A Security Mechanism to Increase Confidence in M-Transactions

David Pequegnot Laurent Cart-Lamy Aurélien Thomas
Thibault Tigeon Julien Iguchi-Cartigny Jean-Louis Lanet

Smart Secure Devices (SSD) Team - Xlim Labs - University of Limoges
david.pequegnot@xlim.fr
http://secinfo.msi.unilim.fr

JTE NFC - February 6, 2012

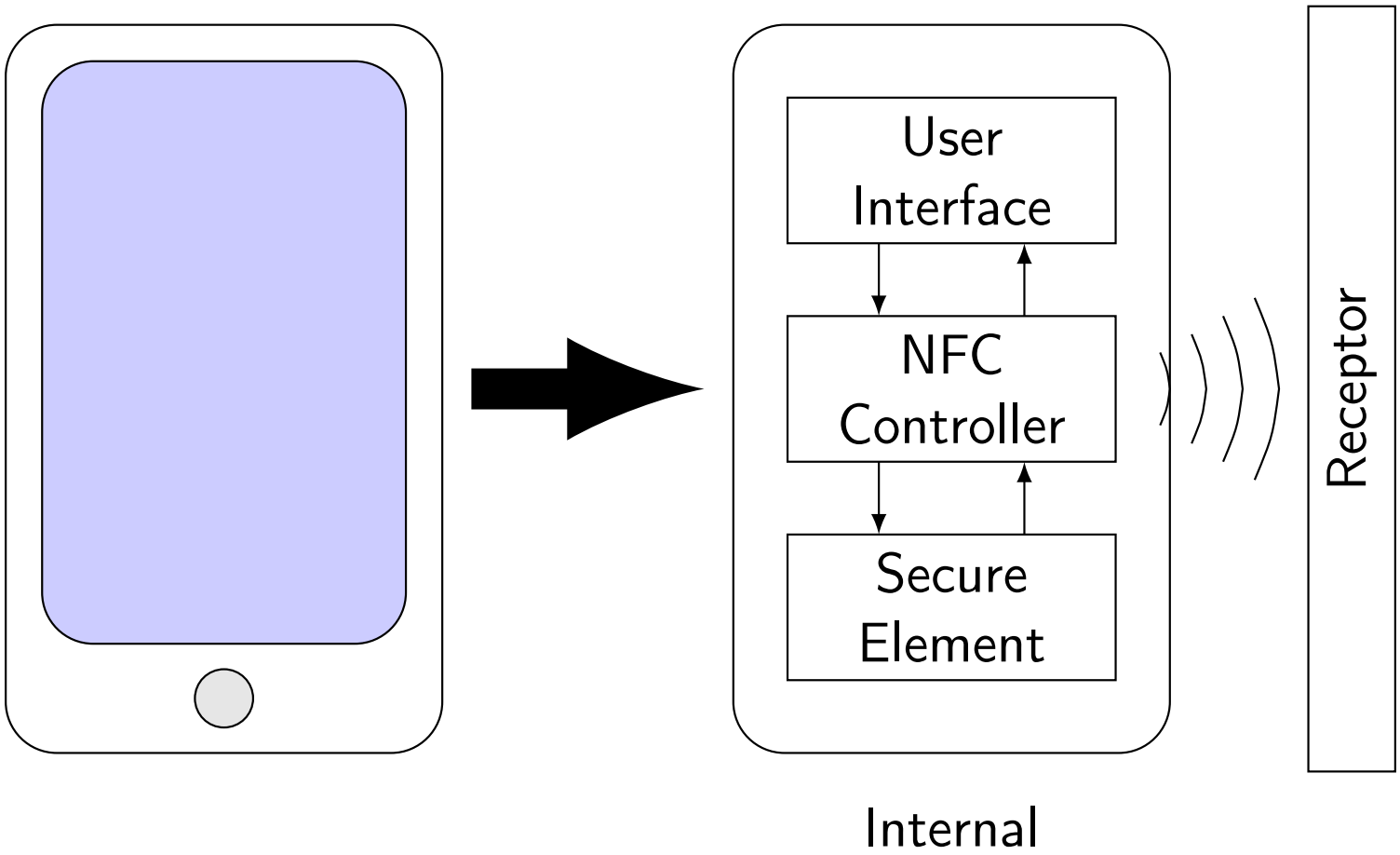
About the Smart Secure Devices team

- Led by Pr. Jean-Louis Lanet (jean-louis.lanet@xlim.fr),
- Xlim Labs, University of Limoges,
- **Security of embedded systems**,
- Research directions:
  - logical security,
  - hardware attacks,
  - network security.

Outline

Outline

Simplified diagram



Uses

Several possible uses:

- Payment,
- M-Ticketting,
- Authentication,
- Pairing/Data exchanges,
- *etc.*

Outline

What are we targeting?

- Feature phones and smartphones
  - evolved handheld devices,
  - allowing to download and install third-party applications.
- Take advantage of the mobility
  - thanks to an improved connectivity,
  - and a set of applications targeting the new needs of customers.

Problem

Applications, and especially mobile payment ones, trust too much handheld devices.

Mobile Payment

- Performing payments thanks to a handheld device
- Security:
  - the communication between the mobile phone and terminals is encrypted,
  - need credentials (login, password and/or a PIN code) to validate the payment.

Existing mobile payment solutions

But...

What happens if the terminal is the target of a malware?

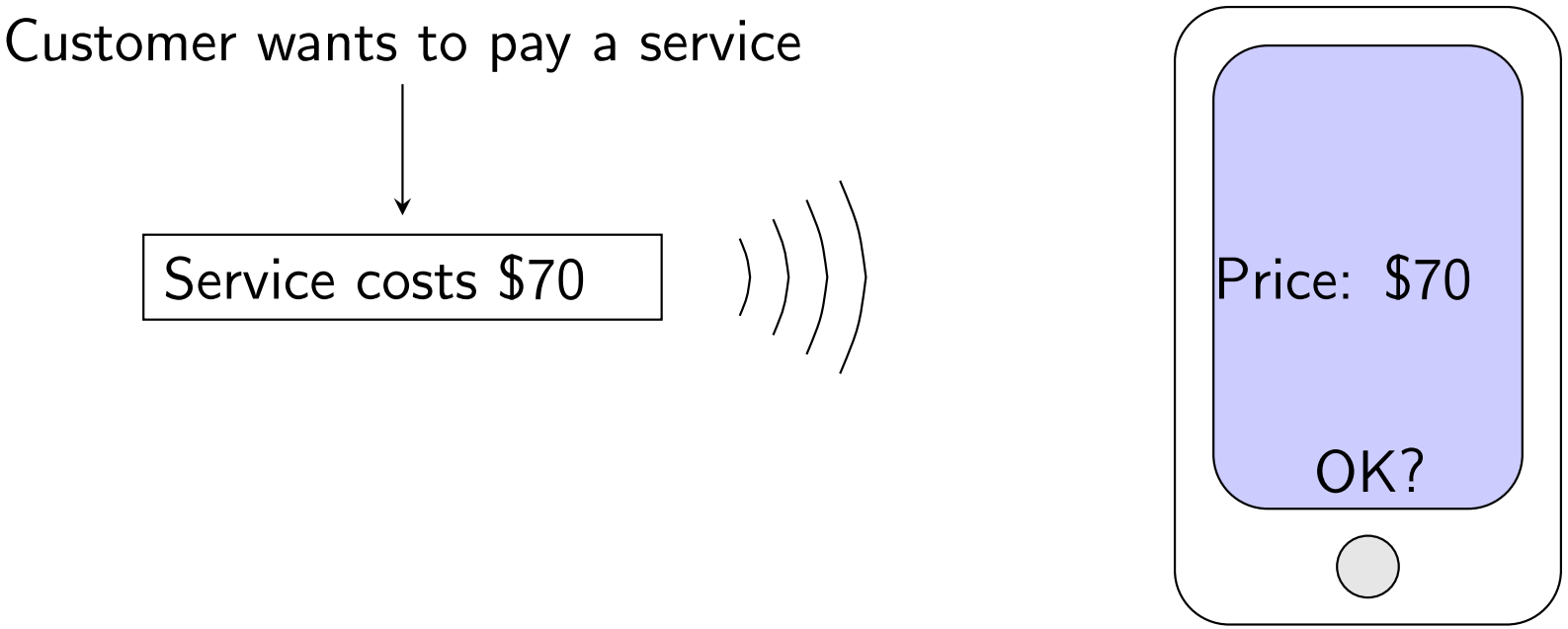
Main security issues

Two main potential security issues:

- **The amount displayed on the screen** of the mobile phone
  - amount detection,
  - amount modification.
- **"Key loggers"**
  - listen the keyboard to retrieve keys entered by the user,
  - send the retrieved keys to a hacker.

Trusting the display

Expected behavior

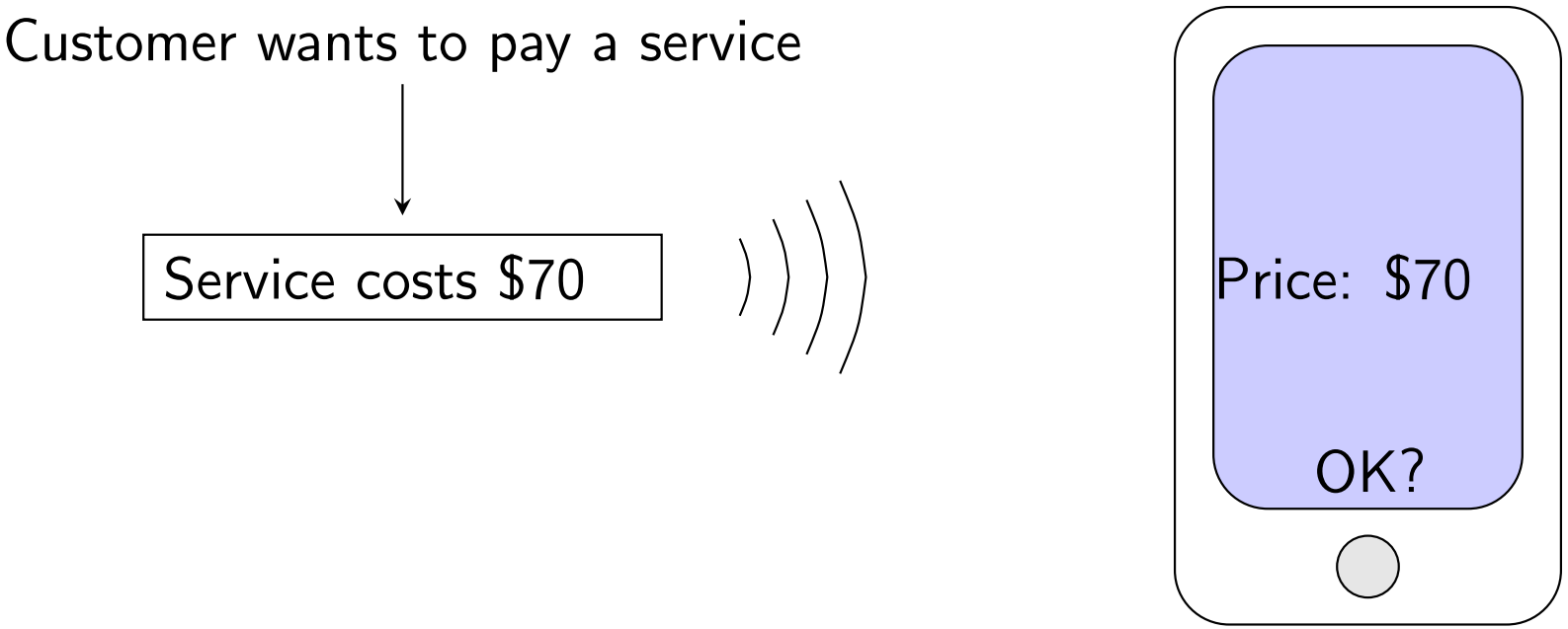


Problem

A malware may modify the display...

Trusting the display

Expected behavior

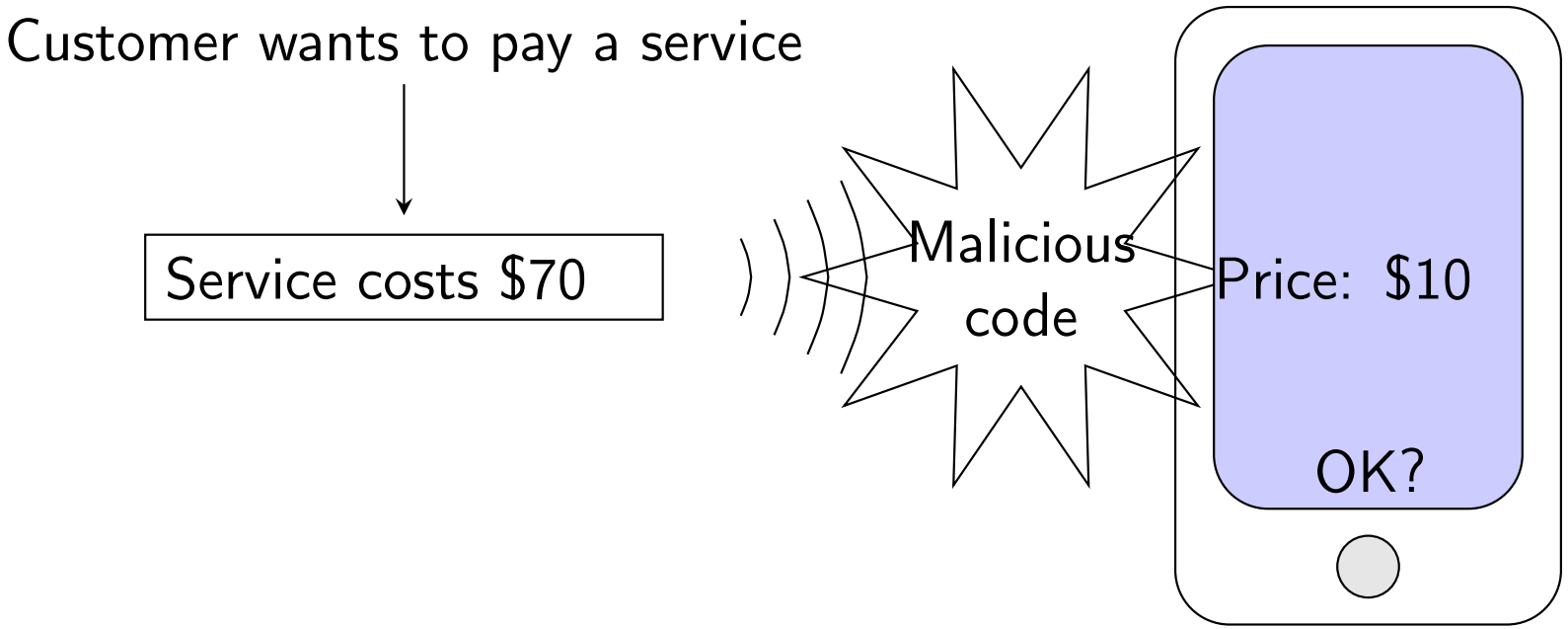


Problem

A malware may modify the display...

Trusting the display

Abnormal behavior

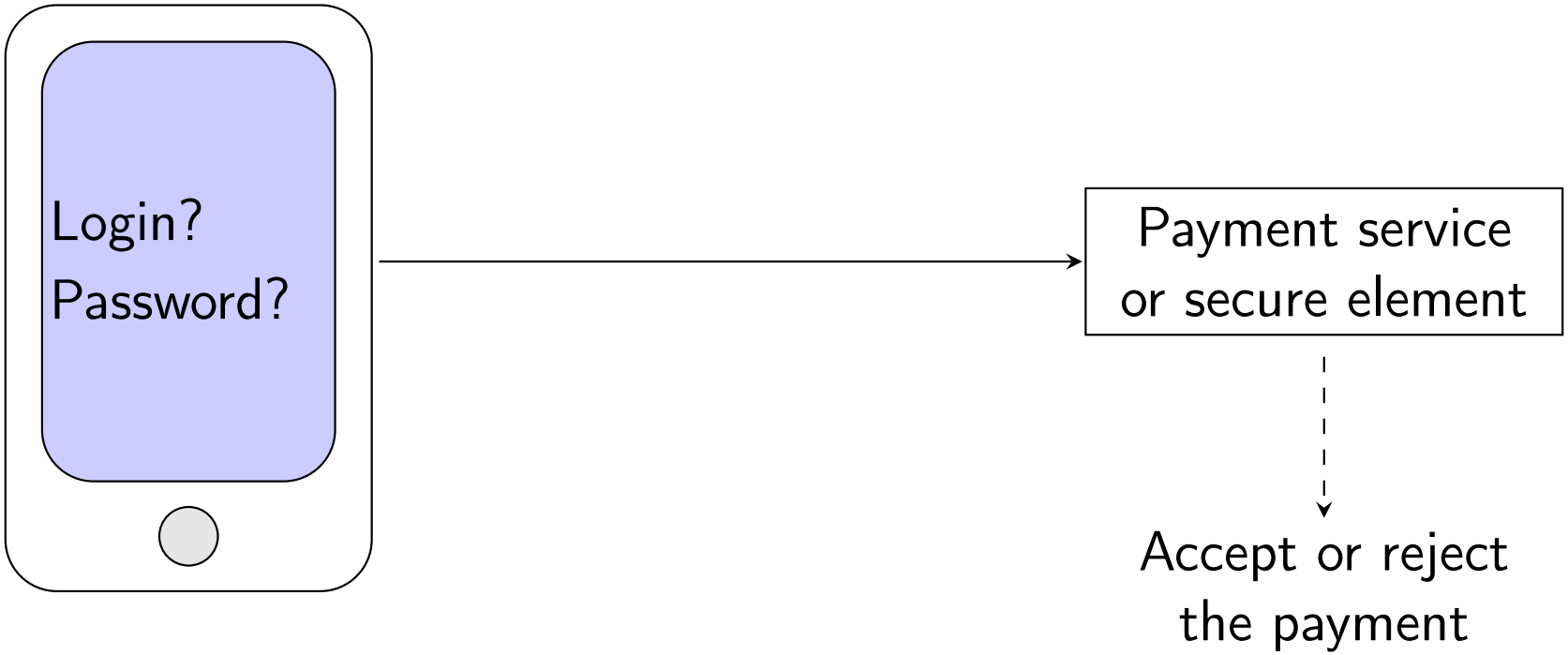


Problem

A malware may modify the display...

Trusting the keyboard

Expected behavior

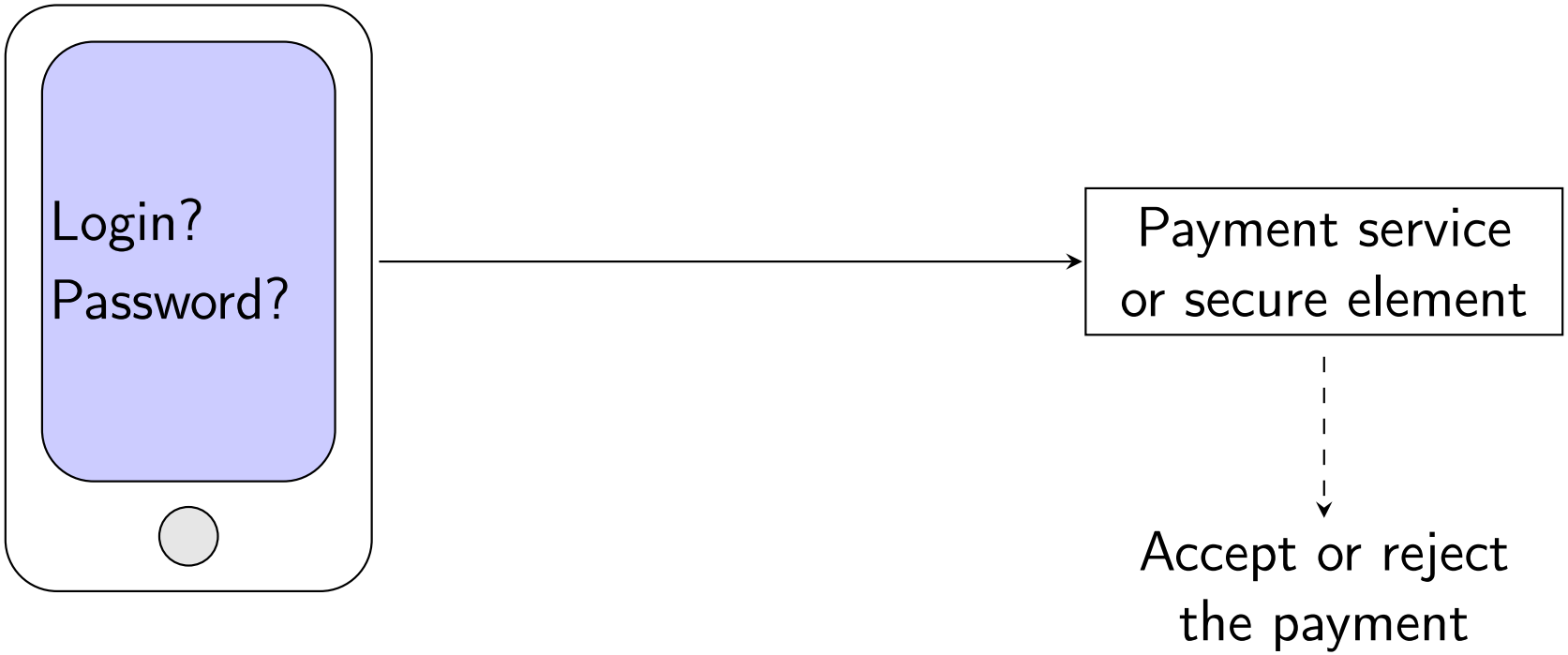


Problem

A malware may listen to the keyboard...

Trusting the keyboard

Expected behavior

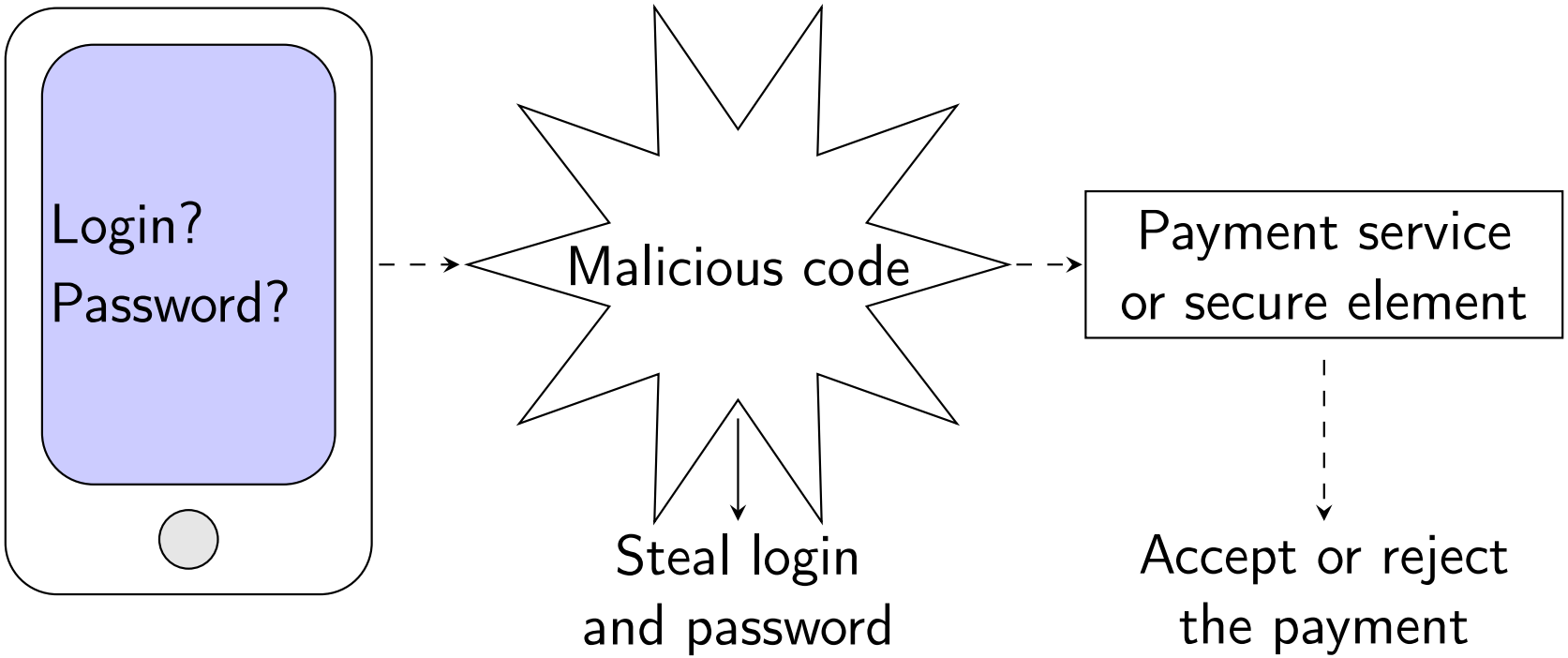


Problem

A malware may listen to the keyboard...

Trusting the keyboard

Abnormal behavior

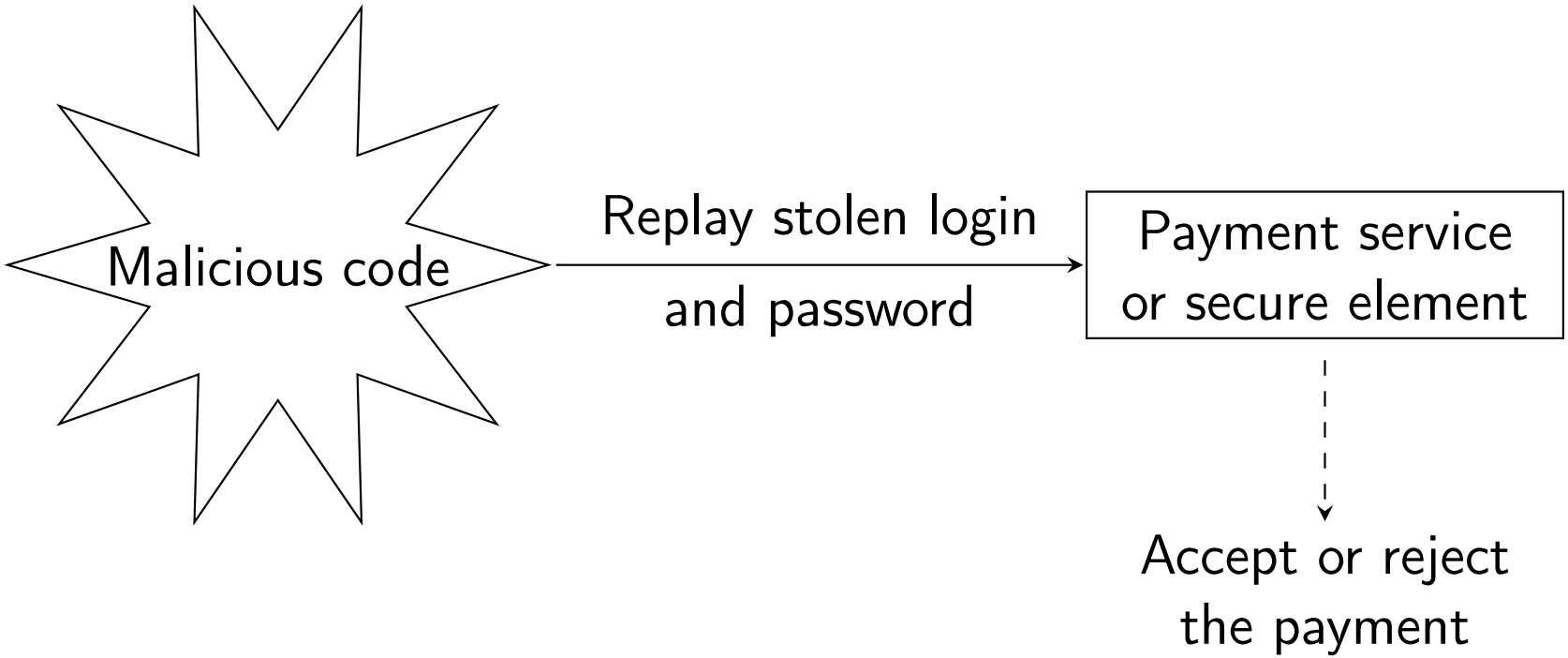


Problem

A malware may listen to the keyboard...

Trusting the keyboard

Abnormal behavior



Problem

A malware may listen to the keyboard...

Questions

- How can we ensure that a malware cannot modify the display?
- How can we ensure that there is no "key logger" on the mobile phone?

Solutions?

- Developing a secure mobile phone, in which we cannot install third party applications,
- Certifying the application code,
- *etc.*

Outline

Introducing the security mechanism

Requirements

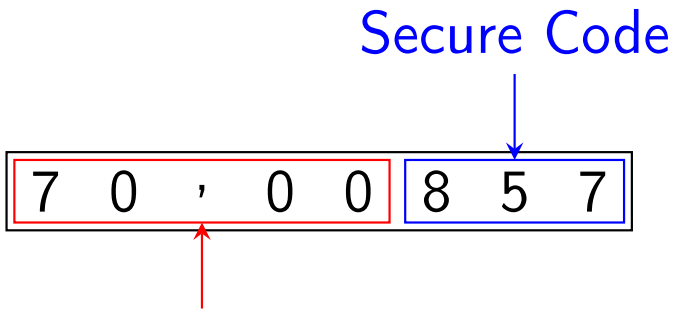
- **Payment service**: the payment request should not come from a malware or an unauthorized person,
- **Customer**: the amount displayed on the device should not been modified by a malware or intentionally.

Security based on

- a **Secure Element**, which contains all the secrets to perform payments,
- a **Graphical Turing Test**, adapted to suit our needs.

The security mechanism

- Graphical Turing Test: a modified CAPTCHA
- The picture contains two information:
  - the **payment amount**,
  - a randomly generated **Secure Code**.

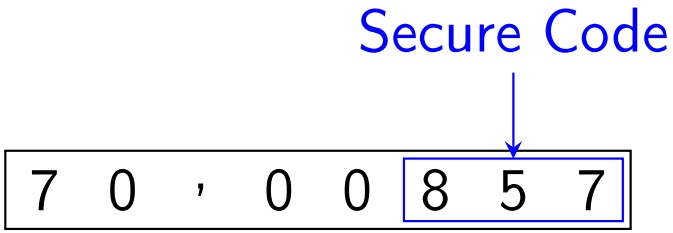


Example of CAPTCHA layout
(does not include protections)

Goal

Malicious code should not be able to read the picture and modify its content.

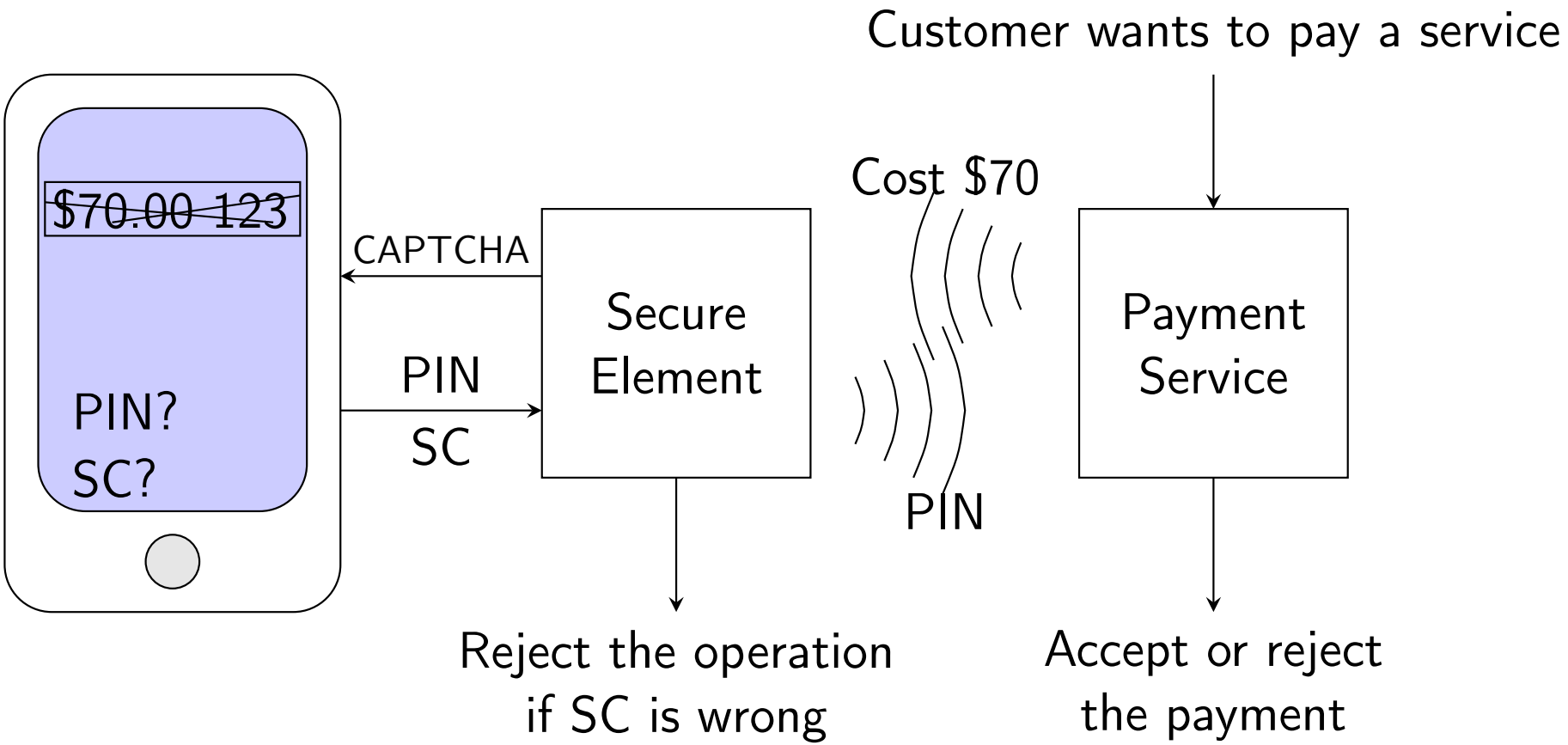
Performing a payment



To perform a payment, the customer must enter:

- the set of credentials needed by the payment service,
- the generated Secure Code.

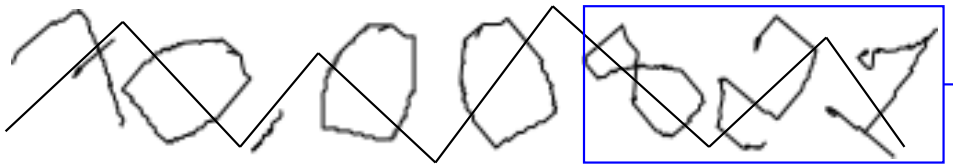
Payment process



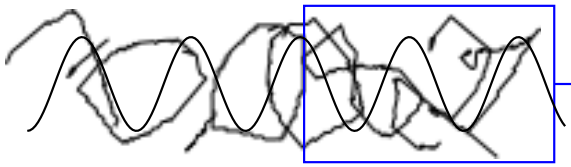
Nota Bene: Credentials can be validated by both the PS or the SE.

Samples

A first CAPTCHA implementation:

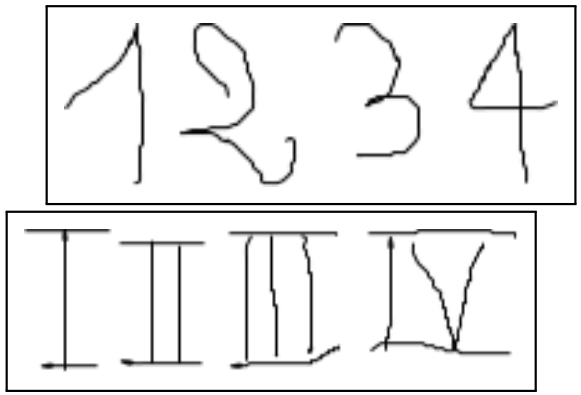


A second, more complex, CAPTCHA implementation:



Semantics

- Use of customer's handwriting, instead of a default font
  - optional,
  - the customer is able to recognize his handwriting,
  - malicious code cannot be designed to recognize a single font.
- Possibility to define properties on the picture generation

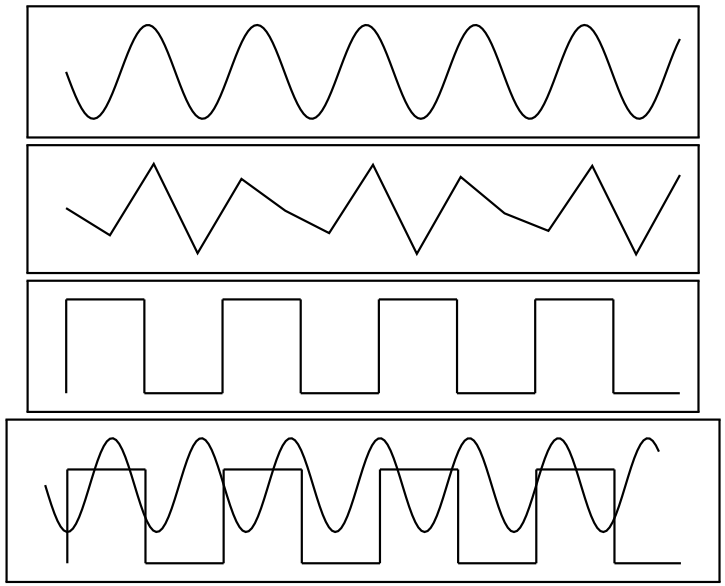


Advantage

Personalization makes character recognition and picture modification difficult.

Semantics

- Use of customer's handwriting, instead of a default font
- Possibility to define properties on the picture generation
  - **the parasite curve shape**.

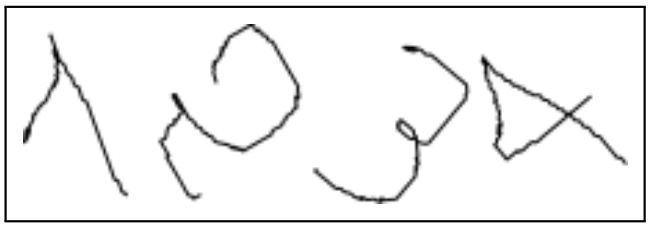


Advantage

Personalization makes character recognition and picture modification difficult.

Semantics

- Use of customer's handwriting, instead of a default font
- Possibility to define properties on the picture generation
  - the parasite curve shape,
  - **rotation of the digits**.

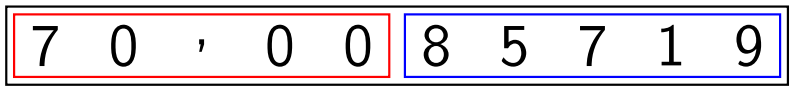


Advantage

Personalization makes character recognition and picture modification difficult.

Semantics

- Use of customer's handwriting, instead of a default font
- Possibility to define properties on the picture generation
  - the parasite curve shape,
  - rotation of the digits,
  - **Secure Code length**.

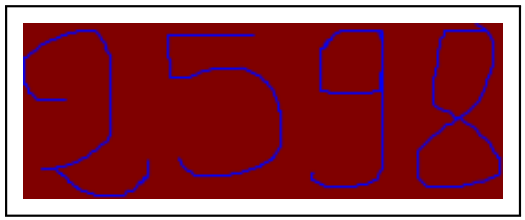


Advantage

Personalization makes character recognition and picture modification difficult.

Semantics

- Use of customer's handwriting, instead of a default font
- Possibility to define properties on the picture generation
  - the parasite curve shape,
  - rotation of the digits,
  - Secure Code length,
  - **foreground and background colors,**
  - **timestamp,**
  - ***etc.***



Advantage

Personalization makes character recognition and picture modification difficult.

Protections

- To modify the amount, the malware must:
  - recognize the amount part,
  - be able to modify the amount without altering the semantics.
- To perform a payment, the malware must:
  - recognize the Secure Code in the picture.
- The CAPTCHA complexity, thanks to the semantics, makes attacks difficult
- If credentials have been stolen, the Secure Code is still needed

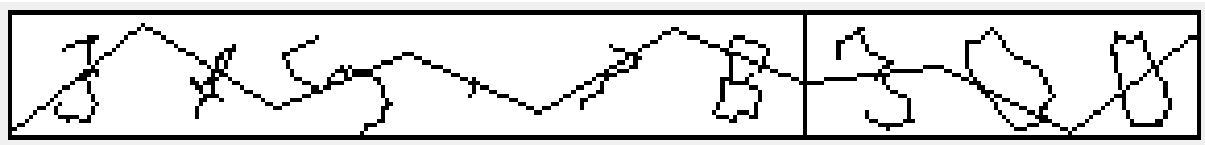
Outline

First implementation

- Proof of concept
- Secure Element:
  - SIM card,
  - Java Card environment,
  - we must deal with environment limitations...
- Only a subset of the semantics and protections previously listed were implemented

Protections and sample

- Rotations
- Digits deformation
- Scaling
- One parasite curve across the whole image
- Timestamp



A CAPTCHA generated by the Secure Element with our implementation

Metric

- Hardware: Gemalto GemXplore 3G v3 SIM Card
- First naive implementation: 1'46" to generate and receive the entire CAPTCHA
  - 5-digit amount,
  - 3-digit Secure Code.
- Optimizations: 16"
  - same CAPTCHA properties.
- On a more recent card, a Gemalto TOP GX4, with further improved optimizations:
  - 2" (the 16" implementation took 9" on this card).

Outline

Limitations

Implementation

- Not a full implementation:
  - the user cannot define the font,
  - only one parasite curve,
  - all protections have not been implemented.
- Timestamp verification not secure
- May be subject to time and power analysis

Limitations

Security mechanism

- The Secure Element is the Trusted Computing Base
- If a malware is able to read the CAPTCHA, then it may perform payments (need to know credentials)
- Do not prevent the theft of logins, passwords, *etc.* by a keylogger

Advantages

- Works on NFC-based transactions...
- ... and Internet-based ones!
- Application areas:
  - Ticketting,
  - Payment,
  - *etc.*

Future works

- Full implementation of the security mechanism
- Secured implementation of the timestamp mechanism
- Implementation on a native card instead of a Java Card

Conclusion

- A simple security mechanism
- Able to introduce more confidence in m-transactions on untrusted devices
- Difficult for a malware:
  - to perform payments without the user being aware,
  - to display a wrong amount.

Thank you for your attention!
Any questions?

?

david.pequegnot@etu.unilim.fr
http://secinfo.msi.unilim.fr