Introduction to Security of Embedded Devices

Dr. Benedikt Gierlichs

ATHENS course, Leuven, Nov 2013

Goal of this lecture

• Basics of cryptography
• Difference between cryptanalysis and implementation attacks
• Why embedded security?
• Main principles of implementation attacks
• Basics of power analysis
• Exercise on simple power analysis

Background: cryptography

• Cryptology = Cryptography + Cryptanalysis
  • making breaking

• Original problem: two parties, each in a secure environment, want to communicate over an insecure channel
• The bad guy wants to listen in, modify, etc.

Background: cryptography

• Mathematical algorithms to achieve security goals:
  • Confidentiality – Encryption (nobody other than the intended recipient can read the message)
  • Data origin authentication – Message authentication codes (nobody other than the indicated sender could have sent the message, and the message is unmodified)
  • Integrity – Cryptographic hash functions (nobody has modified the message during transmission)
  • Many more

• In the remainder we focus on encryption as example
Background: encryption

• Encryption algorithm: (plaintext, key) → ciphertext
  – Goal (loosely): nobody other than the intended recipient is able to recover the plain message
• Kerckhoffs’ principle: the security of a cryptosystem should exclusively rely on the secrecy of the key
  – No security by obscurity, more later
  – Publicly known algorithm ENCRYPT transforms plaintext into ciphertext under a key, algorithm DECRYPT does the inverse
  – DECRYPT(ENCRYPT(plaintext,key),key) = plaintext

Background: cryptanalysis

• Goal: break the security provided by crypto algorithms
• Breaking a cryptographic algorithm affects every implementation of the algorithm
• Many different contexts and scenarios, e.g.:
  – Given ciphertext, find plaintext
  – Given ciphertext, or plaintext or both, find key
• Most important here:
  – Algorithm is known
  – Cryptanalyst can only observe exchanged messages, e.g. inputs and outputs of cryptographic algorithm
  – Algorithm itself is implemented in a black box
What is an embedded system?

- Special purpose computer system
- Designed to perform one or a few specific tasks
  - Opposed to general-purpose computers
- Engineers can optimize:
  - Reduce size and cost, etc.
  - Increase reliability and performance, etc.
- Range from small portable devices to systems controlling nuclear power plants
- Focus: small, portable devices with relatively low complexity

What is embedded security?

- Typically relates to an embedded system that performs one or several security related tasks
- Cryptography is an important building block for embedded security
  - Embedded cryptographic systems
- Cryptography is not a "magic" wild-card that solves all your problems (more later...)

Embedded Cryptographic Devices

- A cryptographic device is an electronic device that implements a cryptographic algorithm and stores a cryptographic key. It is capable of performing cryptographic operations using that key.
- Embedded (here): it is small, typically portable, mass product

Why Embedded Cryptography?

- Predominant application: authentication
- Physical access control: e.g. remote keyless entry
  - Early days: send identifier in clear
    - Problem: all transmitters use same identifier and same frequency!
    - "Solution": small set of (e.g. 256) identifiers and / or a frequency range
    - Attack: try all identifiers, frequency scanners
Why Embedded Cryptography? (2)

- **Next step**: rolling codes (code hopping)
  - Key and car share a common secret (e.g. 40bits)
  - Next secret := algorithm(secret)
  - Update: secret = next secret
- Prevents replay and exhaustive search
- Problem 1: car and key need to be synchronized
  - Solution: car checks against the 256 next possible secrets
- Problem 2: algorithm() needs to be secret

Why Embedded Cryptography? (3)

- **Better**: challenge and response protocol
  - Key and car share a common secret key k (e.g. 128bits)
  - Publicly known (secure) algorithm E()

Opportunities

- A password yields limited security
  - Humans tend to pick "easy" passwords
  - Transmission of password can be eavesdropped
- Cryptography yields better security
  - But a cryptographic key is difficult to remember
  - And humans are not very good at computing cryptographic algorithms
- Have a device store the key and do the crypto for you
  - Sometimes: password protects access to device
  - Two-way security:
    - You need to have the device
    - You need to know the password

Applications

- Embedded cryptographic devices enable(d) a whole range of novel security applications
  - Authentication
    - Physical access control: buildings, cars, public transport, border control
    - Logical access control: computer system, digital content (e.g. DVD, PayTV, iTunes), mobile phone, eBanking, banking card
    - eGovernment, eVoting, etc.
  - Confidentiality (encryption)
    - Secure storage, eHealth, secure data transfer, eBanking
  - Other
    - Payment: electronic purse, public payphone
    - Loyalty cards
Smart Cards Are Used in Daily Life

Which smart cards do you have in your pocket?

What is a smart card?

- An embedded system integrating
  - A microcontroller that runs software
  - Memory: RAM, ROM, EEPROM
  - Input/Output interface
  - Optionally: cryptographic co-processor

- Stores cryptographic keys and performs cryptographic computations
- Secure, tamper resistant environment
- Security relies on physical possession
  - And optionally on a PIN or password

What is a smart card?

A piece of silicon and a plastic body

max. 25 mm²

PART II: IMPLEMENTATION ATTACKS
The Old Model (simplified view)

- Attack on channel between communicating parties
- Encryption and cryptographic operations in black boxes
- Protection by strong mathematic algorithms and protocols
- Computationally secure

November 2013 Athens course

Or so you think...

- Tempest: refers to investigations and studies of compromising emanations
  - Primarily: Electromagnetic radiation
  - Exploitation of signals and prevention
  - Term coined in the late 1960s (NSA)
  - Documents remain secret until today
  - Basic and redacted versions publicly available in the late 1990s

- Public research:
  - Van Eck phreaking (1985): reading computer screens from a “large” distance (also electronic voting machines)
  - Vuagnoux, Pasini (2009): keystroke logging from a distance (up to 20 meters), works on wireless and wired keyboards

Screen reading from distance


Embedded Cryptographic Devices

- A cryptographic device is an electronic device that implements a cryptographic algorithm and stores a cryptographic key. It is capable of performing cryptographic operations using that key.
- Embedded: it is exposed to adversaries in a hostile environment; full physical access, no time constraints
  - Note: the adversary might be a legitimate user!

November 2013 Athens course
How is Embedded Security affected?

- New Model (also simplified view):
  - Attack on channel and endpoints
  - Encryption and cryptographic operations in gray boxes
  - Protection by strong mathematic algorithms and protocols
  - Protection by secure implementation
- Need secure implementations not only algorithms

Always keep in mind

Your system is as secure as its weakest link

Your system is as secure as its weakest link

And this house is even more secure! The front door is four feet thick and made of solid titanium...

Source: P. Kocher
**Your system is as secure as its weakest link**

- The adversary will go for the weakest entry point
  - Disable or go around security mechanisms
  - Guess / spy on passwords
  - Bribe the security guard

- If you use crypto, he will try to go around it
  - System designer: thinks of the "right" way to use the system
  - Adversary: does not play by the rules
  - Designer has to think like the adversary, anticipate attacks, protect against them
  - There is no way to protect against all attacks
    - Do you know all attacks?

**Security for Embedded Systems**

"Researcher has a new attack for embedded devices
Vulnerability lies in ARM and XScale microprocessors"
Computerworld – security
April 4, 2007
How: Use JTAG interface

"Secustick gives false sense of security"
April 12, 2007
http://tweakers.net/reviews/683
Security completely broken

**Physical security of embedded cryptographic devices**

- Let us assume that the system is well designed
  - Adversary cannot go around / disable security features
  - Cryptography is used

- Embedded context
  - Adversary can "look" at the device under attack
    - Measure physical quantities
  - Adversary can manipulate the device under attack
    - Expose it to physical stress and "see" how it behaves
## Classification of Physical Attacks

- **Active versus passive**
  - Active: Perturbate and conclude
  - Passive: Observe and infer
- **Invasive versus non-invasive**
  - Invasive: open package and contact chip
  - Semi-invasive: open package, no contact
  - Non-invasive: no modification
- **Side channel: passive and non-invasive**
  - Very difficult to detect
  - Often cheap to set-up
  - Often: need lots of measurements automating
- **Circuit modification: active and invasive**
  - Expensive to detect invasion (chip might be without power)
  - Very expensive equipment and expertise required

## Side-Channel Leakage

- Physical attacks ≠ Cryptanalysis
  - (gray box, physics)
  - (black box, maths)
- Does not tackle the algorithm’s math. security

- Observe physical quantities in the device's vicinity and use additional information during cryptanalysis

## Some Side-Channels (not exhaustive)

- **Timing**
  - Overall or "local" execution time

- **Power, Electromagnetic radiation**
  - Predominant: CMOS technology
  - Consumes power when it does something, transistors switch
  - Electric current induces and EM field

- More exotic but shown to be practical
  - Light, Sound, Temperature
Examples: measurement setups

- Smart cards
- Phones
- FPGA, ASIC
- Etc.

Side-Channel leakage

- Side-channel leakage
  - Is not intended
  - Information leakage was not considered at design time
  - Leaked information is not supposed to be known
  - Can enable new kind of attack

- Often, optimizations enable leakage
  - CMOS, cache memory, etc.

- Device under attack is operated in normal conditions
  - Adversary is passive an solely observes

Principle is nothing new...

"Breaking into a Safe is hard, because one has to solve a single, very hard problem..."

"Divide et impera!"

"Things are different if it is possible to solve many small problems instead..."

A timing attack

- 4 digit PIN verification
  - 10000 combinations possible
  - On average 5000 attempts necessary (10000 worst case)
  - Typically only 3 attempts allowed (counter)
  - Probability of correct guess: about 3/10000
A timing attack (2)

- Function check_pin(user_PIN, correct_PIN, length)
  - For (i = 1 to length)
    - IF (user_PIN[i] ≠ correct_PIN[i])
      - Return -1
    - ENDIF
  - ENDFOR
  - Return 0

- If (check_pin(...) == -1)
  - increase counter
- Else
  - counter = 0

A timing attack (3)

- Execution time of the algorithm leaks information!
- The adversary can learn a lot
  - Test a random PIN, say this takes time N
  - Change the first PIN digit and try again
  - If the running time is == N, the initial and the new first digit are both wrong
  - If the running time is < N, the initial first digit was correct
  - If the running time is > N, the new first digit is correct
- This takes on average 5 (worst case 10) attempts per digit: on average 20 (worst case 40) for the entire PIN!
- But only 3 attempts allowed...

Concept of Side Channel attacks

- Some cryptographic algorithms gain their cryptographic strength by repeating a "weak" function many times
  - Classical model: adversary sees only final and secure result
- Other algorithms are more complex but their implementations follow a similar idea
- Side Channels leak information about the "weak" intermediate results
- Side Channel attacks can "see" inside the implementation and attack "weak" intermediate results

Invasive attacks

- Passive: micro-probing
  - Probe the bus with a very thin needle
  - Read out data from bus or individual cells directly
  - Several needles concurrently
- Active: modify circuits
  - Connect or disconnect security mechanism
  - RNG stuck at a fixed value
  - Reconstruct blown fuses
  - Cut or paste tracks with laser or focused ion beam
  - Add probe pads on buried layers

[Helena Handschuh]

[www.fa-mal.com]
Active attacks: fault injection

Apply combinations of strange environmental conditions

- Vcc
- Glitch
- Clock
- Temperature
- UV
- Light
- X-Rays

and bypass or infer secrets

input error

Active attacks (semi invasive) fault injection

- Exploit faulty behavior provoked by physical stress applied to the device
- Semi-invasive: open package but no contact

Microscope view

Exploitation of faulty computation results

- Simple example: set return value of check_pin() to 1 regardless of the value of the user-PIN
Embedded Security

**NEED BOTH**

- **Efficient Implementation**
  - Within power, area, timing budgets
  - Public key: 1024 bits RSA on 8 bit μC and 100 μW
  - Public key on a passive RFID tag

- **Trustworthy implementation**
  - Resistant to attacks
  - Active attacks: probing, power glitches, JTAG scan chain
  - Passive attacks: monitor electromagnetic radiation

Why a hard engineering problem?

- More difficult to guarantee that something will not happen (attacks) than that something will happen
- Engineers are trained to make something happen
- Envision attack paths, fix the problems
  - ... and hope for the best?

Back to the PIN example (problem was the timing...)

- Function `check_pin(user_PIN, correct_PIN, length)`
  - `temp = 0`
  - For (i = 1 to length)
    - IF (user_PIN[i] ≠ correct_PIN[i])
    - `temp += 1`
    - ENDIF
  - ENDFOR
  - If (`temp ≠ 0`)  
    - Return -1
  - ...

- This version should be better. Right?

Back to the PIN example

- Yes, but actually it does not really matter
  - See how the function is used
  - If (check_pin(...) == -1 )
    - increase counter
  - Else
    - counter = 0
  - Better: defensive programming
    - Increment counter
    - Check
    - Decrement counter if PIN correct

Now that you know about active attacks...
Cut power supply here and try as often as you want.
Security as a design dimension

- Energy, peak power
- Chip size, Code size, Memory
- Execution time
- Security

Challenges in embedded security
- Efficient implementation of cryptographic algorithms
- Secure implementation
- Implementation of security ≠ Secure implementation

Physical attacks and countermeasures
- Arms-race
- Perfect security does not exist
- Goal: make it too expensive to attack
- Algorithm security still orders harder than side-channel attacks

Conclusion

Measuring power consumption
- Not average power over time, not peak power
- Instantaneous power over time
  - Trace or curve, many samples
- Typical setup:
  - Oscilloscope
  - Device under attack
  - Central computer
  - Clock generator
  - Power supply
  - Probes

PART III: INTRODUCTION TO POWER ANALYSIS
Measuring power consumption (2)

- Logic: constant supply voltage, supply current varies
- Predominant technology: CMOS
  - Low static power consumption
  - Relatively high dynamic power consumption
  - Power consumption depends on input
- CMOS inverter:

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
<th>Current</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
<td>1 1</td>
<td>Low</td>
</tr>
<tr>
<td>0 1</td>
<td>1 0</td>
<td>Discharge</td>
</tr>
<tr>
<td>1 0</td>
<td>0 1</td>
<td>Charge</td>
</tr>
<tr>
<td>1 1</td>
<td>0 0</td>
<td>Low</td>
</tr>
</tbody>
</table>

Measuring power consumption (3)

- Oscilloscope can only measure voltage
  - Generate voltage signal, proportional to current
- Measure in VDD or GND line
  - Resistor (Ohm's law: $U = R \times I$), measure $U$ over resistor
  - Current probe: current field voltage
  - Dedicated measurement circuits
- Measure 'global' E or H field of the device
  - Field intensity proportional to power consumption
  - Field orientation depends on current direction

Power analysis

- What can we see looking at a curve?
- Information in:
  - Repetitive patterns: typically coarse, structure of algorithm and implementation (e.g. loops)
  - Time: what happens when, program flow
  - Amplitude: what happens at a given moment in time, data flow
    - the same operation, executed with different operand values, consumes more or less power

SPA: Simple power analysis attacks

- Anything but simple (except in examples)
- Visual inspection of few traces, worst/best case: single shot
- Often exploitation of direct key dependencies, input and output data need not be known (but they are useful for verification)
- Require: expertise, experience, detailed knowledge about target device and implementation
Insecure RSA implementation

RSA modular exponentiation
In: message m, key e (l bits)
Output: $m^e \mod n$

$A = m$
for $j = 1 - 2$ to $0$
    $A = A^2 \mod n$ /* square */
    if (bit j of k) is 1 then
        $A = A \times m \mod n$ /* multiply */
Return $A$

Side-Channel

Power Attacks on Public-Key Coprocessors

- RSA-like algorithms use modular exponentiation
  - the secret is the exponent
  - implemented as square and multiply algorithm
    - exponent bit 0: square
    - exponent bit 1: square and multiply
  - coprocessor optimized for square
  - observed difference between both operations reveals the key!