_{Pub(I)} \qquad I \rightarrow B : \{A, N_A\}_{Pub(B)}$$

$$A \leftarrow I : \{N_A, N_B\}_{Pub(A)} \qquad I \leftarrow B : \{N_A, N_B\}_{Pub(A)}$$

$$A \rightarrow I : \{N_B\}_{Pub(I)} \qquad I \rightarrow B : \{N_B\}_{Pub(B)}$$

# Cryptography is not sufficient !

## Example : Needham Schroeder Key Echange

$$A \rightarrow B : \{A, N_A\}_{Pub(B)}$$

$$B \rightarrow A : \{N_A, N_B\}_{Pub(A)}$$

$$A \rightarrow B : \{N_B\}_{Pub(B)}$$

Broken 17 years after, by G. Lowe

$$A \rightarrow I : \{A, N_A\}_{Pub(I)}$$

$$I \rightarrow B : \{A, N_A\}_{Pub(B)}$$

$$A \leftarrow I : \{N_A, N_B\}_{Pub(A)}$$

$$I \leftarrow B : \{N_A, N_B\}_{Pub(A)}$$

$$A \rightarrow I : \{N_B\}_{Pub(I)}$$

$$I \rightarrow B : \{N_B\}_{Pub(B)}$$

Computer-Aided Security

# Formal Verification Approaches

Designer

Attacker

# Formal Verification Approaches

Designer

Attacker

Security Team

# Formal Verification Approaches

Designer

Attacker

Give a proof

Security Team

# Formal Verification Approaches

Designer

Attacker


Give a proof

Find a flaw

Security Team

# Necessity of Tools to Analyze Cryptographic Protocols

- Protocols are small recipes.
- Non trivial to design and understand.
- The number and size of new protocols.
- Out-pacing human ability to rigourously analyze them.

GOAL : A tool is finding flaws or establishing their correctness.

- completely automated,
- robust,
- expressive,
- and easily usable.

Existing Tools: AVISPA, Scyther, Proverif, Hermes, Casper/FDR, Murphi, NRL ...

# Questions?

**How can we find such attacks automatically?**

- Models for Protocols
- Models for Properties
- Theories and Dedicated Techniques
- Tools
  - Automatic
  - Semi-automatic

# Why is it difficult to verify such protocols?

- Messages: Size not bounded
- Nonces: Arbitrary number
- Intruder: Unlimited capabilities
- Instances: Unbounded numbers of principals
- Interleaving: Unlimited applications of the protocol.

# Complexity

Complexity depends of intruder capabilities.

- Passive Intruder
  Problem is polynomial
- Bounded Number of sessions
  Problem is NP-complete
  Tools can verify 3-4 sessions: useful to finds flaws ! OFMC, Cl-Atse, SATMC, FDR, Athena...
- Unbounded Number of sessions
  Problem is in general undecidable
  Tools are corrects, but uncomplete (can find false attacks, can not terminate) Proverif, TA4SP, Scyther, Tamarin.

# Outline

# Outline

# Related Work

- **CryptoVerif [BP06]:**
  - tool that generates proofs by sequences of games
  - has automatic and manual modes
- **CIL [BDKL10]:** Computational Indistinguishability Logic for proving cryptographic primitives.
- **CertiCrypt [BGZB09] /EasyCrypt [BGHB11]:**
  - Framework for machine-checked cryptographic proofs in Coq
  - Improved by EasyCrypt: generates CertiCrypt proofs from proof sketches

# Our Approach

Automatically proving security of cryptographic primitives

1. Defining a language
2. Modeling security properties
3. Building a Hoare Logic for proving the security

# Our Approach

Automatically proving security of cryptographic primitives

1. Defining a language
2. Modeling security properties
3. Building a Hoare Logic for proving the security

Correct but not complete.

# Our Approach

Automatically proving security of cryptographic primitives

1. Defining a language
2. Modeling security properties
3. Building a Hoare Logic for proving the security

Correct but not complete.

- Asymmetric Encryption Schemes [CDELL'08,CDELL'10]
- Encryption Modes [GLLS'09]
- Message Authentication Codes (MACs) [GLL'13]

# Our Approach

Automatically proving security of cryptographic primitives

1. Defining a language
2. Modeling security properties
3. Building a Hoare Logic for proving the security

Correct but not complete.

- Asymmetric Encryption Schemes [CDELL'08,CDELL'10]
- Encryption Modes [GLLS'09]
- Message Authentication Codes (MACs) [GLL'13]

# Verification Technique: Hoare Logic

Set of rules $(R_i)$ : $\{P\}$ $cmd$ $\{Q\}$

# Verification Technique: Hoare Logic

Set of rules $(R_i) : \{P\}\ cmd\ \{Q\}$

$$c_1$$
$$c_2$$
$$\vdots$$
$$c_n$$

# Verification Technique: Hoare Logic

Set of rules $(R_i)$ : $\{P\}\ cmd\ \{Q\}$

$$\begin{array}{ll} \{P_0\} & c_1 \\ & c_2 \\ & \vdots \\ & c_n \end{array}$$

# Verification Technique: Hoare Logic

Set of rules $(R_i) : \{P\}\ cmd\ \{Q\}$

$$
\begin{array}{ll}
\{P_0\} & c_1 \\
& c_2 \\
& \vdots \\
& c_n\ \{Indis(out_e)\}\ ?
\end{array}
$$

# Verification Technique: Hoare Logic

Set of rules $(R_i) : \{P\}\ cmd\ \{Q\}$

$$
\begin{array}{ll}
(R_5)\{P_0\} & c_1\ \{Q_0\} \\
 & c_2 \\
 & \vdots \\
 & c_n\ \{Indis(out_e)\}\ ?
\end{array}
$$

# Verification Technique: Hoare Logic

Set of rules $(R_i)$ : $\{P\}\ cmd\ \{Q\}$

$(R_5)\{P_0\}\ c_1\ \{Q_0\}$
$(R_2)\{P_1\}\ c_2\ \{Q_2\}$, where $P_1 \subseteq Q_0$

$\vdots$

$c_n\ \{Indis(out_e)\}$ ?

# Verification Technique: Hoare Logic

Set of rules $(R_i) : \{P\}\ cmd\ \{Q\}$

$(R_5)\{P_0\}\ c_1\ \{Q_0\}$
$(R_2)\{P_1\}\ c_2\ \{Q_2\}$, where $P_1 \subseteq Q_0$
$\vdots$
$(R_8)\{P_n\}\ c_n\ \{Indis(out_e)\}$ ?

# Verification Technique: Hoare Logic

Set of rules $(R_i)$ : $\{P\}\ cmd\ \{Q\}$

$(R_5)\{P_0\}\ c_1\ \{Q_0\}$
$(R_2)\{P_1\}\ c_2\ \{Q_2\}$, where $P_1 \subseteq Q_0$
$\vdots$
$(R_8)\{P_n\}\ c_n\ \{Indis(out_e)\}$ ?

**Examples of rules:**

(X2): $\{Indis(w; V_1, y, z; V_2)\}\ \ x := y \oplus z\ \ \{Indis(w; V_1, x, y, z; V_2)\}$

(H6): $\{WS(y; V_1; V_2, y) \wedge H(H, y)\}\ \ x := H(y)\ \ \{WS(y; V_1, x; V_2, y)\}$

# Outline

Motivation

Formal Verification
- Verification of Cryptogrpahic Primitives
- Verification of Cryptographic Protocols

Challenges
- Cryptography
- Properties
- Applications

Conclusion

# E-exam: Players and Organization

## Three Roles:

| Candidate | Examination Authority | Examiner |
|---|---|---|
|  |  |  |

# E-exam: Players and Organization

**Three Roles:**

Candidate

Examination Authority


Examiner

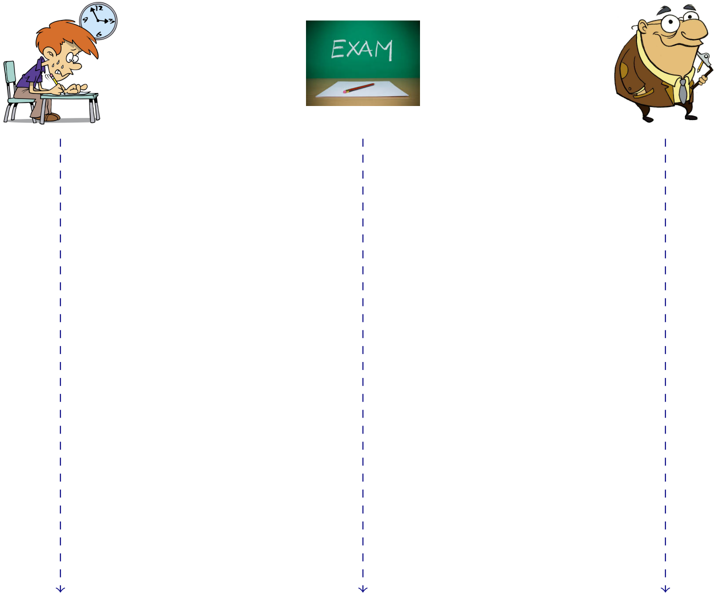
**Four Phases:**

1. Registration 2. Examination 3. Marking 4. Notification

# Model

- **Processes** in the applied $\pi$-calculus
- Annotated using **events**
- **Authentication** properties as **correspondence** between events
- **Privacy** properties as **observational equivalence** between instances
- **Automatic** verification using ProVerif

# Model



# Model


1. Registration

# Model

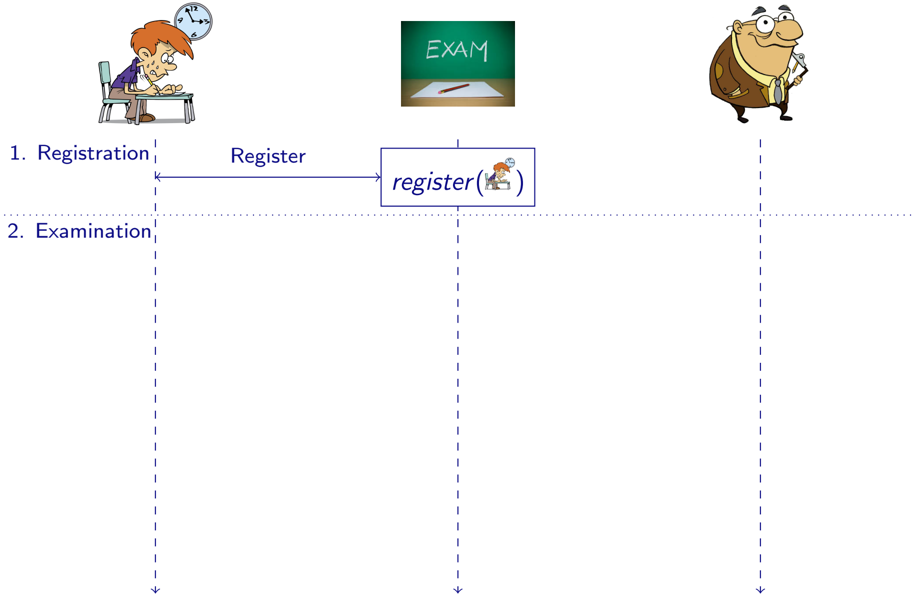


1. Registration

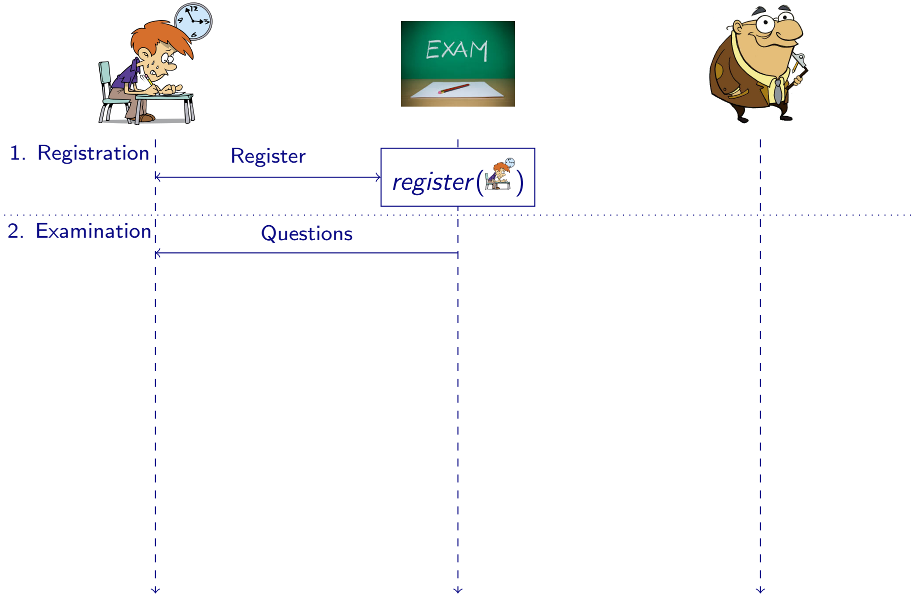
Register

*register*(  )

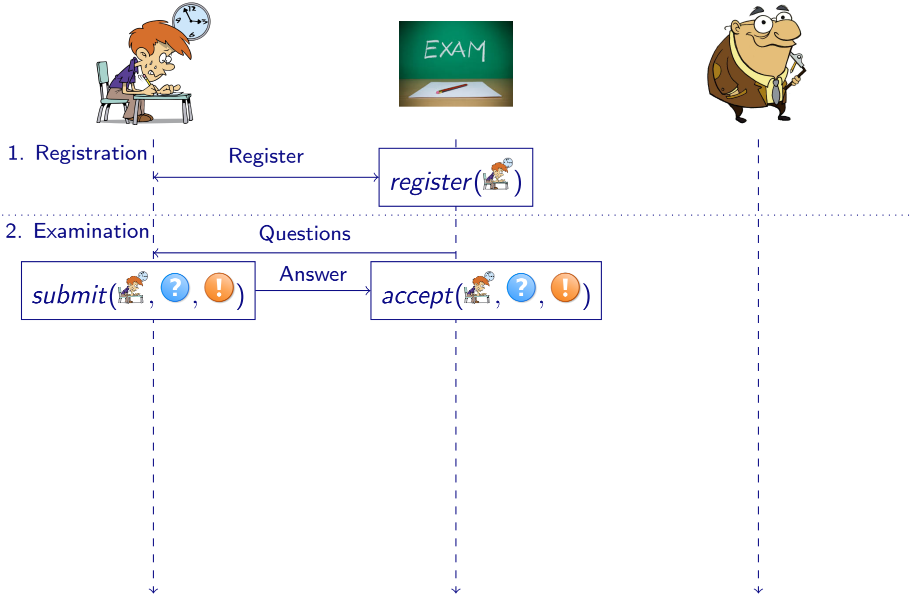
# Model



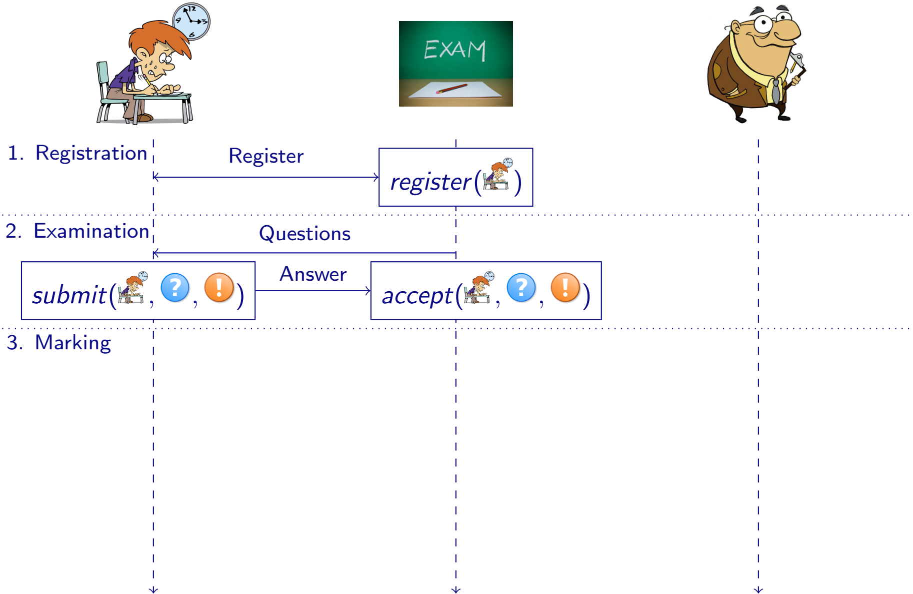
# Model


EXAM
1. Registration
Register
register( )
2. Examination
Questions

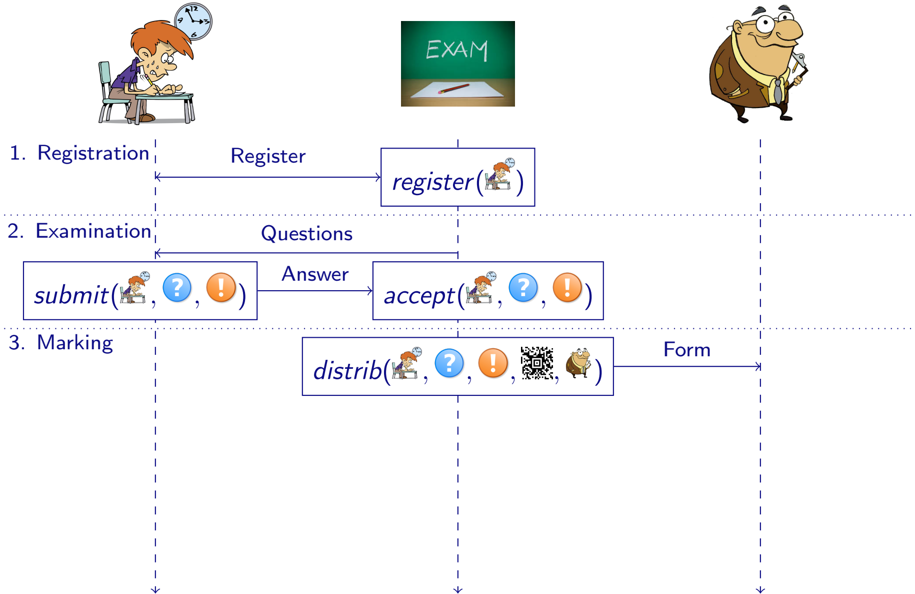
# Model


EXAM
1. Registration
Register
register( )
2. Examination
Questions
submit( , ? , ! )
Answer
accept( , ? , ! )

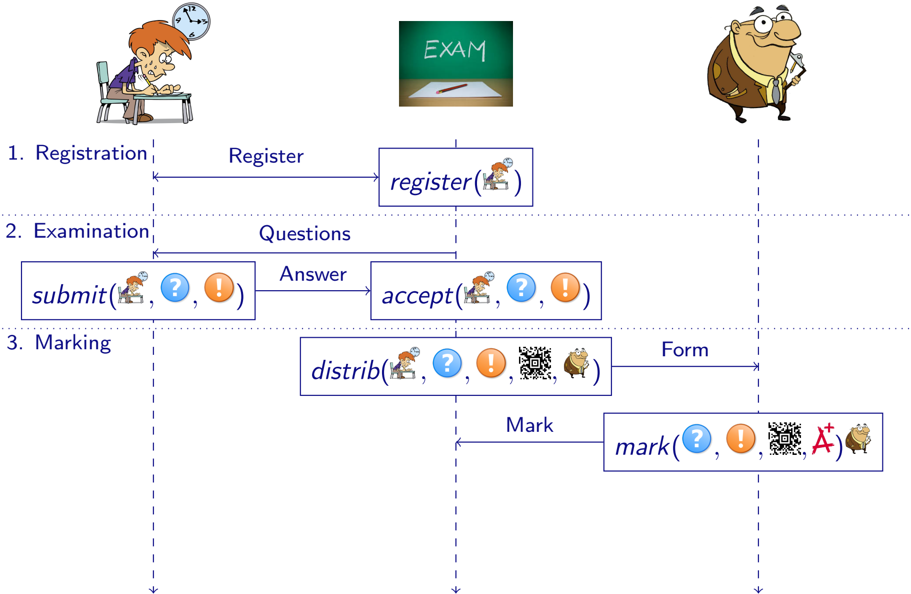
# Model


EXAM
1. Registration
Register
register( )
2. Examination
Questions
submit( , ? , ! )
Answer
accept( , ? , ! )
3. Marking

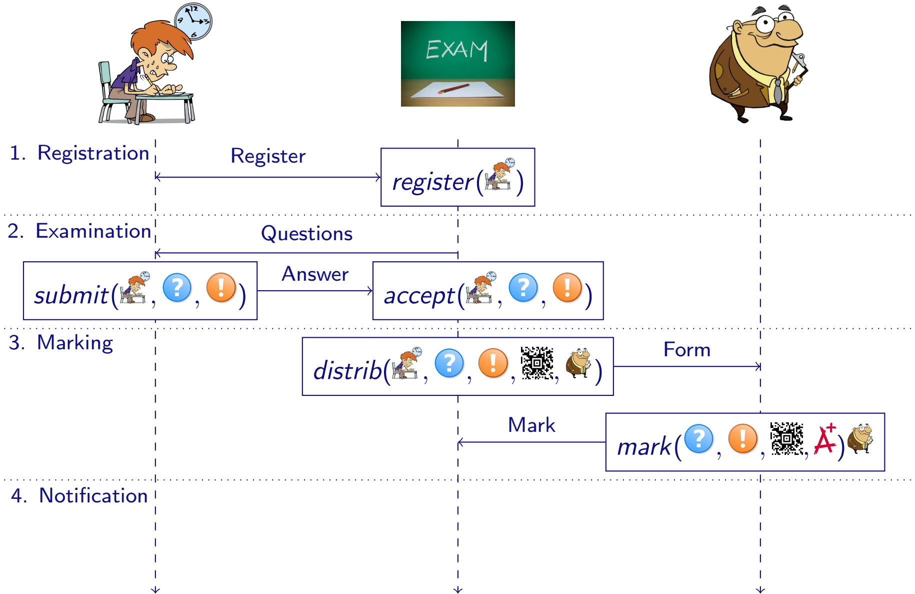
# Model


EXAM
1. Registration
Register
register( )
2. Examination
Questions
submit( , , )
Answer
accept( , , )
3. Marking
distrib( , , , , )
Form

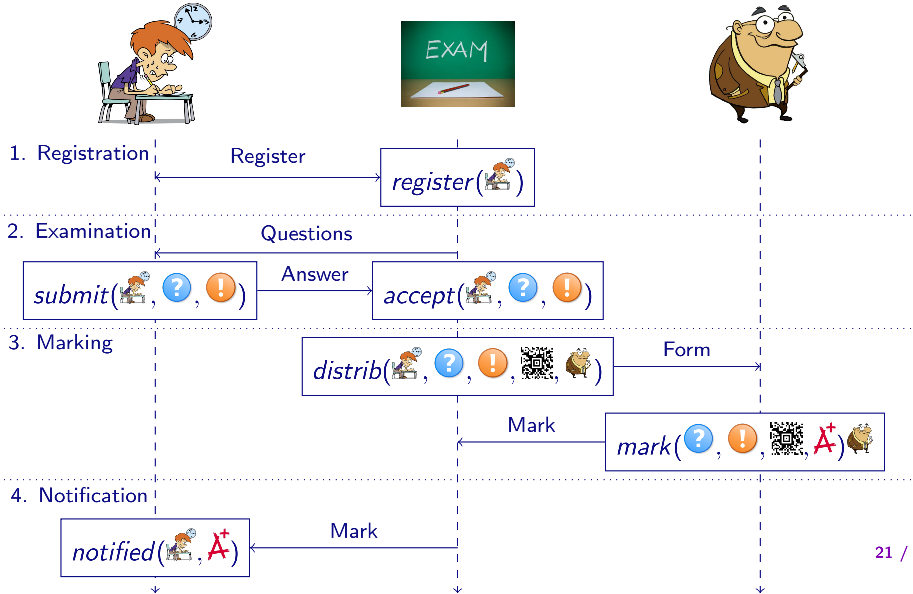
# Model


EXAM
1. Registration
Register
register( )
2. Examination
Questions
submit( , , )
Answer
accept( , , )
3. Marking
distrib( , , , , )
Form
Mark
mark( , , , )

# Model


EXAM
1. Registration
Register
register( )
2. Examination
Questions
submit( , , )
Answer
accept( , , )
3. Marking
distrib( , , , , )
Form
Mark
mark( , , , )
4. Notification

# Model

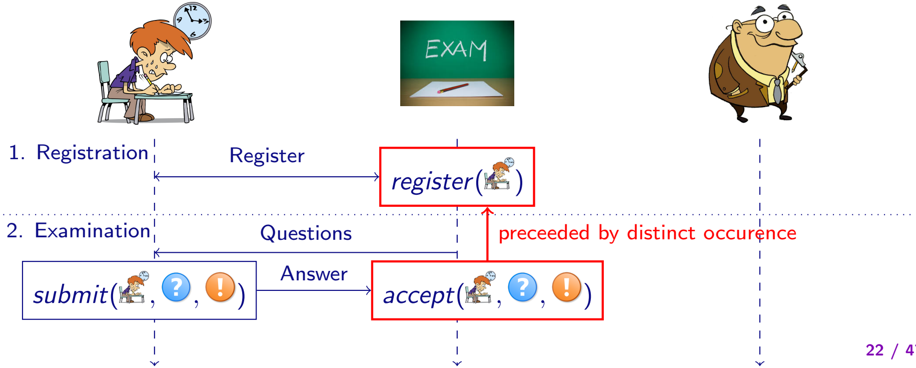

EXAM
1. Registration
Register
register( )
2. Examination
Questions
submit( , , )
Answer
accept( , , )
3. Marking
distrib( , , , , )
Form
Mark
mark( , , , )
4. Notification
notified( , )
Mark

# Answer Origin Authentication

All collected answers originate from registered candidates, and only one answer per candidate is accepted.

## Definition:

On every trace:

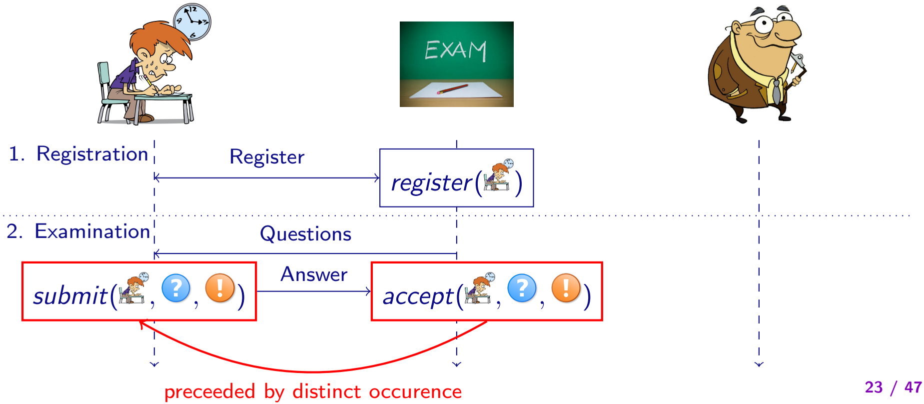


# Form Authorship

Answers are collected as submitted, i.e. without modification.

**Definition:**

On every trace:

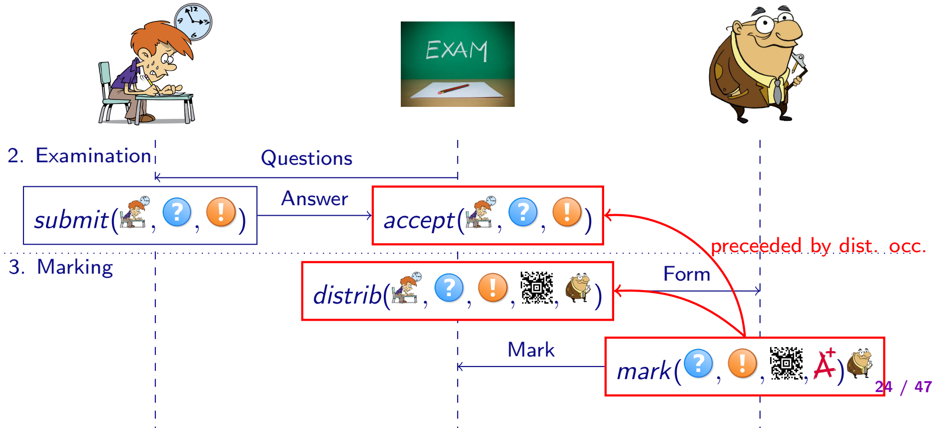


# Form Authenticity

Answers are marked as collected.

## Definition:

On every trace:

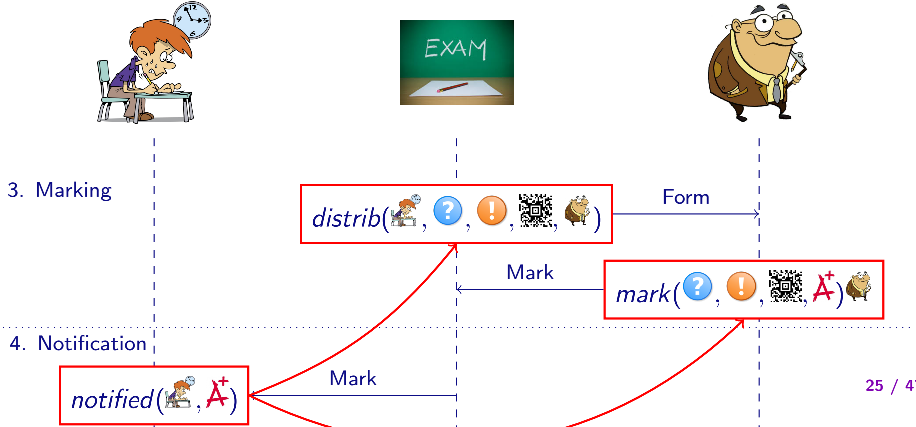


# Mark Authenticity

The candidate is notified with the mark associated to his answer.

## Definition:

On every trace:

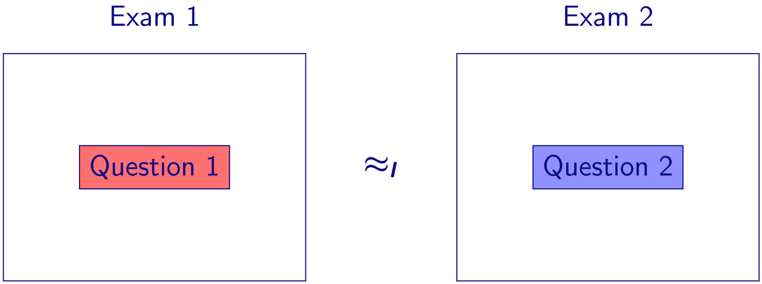


# Question Indistinguishability

No premature information about the questions is leaked.

**Definition:**

Observational equivalence of two instances up to the end of registration phase:

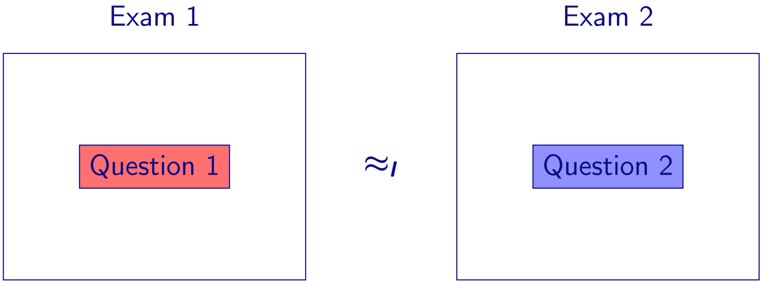


# Question Indistinguishability

No premature information about the questions is leaked.

**Definition:**

Observational equivalence of two instances up to the end of registration phase:




Can be considered with or without dishonest candidates.

# Anonymous Marking

An examiner cannot link an answer to a candidate.

**Definition:**

Up to the end of marking phase:

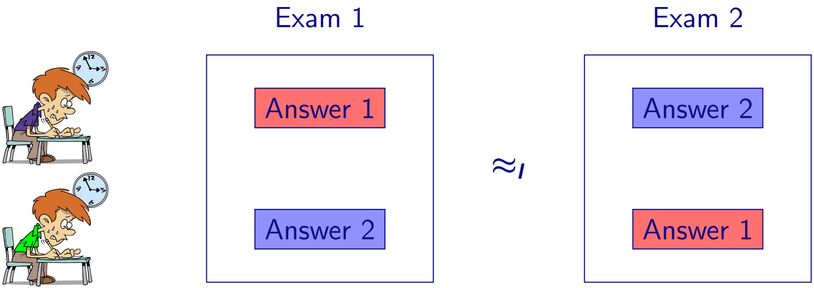
| Exam 1 | | Exam 2 |
| --- | --- | --- |
| Answer 1 | | Answer 2 |
| | $\approx_l$ | |
| Answer 2 | | Answer 1 |

# Anonymous Marking

An examiner cannot link an answer to a candidate.

**Definition:**

Up to the end of marking phase:

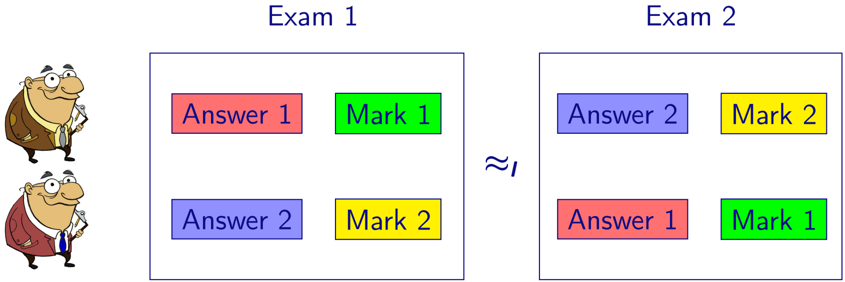



Can be considered with or without dishonest examiners and authorities.

# Anonymous Examiner

A candidate cannot know which examiner graded his copy.

**Definition:**

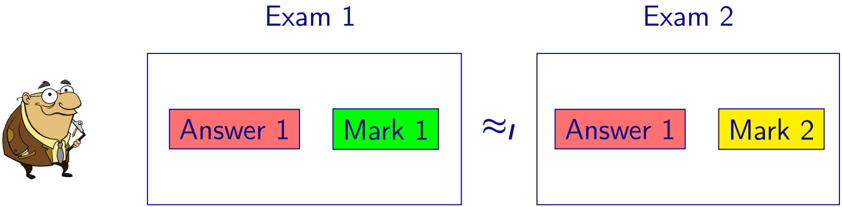



Can be considered with or without dishonest candidates.

# Mark Privacy

Marks are private.

**Definition:**

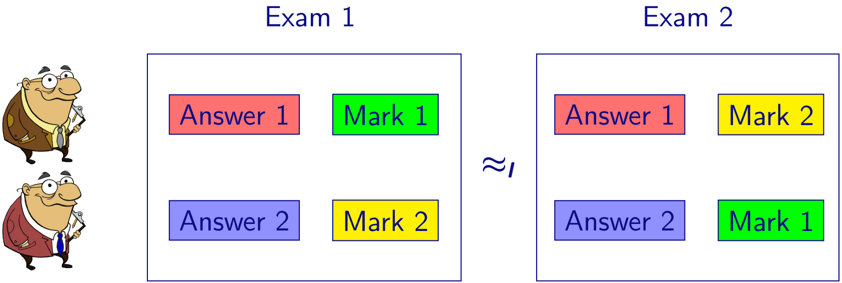



Can be considered with or without dishonest candidates, examiners and authorities.

# Mark Anonymity

Marks can be published, but may not be linked to candidates.

**Definition:**




Can be considered with or without dishonest candidates, examiners and authorities.
Implied by Mark Privacy.

# Application: Huszti & Pethő's Protocol

**"A Secure Electronic Exam System"** using

- ElGamal Encryption
- a Reusable Anonymous Return Channel (RARC) for **anonymous communication**
- a network of servers providing a timed-release service using Shamir's Secret Sharing:
  A subset of servers can combine their shares to **de-anonymize a candidate** after the exam

**Goal:** ensure

- authentication and privacy

in presence of **dishonest**

- candidates
- examiners
- exam authorities

# Results

Formal Verification with ProVerif:

| Property | Result | Time |
|---|---|---|
| Answer Origin Authentication | × | $< 1$ s |
| Form Authorship | × | $< 1$ s |
| Form Authenticity | × | $< 1$ s |
| Mark Authenticity | × | $< 1$ s |
| Question Indistinguishability | × | $< 1$ s |
| Anonymous Marking | × | 8 m 46 s |
| Anonymous Examiner | × | 9 m 8 s |
| Mark Privacy | × | 39 m 8 s |
| Mark Anonymity | × | 1h 15 m 58 s |

# Main reason

Given its security definition, the **RARC**

- provides anonymity, but not necessarily secrecy
- does not necessarily provide integrity or authentication
- is only secure against **passive attackers**

Corrupted parties or active attackers can **break secrecy and anonymity**, as the following attack shows.

# Application: Remark! Protocol

A recent protocol using

- ElGamal encryption
- an **exponentiation mixnet** to create **pseudonyms** based on the parties' public keys
  $\Rightarrow$ allows to encrypt and sign anonymously
- a public append-only **bulletin board**

**Goal:** ensure

- authentication and integrity
- privacy
- verifiability

in presence of **dishonest**

- candidates
- examiners
- exam authorities

# Results

Formal Verification with ProVerif:

| Property | Result | Time |
|---|---|---|
| Answer Origin Authentication | ✓ | $< 1$ s |
| Form Authorship | ✓ | $< 1$ s |
| Form Authenticity | ✓[1] | $< 1$ s |
| Mark Authenticity | ✓ | $< 1$ s |
| Question Indistinguishability | ✓ | $< 1$ s |
| Anonymous Marking | ✓ | 2 s |
| Anonymous Examiner | ✓ | 1 s |
| Mark Privacy | ✓ | 3 m 32 s |
| Mark Anonymity | ✓ | -[2] |

[1] after fix

[2] implied by Mark Privacy

# Outline

# Main changes

- Fully homomorphic encryption
- Post-quantum cryptogrpahy
- Lattice based cryptgraphy
- Privacy primitives

# Main changes

- Fully homomorphic encryption
- Post-quantum cryptogrpahy
- Lattice based cryptgraphy
- Privacy primitives

Are they really secure ?

# Main changes

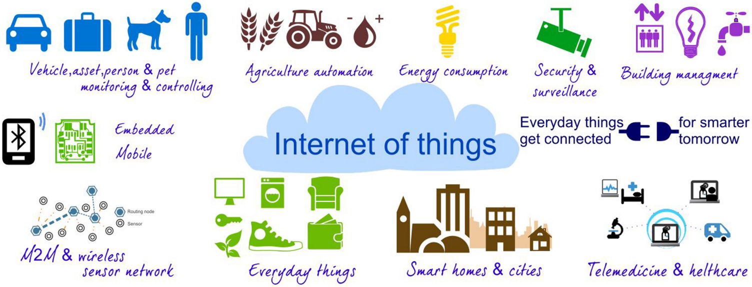
- Fully homomorphic encryption
- Post-quantum cryptogrpahy
- Lattice based cryptgraphy
- Privacy primitives

Are they really secure ?
How to model them in formal verification ?

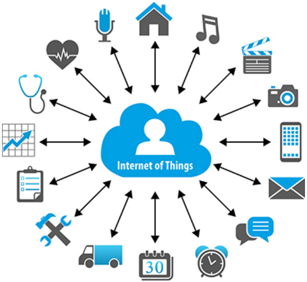
# More Properties

- Privacy
- Traceability
- Accountablility
- Fairness

# Near Future


Vehicle,asset,person & pet monitoring & controlling
Agriculture automation
Energy consumption
Security & surveillance
Building managment
Embedded Mobile
Internet of things
Everyday things get connected for smarter tomorrow
Routing node
Sensor
M2M & wireless sensor network
Everyday things
Smart homes & cities
Telemedicine & helthcare

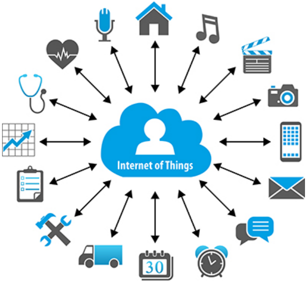
# Reasons of the Succes of IOT




## Technology

- Wireless Communications:
  Wifi, 3G, 4G, Bluethooth, Sigfox ...
- Batteries
- CPU
- Sensors
- Price

# Reasons of the Succes of IOT




## Technology

- Wireless Communications:
  Wifi, 3G, 4G, Bluethooth, Sigfox ...
- Batteries
- CPU
- Sensors
- Price

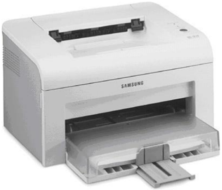
## Usage

- Monitoring services
- Hyperconnectivity
- Avaibility

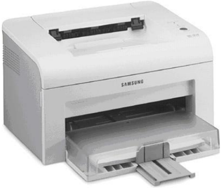
# Real attacks on IoT from 2007 ...



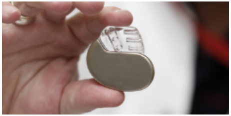
# Real attacks on IoT from 2007 ...

# Real attacks on IoT from 2007 ...





# Is it preserving your privacy?

# Is it preserving your privacy?


4096 RSA encryption

# Is it preserving your privacy?


4096 RSA encryption

Around 60 possible temperatures: 35 ... 41

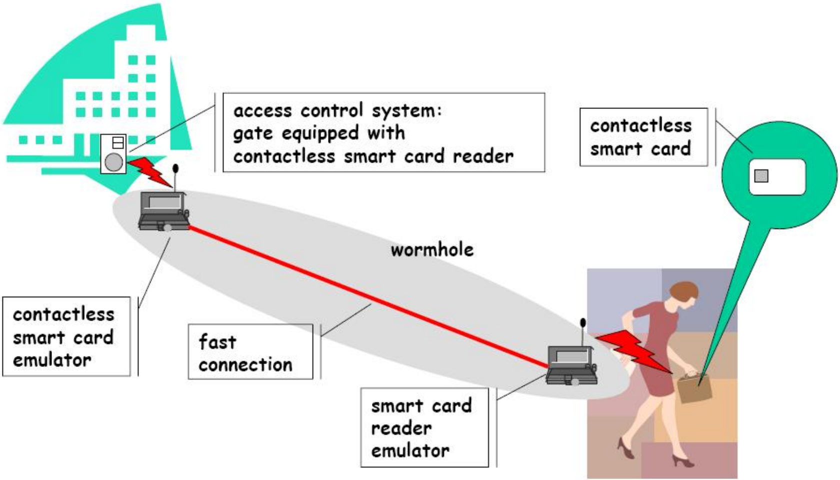
# Is it preserving your privacy?


4096 RSA encryption

Around 60 possible temperatures: 35 ... 41

$$\{35\}_{pk}, \{35, 1\}_{pk}, \ldots, \{41\}_{pk}$$

# Wormhole Attack



# Several Possible Attackers to Consider

- Insider vs Outsider
- Active vs Passive
- Local vs Extended
- Single vs Multiple
- Laptop vs Server






# Outline

# Things to bring home

Several **challenges** in **security**.

- Designing secure protocols is difficult
- Formal methods are useful for designing secure protocols



*Protocol + Properties + Intruder ⇒ Security*

Thanks for your attention

Questions ?