Cyber Security Body of Knowledge:

Hardware Security

Ingrid Verbauwhede
KU Leuven COSIC
Outline

• What: secure hardware from cloud to IOT
• Define: root of trust
• How: design methodology
• Walk through the different HW design abstraction layers
• Conclusions
NEXT GENERATION EMBEDDED SYSTEMS
Cyber physical systems

“Networked embedded systems interacting with the environment”

Ed Lee, after H. Gill, NSF

Figure 6. Displaying an arbitrary message and a false speedometer reading on the Driver Information Center. Note that the car is in Park.

[IEEE Symp Security and Privacy, 2010]
Cyber physical systems

[Source photograph: J. Rabaey: A Brand New Wireless Day]
There will be no E-... and no smart ... without security
E-....: e-health, e-commerce, e-voting, bitcoin, litecoin
Smart-....: smart grid, smart home, smart phone, smart car
Internet of Things, Internet of Everything, Industry4.0
Design challenge: provide security and privacy within

Area, time, power and energy budgets!
TRUST AND TRUST BOUNDARIES
Trust – definition

Trust (R. Anderson in “Security Engineering”, after NSA):
• “Trusted system or component is one whose failure can break the security policy, while a trustworthy system or component is one that won’t fail.”

Trust (Trusted Computing Group):
• “An entity can be trusted if it always behaves in the expected manner for the intended purpose.”

Loosely stated: if trusted system or component fails, then bad things can happen.

Goal of security: minimize what needs to be trusted
Old attack model (simplified view):
- Attack on channel between communicating parties
- Encryption and cryptographic operations in black boxes
- Protection by strong mathematic algorithms and protocols

Focus on communication link: EFFICIENCY
Cost definition

• Area
• Time: throughput versus latency
• Power, Energy
• Physical Security
• NRE (Non Recurring Engineering) cost
Area

- **ASIC = Application Specific Integrated Circuit**
  - Gate count
  - Unit = NAND gate = 4 transistors
- **FPGA = Field Programmable Gate Array**
  - Look-up-Tables, Memory
- **Embedded micro-controllers**
  - Memory size = program size + data size
- **Micro-processors, SOC,**
  - Number of cores, memory size
TIME

Clock frequency versus sample frequency
Throughput versus latency
Real-time, throughput, latency

- **Throughput** = associated with *application*
  - Amount of data processed per time unit
  - Video: Gbits/sec, Internet: Gpackets/sec
  - *Real-time sample rate*: HW has to work as fast as application dictates

- **Latency** = associated with *application*
  - Delay from input to output
  - Measure of reaction speed or turn-around time
  - E.g. Brakes of car, memory encryption

- High throughput and low latency don’t go together
Past: design for efficiency – e.g. DES

Efficiency

- Data Encryption Standard
- Programmable co-processor
- Enc/Dec
- 3DES
- PRNG
- MAC generation
- Modes of operation

[EuroAsic 1991]
Power and energy added!

• Power is limited:
  – RFID tags
  – Cooling: for the small and the server!!
  – Implanted devices only temperature $\Delta < 1 \, ^{\circ}\text{C}$

• Energy Battery is limited
  – Pace maker battery is not rechargeable
  – One AAA battery is $< 5000$ Joule
  – Bitcoin mining = ecological disaster

• **How much crypto in one (micro) Joule?**
Past: efficiency & power - Rijndael

- HW and SW ‘friendly’
- Rijndael AES evaluation
- Enc + Dec
- 0.18 µm CMOS
- Standard cells
- 2.3 Gbits/sec
- Only 56 mW
- Or 11 Gbits/Joule

[JSSC 2003]
## Throughput – Energy Efficiency

<table>
<thead>
<tr>
<th></th>
<th>Throughput</th>
<th>Power</th>
<th>Figure of Merit (Gb/s/W = Gb/J)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES 128bit key</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>128bit data</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.18um CMOS</td>
<td>3.84 Gbits/sec</td>
<td>350 mW</td>
<td>11 (1/1)</td>
</tr>
<tr>
<td>FPGA [1]</td>
<td>1.32 Gbit/sec</td>
<td>490 mW</td>
<td>2.7 (1/4)</td>
</tr>
<tr>
<td>Intel ISA for AES</td>
<td>32 Gbit/sec</td>
<td>95 W</td>
<td>0.34 (1/33)</td>
</tr>
<tr>
<td>ASM StrongARM [2]</td>
<td>31 Mbit/sec</td>
<td>240 mW</td>
<td>0.13 (1/85)</td>
</tr>
<tr>
<td>Asm Pentium III [3]</td>
<td>648 Mbits/sec</td>
<td>41.4 W</td>
<td>0.015 (1/800)</td>
</tr>
<tr>
<td>C Emb. Sparc [4]</td>
<td>133 Kbits/sec</td>
<td>120 mW</td>
<td>0.0011 (1/10.000)</td>
</tr>
<tr>
<td>Java [5] Emb. Sparc</td>
<td>450 bits/sec</td>
<td>120 mW</td>
<td>0.0000037 (1/3.000.000)</td>
</tr>
</tbody>
</table>

1. Amphion CS5230 on Virtex2 + Xilinx Virtex2 Power Estimator
2. Dag Arne Osvik: 544 cycles AES – ECB on StrongArm SA-1110
3. Helger Lipmaa PIII assembly handcoded + Intel Pentium III (1.13 GHz) Datasheet
4. gcc, 1 mW/MHz @ 120 Mhz Sparc – assumes 0.25 u CMOS
5. Java on KVM (Sun J2ME, non-JIT) on 1 mW/MHz @ 120 MHz Sparc – assumes 0.25 u CMOS
Trust boundaries – current model

Modified Attack Model (also simplified view):
- Attack 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*
Design for efficiency AND security

SEMA attack: Simple Electromagnetic Attack on Elliptic Curve Public Key implementation.

[E. Demulder EUROCON 2005]
Embedded Security

NEED BOTH

Efficient, light-weight Implementation
- Within power, area, timing budgets
- Public key: 1024 bits RSA on 8 bit \( \mu \)C and 100 \( \mu \)W
- Public key on a passive RFID tag

Trustworthy implementation
- Resistant to attacks
- Active attacks: probing, power glitches, JTAG scan chain
- Passive attacks: side channel attacks, including power, timing and electromagnetic leaks
DESIGN METHODS

For low power/low energy, high security
Design methodology: design abstraction levels

Application: e-commerce, smart energy

Security protocol: authentication, privacy, ...

Crypto Algorithm/Protocol: crypto, entity authentication

Architecture: Co-design, HW/SW, SOC

Micro-Architecture & micro-archi attacks: co-processor design

Circuit: Circuit techniques to combat side channel analysis attacks

WHY: 1. To get low power/low energy
2. To be secure
"A root of trust is a component at a lower abstraction layer, upon which the system relies for its security."
BODY OF KNOWLEDGE AT
DIFFERENT ABSTRACTION LAYERS

Illustrations: for more formal definitions and details see document
Level 1: Secure platforms

- Highest abstraction level from HW perspective
- What is visible to SW designer or security protocol designer
- Examples:
  - HSM: Hardware Security Module
  - Secure Element and Smartcard
  - Trusted Platform Module
  - ...
Smart card IC

- Dedicated **semiconductor** based component
- Contains memory
  - ROM
  - RAM
  - EEPROM
  - FLASH
- Contains a CPU
- Crypto accelerations
- Physical countermeasures

[source: H. Handschuh]
Level 2: Support for Software security

• Provide
  – Isolation
  – Attestation
  – ...

• Concept of TEE Trusted Execution Environment
  – ARM Trustzone
  – Intel SGX
  – KU Leuven Sancus
SANCUS, Protected Module Architecture

Modify micro-controller architecture
Example: MSP430 embedded micro-controller
Program-counter based memory access control
Dedicated set of instructions

[Usenix 2013, ACM Transactions on Privacy and Security 2017]
Level 3: co-processors for cryptographic algorithms

- Secret key algorithms:
  - AES, 3xDES,
  - Lightweight, authenticated encryption
  - NIST call for lightweight
- Public key algorithms:
  - RSA, ECC,
- Post-quantum
Elliptic curve Public key for RFID

- Public key ECC: one point multiplication less 5 microJoule
- Combination full-custom – standard cells
- HW and SW co-design
- Fits power budget of **PASSIVE** RFID-tag
- Side channel attack countermeasures
Post-quantum crypto

Mathematical foundation of existing public key algorithms disappears
Quest for novel post-quantum secure public key algorithms.
Lattice based:

New Algorithms:
- Efficient
- Side-channel attack resistant
- Post-quantum secure

New Hardware Architectures
Level 4: side-channel attacks

Micro-architectural side-channels:
• Spectre, Melt-down, Foreshadow

Physical side side-channel:
• Power
• Timing
• Electro-magnetic

Active attacks:
• Laser attacks
Design for efficiency AND security

SEMA attack: Simple Electromagnetic Attack on Elliptic Curve Public Key implementation.

[E. Demulder EUROCON 2005]
Timing Leakage

Elliptic Curve Public Key
“Point Multiplication” algorithm
Top level description

In: point P, key k (W bits)
Output: Q = k.P

for j = 0 to W - 1
  Q = 2.Q /* double */
  if (bit j of k) is 1 then
    Q = Q + P /* add */

Return Q

Timing Side-Channel
Also basis for cache timing attacks
Level 5: countermeasures against attacks

Hiding: make power consumption variations independent of data processed
- Dynamic differential circuit styles
- Balanced place and route
- Examples: Sense Amplifier Based Logic, Wave Dynamic Differential Logic

Randomization, masking
- Algorithm level randomness: e.g. public key
- Circuit level randomness: masked logic styles
- Secret sharing, multi-party computation
- NEEDS random number generation on chip!
AES with DPA countermeasures

- AES, 2nd generation
- Regular & **WDDL** based implementation
- Standard cells
- 1 Gbit/sec @ 50MHz
- to 3.8 Gbits/sec @ 330MHz
- 50mW unprot
- to 200mW prot

[CHES2005]
Level 6: Transistor level roots of trust

- PUFs: Physically Unclonable Functions
  - Replacement for key storage
  - Cheap alternative for IC authentication

- TRNG: True Random Number Generators
  - Key generation
  - Nonces
  - ...
Silicon PUFs - Variability

- Silicon Biometrics
- Variability in transistors and interconnect
- In general undesired, except for PUFs
- Random dopant fluctuation
- Line edge/width roughness
- Crucial design challenge with CMOS down scaling (Moore’s law)
  Pelgrom’s law: $\sigma^2 \sim 1/\text{WL}$ (Marcel Pelgrom, Dutch engineer)
The ideal PUF?

IDEAL PUF is without noise
PUF (F = Function)

- Dream 1: IDEAL PUFs don't exist...
- Practical example of a (weak) PUF - SRAM

Most “strong” PUFs broken: focus on weak PUFs for key generation
PUF behavior of SRAM in commodity micro-controllers

Black box approach (off the shelf micro-controllers)

- PIC16F1825
- STM32F100R8

[PhD thesis Anthony VH, PUFFIN]
PUF behavior of SRAM in commodity micro-controllers

Black box approach (off the shelf micro-controllers)

- PIC16F1825
- STM32F100R8

Not yet useful: needs post-processing to create ID or key!
Cryptographic Key Generation from PUF

Physically Unclonable Function

PUF

challenge → response

INTRA distance

INTER distance

PUF based Key Generation

Cryptographic Key

Uniform
True Random Number Generators – compliant with AIS 31 and NIST
TRNG On-line test challenge

1. Data Collection (normal operation and under attack)
2. Preliminary selection of useful features
3. Feature verification
4. Attack impact analysis

5. HW implementation

6. HW verification

Example: TOTAL
• Shows attack on TRNG
• Shows how some statistical features can detect an attack on the fly while other not
“A root of trust is a component at a lower abstraction layer, upon which the system relies for its security.”

Hardware Security Body of Knowledge:

- Many design abstraction layers
- Root of trust is one layer down
- Weakest link decides the security