# *e-Commerce Security, Challenges in Secure Element Security*

## Part I Smart Card & Java Card

Challenges in Cyber Security - from paradigms to implementations

Bucharest, Romania, 17-25 Aug. **2012**

**Jean-Louis Lanet**

Jean-louis.lanet@unilim.fr

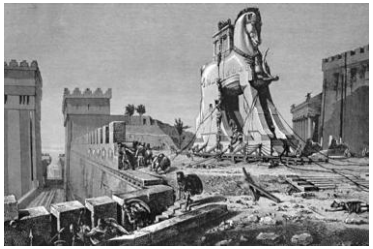
Université de Limoges

SUFOP Service Universitaire de Formation Permanente

# *e-commerce*

- Development of e-commerce relies on the confidence customers have on the service,
- Increasing demand on mobile e-commerce implies the a confidence into the mobile device,
- The seed of trust of a mobile device is the secure element,
- **Can we trust the Secure Element ?**

# Presentation Goal



Learn how smart cards are vulnerable to Trojan.

# Agenda

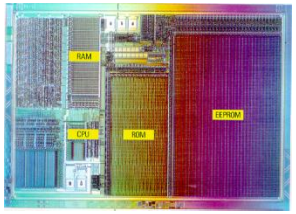
- Saturday, Bucharest
  - Part I  Java Card Security
    - Virtual Machine architecture
    - Hardware attacks
  - References
- Monday, Busteni
  - Part II Logical Attacks
    - Type confusion, Control flow deviation
    - Executing arbitrary code, and countermeasures
    - Laser beam as an enabling technology for combined attacks

# Agenda

- Part I  Java Card Security
  - Introduction to Smart Card
  - Virtual Machine architecture
  - Hardware attacks
  - Assets

# What is a Smart Card?

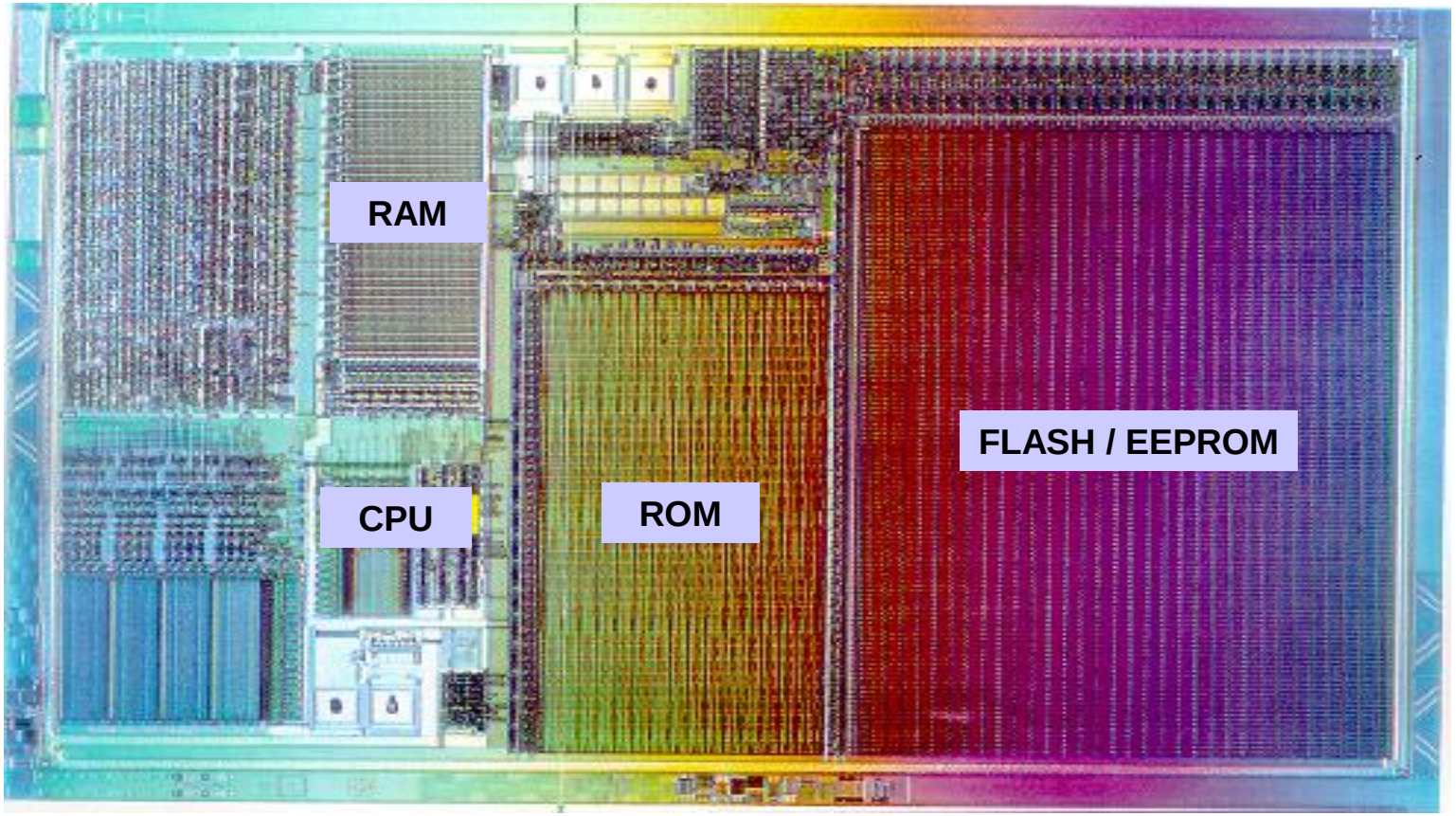
A piece of silicon on a plastic body




Chip

A very secure way of storing a small amount of sensitive data

# Microcontroller of the card



# Different Types of Memory ...

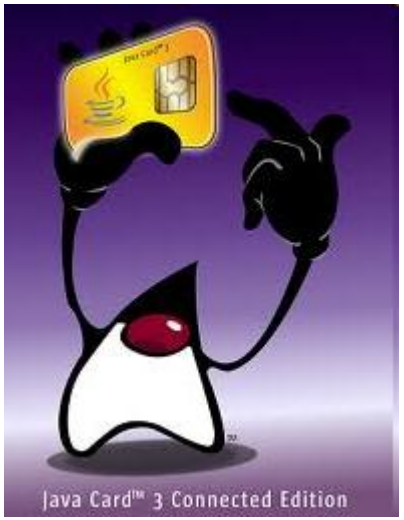
- ROM : CPU only NO ACCESS !
  - used for embedded Operating System
- EPROM : Write once, read FOR EVER !
  - Used for initialization area (e. g. Lock bytes)
- EEPROM : Write, erase, read FLEXIBLE !
  - used to store applicative data or added functionalities
- RAM : Write, read TEMPORARY !
  - used during power on sessions only

# OS: Open cards vs. Native cards

- From a developer point of view:
  - Until now, writing an application required a specific knowledge,
  - No need of smart card specialists,
  - Solution: use general purpose programming language (C, Java, …)
  - Much more easier to integrate applications,
  - More tools to test applications,
- From an end-user point of view
  - Several application on a single card,
  - Possibility to load/unload application when required.

# Smart cards of the present days

- Java Card
  - Embedded virtual machine, post issuance,
  - Open standard (Java Card 3.0),
  - Wide support of the industry and customers,
  - Reduction of development time,
  - Flavor of security
  - Interoperability and multi applicative cards.




- Multos
  - Based on the MEL (Multos Executable Language) interpreter.
  - Operating system and memory firewalls, virtual Machine layer to provide abstraction
  - Application management including secure loading and deleting methods

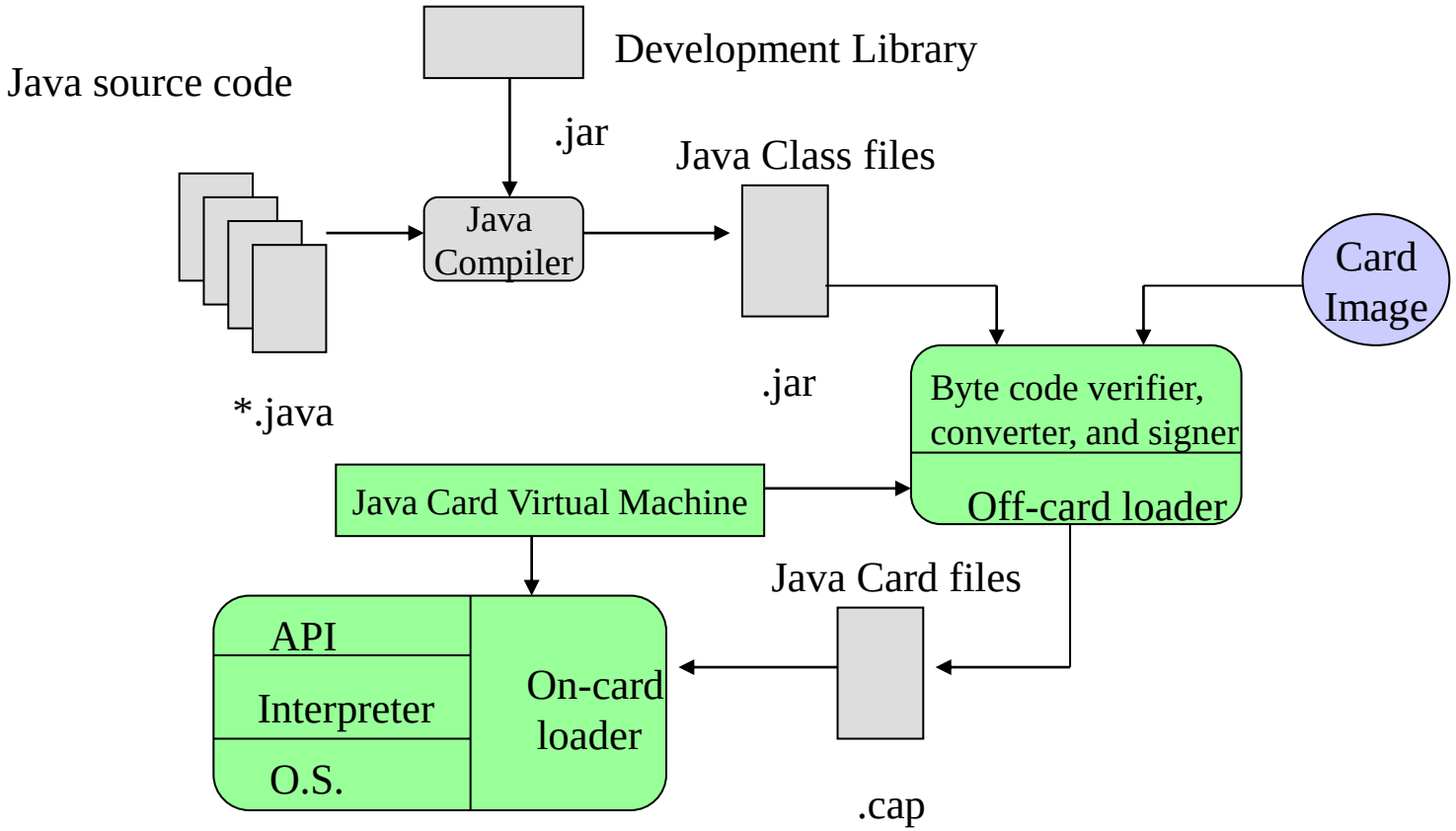
# Java Card Applet development

- Write your Java code
- Compile it
- Debug it (simulator)
- Verify and Convert it (specific byte code)
- Load it
  - Personalization center
  - Point of sale
  - Over the Internet

# Agenda

- Part I  Java Card Security
  - Introduction
  - Virtual Machine architecture
  - Hardware attacks
  - Assets

# Java Card Architecture



# Two specific file formats

- The CAP (Converted Applet) file format
  - Contains all the classes from one package
  - Semantically, is equivalent to a set of class (.class) files
  - Syntactically, differs a lot from class (.class) files
    - All “string names” are replaced by “token identifiers”,
    - Byte codes are different
- The EXP (Export) file format
  - Maintains the consistency between the originated class (.class) files and the resulting CAP file
  - Not loaded into the card

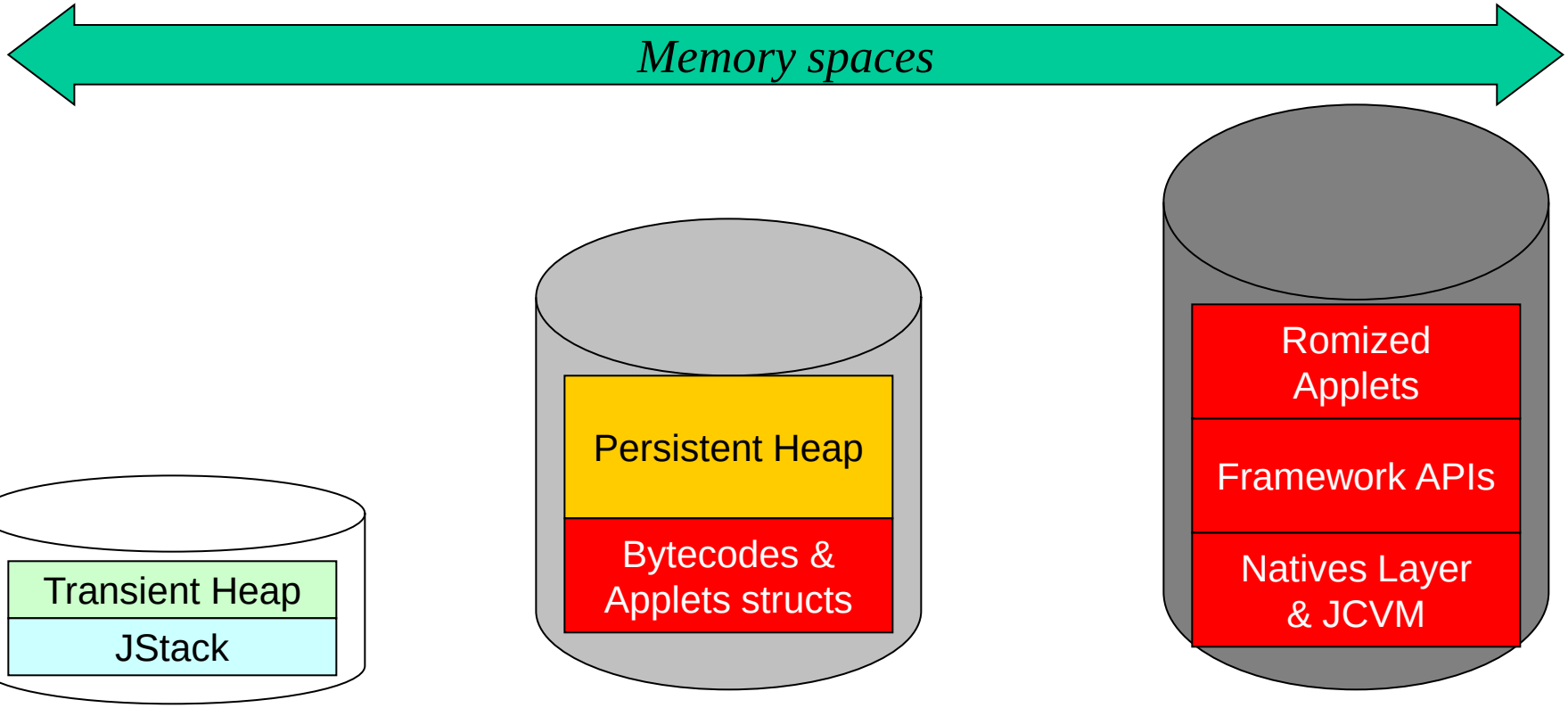
# The CAP file

- Contains an executable representation of package classes
- Contains a set of components (11)
- Each component describes an aspect of CAP file
  - Class info
  - Executable byte code
  - Linking info,…
- Optimized for small footprint by compact data structure
- Loaded on card

# Java Card Memory Model

- By default, all objects are *implicitly* persistent
  - Because we have few RAM
  - Objects must survive between two sessions
- Some arrays can be *transient*
  - For efficiency and security reasons
- Transactional mechanisms are provided
  - All write operations on persistent memory are atomic
  - At the programming level a mechanism to handle transactions is also available

# Java Card Memory



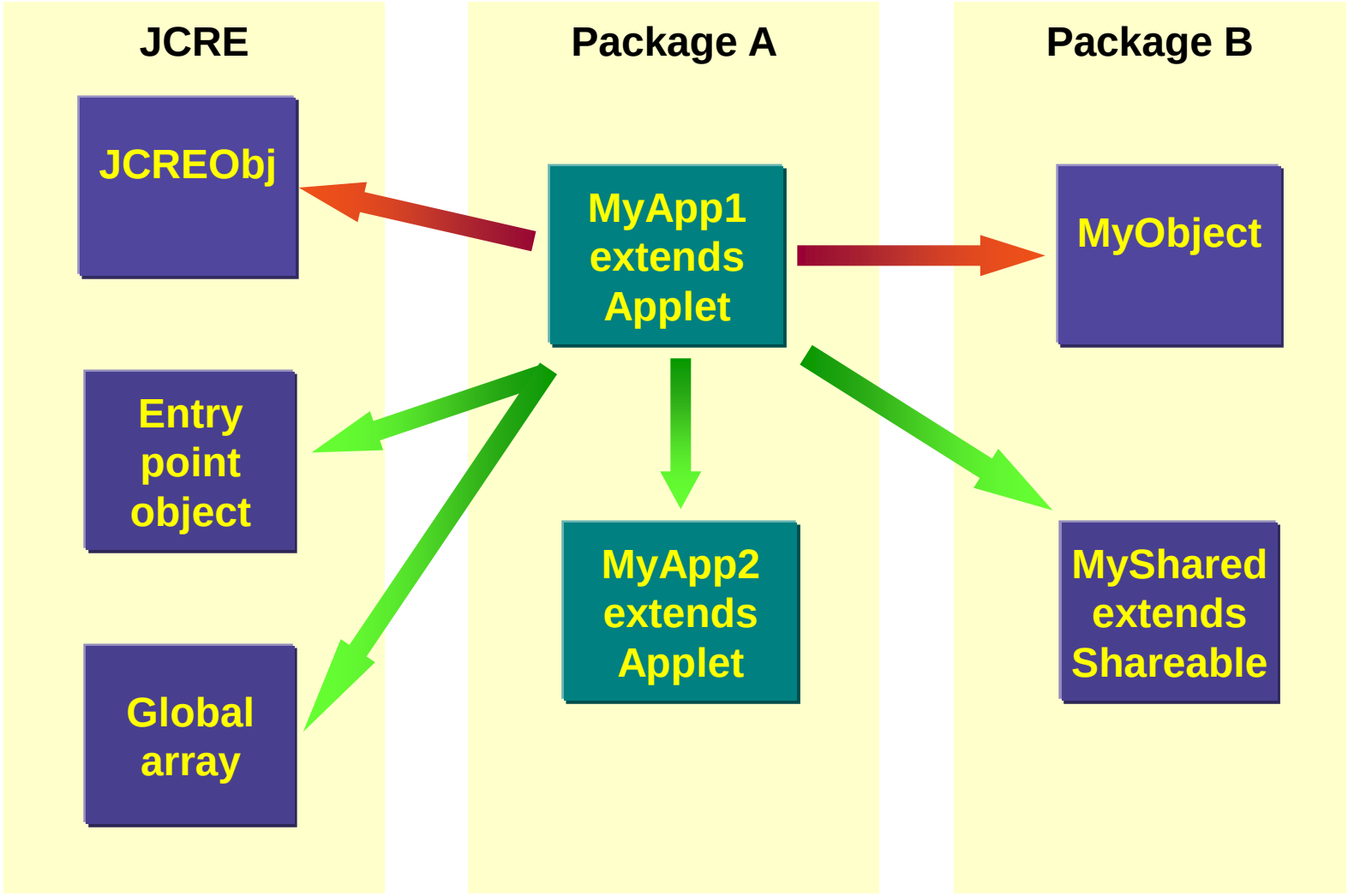

**RAM (~1Kb)**

EEPROM (~32Kb)

ROM (~64Kb)

# Execution environment (JCRE)

- Define how a Java Card manages its resources
- Define constraints on the Java Card operating system
  - Applet lifetime (installation, register and deletion)
  - Logical channels and applet selection,
  - Transient objects,
  - Applet isolation (firewall) and sharing based on security context,
  - Transaction and atomicity,
- The JCRE is at the heart of a Java Card


JCRE
Package A
Package B
JCREObj
MyApp1 extends Applet
MyObject
Entry point object
MyApp2 extends Applet
MyShared extends Shareable
Global array

# BC interpretation

- It is the execution engine for the byte code loaded into the card,
- It controls byte code execution, memory allocation and participate to the security through the firewall,
- If the interpreter is a defensive one (run time type verification) it cost too much memory and CPU,
- If the interpreter is the one defined by Sun …
- Often it includes more tests than the firewall due to the absence of BC verifier…

# Agenda

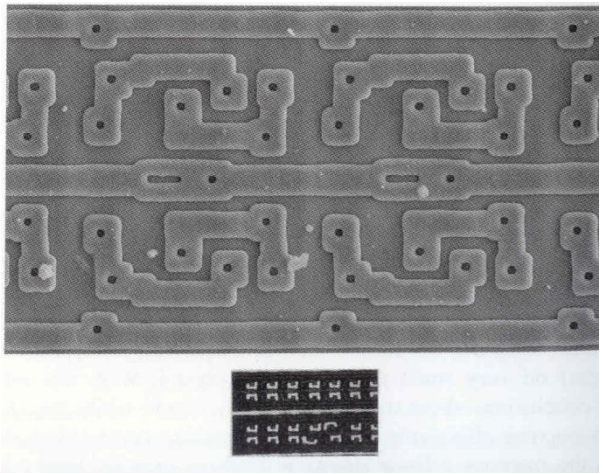
- Part I  Java Card Security
  - Introduction
  - Virtual Machine architecture
  - Hardware attacks
  - Assets

# Target

- Hardware attacks aim at recovering assets with some physical means.
  - Invasive attacks,
  - Non invasive attacks,
- Invasive attacks need a lot of costly equipment, training,…
  - For institutional labs, destruction of the samples
- Non invasive attacks
  - Affordable for public labs.

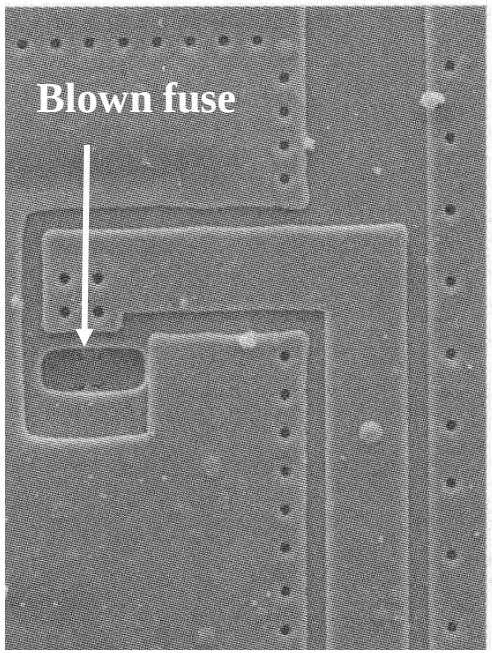
# Invasive attacks

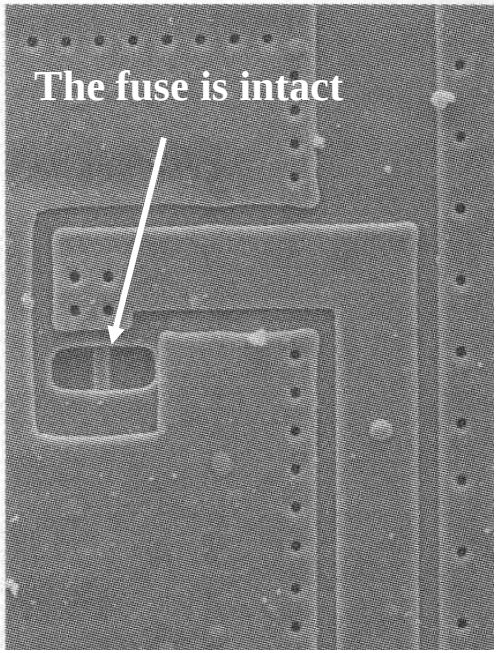
- Chip is physically and irreversibly modified (remove the glue, can be visually detected later)
  - Passive attacks :
    - off line : reverse engineering of ROM code, but the chip structure (0.35µm) reduce it, the ROM is on deep level to avoid optical reading, dummy structure
    - in line : information reading (bus, memory, etc…) by probing or analysis of electrical potential. Counter measure scrambling, protective metallic layers see below with an electron beam tester
  - Active attacks :
    - off line : modification of the component,
    - in line : injection of information.

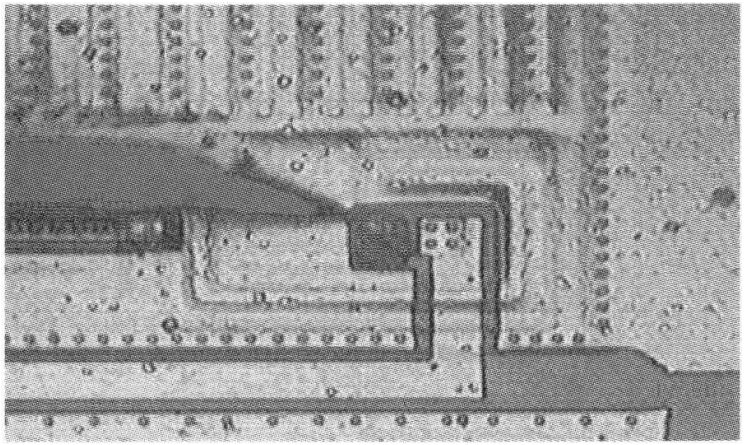


# Irreversible switching from test mode to user mode !!!

- All chip have a test mode and use poly silicon fuse on the chip to switch to user mode.



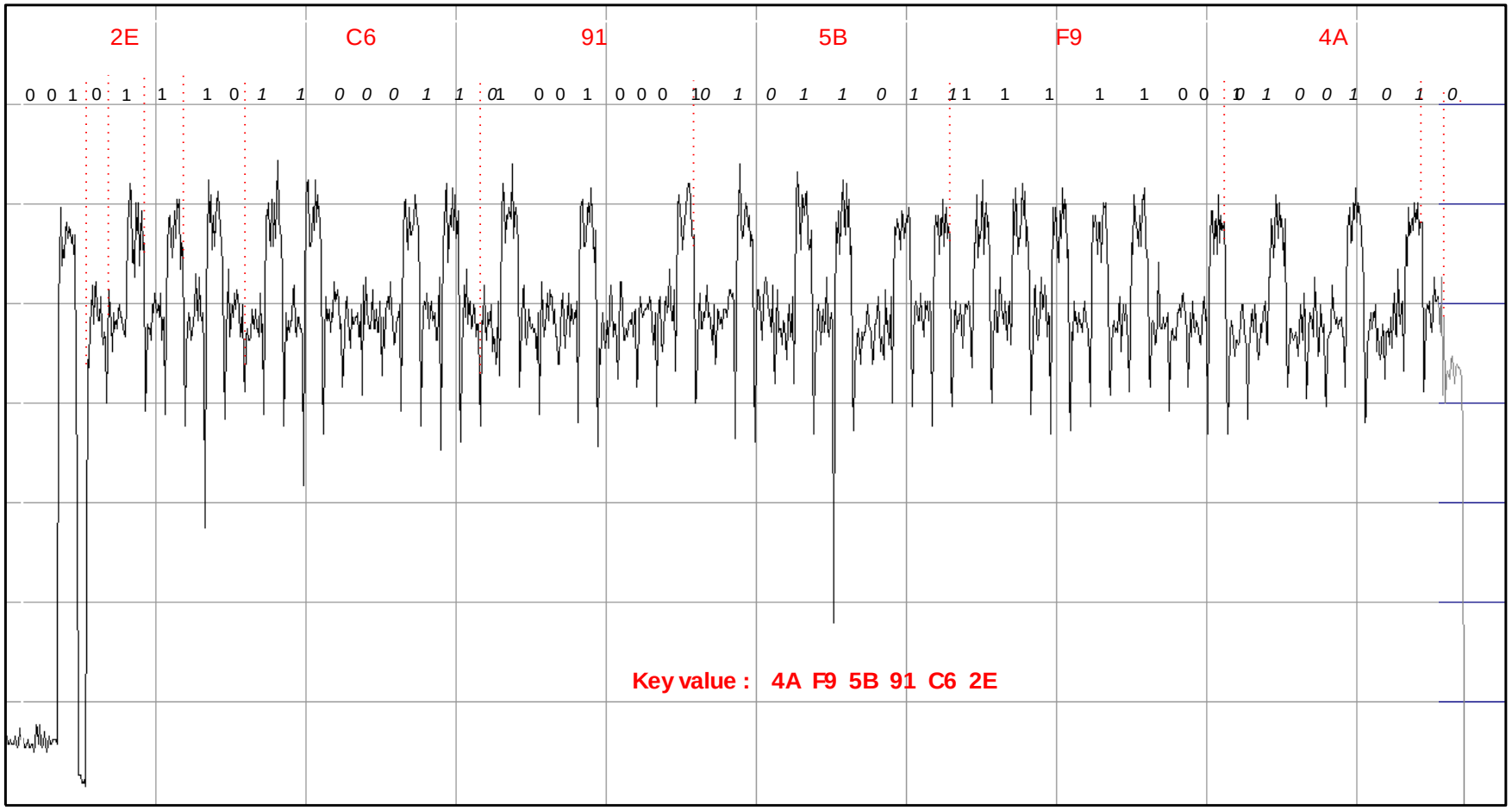



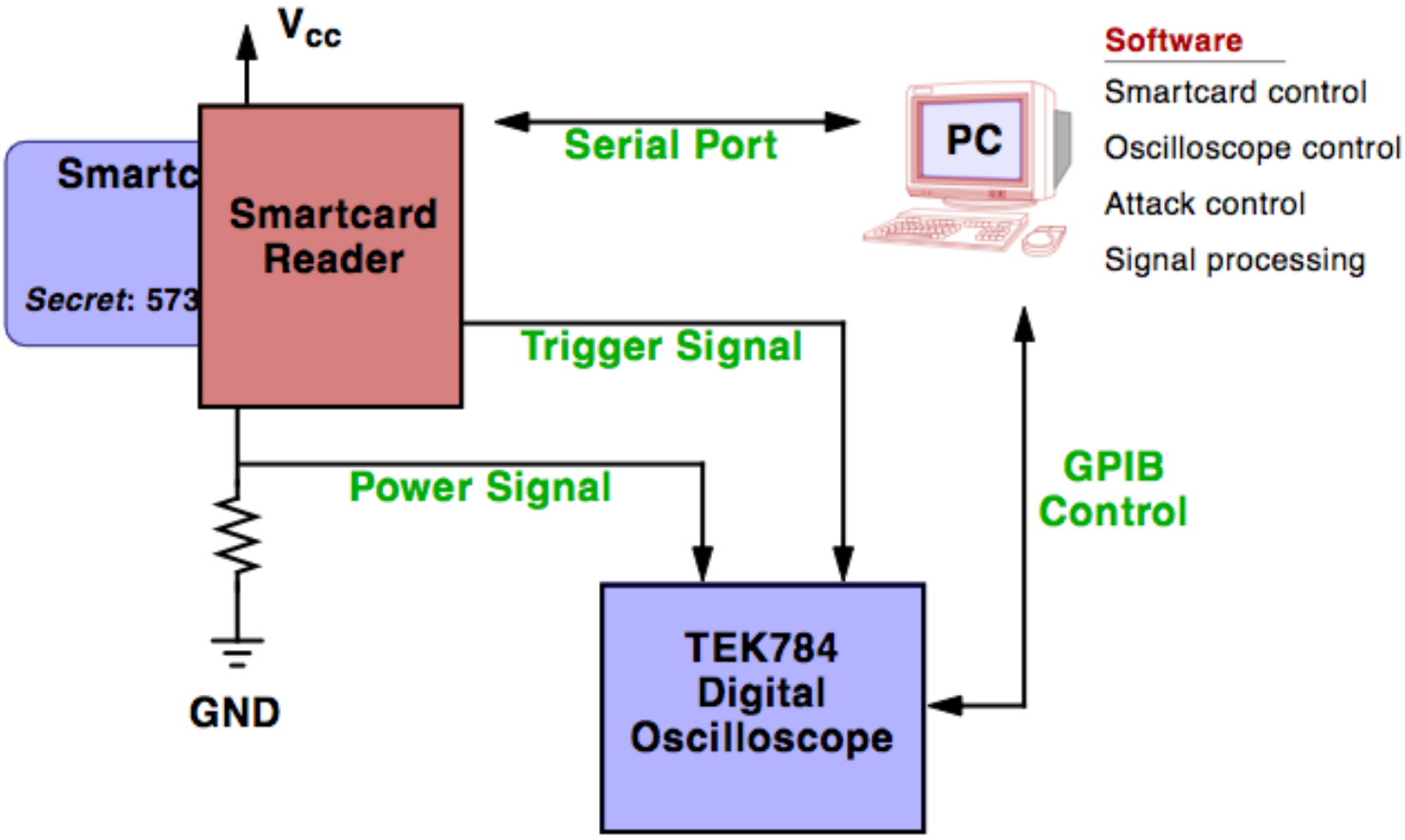
# Non invasive attacks

- Simple Power Analysis,
  - Khokar et al., June 1998: Measure instantaneous power consumption of a device while it runs a cryptographic algorithm
  - Measurable variation in current consumption depending on instruction and data processed,
  - Different power consumption when operating on logical ones vs. logical zeroes.
- Differential Power Analysis (same but statistical…).
- Glitch attack & **Fault induction.**
- Random number generator attack.

# RSA attack - private key



# Equipment used



# Agenda

- Part I  Java Card Security
  - Introduction
  - Virtual Machine architecture
  - Hardware attacks
  - Assets

# Assets

- Products can only be attacked with hardware means,
  - Allows to retrieves keys, retro-engineering of the code, modification of the control flow,
  - Needs specific knowledge, often heavy investment,
- Software attacks (upload of hostile code) can only be done on development cards,
  - Allows to retrieves keys, retro-engineering of the code, modification of the control flow,
  - Needs good knowledge on Java, affordable for students (hacker kit 20€),
  - Warning :
    - `development card == a product card – upload;`

# Assets

- What you discover on a development card can help to attack products,
  - Banking cards share the same OS-VM that dev. cards,
- Reverse the operating system of a card then you obtain all the algorithms used in a product:
  - Discover the embedded countermeasures,
  - Attack with a white box approach a product,

# Software attacks

- Two different approaches:
  - Type confusion
  - Modification of the control flow
- Two main hypotheses :
  - The attacker has the right to upload code, he has the keys to authenticate and to sign the code,
  - The byte code verifier is not embedded: pure software attack,
  - There is a byte code verifier, a physical mean can be used.

# References

- All the papers are available on the web site: **`http://secinfo.msi.unilim.fr/~lanet`**
- Smart Card attacks
  - *Developing a Trojan applet in a Smart Card*, J-L. Lanet, J. Iguchi-Cartigny, Journal in Computer Virology, Vol. 6, Issue 4, pp. 343-351, **2010**
  - *The Next Smart Card Nightmare Logical Attacks, Combined Attacks, Mutant Applications and other Funny Things*, G. Bouffard, J.-L. Lanet, Cryptography and Security: From Theory to Applications Lecture Notes in Computer Science, 2012, Volume 6805, pp. 405-424
  - *Smart Card Reverse-Engineering Code Execution Using Side-Channel Analysis*, NTCCCS, Théorie des Nombres, Codes, Cryptographie et Systèmes de Communication, Oujda, Maroc, April **2012**,
  - *A friendly framework for hidding fault enabled virus for Java based smartcard*, T. Razafindralambo, G. Bouffard and J.-L. Lanet, 26th Annual IFIP WG 11.3 Working Conference on Data and Applications Security and Privacy DBSEC 2012, LNCS vol. 7371, pp 122-128, Paris, July 11-13, **2012**
  - *Combined Software and Hardware Attacks on the Java Card Control Flow*, **CARDIS'11**, G. Bouffard, J. Iguchi-Cartigny, J.-L. Lanet, pp. 283-296, Leuven, Belgium, 14-16 September **2011**,

# References

- Design of countermeasures
  - *Evaluation of Countermeasures Against Fault Attacks on Smart Cards*, A. Sere, J-L. Lanet, J. Iguchi-Cartigny, International Journal of Security and Its Applications Vol. 5 No. 2, pp 49-61, April, **2011**
  - *A Dynamic Syntax Interpretation for Java Based Smart Card to Mitigate Logical Attacks*, T. Razandralambo, G. Bouffard, B. N Thampi, J.-L. Lanet, SNDS 2012, Trivandrum, India, 11-12 October **2012**
  - *Type classification against Fault Enabled Mutant in Java based Smart Card*, J. Dubreuil, J.-L. Lanet, G. Bouffard, J. Cartigny, SecSE 2012, Prague, Czech Republic, 20-24 August **2012**
  - *Incremental Dynamic Update For Java-based Smart Cards*, ICONS 2010, A. Noubissi, J. Cartigny, J.-L. Lanet, April **2010**, Les Menuires
  - *Automatic detection of fault attack and countermeasures*, WESS'2009, A. Sere J., Iguchi-Cartigny, J-L. Lanet, 4th Workshop on Embedded Systems Security, October 15, **2009**, Grenoble

# References


- Fuzzing
  - *Enhancing fuzzing technique for OKL4 syscalls testing*, International Workshop on Secure Software Engineering, SecSE, A. Gauthier, C. Mazin, J. Iguchi-Cartigny, J.-L. Lanet, August 22-26, **2011**, ARES Conference publication pp.728-733, Vienna, Austria,
  - *Fuzzing on the HTTP protocol implementation in mobile embedded web server*, M. Barreaud, G. Bouffard, N. Kamel, J.-L. Lanet, C&ESAR11, Rennes, France, November **2011**
- Formal method for vulnerability discovering
  - *VTG : Vulnerability Test cases Generator, a Plugin for Rodin*, A. Savary, J.-L. Lanet, M. Frappier, T. Razafindralambo, Rodin User and Developer Workshop, Fontainebleau, France, 28th February **2012**,
  - *Automatic generation of vulnerability test suite for the Java Card verifier*, A. Savary, M. Frappier, J.-L. Lanet, E-smart 11, 21-23 September **2011**, Sophia Antipolis

Virtual machine paradigm brings security…

Its implementation in a constrained device brings insecurity !!!

University of Limoges is interested as an output of this summer school:

- To motivate skilled student to apply for an internship, second year of the master in security: Cryptis, fo apply for a PhD or a post doc,
- To establish links with Romanian University either for research or teaching in the smart card domain,
- To join a proposal of an European Project dedicated to security