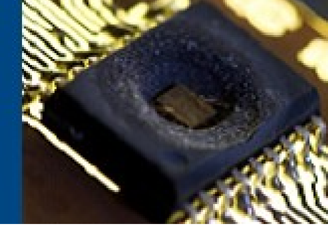


Smart Card Laboratory
Introduction

Prof. Dr.-Ing. Georg Sigl
Institute for Security in Information Technology
Lecturer: M.Sc. Michael Gruber

Summer Semester 2019

What is the Smart card Laboratory (in a nutshell)?

Practical embedded-systems project

Focus on **hardware security,** specifically side-channel (power) analysis

Work conducted in **teams** of 4 people

The course is designed to let you practice how to…

Design, **implement** and **debug** a complex embedded system

Plan, **manage** and **execute** a sizable project

Evaluate cost/benefit **tradeoffs** of security measures

Work effectively in a **team**

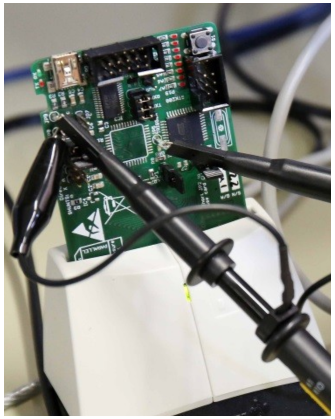
Project Description

Objective:

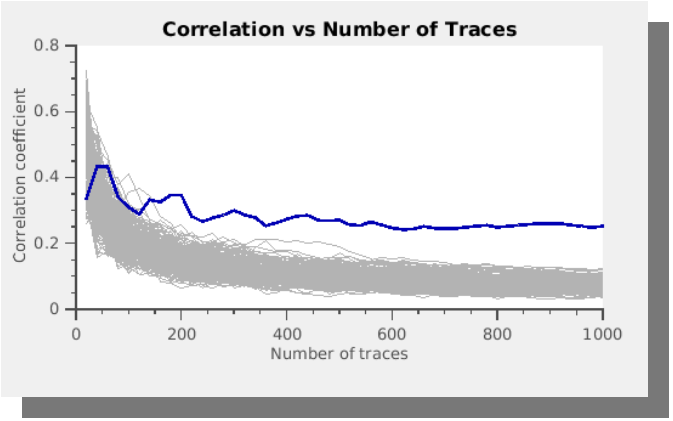
- Design a differential power analysis (DPA) resistant smartcard

Two main sub-projects:

1. Smart card emulator



2. Side-channel analysis tools



Agenda

- Smart Card Laboratory
  - Administrative topics
  - Why is security important?
  - Introduction to smart cards
  - Objectives of the laboratory
  - Work plan
  - Project management
- Group assignment (second lecture)

Section 1 – Organization and

ADMINISTRATIVE TOPICS

Contact

Lab Instructor

- **M.Sc Michael Gruber**

E-Mail

- m.gruber@tum.de

Consultation hours

- Schedule an appointment

Room N1007

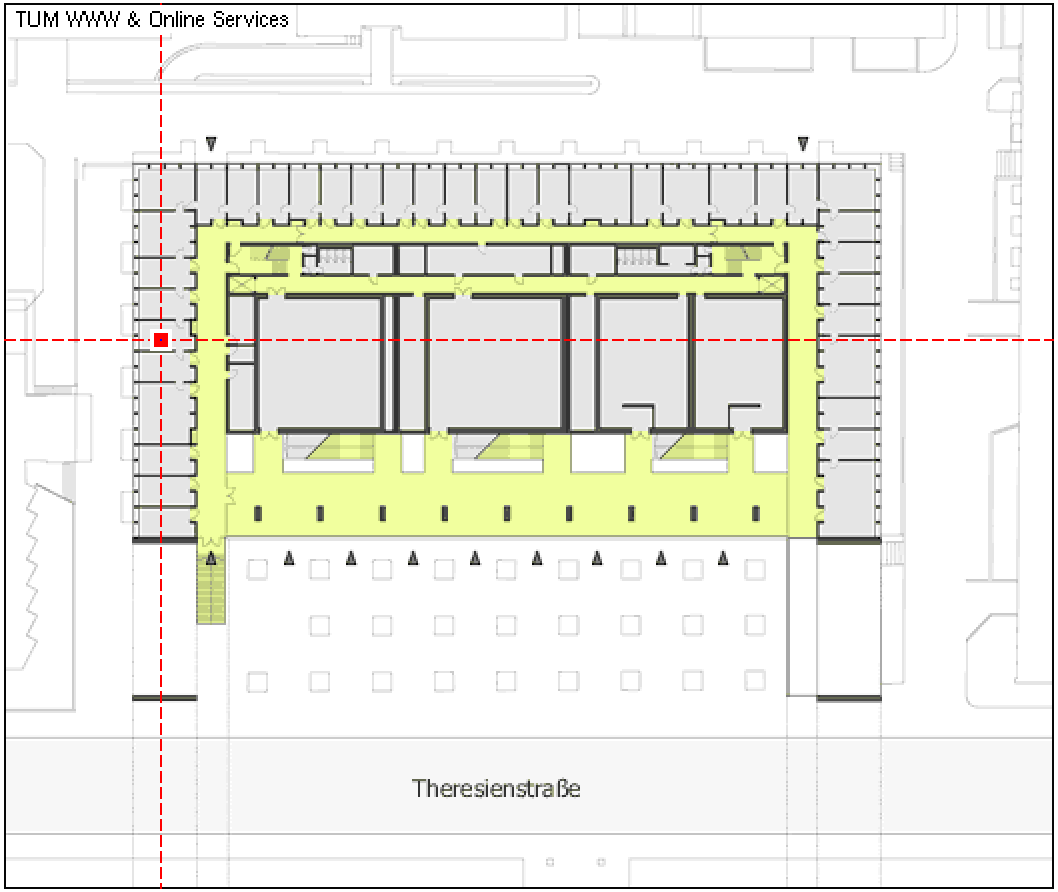


image source: http://www.clker.com/cliparts/d/L/P/X/z/i/no-image-icon-hi.png

Contact

Lab Tutors

- **Moritz Wettermann**

E-Mail

- moritz.wettermann@tum.de

Room N1003

Consultation hours

- TBA

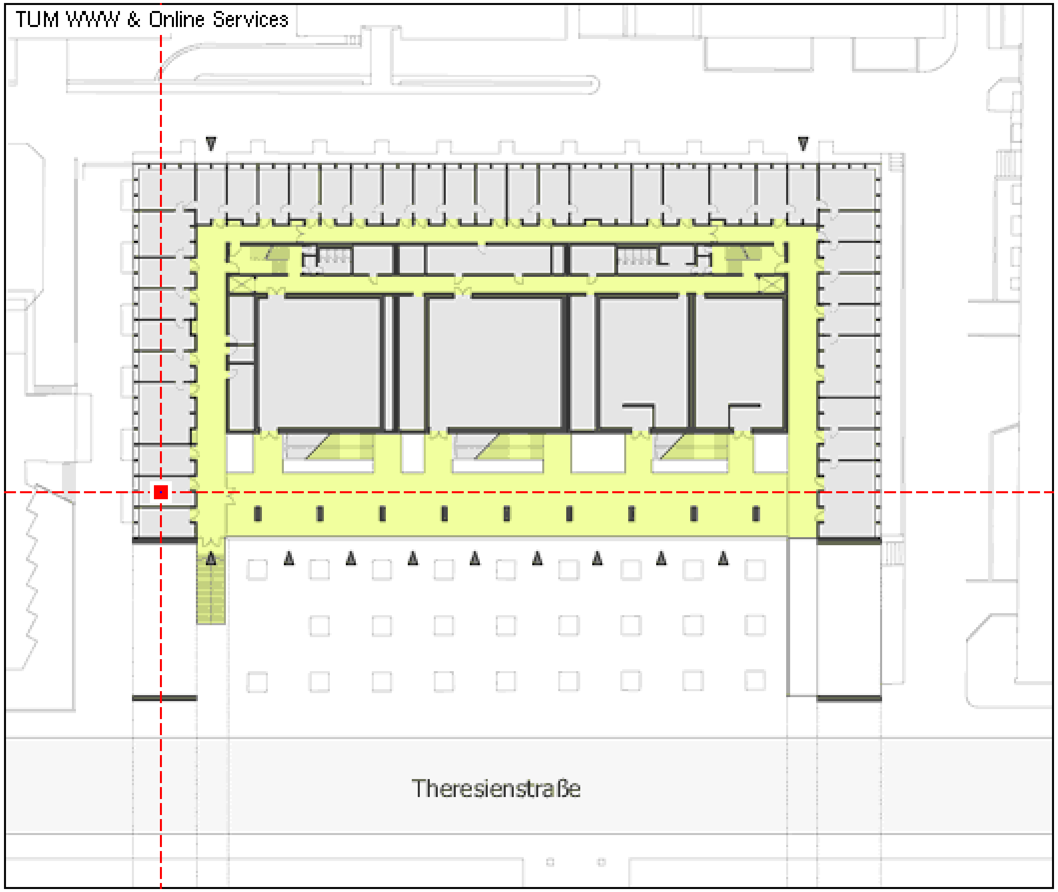


image source: http://www.clker.com/cliparts/d/L/P/X/z/i/no-image-icon-hi.png

Laboratory hours

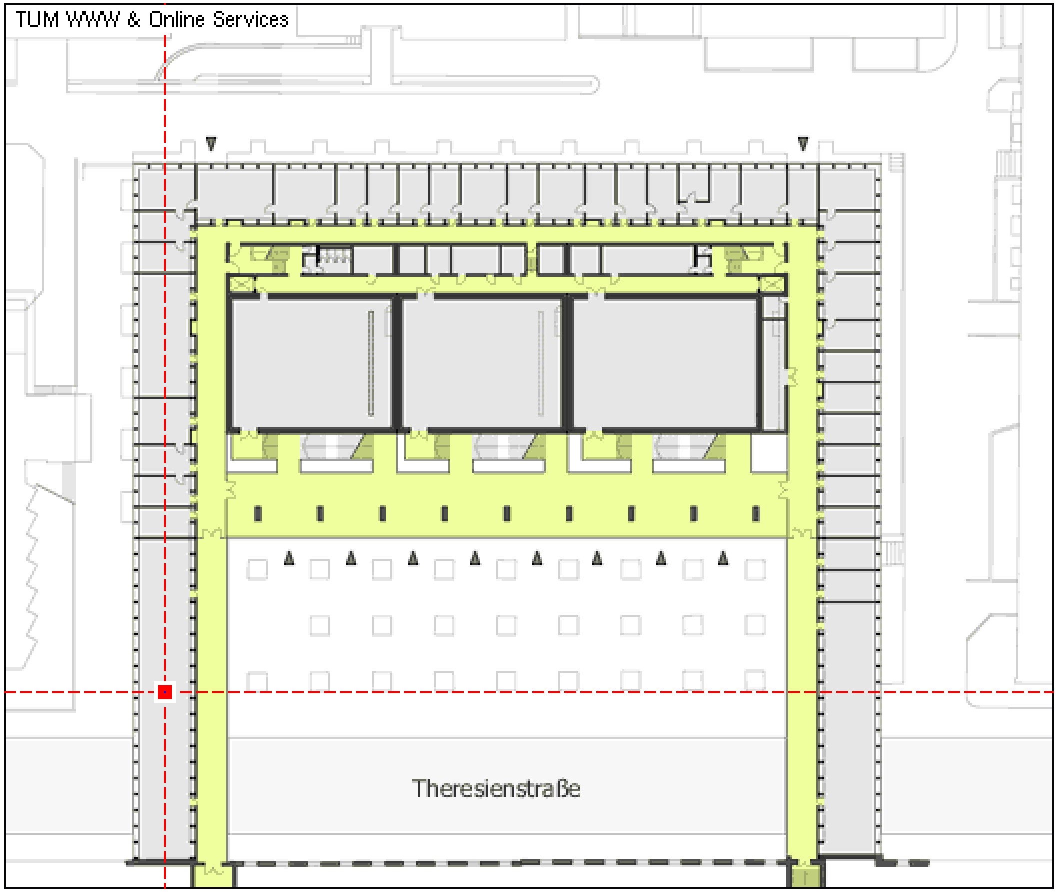
- The Smart card Laboratory can be performed on **your own schedule**.

Dates

- Monday – Friday

Time

- 7:00 – 21:00



Locations

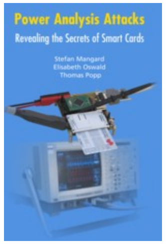
N1003 (programming), N1110D (measurements) both rooms shared with SIKV

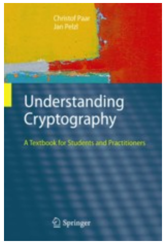
Pre-requisites

- What is **needed** for the course?
  - A good understanding of digital circuits
  - Experience coding in C
  - Experience with scripting languages (Matlab or Python)
  - Being open to work in a team

- What is **nice to have**?
  - Knowledge of computer (microcontroller) architectures
  - Having taken the course on Secure Implementation of Cryptographic Algorithms (a.k.a. “SIKV”)

Literature





Main Literature:

- ISO/IEC 7816 Standard
- Power Analysis Attacks: Revealing the Secrets of Smart Cards
  Stefan Mangard, Thomas Popp, Elisabeth Oswald, ISBN-13: 978-0387308579

Additional Literature:

- Smart Card Handbook
  Wolfgang Rankl und Wolfgang Effing, ISBN-13: 978-3-446-40402-1
- Understanding Cryptography
  Christof Paar and Jan Pelzl, ISBN-13: 978-3-642-04100-6

Laboratory Logistics

- Lectures
  - Introduction to the Smart Card Lab (this lecture)
  - Introduction to Side-Channel Analysis (SCA) attacks

- Project Milestones
  - Midterm Presentation
  - Final Presentation
  - Oral exam

Important Dates

Lectures:

Date	Time	Place	Description
25.04.2019	13:15 – 14:45	N1005	Introductory lecture
02.05.2019	13:15 – 14:45	N1005	SCA-Introduction + Demo

Presentations / Exam (remember to register!):

Date	Time	Place	Description
06.06.2019	12:30 – 15:30	N1005	**Midterm** Presentation
11.07.2019	09:00 – 12:00	N1005	**Final** Presentation
18.07.2019	09:00 – 13:30	N1011	Oral **exam**

Teamwork

- Work is to be carried out in **groups** of 4 people
- Each student must keep a lab protocol (template will be given)

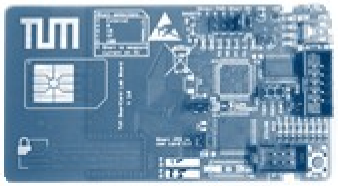
- Suggestions for effective teamwork:
  - Share the workload in a fair manner
  - Contribute with ideas and listen to the opinions of others
  - Work on your task and be open to help others
  - **Keep in touch** with your team members

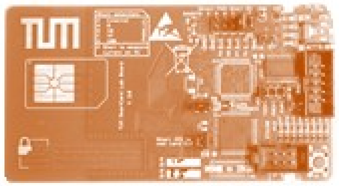
- Groups will be formed at the end of the lecture today
- You have **until the next lecture** to make any **changes** to your group

Tools

Smart cards

- Reference card
- Blank card (for your clone)





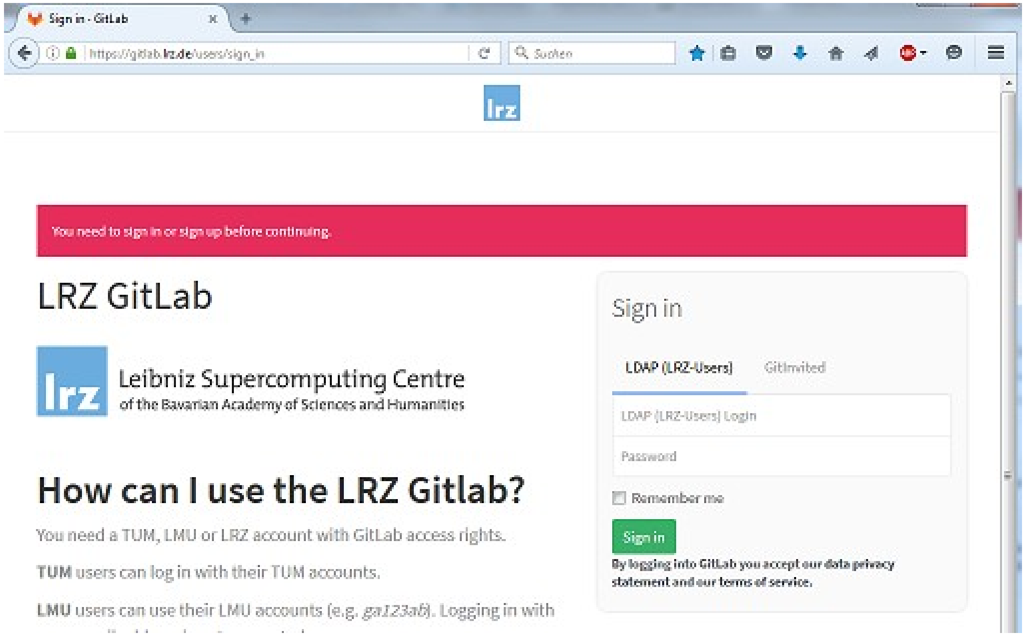
Tools

- Programmer (AVR MKII ISP)
- Programmer helper
- Logic analyzer (Saleae Logic)
- Oscilloscope (Picoscope 5204)

Project management in the Laboratory

GitLab (LRZ) - Git

- Version control
  - Tracking your changes
  - Synchronization between developers
- Project management
  - Wiki (for documentation)
  - Milestones

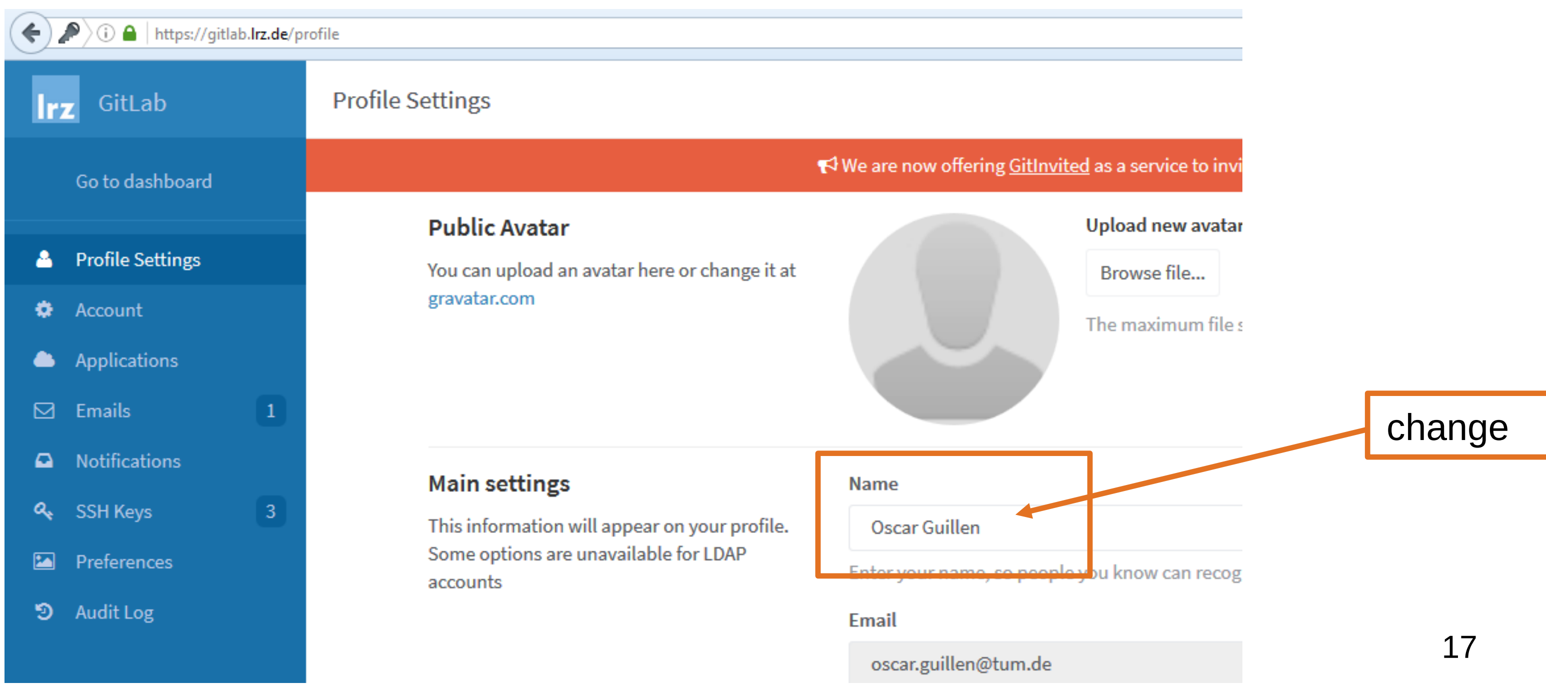


https://gitlab.lrz.de/

Project management in the Laboratory

Please:

- Log into GitLab (to activate your account)
- Change your “Name” to your real name
- Leave your “Username” **unchanged** (your LRZ account)



Grading

Lab work

- Work is to be carried out and documented in groups.
- Each student must keep a lab protocol.

Midterm / Final presentations (50%)

- 20-30 min. Presentation on the project
- Subsequent discussion

Oral examination (50%)

- 10 min. Oral examination.
- Theoretical knowledge on the topics learned

Academic Integrity

- All **work** you submit for this laboratory **must be your own**

- If you have to use of **code/designs/ideas** from **other people**
  - Must be **clearly** and **explicitly noted**
  - Must have a proper and **complete citation**

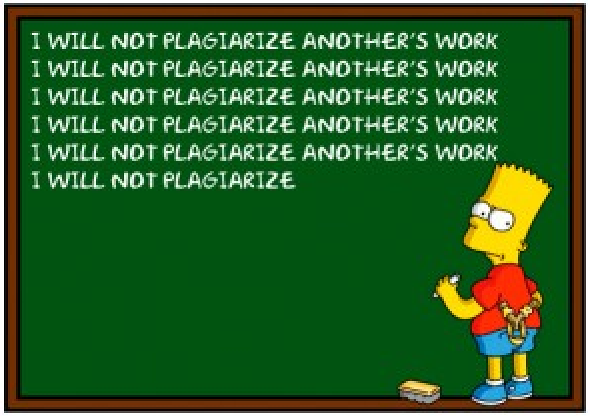


image source: http://exmss.org/files/2012/09/Bart.jpg

Section 2 – Motivation

WHY IS SECURITY IMPORTANT?

Use of Cryptography

- Historically: Military Communications
- since 1970: Industrial Data Transmission (DES)
- since 1980: Mobile Communications (GSM, Chipcards)
- since 1990: Everyday Life (WLAN, HTTPS, PGP)

military

commercial

Caesar Cipher (100 BC)	Vigenère Cipher (1467 AD)	Herbern Rotor Machine (1908)	Enigma Machine (1920)	Data Encryption Standard (1977)	**Advanced Encryption Standard** (2001)	CEASAR Auth. Encryption (ongoing)
• Rotation of the alphabet by 13 letters	• First cipher to use a key. • Key addition modulo 26	• Electro mechanical. Typewriter-like. • Single rotor. • Key embedded in the disc.	• Multiple rotors. • Developed after WWI • Widely used during WWII	• IBM's "Lucifer" cipher. • Standardized by the National Bureau of Standards	• NIST • 50 submissions • Rinjdael cipher	

Use of Cryptography - Trends

The use of cryptography in everyday devices is becoming more important

Examples:

- Smartcards
- Internet of Things
- Machine-to-Machine
- Car-to-X
- Advance Metering Infrastructure (Smart Grid)
- Medical Monitoring / Telemetry
- Cloud Computing
- Industry 4.0

Use of Cryptography – Everyday use

Chipcards as an example

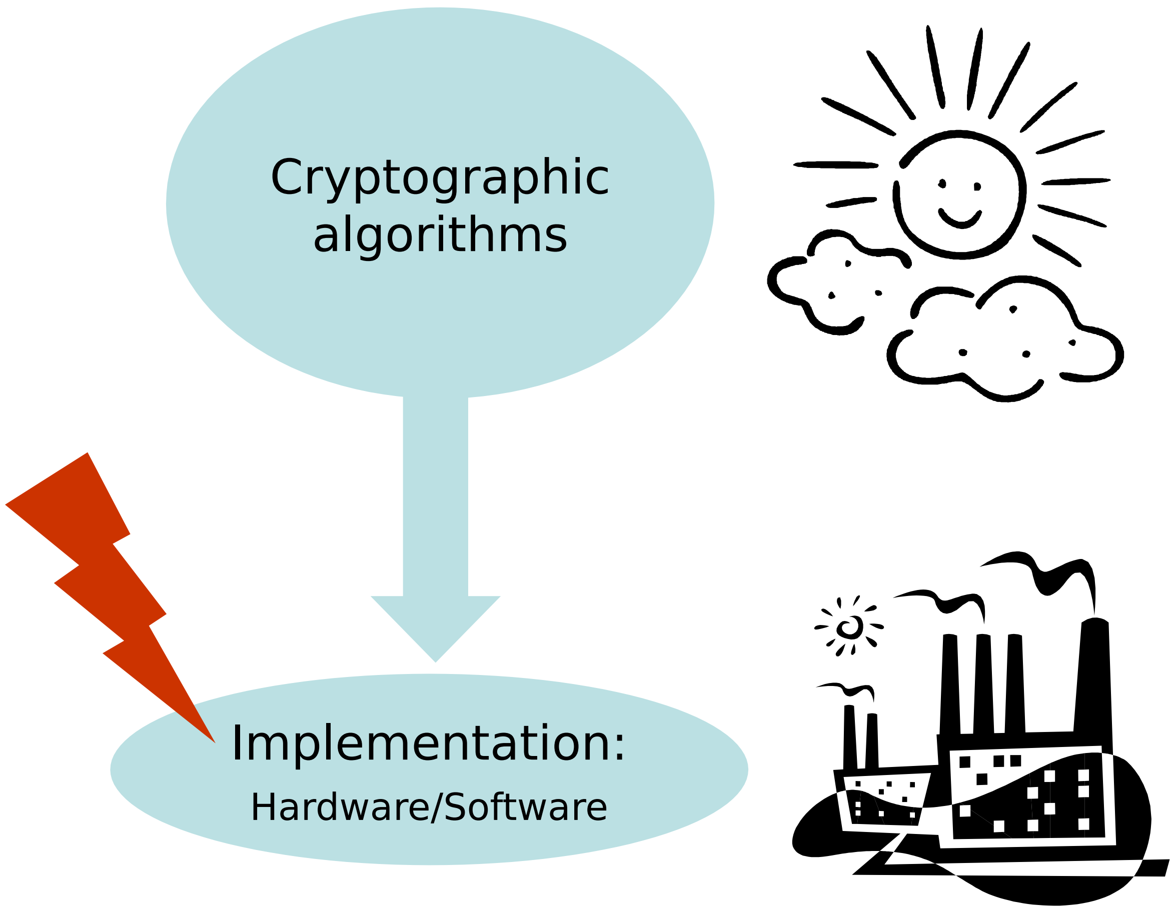
- Telecommunication
  - Telephone cards, SIM-Cards
- Payment
  - e-Purses, Credit cards
- Access control
  - Access ID, Public transportation cards
- Identification
  - Passport, Driving license, Medical cards
- Digital Rights Management
  - Pay TV

image source: http://siliconangle.com

Challenges

- Every designer will somehow be involved with the topic of security when designing a system (e.g. piracy prevention).
- Two main challenges:
  - The commercial benefit for an attacker can be really high (e.g. Pay TV, product piracy), this is also true for the amount of time and money that someone can invest in order to attack a system.
  - Devices that make use of cryptography are in the hands of many users (and attackers).

Cryptography in Engineering



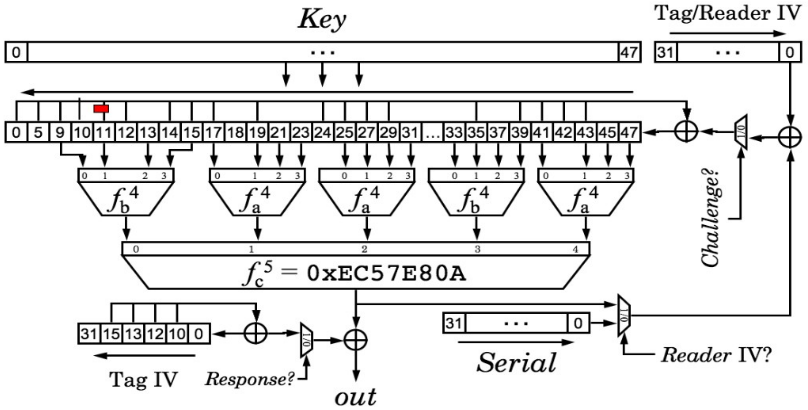
Attacking the weakest link

Hardware Security: Mifare Classic

Access control and ticketing systems (e.g. Oyster Card in London)
Contactless memory card, crypto in Hardware (LFSR-based)

Their security was broken in 2007 by researchers at the Humboldt-Universität Berlin and Radboud Universiteit Nijmegen by making use of:

- Gate-level Reverse Engineering
- Protocol Analysis
- Emulators



Weaknesses:

- Proprietary Cryptography (Crypto-1)
- Weak pseudo-random-numbers generator (PRNG)

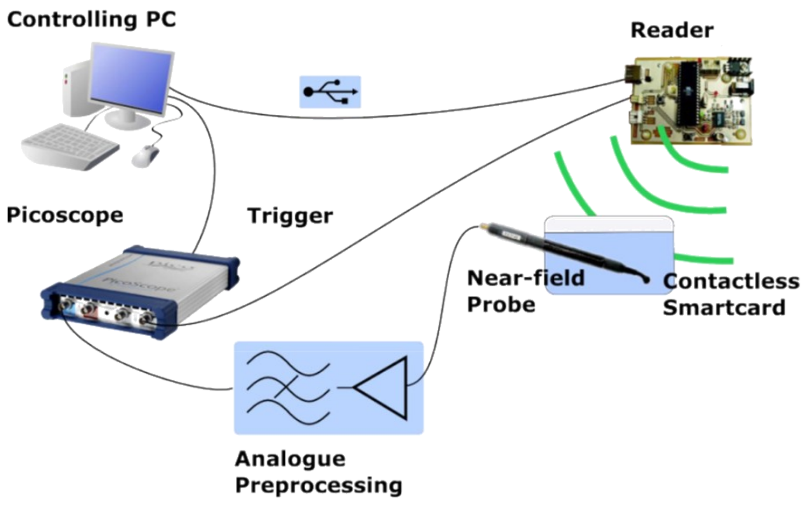
Details: http://en.wikipedia.org/wiki/MIFARE

Hardware Security: Mifare DES Fire

Access control and ticketing systems (Prague, San Francisco, London,…)
Contactless memory card, (strong) crypto (3DES)

Their security was broken in 2011 by researchers at Ruhr-University Bochum:

- Home-brewed RFID reader
- Low-cost USB oscilloscope
- Near field probes



Weaknesses:

- EM Emanation (Side Channel Analysis)

Details: https://www.iacr.org/workshops/ches/ches2011/presentations/Session%205/CHES2011_Session5_1.pdf

Hardware Security: Kee-Loq

“Remote Keyless Entry” Systems e.g. Car keys, Garage door openers

Algorithm implemented in Hardware (NLFSR)

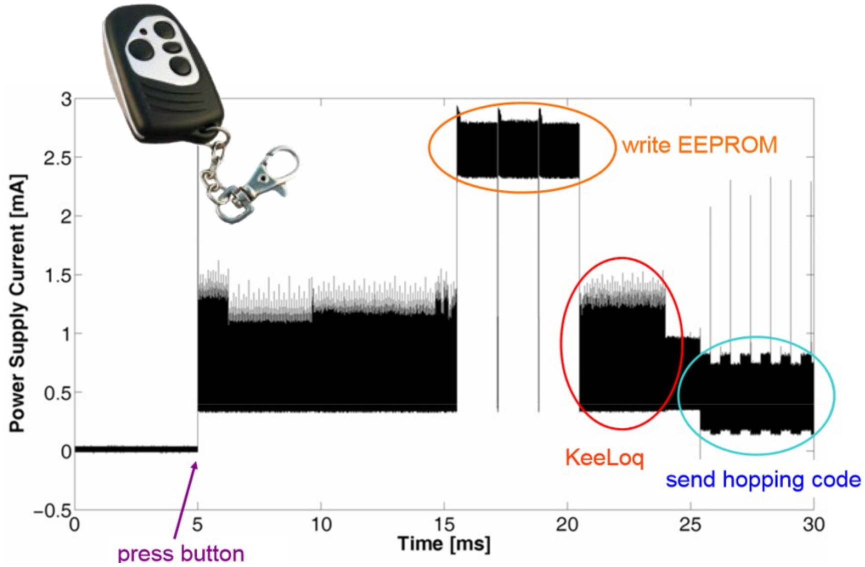
Their security was broken by researchers at the Ruhr-Universität Bochum by making use of:

- Mathematical cryptanalysis
- Side-channel attacks (DPA, SPA)

Weaknesses:

- Proprietary cryptography
  (in 2006 their algorithm was leaked in Internet)
- Susceptibility to side-channel attacks

Details: http://www.crypto.rub.de/keeloq/



Hardware Security: USIM-Cards

3G UMTS / 4G LTE Cards Cloning

MILEANAGE algorithm (AES-based)

Mutual authentication protocol designed to remediate problems found in GSM (base station spoofing)

Weaknesses:

- Cheap USIM cards do not provide resistance to Side Channel Attacks

Details: http://perso.uclouvain.be/fstandae/PUBLIS/161.pdf
https://youtu.be/x8exHMhGy1Q (Black Hat 2015)

Hardware Security: Locking System

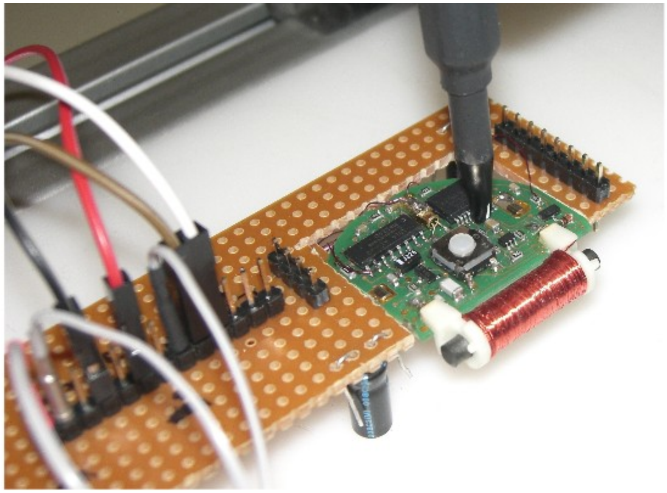
Access control

Strong Cryptography (3DES)

The security was broken by a collaboration between researchers/students from TUM, LMU, TU Darmstadt and TU Kaiserslautern

Weaknesses:

- General purpose MCU
- Weaknesses in RNG
- Susceptible to Side Channel Analysis
- Susceptible to Fault Injection attacks



Hardware Security: USIM-Cards

3G UMTS / 4G LTE Cards Cloning

MILEANAGE algorithm (AES-based)

Mutual authentication protocol designed to remediate problems found in GSM (base station spoofing)

Weaknesses:

- Cheap USIM cards do not provide resistance to Side Channel Attacks

Details: http://perso.uclouvain.be/fstandae/PUBLIS/161.pdf
https://youtu.be/x8exHMhGy1Q (Black Hat 2015)

Take away notes on security

Robust security is becoming more critical in everyday-use products

- Stark rise in use of embedded systems
- High number of people with access to these systems
- Valuable information stored within or transmitted by them

Cryptography has evolved in the past twenty years from being a secret science practiced only by a small group of mathematicians to a fundamental discipline for engineers

Designing secure systems and **securely implementing** cryptographic algorithms are skills which engineers require more than ever

Side-note: Security is not the same as Safety
(…although in German the same word is used for both: *Sicherheit*):

- Safety: The system must not represent a hazard (to people)
- Security: The system must be resistant to attacks (to the system)

Section 3 – A brief introduction to

SMART CARDS

Introduction to Smart cards

What is a Smart card?

- Embedded computer (Microcontroller)
- Limited resources
- Embedded in a plastic card
- Low cost
- Tamper-resistant

Typical uses of a Smart card

- Secure data storage
- Secure data processing
- Authentication

image source: http://crystalequityresearch.blogspot.com

Smart card hardware components

Non-Volatile Memory
- EEPROM
- ROM

Volatile Memory
- SRAM

Crypto Functions
- Symmetric (3DES, AES,…)
- Asymmetric (e.g. RSA)

Analog components
- Voltage regulators
- Anti-tamper sensors

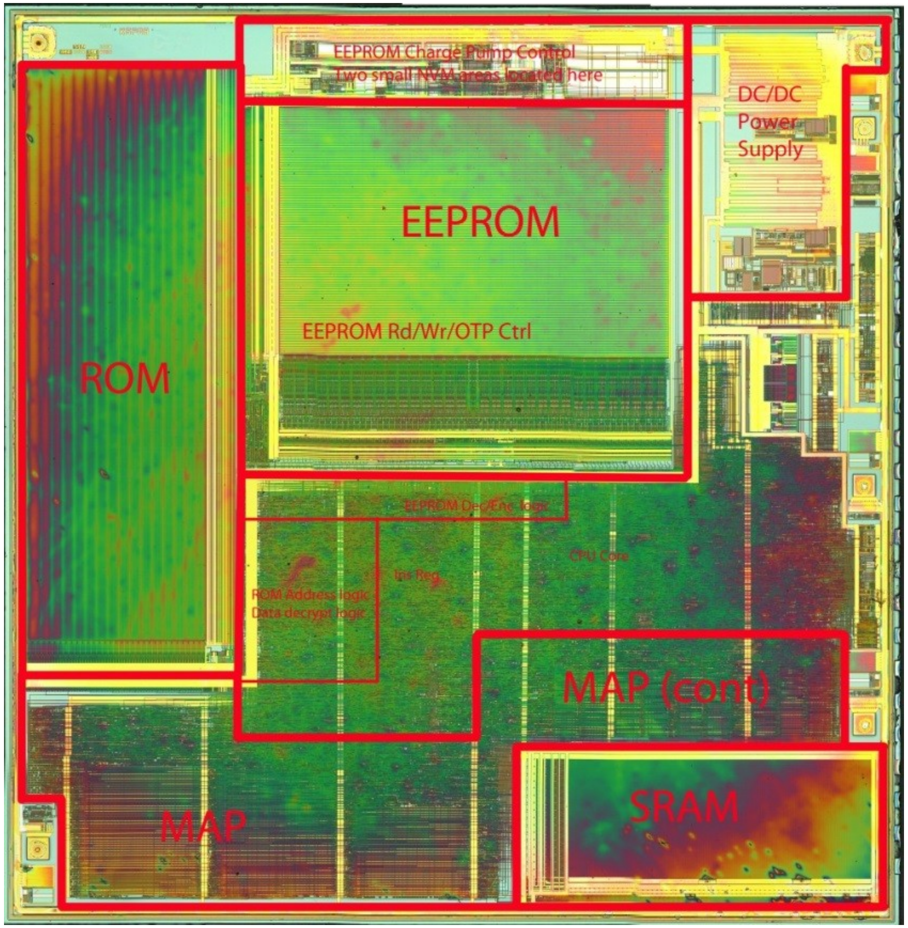


image source: http://www.flylogic.net

Attacks on Smart cards

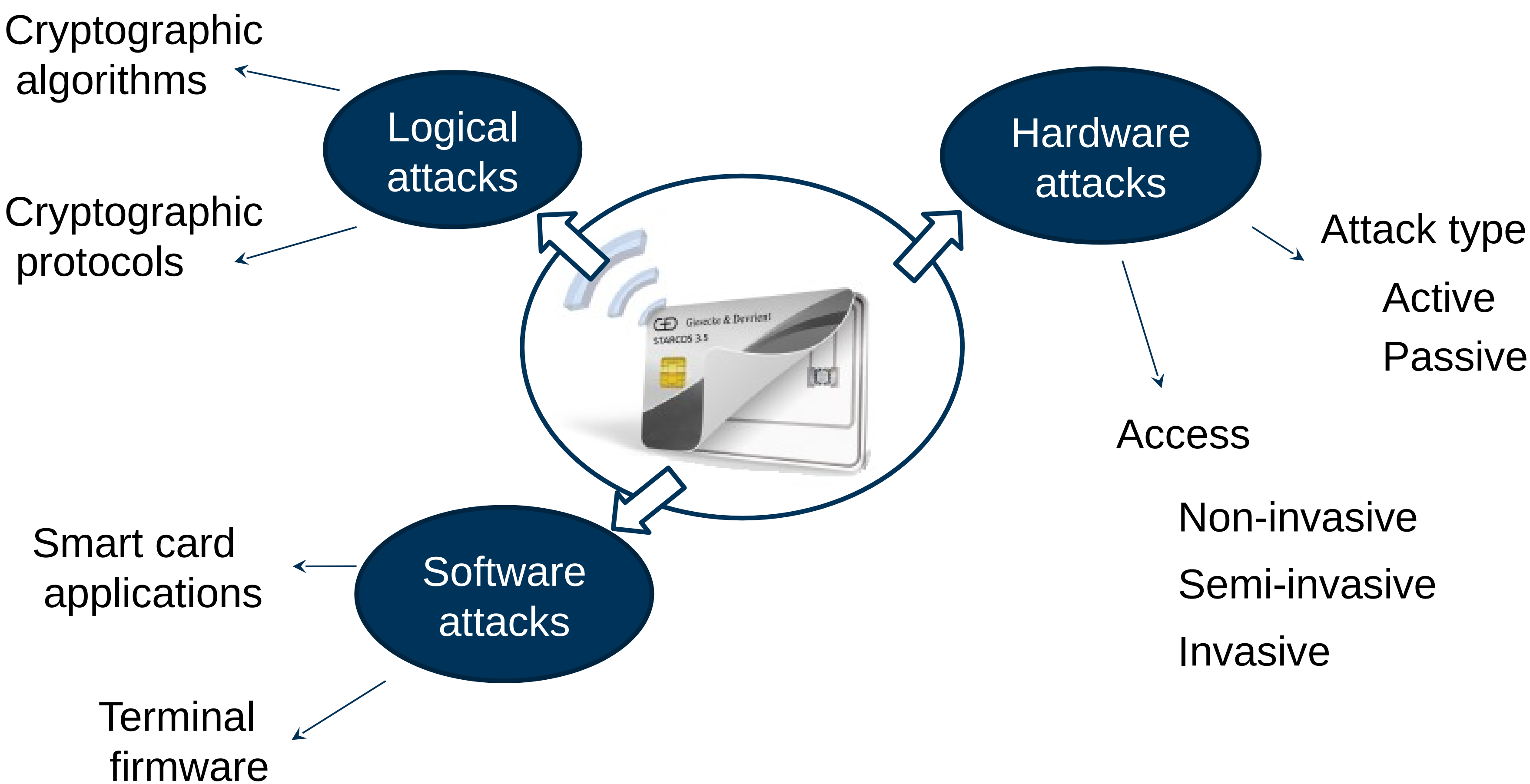


image source: https://www.gi-de.com/gd_media/media/press/images_1/pis_2012/Starcos35.png

Attacks on Smart cards

Logical attacks
- Cryptographic algorithms
- Cryptographic protocols

Hardware attacks
- Attack type
  - Active
  - Passive
- Access
  - Non-invasive
  - Semi-invasive
  - Invasive

Software attacks
- Smart card applications
- Terminal firmware

image source: https://www.gi-de.com/gd_media/media/press/images_1/pis_2012/Starcos35.png

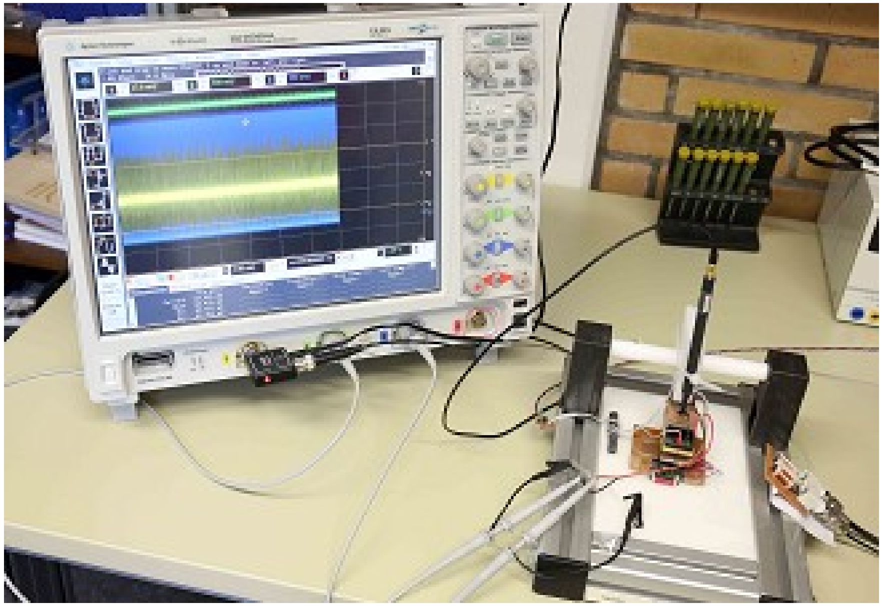
Side-channel Attacks

Advantages for an attacker

- Non-invasive
- Passive
- Relatively low-cost
- Powerful and relatively fast

Possible Side-channels

- Timing
- Power consumption
- EM emission
- others…



Section 4 – Smart card Lab

LABORATORY OBJECTIVES

Laboratory Description

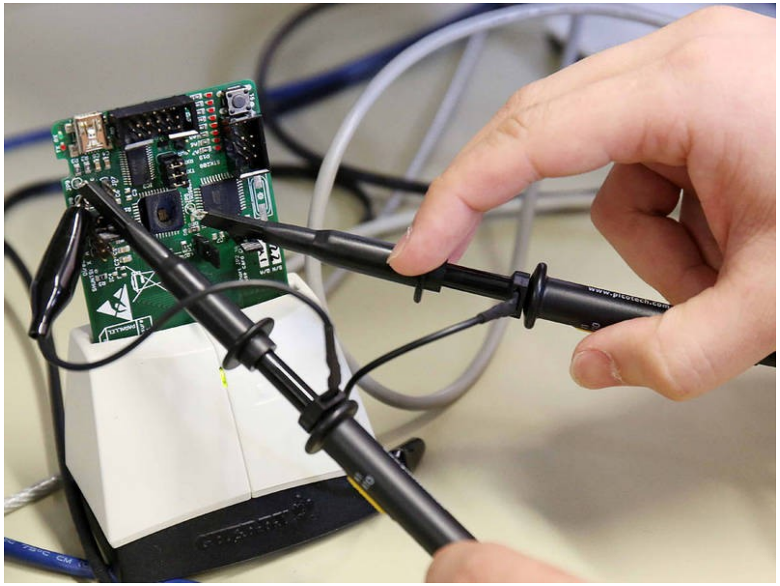
Pay TV system

- A smart card is used to decrypt a video data stream
- The right cryptographic key needs to be present in the card for that to occur
- Students take the role of the attacker to compromise the system
- …and also the role of developer to provide a secure solution against attackers

Laboratory's Objectives

Phase 1: Attack and **clone** a Smart card

- Analyze an existing Smart card (emulator)
- Extract the crypto key
  - Differential Power Analysis
- Create your own Smart card OS
- Make use of the extracted key in your own card

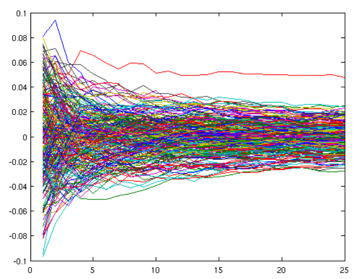


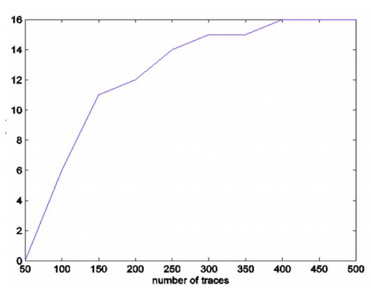
Phase 2: **Protect** your Smart card

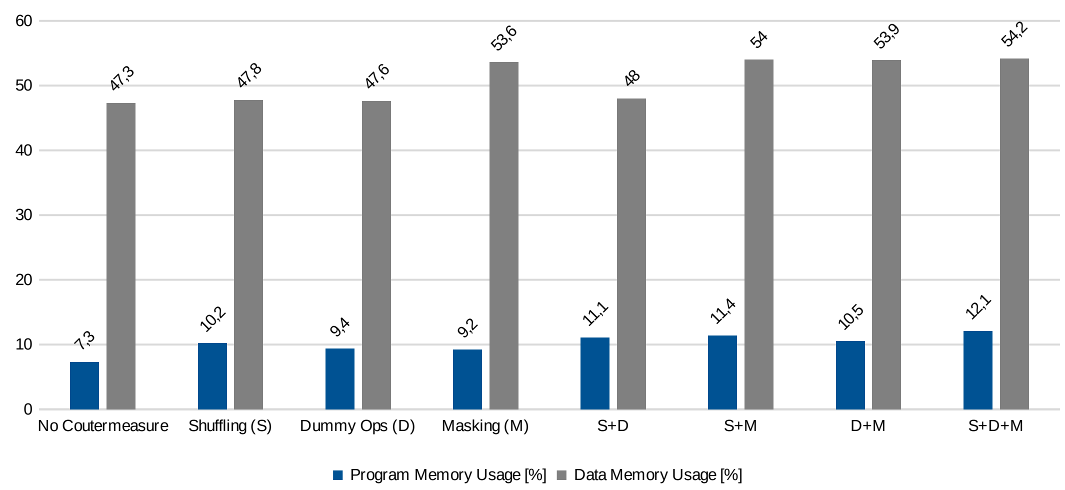
- Implement countermeasures against DPA
- Attempt to break your own countermeasures
- Evaluate the resistance of the different countermeasures

Learning Objectives

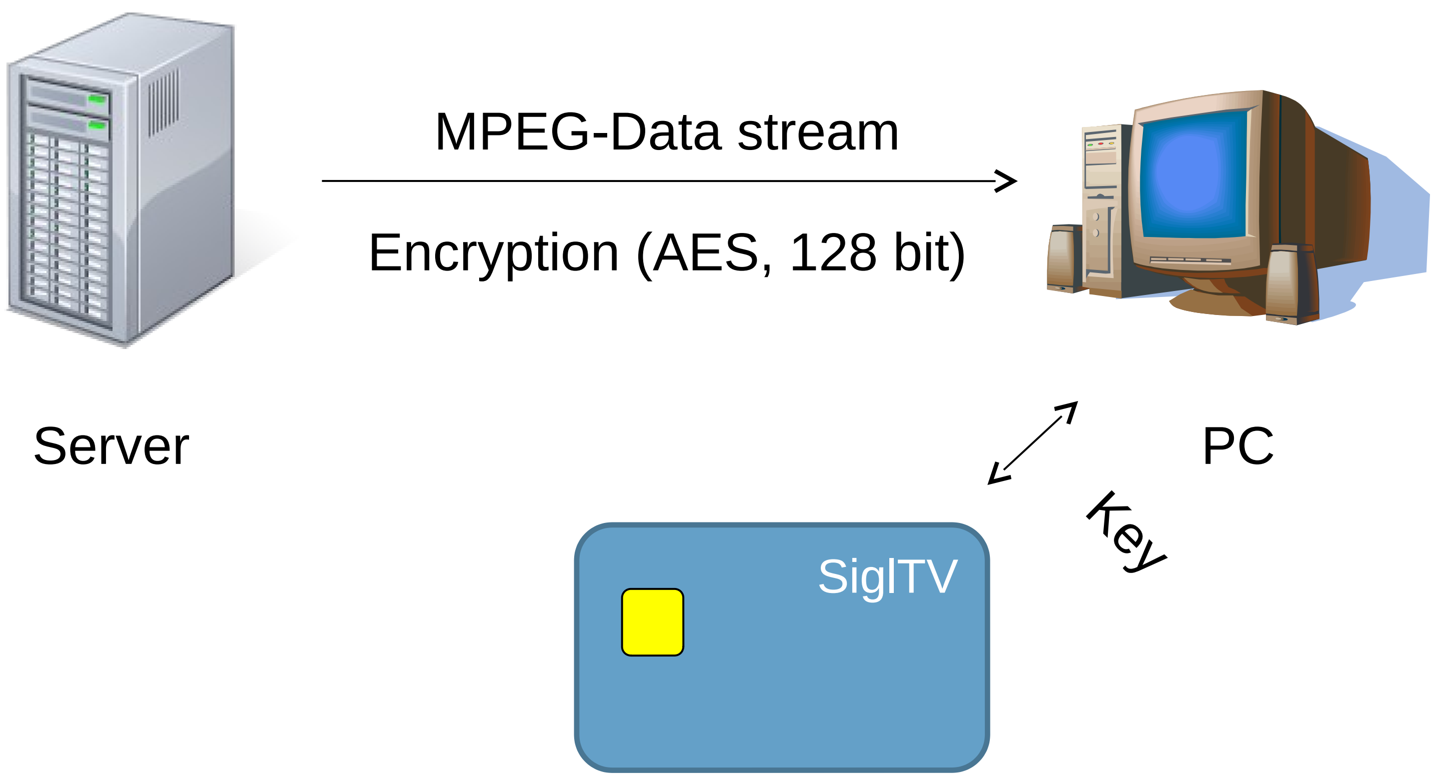
The main objective of the laboratory is to analyze the tradeoffs between different secure implementations and their cost







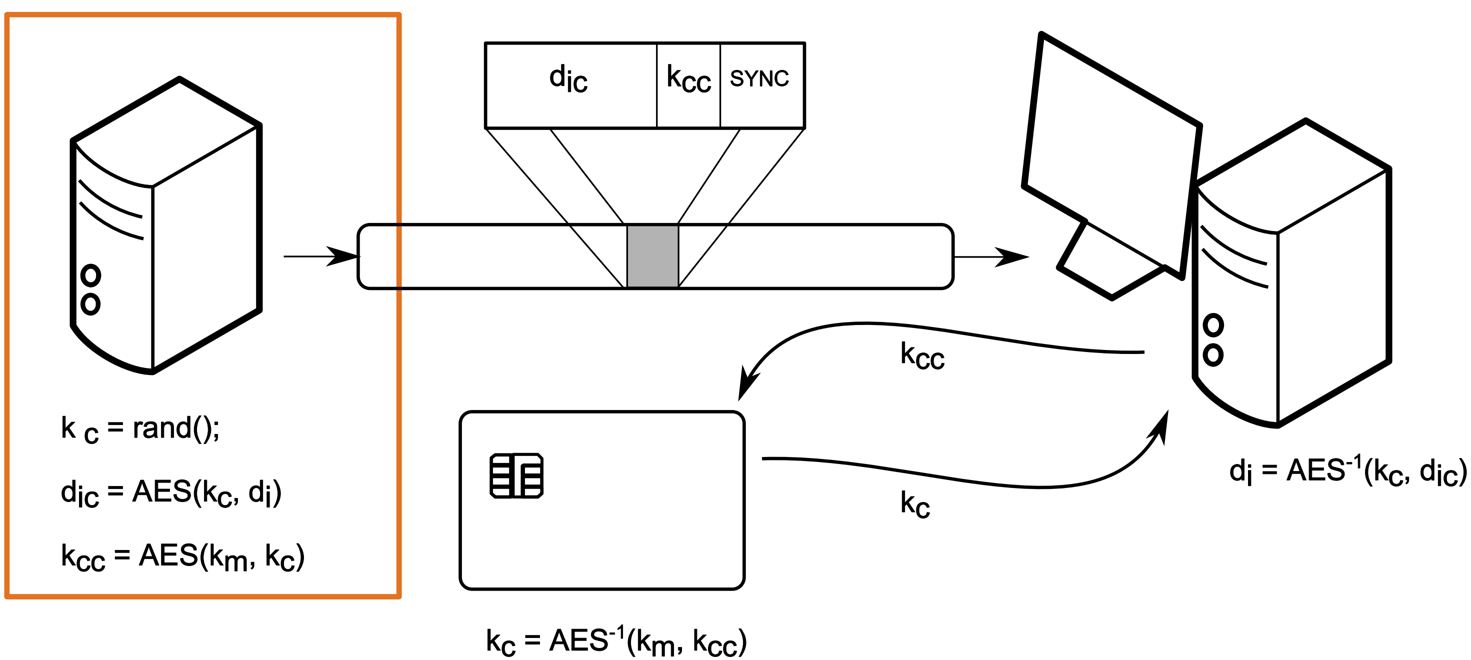
Structure of the PayTV-System



Structure of the PayTV-System

Server:

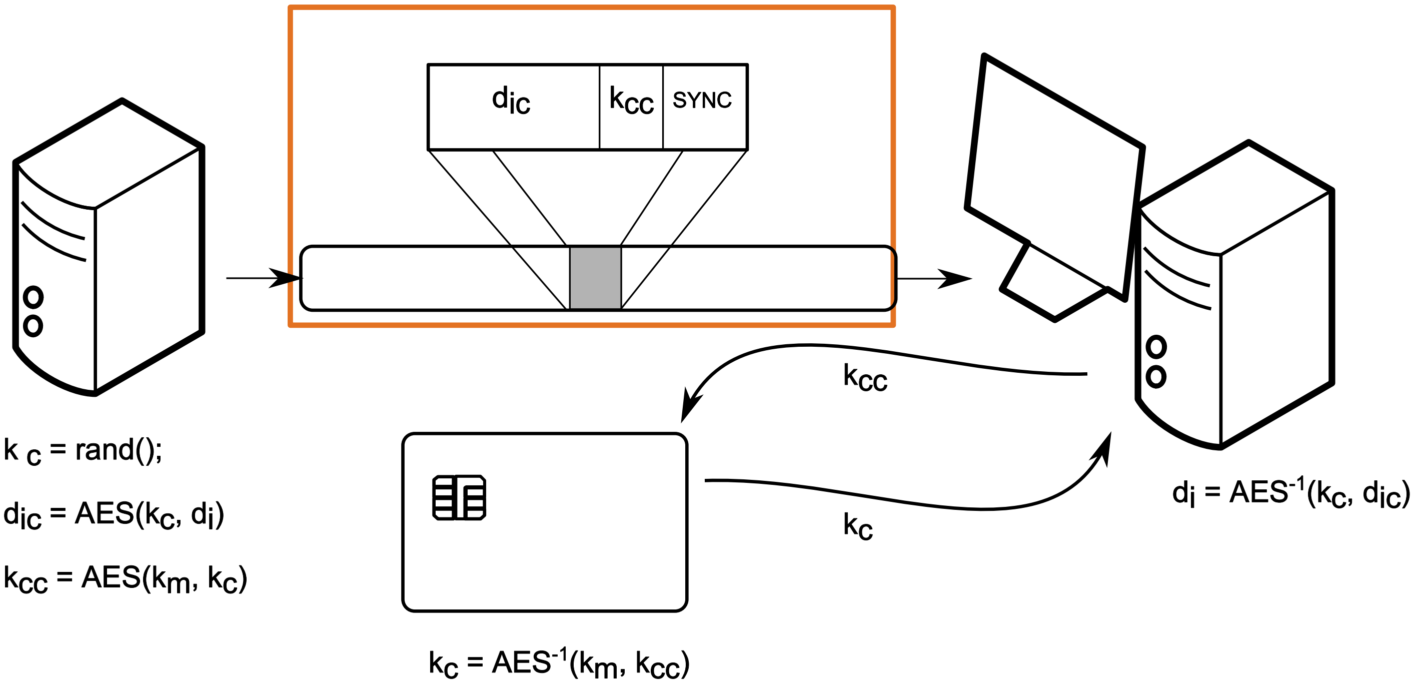
- Video data stream is divided into chunks d_i
- A random key k_c encrypts each data chunk
- The random key k_c is then encrypted with a master key k_m to generate k_{cc}
  - k_m is known only by the server and the card



Structure of the PayTV-System

Server:

- Sends packets that include
  - An encrypted data chunk, d_{ic}
  - The encrypted random key, k_{cc}
  - Synchronization data



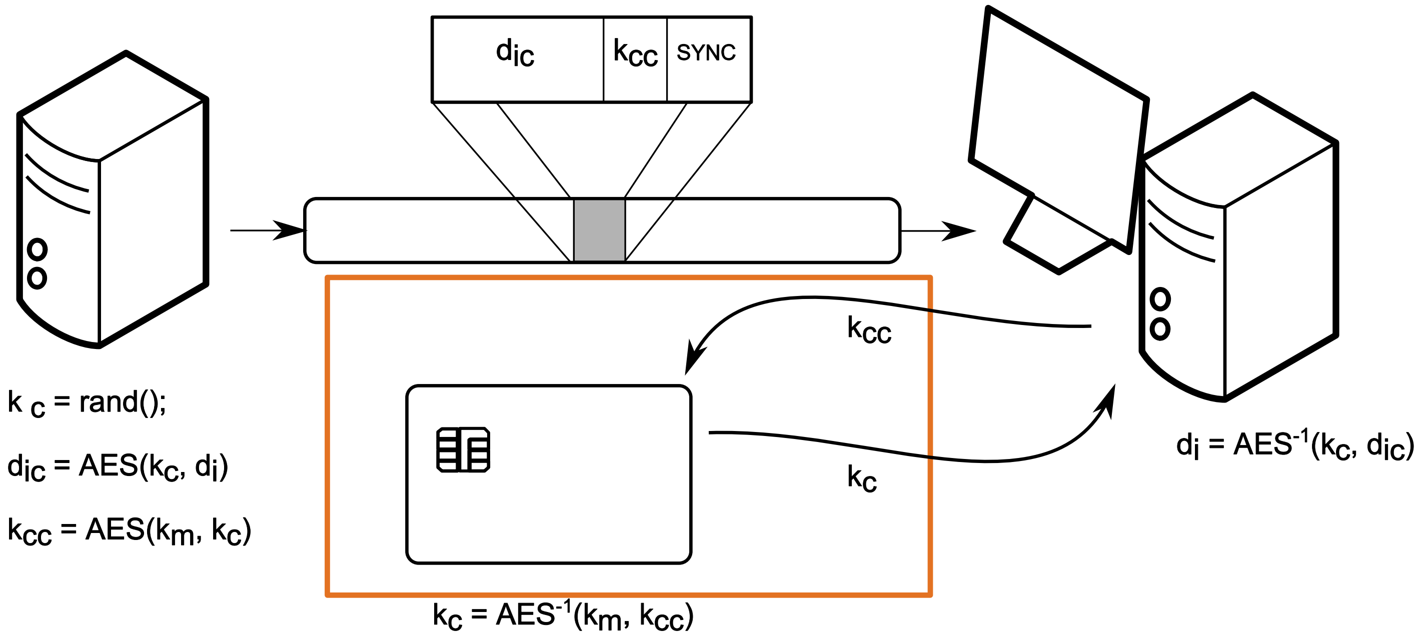
Structure of the PayTV-System

PC:

- Sends the encrypted random key, k_{cc} , to the smart card

Smart card:

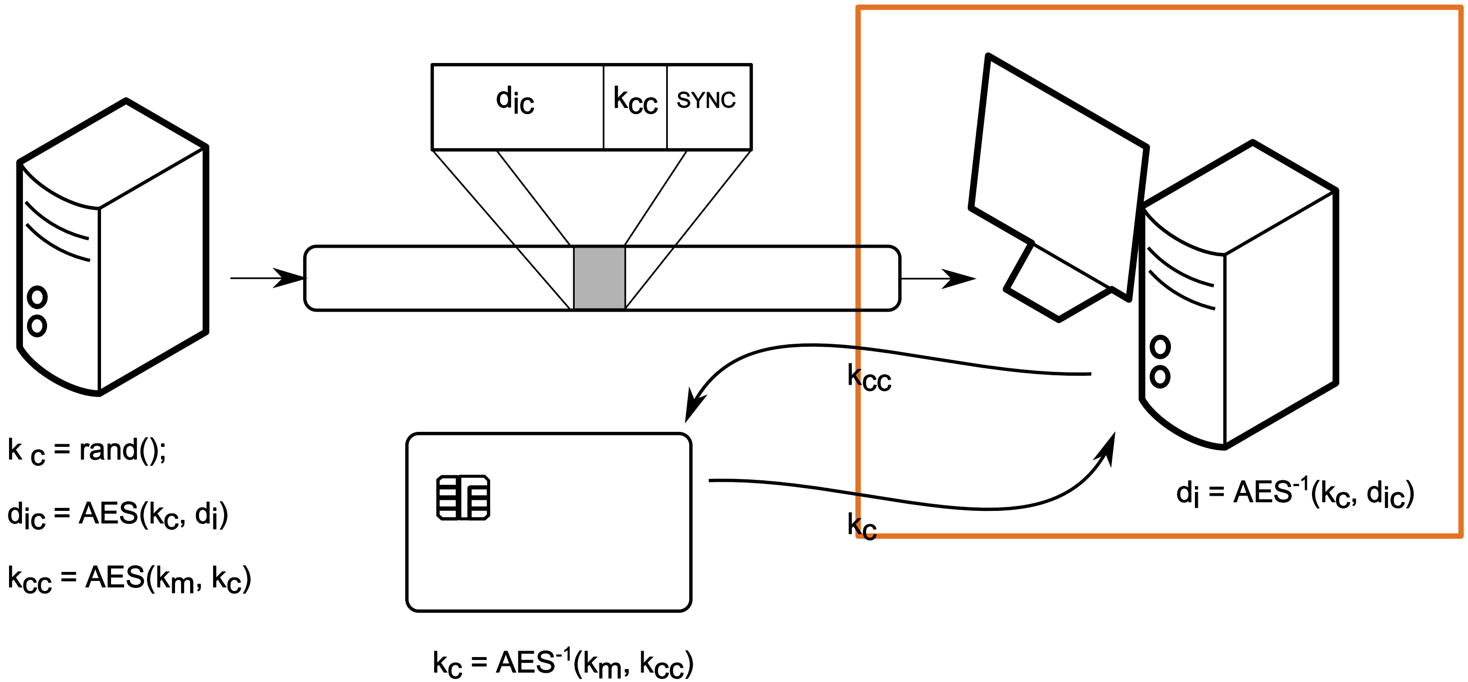
- Decrypts k_{cc} with k_m to obtain k_c



Structure of the PayTV-System

PC side:

- Uses k_c to decrypt d_{ic}
- Displays the plaintext data chunk d_i



Structure of the PayTV-System

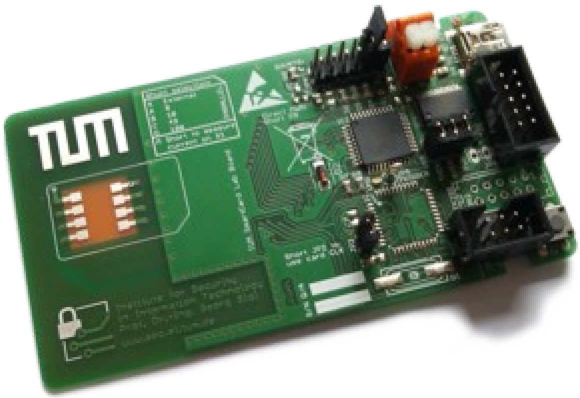
Reference implementation

- Smart card emulator using an ATMega644
- Preloaded master key

PC-Software

- Some lines of Python code
- LAN video client

Server

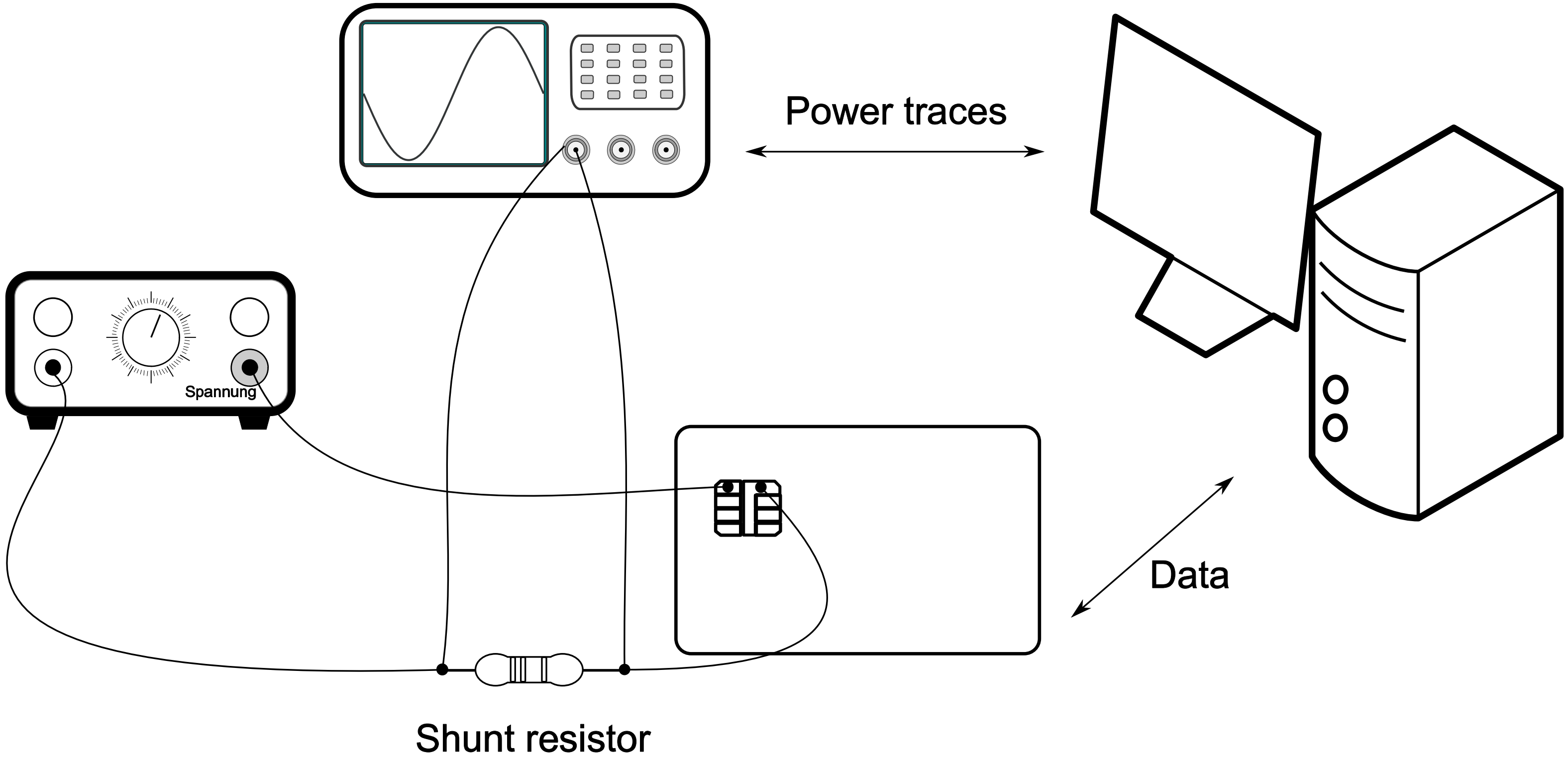


Side-channel Analysis

Differential Power Analysis

- The key will be extracted using the current profile
- Measurement of the current profile:
  - Send data to the smart card crypto function
  - Measure the current of the operation with a shunt-resistor
  - Record the power traces with an oscilloscope
  - Analyze and process the traces offline (Matlab / Python)

Implementing the DPA



Inferring the key with DPA

Steps

- Online:
  1. Generate or eavesdrop upon data sent to the device
  2. Measure power consumption of a device
- Offline:
  1. Assume a power model
  2. Create a Key hypothesis
  3. Check the correlation: Hypothesis equals Measured power profile
  4. Find out the key with the highest probability

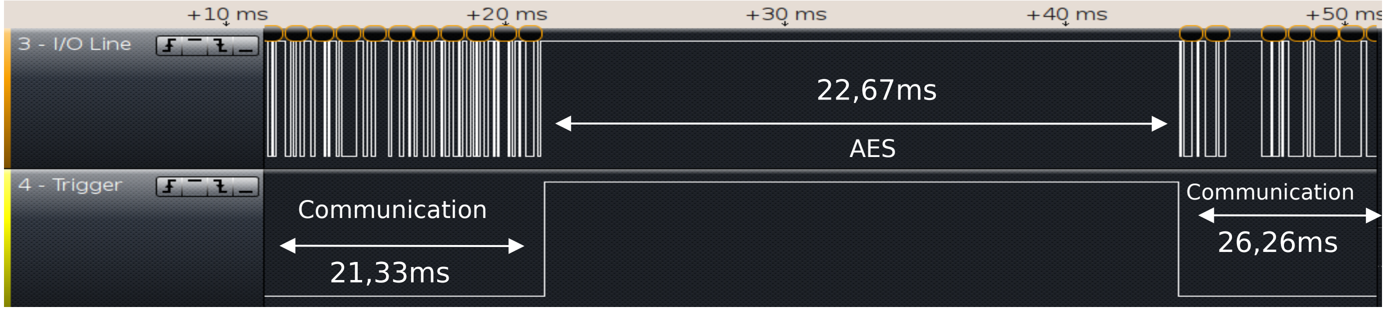
Details will be given on the second lecture

- Introduction to Differential Power Analysis

Smart card Clone

Program the ATmega644 (ATmega64) of your emulator card

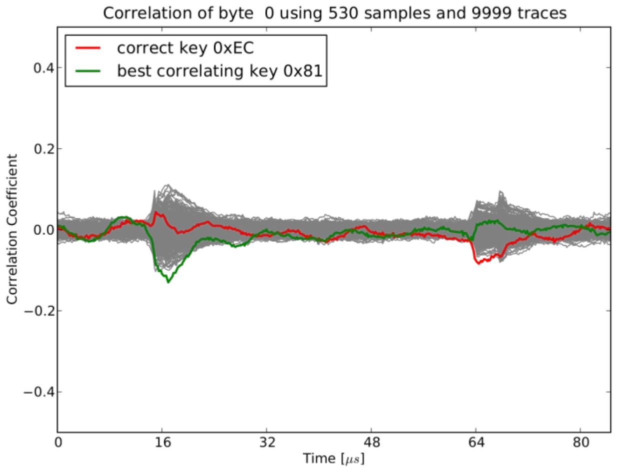
1. Create a basic Smart card OS following the ISO7816 standard
2. Implement AES-128 in software
3. Use the previously extracted key as master key
4. Test the functionality of the smart card clone



Improving the Security

Starting point: Your own implementation

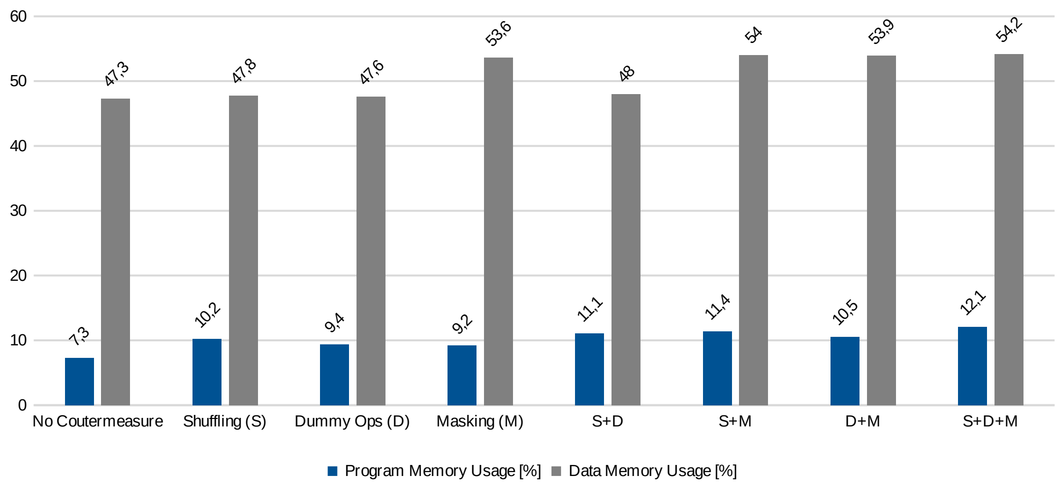
Harden your code to protect it against DPA
- Leakage hiding techniques
  - Random wait states
  - Shuffling operations
- Masking intermediate values



Evaluation of the Countermeasures

Attempt to break your own countermeasures

1. Measure the resistance against your attack techniques
2. Adapt your attacks
3. Document the results
4. Improve your countermeasures
5. Rinse and repeat!



Section 5 – smart card Lab

WORK PLAN

Work plan

Pre-Lab Assignment (on your own)

- Matlab / Python
- AVR
- Project administration and milestones planning

<u>(Basic) Checklist for Matlab/Python:</u>

- ❑ Working with HDF5 files
- ❑ Vectors manipulation
- ❑ Efficient use of memory
- ❑ Plotting

<u>(Basic) Checklist for AVR:</u>

- ❑ Ports
- ❑ Interrupts
  - ❑ Timers
  - ❑ Pin change
- ❑ UART (testing)
- ❑ Downloading the program

Work plan

Phase 1

Team A
- Perform DPA
- Extract key
- Document

Team B
- Implement card on MCU
- Test communication
- Document

Integration
- Use the extracted key to create your clone card
- Test that the clone behaves like the original

Milestone
- Present your results
- Get feedback for the next phase

Work plan

Phase 2

Team B
- Perform DPA on clone card
- Measure resistance
- Document

Team A
- Implement countermeasures
- Benchmark tradeoffs
- Document

Assessment
Final Presentation / Oral Examination

Milestone presentation

What is expected for the **milestone presentation**?

- Differential Power Analysis
  - Details of the implementation / optimizations
  - How many traces did it take to break the key?
  - How fast can you obtain all the key bytes?

- Smart card clone
  - ISO UART (implementation, sampling strategy)
  - Size of the complete smart card OS
  - Size of your AES implementation
  - Speed of your AES implementation (compare against reference card)

- Project management (who is doing what and when?)
  - Plan
  - Reality

Final presentation

What is expected for the **final presentation**?

- Countermeasures
  - Which countermeasures where tested?
  - What is the impact in size / speed?
  - What is the resistance provided by each countermeasure?

- Attack improvements
  - Which type of improvements were made?
  - How are you attacking specific countermeasures?
  - How do the new techniques compare against the simple DPA?

- Project management
  - Plan
  - Reality

Section 6 – smart card Lab

PROJECT MANAGEMENT

Which qualities must a project have?

Definition taken from DIN 69901:

“Project is an undertaking characterized essentially through the uniqueness of the conditions, for instance,
goal,
time, money, personnel, and other ***restrictions***,
the ***scope*** compared with other undertakings,
and project specific ***organization***”

Project Phases

Definition	Planning	Realization	Conclusion
Collect Ideas	Define the Architecture	Execute the work plan	Final Report
Define Goals	Create a Work Plan		Presentation

Phase1: Definition

Requirements Specification
- Requirements of the client
- Common Problem: The client often does not know himself exactly what he wants

Feature Specification
- First draft of the plan
- Describes how the contractor will implement the requirements

Project Goals Definition
- Goals must be “SMART”

  (**S**pecific, **M**easurable, **A**ccepted, **R**ealistic, **T**imely)

Phase 2: Planning

Work-breakdown: Structure of subtasks, Tree structure
Process list:

[ID-Nr., Process description, Duration, Predecessors, Resources]

Gantt Charts
Milestones: Important events of the project

System architecture: Relationship between the different components
Interface definitions: Function calls between components

The first plan will still differ from reality,
nevertheless, planning in the first stages is very important!

Phase 3: Realization

Perform the planned operations

Documentation

- As much as needed, as little as possible
- Architecture and steps

Project-Monitoring (Cycle)

- Test
- Check if what it is, is what it is supposed to be
- Correct discrepancies
- Adapt the plan

Definition | Planning | **Realization** | Conclusion

Phase 4: Conclusion

Final Report

Final Presentation, Demonstration

Client's Approval

Evaluation - „Lessons Learned"
- What has been achieved?
- What problems where there?
- What could be improved in the future?

Definition → Planning → Realization → **Conclusion**

QUESTIONS SO FAR?

GROUPS ASSIGNMENT

Questions?

THANK YOU FOR YOUR ATTENTION!

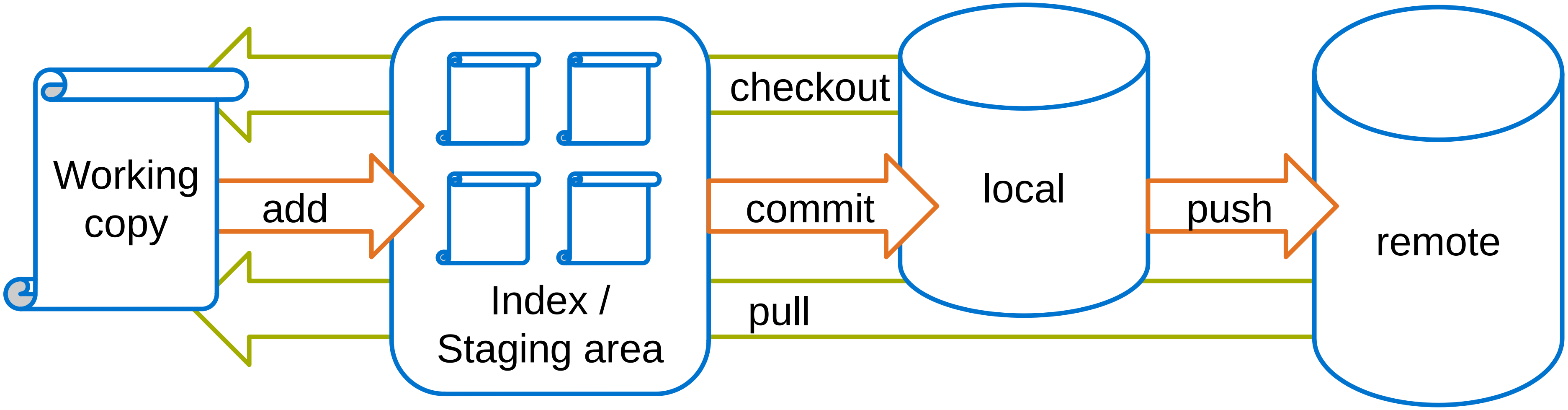
BACKUP SLIDES

Backup Slides

MANAGEMENT TOOLS

Crash-course Git

Simplified data-flow



based on: https://en.wikipedia.org/wiki/Git_%28software%29

Crash-course Git

Important commands

Create a working copy: `git clone username@host:/path/to/repository`

Update your local copy: `git pull`

Print the status of working copy files and directories: `git status`

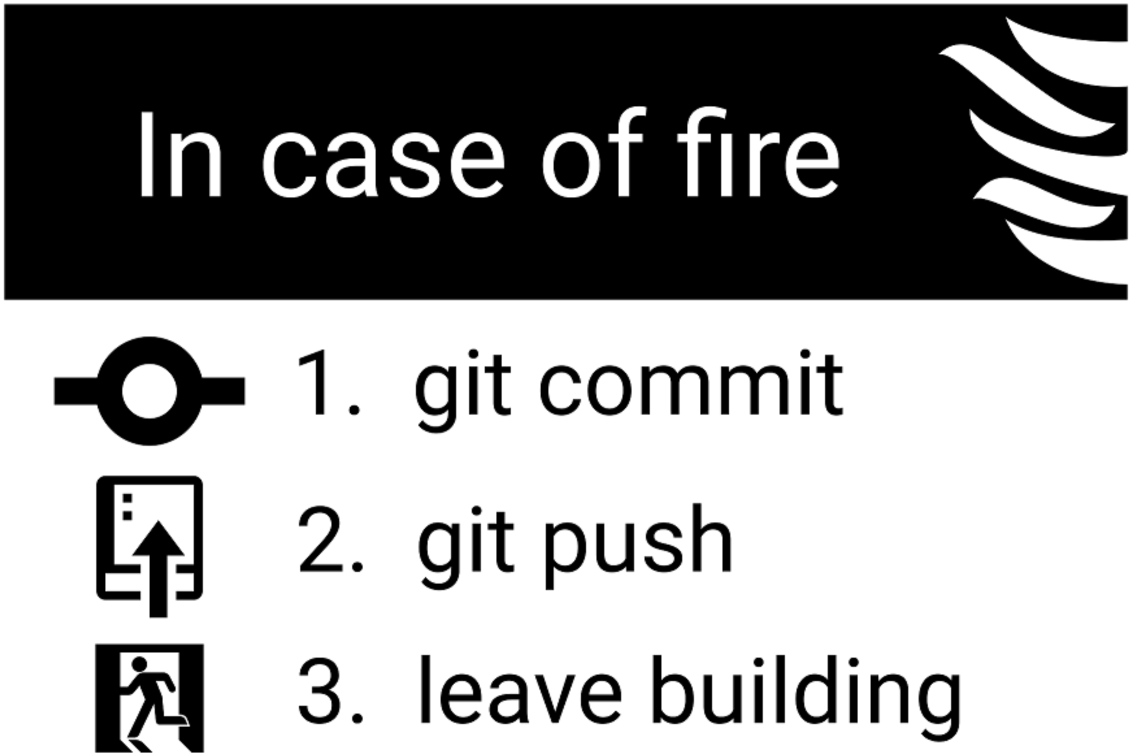
Add files: `git add <filename>`

Commit your changes: `git commit –m “descriptive comment“`

Send your changes to the repository: `git push origin master`

Display the history of changes: `git log`

source: http://rogerdudler.github.io/git-guide/
Img source: http://abload.de/img/in_case_of_fireirrtb.jpg



Or add an alias to use on those special occasions:

alias fire='git add -A . && git commit -a -m "FIRE!" ; git push -f origin --all'

img source: http://abload.de/img/in_case_of_fireirrtb.jpg
alias source: https://www.reddit.com/r/ProgrammerHumor/comments/3nc531/in_case_of_fire/