Requirements, Threat Models and Report on Building Blocks for Hardware Security

Project number:	238811
Project acronym:	**UNIQUE**
Project title:	Foundations for Forgery-Resistant Security Hardware
Start date of the project:	01.09.2009
Duration:	30 months

Deliverable type:	Report
Deliverable reference number:	238811/ D1.1 / Final |1.1
Deliverable title:	Requirements, Threat Models and Report on Building Blocks for Hardware Security
WP contributing to the deliverable:	WP1
Due date:	2010-08-31
Actual submission date:	2010-08-31

Responsible organization:	TUD (Heike Schröder, Stefan Katzenbeisser)
Authors:	All partners contributed
Abstract:	Deliverable D1.1 provides the basis for the foundation of the technical work in the UNIQUE project. For this purpose the identification of requirements, threat models, and building blocks has been done.
Keywords:	use cases, requirements, building blocks, hardware level technology, design technique

Dissemination level:	Public
Revision:	FINAL | 1.1

Instrument:	STREP
Thematic Priority:	ICT

Table of Contents

1 Introduction

The first part of this document contains a summary of requirements, a description of the threat model and guidelines for security evaluation identified in Task 1.1. In particular, Section 2 specifies relevant scenarios as part of UNIQUE internal deliverable iD1.1 "Requirements and Threat Model for Anti-Counterfeiting and Anti-Tampering solutions". The information within reflects contributions from various UNIQUE project participants. It is not suggested that it is possible to mitigate every threat listed in this document.

The second part of the document includes a list of promising building blocks in conjunction with an analysis in respect to the identified requirements. The structure of the second part, representing UNIQUE internal deliverable iD1.2 "Preliminary Report on Building Blocks for Hardware Security Features", is as follows: Section 3 gives an overview of the most important cryptographic primitives and protocols, which are available in the field and used in practice. Subsection 3.1 gives an overview of the most common symmetric cryptographic primitives, while Subsection 3.2 focuses on asymmetric primitives. For each of the touched subjects in these subsections, we will try to provide the following information: Definition, functionality, parameters, security and important instantiations. Section 4 covers hardware-level technologies and design techniques. Finally, Section 5 gives an overview of existing technologies and building blocks based on Physical Unclonable Functions (PUF). The scope is limited to electronic PUFs that can be used on Integrated Circuits (IC).

2 Threat Scenario

In the following we specify the requirements, description of the threat model and guidelines for security evaluation identified in Task 1.1.

2.1 Counterfeiting

A counterfeit product is defined as bearing a validly registered trademark without the authorization of the trademark owner. The counterfeiter aims to make the product indistinguishable from the original and achieves this with varying degrees of success.

Counterfeiting and forgery are very serious global issues causing significant economic losses as well as other important negative impacts. We distinguish between two types of counterfeiting. The first type is called overproduction or overbuilding. In some cases (e.g. outsourcing), manufacturers who are not technology developers or owners may be in possession of the blueprints of the original products. The easiest way to create perfect forgeries is to overproduce the product in an unauthorized manner, by manufacturing unauthorized exact copies after hours. Since the product is unchanged, we consider this model as **passive** counterfeiting. There are several types of **active** counterfeiting. The first sub-class of active counterfeiting is also related to outsourcing in manufacturing. Overseas manufacturers may try to cut costs by omitting or reducing the defined set of features from the original design, with possible effects on security. There is also a risk that the functionality on a chip has been deliberately modified in a malicious way or supplemented with a hidden trapdoor circuit such as a hardware Trojan. For instance, a circuit might be added such that keys that were never supposed to leave a security component could be leaked (e.g., via padding or a subliminal channel). Additionally tamper or leakage protection circuits may be disabled or weakened, the True Random Number Generator (TRNG) may be biased or the IC might have a kill switch that makes it stop functioning under certain conditions. The second sub-class of active counterfeiting refers to cloning and/or reverse engineering of products to obtain Intellectual Property (IP) in an unauthorized way.

Finally, in the following subsections we classify counterfeits into device remarking, IC overproduction, higher-level overproduction, IC reverse engineering by delayering, and counterfeit higher-level assemblies.

2.1.1 Semiconductor Device Re-marking

Threat Type:

- Counterfeiting.
- Active.

Use Case Definition:

> Remarking is the most common counterfeiting problem today [1]. A device's product markings are removed and replaced with some combination of trademark, speed grade, temperature range, date code, serial number or other markings. The device may not be functional such as in the case of a remarked mechanical sample or a dead device. Remarked devices enter the black or grey markets and are sold to unwary end users.

Threat:

- Entity: Third party.
- Budget: Low to medium depending on marking technology.
- Goal: Increase semiconductor device value for resale.

Impact:

- Reputational damage and revenue loss for brand owner.
- Reliability and safety issues ranging from data loss to loss of life.
- Failure of defense systems, critical infrastructure, threats to national security.

Risk:

- High.

Security Requirement:

- Tamper evidence device packing.
- Silicon-level device identification mechanism.

State-of-the-Art:

- Remarking resistant packages.
- Laser marking.
- 2D Codes.

2.1.2 IC Overproduction

Threat Type:

- Overproduction.
- Passive.

Use Case Definition:

> It is common for semiconductor companies to use a fabless business model where IC manufacturing is outsourced to an external foundry. An unlicensed manufacturing overrun is possible if the foundry processes excess wafers and the foundry or an external packaging house packages the resultant dice. These devices can be considered 'perfect counterfeits'.

Threat:

- Entity: Foundry and packaging house (note: at the moment masks are not portable to other fabs).
- Budget: Low (no non-recurrent engineering).
- Goal: Produce excess ICs to be sold on alternative markets.

Impact:

- Loss of revenue for fabless/Original Equipment Manufacturer (OEM).
- Loss of market share for fabless/OEM.
- Reputation damage for foundry and packaging house.

Risk:

- Low for cutting edge technologies (few well-known companies: TSMC…).
- Higher for old technologies ("boutique" foundry – they provide customized and specialized design and manufacturing services in low temperature cofired ceramics (LTCC) design and modeling).

Security Requirement:

- Traceability mechanism allowing the IP owner to verify whether a device was legitimately manufactured is required.
- Authenticate genuine chips.

State-of-the-Art:

- IP activation mechanism.

2.1.3 Higher Level Overproduction

Threat Type:

- Overproduction.
- Passive.

Use Case Definition:

> It is common for electronics companies to outsource the manufacturing of boards or higher-level assemblies to a contract manufacturer. The contract manufacturer may produce more products than defined by the contract. These unauthorized units then enter the black or grey market and can be considered as "perfect counterfeits".

Threat:

- Entity: Manufacturer.
- Budget: Low.

- Goal: Produce additional legal products in order to sell them on white/grey/ black markets.

Impact:

- The IP owner suffers loss of revenue.

Risk:

- High.

Security Requirement:

- A traceability mechanism allowing the IP owner to verify whether a board or higher-level assembly was legitimately manufactured is required.
- Hardware/software binding.
- A genuine IC should recognize "illegal" surroundings.

State-of-the-Art:

- Hardware/software bindings.

2.1.4 IC Reverse Engineering by Delayering

Threat Type:

- Production/IP Cloning.
- Active.

Use Case Definition:

> An IC's transistor- or gate-level net list is extracted using state of the art IC analysis techniques. This is a destructive process performed on a number of device samples, with the objective of obtaining a set of images representative of the layer stack-up of the device. Polygon feature extraction is used to identify transistors (or gates) and metal interconnects. Special software can then combine this data into a viable net list. The net list can be targeted to the original or a different process technology and the resultant die packaged to create counterfeit devices.

Threat:

- Entity: Government agency, criminal organization, market competitor, rogue foundry/packaging house.
- Budget: High.
- Goal: Produce counterfeit ICs to be sold on white/grey/black market.

Impact:

- It is likely that the quality and reliability of a counterfeit IC produced by reverse engineering will be reduced.
- Implementation errors resulting from incomplete knowledge of the design or due to the net list extraction process itself may be introduced.
- Lack of access to the original verification regime will limit test vectors to only those, which can be captured during device operation.
- The IP holder will suffer loss of revenue and reputational damage if such devices fail in the field.
- For safety-critical applications there may be a risk to life.

Risk:

- Low (large transistor count ICs implemented in state-of-the-art technology nodes).
- Medium (niche products and obsolete devices)

Security Requirement:

- It should be possible to establish the authenticity of an IC thus allowing counterfeit devices to be detected.
- IP Activation

State-of-the-Art:

- Mass serialization techniques using unique identifiers implemented in Non-Volatile Random-Access Memory (NVRAM) or fuse technologies. These approaches are susceptible to cloning. Note that there is a precedent for public opposition to unique identifiers in general purpose processors because of privacy concerns.
- Obfuscation techniques (dummy circuits, flat chip layouts)
- Layout features which when copied verbatim result in a non-functional device (example two step contacts)

2.1.5 Counterfeit Higher Level Assemblies

Threat Type:

- Production/IP Cloning.
- Active.

Use Case Definition:

Board level products, system components and complete systems comprising subassemblies such as boards, mechanical components and casings are vulnerable to counterfeiting. A counterfeit board attempts to mimic the form factor, layout and routing of an authentic board as well as its functionality.

Silkscreen, labels and packaging may be indistinguishable from the original. Complete counterfeit systems comprising subassemblies such as boards, mechanical components and casings are a threat. Any required firmware, Field Programmable Gate Array (FPGA)/ Complex Programmable Logic Device (CPLD) configuration data or other software required will typically be copied from the original.

Threat:

- Entity: Contract Manufacturer, OEM, criminal organization.
- Budget: Medium.
- Goal: Produce counterfeit product, sell on white/grey/black market.

Impact:

- Loss of revenue for the IP owner. Since the quality and reliability of the counterfeit product is likely to be lower the IP owner will suffer reputational damage when such product fails in the field. For safety critical applications there may be a risk to life.

Risk:

- High.

Security Requirement:

- It should be possible to verify the authenticity of a Printed Circuit Board (PCB), component or system.

State-of-the-Art:

- Mass serialization techniques using unique identifiers implemented in NVRAM or fuse technologies.
- Authentication ICs are available, which provide this function and are intended for inclusion in the assembly. However, such approaches are susceptible to cloning.
- Product marking techniques such as laser etching, 2D codes, and holograms.

2.1.6 Summary of Counterfeiting

	Semiconductor Device Re-marking	IC Overproduction	Higher Level Overproduction	IC Reverse Engineering by Delayering	Counterfeit Higher Level Assemblies
Threat Type	Counterfeiting, active	Overproduction, passive	Overproduction, passive	Production/IP cloning, active	Production/IP cloning, active
Threat Entity	Third party	Foundry/packaging house	Manufacturer	Government agency, criminal organization, market competitor, rogue foundry/packaging house	Contract manufacturer, OEM, criminal organization
Threat Budget	Low to medium	Low	Low	High	Medium
Threat Goal	Increase semiconductor device value for resale	Produce excess ICs; sell on alternative market	Produce additional illegal products; sell on white/grey/ black market	Produce counterfeit ICs; sell on white/grey/black market	Produce counterfeit product, sell on white/grey/black market
Impact	Reputational damage and revenue loss for brand owner; Reliability and safety issues ranging from data loss to loss of life; Failure of defence systems, critical infrastructure, threats to national security	Loss of revenue for fables/OEM; loss of market share for fables/ OEM; reputation damage for foundry/packaging house	The IP owner suffers loss of revenue and market share	Reduction of quality/reliability of a counterfeit IC produced by reverse engineering; introduction of implementation errors resulting from incomplete knowledge of the design or due to the net list extraction process itself; limitation to test vectors to only those, which can be captured during device operation; IP holder suffers loss of revenue/ reputational damage; risk of life for safety critical applications	Loss of revenue for the IP owner; Since the quality/ reliability of the counterfeit product is lower IP owner suffers reputational damage; risk of life for safety critical applications
Risk	High	Low for cutting edge technology; high for old technologies	High	Low for state-of-the-art technology devices; medium for niche products and obsolete devices	High
Security Requirement	Tamper evident device packaging; silicon-level device identification mechanism	Traceability mechanism; authentication of genuine chips	Traceability mechanism; binding software to hard-ware; genuine ICs recognize "illegal" surrounding	Verify authenticity of an IC thus allowing counterfeit devices to be detected	Verify the authenticity of PCB, component or system
State-of-the-Art	Remarking resistant packages; laser marking; 2D codes	IP activation mechanism	Hardware/software binding	Mass serialization techniques using unique identifiers implemented in NVRAM or fuse technologies	Mass serialization techniques using unique identifiers implemented in NVRAM or fuse technologies; product marking techniques such as laser etching, 2D codes, and holograms

2.2 Cloning

A cloned product is one, which attempts to replicate the functionality of the original and is intended to compete in the market alongside the original. A trademark, if used, will not be that of the original although in some cases it may be similar in nature allowing for a subtle deception of the consumer.

2.2.1 IC Clones

Threat Type:

- Production/IP Cloning.
- Active.

Use Case Definition:

> Reverse engineering techniques (outlined in 2.1.4) can be used to develop an IC clone with identical functionality to that of the original and which is intended to compete alongside the original in the marketplace under a different brand name.

Threat:

- Entity: Criminal organization/market competitor.
- Budget: High.
- Goal: Put a competing IC on the original market.

Impact:

- Loss of revenue and market share for brand/IP owner.

Risk:

- Low for most common ICs (state-of-the-art). Medium for niche products and obsolete devices.

Security Requirement:

- Activation of ICs.

State-of-the-Art:

- Activation mechanisms.

2.2.2 Mass Serialization IC Clones

Threat Type:

- Production/IP Cloning.
- Active.

Use Case Definition:

> Electronic identification ICs (e.g. Radio-Frequency Identification (RFID) tags used for mass serialization applications) can be cloned by reading the relevant data from a genuine device and writing it back to another device of equivalent functionality.

Threat:

- Entity: Criminal organization/market competitor.
- Budget: Medium, depending on the difficulty of obtaining programmable target devices with equivalent functionality.
- Goal: Make a product counterfeit or clone that is identified as a genuine product when the IC is verified in the application.

Impact:

- Loss of revenue and market share for brand/IP owner of original product.
- Possible safety concerns (e.g. in the case of counterfeit parts that are not compliant with safety standards).
- Identity theft and possible privacy concerns, e.g., in case of cloned RFID-based passports.

Risk:

- Medium.

Security Requirement:

- Strong authentication mechanism in combination with secure on-chip storage of secrets.

State-of-the-Art:

- Cryptographic protocols (see, e.g., [52] for an example).

2.2.3 Identification IC Clones

Threat Type:

- Production/IP Cloning.
- Active.

Use Case Definition:

> ICs for identification applications that use cryptographic secrets to protect data or execute cryptographic authentication protocols. Invasive attacks or side-channel attacks can recover the secret even when stored in secure on-chip non-volatile memory. Once the secret is recovered, the attacker can clone the IC.

Threat:

- Entity: Criminal organization/market competitor.
- Budget: High, if invasive attacks need to be performed.
- Goal: Make a product counterfeit or clone that is identified as a genuine product when the IC is verified in the application.

Impact:

- Loss of revenue and market share for brand/IP owner of original product. Possible safety concerns (e.g. in the case of counterfeit parts that are not compliant with safety standards). Possible security concerns (e.g. when building or vehicle access control ICs are cloned).

Risk:

- Low.

Security Requirement:

- Strong authentication mechanism in combination with secure on-chip storage of secrets (making invasive attacks difficult).

State-of-the-Art:

- Active tamper detection mechanisms that clear the key storage as soon as tampering are detected.
- Randomizing cryptographic operations, use of dual-rail logic, and asynchronous technology to thwart side channel analyses.

2.2.4 Higher Level Clones

Threat Type:

- Production/IP Cloning.
- Active.

Use Case Definition:

Higher level clones such as boards and systems, which attempt to replicate the functionality of the original without a strong emphasis on deceiving the consumer. Any required firmware, FPGA/CPLD configuration data or other software required would typically be copied from the original. The boards/systems will be sold under a different brand name.

Threat:

- Entity: Criminal organization, market competitor.
- Budget: Medium (reverse engineering of boards/systems is easier compared to ICs).

- Goal: Put a competing board/system on the original market.

Impact:

- Loss of revenue and market share for brand owner.
- Safety and reliability concerns

Risk:

- Medium.

Security Requirement:

- Activation of ICs to make sure that board/system counterfeits will not function or binding software to hardware in a way that counterfeits will not be able to operate.

State-of-the-Art:

- Firmware updates that only work on the original product, e.g., based on features reserved for future use that were not (successfully) cloned.

2.2.5 Summary of Cloning

	IC Cloning	Mass Serialization IC Clones	Identification IC Clones	Higher Level Clones
Threat Type	Production/IP cloning, active	Production/IP cloning, active	Production/IP cloning, active	Production/IP cloning, active
Threat Entity	Criminal organization, market competitor	Criminal organization, market competitor	Criminal organization, market competitor	Criminal organization, market competitor
Threat Budget	High	Medium	High	Medium
Threat Goal	Put a competing IC on the original market	Product counterfeit or clone	Product counterfeit or clone	Put a competing board/ system on the original market
Impact	Loss of revenue and market share for brand/IP owner	Loss of revenue and market share for brand/IP owner; loss of safety; identity theft and privacy	Loss of revenue and market share for brand/IP owner; loss of safety; identity theft and privacy	Loss of revenue and market share for brand owner
Risk	Low for most common ICs; medium for niche products and obsolete devices	Medium	Low	Medium
Security Requirement	Activation of ICs	Strong authentication mechanism in combination with secure on-chip storage of secrets	Strong authentication mechanism in combination with secure on-chip storage of secrets	Activation of ICs; binding software to hardware
State-of-the-Art	Activation mechanisms	Cryptographic protocols	Active tamper detection mechanisms; randomizing cryptography operations; dual-rail logic; asynchronous technology	Firmware updates that only work on original product

2.3 Unlicensed IP Use

2.3.1 Hardware IP Cores

Threat Type:

- Production/IP Cloning.
- Active.

Use Case Definition:

Developers of IP cores for integration into Application-Specific Integrated Circuit (ASIC) or FPGA designs use a variety of licensing models. A royalty-based model where the IP owner receives a payment each time the IP is instantiated in a product is vulnerable to abuse when the IP is used without paying royalties. Other threats include the use of obtained IP cores in other products without the knowledge/consent of the IP owner and leaking/reselling IP cores to third parties. All of these violate the legally stated rights of the IP owner.

Threat:

- Entity: (Illicit) system designers, but can also be a single rogue employee of a respected design house leaking/reselling confidential IP designs (industrial espionage).
- Budget: (Very) low.
- Goal: Use (excess) IP cores without paying royalties to the IP owner, make profit at the expense of violating the IP owner's rights.

Impact:

- The rights of the IP owner are violated. He loses revenue and possibly market share.

Risk:

- Medium. On the one hand, respected system designers will not risk losing their good name. On the other hand, this is an "easy" threat (low-overhead), and since detecting/proving IP-infringement is very expensive, the risk of getting caught is low.

Security Requirement:

- The security requirements to prevent/diminish the use of unlicensed IP-cores are twofold:
  - Active: The IP owner has active control over the number of instantiations of a particular IP core. This requires active involvement of the IP owner in the instantiation of every core.

- Passive: The IP owner has an easy/inexpensive way of detecting and proving IP-infringement, greatly increasing the risk of getting caught.

State-of-the-Art:

- Watermarking (passive).

2.3.2 CPLD/FPGA Configuration Data

Threat Type:

- Production/IP cloning.
- Active.

Use Case Definition:

Counterfeit or cloned systems typically copy the original system's FPGA and CPLD configuration data. The Programmable Logic Device (PLD) configuration data is very vulnerable to cloning since it packs a full-fledged design in a digital format (a few MBytes or less), which is stored in a non-volatile way on or near the PLD chip. A digital copy of this data implies a complete and identical clone of the original design. This is much easier than obtaining a clone of an ASIC.

Threat:

- Entity: (Higher level) system cloners/counterfeiters.
- Budget: Low/medium, depending on the particular devices used and possible precautions taken. Snooping the bus between a Flash chip containing an unencrypted configuration file and an FPGA can be done relatively easy with a low-end logic analyzer. Obtaining a copy of a configuration file stored inside the FPGA chip, or of an encrypted configuration file, is more difficult, but not insurmountable.
- Goal: Copying the functionality of the PLD chip, generally as part of the higher goal of copying the functionality of a complete system (see 2.1.5 and 2.2.4).

Impact:

- The IP owner loses revenue and market share. Using a PLD instead of an ASIC facilitates (and hence increases the risk of) higher-level system cloning.

Risk:

- Medium/High, depending on the availability and security of existing precautions such as bit stream encryption.

Security Requirement:

- A mechanism, which binds the PLD data to authentic hardware, is required, i.e., digitally copying the configuration data and loading it on an identical device should not result in a functional copy of the original PLD.

State-of-the-Art:

- Bit stream encryption and/or authentication. This has the disadvantage of having to store a device-specific secret key on the board/chip in a non-volatile way.
- IP Activation

2.3.3 Software

Threat Type:

- IP Cloning.
- Passive.

Use Case Definition:

Counterfeit or cloned systems typically copy the software associated with the system, e.g., host software, embedded software and firmware, which is usually provided in non-volatile memory built into the device or attached to the system. This soft-/firmware can be read out and used with counterfeit or cloned products or adapted for the use with different products with similar functionality (e.g., those of a competitor). In both cases the rights of the original IP owner of the soft-/firmware are violated since his IP is used in an unauthorized way.

Threat:

- Entity: (Higher level system) counterfeiters/cloners or competitors of the original soft-/firmware IP owner.
- Budget: Low to medium, depending on the storage mechanism used and the copy protection mechanisms applied.
- Goal: Copying the functionality of the soft-/firmware of a device or higher-level system for the use with counterfeit/cloned systems or other systems with similar functionality.

Impact:

- The rights of the IP owner are violated. Moreover, the IP owner loses revenue and possibly market share.

Risk:

- Medium to high, depending on the availability and security of existing precautions such as Digital Rights Management (DRM).

Security Requirement:

- A mechanism that binds authentic software to authentic hardware is required, i.e., that ensures that the software cannot be used with a different device or system.

State-of-the-Art:

- DRM or Trusted Computing techniques like Sealing. The disadvantages of these mechanisms are the requirement for each device to securely store a device-specific key and their rather high complexity (and thus unsuitability for many lightweight embedded devices such as RFID tags and wireless sensor nodes).

2.3.4 Malicious Circuits

Threat Type:

- Tampering.
- Active.

Use Case Definition:

A malicious circuit is covertly inserted into an IC (ASIC, Commercial Off-The-Shelf (COTS) or FPGA). The circuit is designed to result in a functional change to the IC under specific conditions. For example a compromised third-party Ethernet IP core, which dumps the contents of internal memory over the network on reception of a trigger packet, can be envisaged. A 'kill switch' might be introduced allowing for deactivation of the IC remotely or after a specific time period. Circuits might be introduced at any point in the design cycle and if the Electronic Design Automation (EDA) tool chain itself cannot be trusted then many attack vectors are possible. Minor changes such as a deliberate routing change introduced at the manufacturing stage could cause a data dependent fault on an arithmetic circuit. Although not strictly a malicious circuit, a targeted thinning of the IC's power grid could result in an early device failure due to electro migration. In general FPGAs are vulnerable to compromised configuration bit streams as a result of malicious third-party IP or EDA tool chain modifications. If the FPGA supply chain cannot be trusted then malicious design or manufacturing level modifications of the FPGA fabric itself cannot be ruled out.

Threat:

- Entity: Criminal organization/ government agency.

- Budget: High.
- Goal: Denial-of-service, theft of data, sabotage.

Impact:

- Risks to security, economies and infrastructure at the national and global level.

Risk:

- Low.

Security Requirement:

- A hardware attestation mechanism is required that can confirm or deny the presence of unauthorized circuitry in an IC.

State-of-the-Art:

- Approaches using measurable circuit characteristics such as path delays, power consumption (quiescent and transient) and electro-magnetic profiles which allow a 'fingerprint' for a known good sample of the device to be established. Formal verification techniques can be used to help rule out compromised synthesis tools by performing an RTL to gate level equivalence check (assuming the tool vendor is trusted).

2.3.5 Additional Information

In many of today's electronic products, embedded software plays an important role. Crucial functional parts of a device are implemented in hardware (i.e. in integrated circuits) but more and more higher layers (drivers, control, user interface, etc.) are implemented in embedded software or firmware that is running on an internal processor or microcontroller. This keeps the design flexible. Changing the software is easier and less costly than making new hardware. Furthermore it makes the design cycle shorter since bugs can easily be repaired in later software upgrades. Another advantage of this software flexibility is the possibility to use it for device differentiation. Instead of producing different hardware platforms for different products, a basic hardware platform is reused for multiple different products by implementing different software versions on top of it. This way a whole product line can be created from "basic functionality" to "fully featured" product, of which the latter version is sold for the highest price.

The product differentiation feature can however be abused by hackers. By copying software from the fully featured device and copying it onto the basic device (possibly after making some minor modifications to the software image), the basic device is turned into a fully featured product for free.

PUFs can be used to lock software to specific hardware (i.e. a specific device) and can therefore potentially solve this problem. Existing solutions also try to bind software to specific hardware by means of device identifiers, but the problem is that hackers often manage to spoof these identifiers. The unclonability feature of PUFs can bring a big advantage here. High-end vehicles already contain megabytes of code distributed over multiple controllers

The goal in this scenario is to provide a secure software hardware binding solution for embedded software based on PUF technology. Methods and cryptographic protocols need to be investigated that allow for securely anchoring software to a specific hardware device and prevent the possibility of illegal upgrading of devices. Processes around this technology also need to be investigated. For example how, where and in which part of the production process is the binding done? Who is controlling it and what is the impact on the production flow?

Besides purely binding software to a specific device, another goal is to hide useful information from such a software image to an attacker. For example by encrypting and signing it with a device specific PUF-based key. This will make it very difficult for the attacker to understand the contents and to modify the software in a sensible way.

Instead of embedded software, also content or data can be bound to the device (see use case example mentioned below). Binding content uniquely to a device can help in solving the issue of illegal content distribution. Consider a device (e.g. an MP3 player) that has several functional blocks implemented in the hardware of an IC. On the same IC a processor is integrated that runs (embedded) software, which provides the functionality to the outside of the IC. With this IC, two different products are sold. Product A only has basic functionality (e.g. playing MP3's, WAV's and other formats). The basic functionality is implemented by means of embedded software version A, which is stored in the EEPROM of the device. Product B is the high-end version of the device, which has a lot of extra features (e.g. equalizer, FM radio, recording) and is sold for a higher price. The features of product B are implemented by means of embedded software version B that is running on the same IC. The system needs to be designed such that copying the software version B to the EEPROM of product A does not lead to a working product.

A nice real-life example comes from a photo camera of Canon (see [56]) Canon introduced an entry-level digital camera (type SLR) that has lots of similarities with another much more expensive camera (type 10D) which is oriented for the professional market. The two types of cameras share for a great part the same hardware technology. On the cheaper model certain features are limited in the software. By the software images of both cameras, hackers found out that both cameras probably share the same code base and managed to circumvent the limitations of the cheaper model by modifying some bytes in the firmware image.

Note that in this case a literal copy of the firmware of the expensive model does not work on the cheaper model.

Consider a navigation system that is bought with a certain set of maps. New map data can be bought via the Internet and installed on the device. Illegally a lot of maps are distributed over the Internet. By small modifications in the software it is possible to use such illegal content on a device. This leads to a loss of revenue for the official providers of this content.

Navigation systems like TomTom have their own software running on a PC that is used to download new maps from the Internet (pay by credit card) and install it on the flash memory of the navigation device. However, hackers have published their own tools on the Internet, which make it possible to overwrite flash memory with maps also available on the Internet (see also [57]). This way the TomTom software is circumvented and the user can illegally install new maps on their device without paying for them.

2.3.6 Summary of Unlicensed IP Use

	Hardware IP Cores	PDL Configuration Data	Software	Malicious Circuits
Threat Type	Production/IP cloning, active	Production/IP cloning, active	IP Cloning, passive	Production/IP cloning, active
Threat Entity	System designer; single rogue employee	System cloners/counterfeiters	Counterfeiter/cloner; competitors of original soft/firmware IP owner	Criminal organization, government agency
Threat Budget	Low	Low/medium	Low to medium	High
Threat Goal	Use IP cores without paying; make profit	Copying the functionality of PLD chip	Copying the functionality of soft/firmware	Denial of service; theft of data; sabotage
Impact	Right of IP owner are violated; loss of revenue and market share	Loss of revenue and market share for IP owner	Right of IP owner are violated; loss of revenue and market share	Risk to security, economics and infrastructure at national and global level
Risk	Medium	Medium/high	Medium to high	Low
Security Requirement	Active: IP owner has active control over number of instantiations of a particular IP core; Passive: IP owner has easy way of detecting and proving IP-infringement	Mechanism, which binds PLD data to authentic hardware	Mechanism, which binds authentic software to authentic hardware	Hardware attestation mechanism
State-of-the-Art	Watermarking (passive)	Bit-stream encryption; authentication	DRM; Trusted Computing techniques	Approaches using measurable circuit characteristics

2.4 Cryptographic Attacks

2.4.1 Cryptographic Key Extraction

Threat Type:

- Active.

Use Case Definition:

The key store for a crypto device is vulnerable to attacks. Off-chip key stores can be simply read out. On-chip key stores in volatile or non-volatile memory can be revealed by invasive techniques (reverse engineering of memory content, micro-probing through Focused Ion Beam (FIB) contacts, scanning electronic microscopy) or non-invasive techniques (usage of an unintended design features, side channel attacks).

Threat:

- Entity: Governmental agencies/crime organizations.
- Budget: Unbounded (governmental agencies) or high (crime organizations).
- Goal: Discover strategic secrets (governmental agencies), create cash (e.g., think of a wide-scale counterfeit of banking smart cards).

Impact:

- Put countries, strategic or sensitive countries at risk.

Risk:

- Difficult to assess (governmental agencies), low up to now (crime organizations have been able to make much money with easier means).

Security Requirement:

- Cryptographic keys should not be stored in the clear.
- The key storage should be protected, e.g., encrypted with a PUF-based key, which is unique on a per-device basis.
- Devices should be protected against attempts to extract the key on a powered-up operational device (for example an invasive micro-probing attack), a key-masking scheme might be used or a coating PUF considered (i.e. the key never appears in clear in the device and cannot easily be rebuilt, the cryptographic algorithm uses a conjunction of the PUF-based key and of a derived cipher).

State-of-the-art:

- Zeroization of key material when an incursion into the cryptographic boundary is detected (FIPS-140 Level 4)

2.4.2 Loss of Cryptographic Keys

Threat Type:

- Active.

Use Case Definition:

Systems containing cryptographic secrets may fall into the wrong hands for example governmental/banking communication systems containing secret keys for authentication and decryption of messages.

Threat:

- Entity: Enemy, thief.
- Budget: Medium (stealing an IP cipher equipment, a payment terminal, etc.).
- Goal: Identity usurpation, secret data stealing, eavesdropping.

Impact:

- Secure communications or transactions are compromised.

Risk:

- High.

Security Requirement:

- A key zeroization mechanism is required. An encrypted key store using a PUF-based key poses a difficulty here. Reconfigurable PUFs have been proposed and could be used for key 'zeroization'. The reconfigurable PUF could have a "crypto-period" (i.e. it expires after a certain period of time if it receives no updates) in case the zeroization could not be triggered when the equipment was stolen.

State-of-the-art:

- Zeroization of key material on demand

2.4.3 Summary of Cryptographic Attacks

	Cryptographic Key Extraction	Loss of Cryptographic Keys
Threat Type	Active	Active
Threat Entity	Governmental agencies/crime organizations	Enemy/thief
Threat Budget	Unbounded	Medium
Threat Goal	Discover strategic secrets; generate cash	Identity usurpation; secret data stealing; eavesdropping
Impact	Put countries, strategic, or sensitive countries at risk	Secure communication or transactions are compromised
Risk	Low	High
Security Requirement	Store encrypted key securely	Key zeroization mechanism
State-of-the-Art	Key zeroization on tamper detection	Key zeroization on demand

3 Cryptographic Primitives

The aim of this chapter is to provide an overview of the most important cryptographic primitives and protocols, which are available in the field and used in practice. The level of detail is limited since for most of the mentioned subjects an abundance of information is available in literature. Sections 3.1 and Section 3.2 of this chapter are largely based on the more comprehensive document "D.SPA.7 - ECRYPT2 Yearly Report on Algorithms and Key sizes (2008-2009)" provided by the European Network of Excellence in Cryptology II (ECRYPT2) [1].
An even more extensive and formal discussion on the major cryptographic primitives, including security requirements, design methodologies and cryptanalytic techniques, can be found in the document "D20-v2 NESSIE security report", a deliverable of the European NESSIE project (EU FP5) [4].

3.1 Symmetric Primitives

Symmetric cryptographic primitives have as a common denominator that they all work under the assumption of a secret key, which can be shared between two or more parties. The only exceptions are hash functions, which are keyless. How such a key is generated and shared in a secure way is a topic on its own and will not be discussed here. The symmetric primitives, which are treated in this section, are respectively block ciphers, stream ciphers, hash functions and message authentication codes.

3.1.1 Block Ciphers

Definition and Functionality

A block cipher is a keyed, invertible transformation (permutation) on fixed-length blocks of bits. In general, it consists of a keyed encryption algorithm E_K and a keyed decryption algorithm D_K such that $D_K(E_K(x)) = x$, where x is a plain text block of n bits and K is the key of length k bits.
Block ciphers are generally used for encryption of large portions of digital data and hence are used to provide confidentiality. In its raw form, block ciphers do not provide integrity protection. To apply a block cipher on a large message text, the message should be chopped in blocks of size n and the block cipher should be used in a particular "mode of operation":

- Electronic Code Book Mode (ECB) encrypts every block of plaintext straightforwardly under the same key. Since this leaks information on the plaintext (e.g. two equal plaintext blocks have the same encryption), this mode is insecure and should hence only be used for messages that are at most the block length size.
- Cipher Block Chaining (CBC) is the most widely used mode of operation. It needs a random initialization vector (IV), which is XOR-ed with the plaintext block before encryption. The IV is updated for every block with the encryption

of the previous block, i.e., the continuous encryption of blocks literally chained. CBC can proven to be secure if the block cipher is secure. A disadvantage of CBC is that random-access decryption is not possible.

- Counter Mode (CTR) turns a block cipher into a stream cipher by encrypting successive values of a counter and XOR-ing them with the plaintext blocks. To be secure, the counter value should be padded with a random IV. No Key-IV pair may be reused. CTR can be proven secure if the block cipher is secure. CTR allows for random-access decryption.

Further modes are for example the Cipher-Feedback mode (CFB), the Output-Feedback Mode, or hybrid modes and we refer the interested reader for more details to [53].

Parameters and Security

In practice, block ciphers with block lengths n of 64-bit, 128-bit and 256-bit are used, and key lengths k are mostly equal to (or a multiple of) the block length. For present-day security, an equivalent key size of at least 80-bit is highly recommended.
Two important security notions of block ciphers are:

- No key-recovery attack of better complexity better than 2^k should be known/exist. Many cryptanalytical attacks aim at recovering the key of a block cipher in less effort than a brute-force search. Two well-known cryptanalytical methods are linear and differential cryptanalysis, but many more exist and new techniques are introduced regularly.
- It should be (very) hard to distinguish a block cipher from a random permutation.

Subdivisions of these security notions are made based on whether or not the adversary has (adaptive) control over the applied plain- and/or cipher texts.
It is important to note that generally, block ciphers, as most cryptographic primitives, are not provably secure. Their security is mostly based on a well-founded design methodology to thwart known cryptanalytical attacks.

Important Instantiations

- DES (Data Encryption Standard) is a 64-bit block cipher with a key-length of 56 bit and was for a long time the standard cipher for most applications. Due to its limited key length and susceptibility to a number of attacks, it is not secure anymore. National Institute of Standard and Technology (NIST) officially withdrew it in 2004. DES is widely deployed, e.g., in Internet Protocol Security (IPSec) and (Third Level Support) TLS.
- 3DES (Triple-DES) is a threefold iterative extension of DES, with a key-size of 112-bit or 168-bit and a block size of 64-bit. Due to its iterative construction and known attacks, 3DES does not offer full key-length security, but is still considered secure by NIST up to 2010 (for 112-bit keys) and 2030 (for 168-bit keys). 3DES is widely deployed, e.g., in many financial applications (112-bit) and IPSec and SSL/TLS (168-bit).

- AES (Advanced Encryption Standard) is the standardized (NIST) 128-bit version of the Rijndael block cipher. AES works on plaintext blocks of 128-bit and uses keys of 128-, 192-, or 256-bit. Since its selection and standardization by NIST in 2001, AES quickly became the most widely used 128-bit block cipher. AES is considered secure up to date, given its sufficient key-length and the fact that no better-than-brute-force attacks have been found on any full-round versions.
- Kasumi. 128-bit key and 64-bit blocks. Used in Universal Mobile Telecommunications System (UMTS).
- Blowfish. 32-448-bit key and 64-bit blocks. Used in IPSec.
- Present. 80-bit key and 64-bit blocks. Present is a lightweight block cipher for use in (hardware) constrained environments. Its current implementation records are
  - [1570GE - 200kbit/s - .18µm] and
  - [1000GE - 11.4kbit/s - .35µm],

  where records are shown as [size in gate equivalents – speed at a clock of 100kHz – technology node (standard cell Complementary Metal Oxide Semiconductor (CMOS))] [5].
- KATAN/KTANTAN. 80-bit key and 64- or 32-bit blocks. KATAN and KTANTAN are lightweight block ciphers for use in (hardware) constrained environments. Its current implementation records are [6]:
  - KATAN-64: [1054GE – 25.1kbit/s - .13µm]
  - KTANTAN-64: [688GE – 25.1kbit/s - .13µm]
  - KATAN-32: [802GE – 12.5kbit/s - .13µm]
  - KTANTAN-32: [462GE – 12.5kbit/s - .13µm]

3.1.2 Stream Ciphers

Definition and Functionality

A stream cipher is a keyed algorithm, which produces an arbitrarily long sequence called the key stream. A stream cipher is used for encryption by combining, usually XOR-ing, the key stream with a plain text to form a cipher text. Encryption with a stream cipher is used for confidentiality. In general, stream cipher do not offer integrity protection, e.g. in the case of XOR-ing with the key stream, flipping a bit in the cipher text will flip the corresponding bit in the plain text. Motivations to use a stream cipher for confidentiality protection instead of a block cipher are its possible higher speed with less computing power, the fact that it does not work on blocks of text, which can increase the bandwidth, and the fact that it does not propagate bit-errors.

Parameters and Security

The most obvious security requirement of a stream cipher is that its key stream should be indistinguishable from a random sequence. In this view, reusing a

key(stream) for encryption of a different plain text leads to a loss of all security and should hence be avoided at all cost.
It is important to note that in general, stream ciphers are less well studied than block ciphers, both in a design and an analysis perspective. To counter this, ECRYPT started the eStream initiative for the development of efficient and secure stream ciphers, both for hardware (80-bit key) and software (128-bit key) purposes [55]. After extensive peer-reviewing, a portfolio of the most efficient/secure candidates was presented.

Important Instantiations

- RC4 is a stream cipher with variable key size. It is widely used, e.g., in Secure Sockets Layer (SSL)/TLS and Wired Equivalent Privacy (WEP) for wireless LAN. It has several known weaknesses and its use is not recommended for general purposes.
- SNOW 2.0 has 128- and 256-bit keys and is used in Display Port. A known weakness reduces its security to 174 bits, but no practical attacks are known.
- eStream Portfolio (software):
  - HC-128
  - Rabbit
  - Salsa20/12
  - SOSEMANNUK
- eStream Portfolio (hardware):
  - Trivium (2599GE in .13µm)
  - Grain v1 (1294GE in .13µm)
  - MICKEY v2 (3600GE in .13µm)

3.1.3 Hash Functions

Definition and Functionality

In cryptography, hash functions are functions from arbitrarily long inputs or messages to fixed-size outputs called hash values or message digests. Requirements for a good hash function are:

- It is easy to evaluate the hash function on any given message.
- It is hard to find a message that has a given hash. This is called *pre-image resistance*.
- It is hard to find a message that has a given hash value, given a different message with that hash value. This is called *second pre-image resistance*.
- It is hard to find two messages with the same hash value. This is called *collision resistance*.

Cryptographic hash functions have all kinds of applications, ranging from integrity protection and checksums to the use in authentication and digital signature schemes, and are therefore sometimes called the "Swiss army knife of cryptography".

Parameters and Security

We denote the output length of a hash function by n. For a cryptographic hash function to be secure, it is required that finding a (second) pre-image needs around 2^n operations and finding a collision needs around $2^{n/2}$ operations due to the birthday attack. The hash value length n should be chosen large enough for these amounts of effort to be practically infeasible. Hash functions used in practice have an output length of at least 128-bits.
It is important to note some recent developments in the world of hash functions. In the last couple of years, the most widely used hash functions, i.e. Message Digest Algorithm 5 (MD5) and Secure Hash Algorithm-1 (SHA-1), have been broken. Although a number of other hash functions are still considered secure, NIST has started an open competition to select a new hash function for standardization comparable to the AES-competition. The SHA-3-competition is ongoing for the moment and 14 out of the 64 submitted functions have made it to the second round. An up-to-date overview of the status of this competition and lots of information about the submissions are given by the SHA-3 Zoo, an initiative of ECRYPT II [54].

Important Instantiations

- MD5 (Message-Digest-5) has a 128-bit output and is widely used, among others in SSL/TLS and IPsec. It is fully broken in the sense that collisions can be found within seconds. MD5 should not be used anymore, and existing applications using MD5 should, if possible, switch to another hash function.
- SHA-1 (Secure-Hash-Algorithm-1) has a 160-bit output and is widely used, among others in IKE and IPsec. SHA-1 shows several weaknesses, and although no practical collisions have been found yet, its security is believed to be very marginal. The use of SHA-1 for new applications is not recommended.
- SHA-224/256/382/512 (SHA-2) has an output length of respectively 224-, 256-, 382-, and 512-bit. It is not (yet) widely used. Although based on SHA-1, its design is quite different and SHA-2 is considered to be fully secure.
- RIPEMD-128/160 has an output length of respectively 128- and 160-bit. RIPEMD-160 is used among others in IPsec and OpenPGP. Both are based on the older RIPEMD algorithm, but the design is significantly different and RIPEMD-128/160 are considered fully secure.
- Whirlpool has a 512-bit hash output, is considered fully secure but is not widely deployed.
- SHA-3 round 2 candidates:
    - BLAKE
    - Blue Midnight Wish
    - CubeHash
    - ECHO
    - Fugue
    - Grøstl
    - Hamsi

- JH
- Keccak
- Luffa
- Shabal
- SHAvite-3
- SIMD
- Skein

3.1.4 Message Authentication Codes

Definition and Functionality

A message authentication code or MAC is a keyed primitive operating on a (arbitrarily long) message and producing a (fixed length) check-value or tag as output. The MAC output is usually appended to the message and its purpose is to provide integrity protection. For a MAC to be secure, it should be infeasible to:

- Recover the used key from a number of MAC outputs: *key recovery*.
- Produce a valid message-tag pair without knowledge of the used key: *insertion*.
- Modify an existing message-tag pair to obtain a new message-tag pair, which is valid under the used key: *modification*.

As mentioned earlier, encryption in general only offers confidentiality and no integrity of the message. Therefore, encryption is often used in combination with a MAC to provide integrity protection. Best practice is to first encrypt a message and to calculate the MAC on the cipher text.

Parameters and Security

For a MAC, we denote the output size as m and key size as n. Succeeding in forging a valid MAC without knowledge of the used key should only be possible with probability 2^{-m}. However, in practice the security level is often only $2^{-m/2}$.

Important Instantiations

- HMAC is a MAC-algorithm based on a hash function. It takes as parameters a hash function and a key. The maximal output length depends on the used hash function. The key size depends on the hash function. Since the internal hash functions are susceptible to the birthday attack, the maximal offered security is $2^{-m/2}$. The security of the HMAC-construction can be proven based on some assumptions of the underlying hash function. HMAC is widely used, e.g. in SSL/TLS and IPsec, especially in combination with the MD5 and SHA-1 hash functions. Note that, since MD5 and SHA-1 are considered broken, using HMAC with these hash functions is highly advised against!
- CBC-MAC-X9.19 uses a 112-bit key and produces 64-bit MAC outputs. It is widely deployed. A known weakness allows to this primitive to be broken when more than 2^{32} MAC operations are done under the same key. Use is not recommended for future applications unless frequent re-keying is used.

- CBC-MAC-EMAC is a MAC-algorithm based on a block cipher, in particular AES. The key lengths are the same as the AES key lengths and the tag length is at most 128-bit. The security level is 2^{-64} for MAC forgery and 2^{-n} for key recovery.
- CMAC is a MAC-algorithm based on a block cipher, typically AES. It takes as parameters a block cipher and a key. The maximal tag length is the size of a block of the block cipher, at least 64-bit is recommended. The probability of a successful forgery can be bounded by a function of the tag length and the number of authenticated blocks b: $b^2/2^{m-2}$. CMAC is becoming widespread and is the recommended AES-based MAC algorithm.

3.2 Asymmetric Primitives

3.2.1 Public Key Encryption

Definition and Functionality

A public key encryption scheme consists of three algorithms:

- A key generation algorithm G generating a public-private key pair (p,k).
- An encryption algorithm E which encrypts a message m under the public key p: $c = E_p(m)$.
- A decryption algorithm D which decrypts a cipher text c under the private key k: $m' = D_k(c)$.

If G generated (p,k), then $D_k(E_p(m)) = m$.
As most asymmetric primitives, a public key encryption scheme is based upon a hard mathematical problem, typically the hardness of factoring a large integer or the hardness of finding a discrete logarithm in a suitable group, but other problems are also possible. The first notion of public key encryption was introduced by the RSA cryptosystem and is based on the factoring problem. Public key encryption schemes based on the discrete log problem exist, but are not widely deployed, mainly due to their lack of efficiency.
The main difference with symmetric encryption is that two parties do not need to share a secret key in order to communicate confidentially. The public key can be made publicly available and anyone can use it to encrypt a message. However, only the party with access to the private key is able to decrypt. The major disadvantage lies in the fact that public key encryption requires a substantial amount of computation and can only be done for relatively short messages. To this end, hybrid encryption can offer a solution. In a hybrid encryption scheme, public key encryption is only used to transport a secret key, which can then be used with symmetric techniques to protect the full data stream. This is called *key encapsulation*. More information on hybrid techniques and key encapsulation methods is given in Section 3.2.4.

Parameters and Security

A number of security notions for asymmetric encryption exist, depending on the amount of control an adversary has in obtaining and/or choosing plain- and cipher texts. The strongest considered notion is called indistinguishability under adaptive chosen cipher text attacks. In short, this states that an adversary cannot link a message to a given encryption, even when he can query a decryption oracle for all but the given encryption.
In asymmetric primitives, the security level is mostly not equal to the length of the public or private key, but only a fraction thereof.

Important Instantiations

- RSA PKCS#1 v1.5 (Public Key Cryptography Standard (#1=RSA) v1.5) is a public key encryption standard based on RSA. It takes as parameters a (large) integer N, which is a product of two primes p and q, from which an encryption exponent e and a decryption exponent d can be deduced. The pair (e,N) serves as the public key, whereas d, or equivalently (p,q) serves as the private key. Although based on the hardness of factoring, the security of RSA PKCS#1 v1.5 could be less due to the lack of a security reduction. Known weaknesses include bad choices for p and q, and adaptive chosen cipher text attacks with decryption error feedback. When used correctly, RSA PKCS#1 v1.5 is considered secure at this moment for a choice of N of at least 1024-bits. For new applications, an N of at least 2432-bits is recommended. RSA PKCS#1 v1.5 is widely deployed, e.g., in TLS and S/MIME.
- RSA-OAEP (Optimal Asymmetric Encryption Padding) is a public key encryption standard based on RSA. It takes the same parameters as RSA PKCS#1 v1.5, and additionally a hash function and a mask-generating function. In contrast to RSA PKCS#1 v1.5, RSA-OAEP has a loose reduction to the factoring problem in the random oracle model. Again, caution is advised to prevent bad choices of the primes and to prevent feedback of decryption errors, since this may lead to attacks. The recommendation for the length of N is the same as for RSA PKCS#1 v1.5. The use of MD5 as a hash function is not recommended, and caution is advised for the use of SHA-1.
- ElGamal is a public key encryption cryptosystem based on the discrete log problem. More information and concrete schemes can be found in Section 3.2.4.

3.2.2 Digital Signatures

Definition and Functionality

A public key signature scheme consists of three algorithms:

- A key generation algorithm G generating a public-private key pair (p,k).
- A signing algorithm S, which can produce a signature σ on a message m under the private key k: $\sigma = S_k(m)$.

- A verification algorithm V which can decide whether a message-signature-pair is valid under a public key p: $V_p(m, \sigma)$ = valid/invalid.

If G generated (p,k), then $V_p(m, S_k(m))$ = valid.
Digital signatures are used to provide both authentication of the source of a message and message integrity. Note that only the owner of the private key can produce a signature on a message, but anyone with access to the public key can validate this signature and hence confirm that it came from the correct source. Also, altering the content of the message will with high probability render the signature invalid. The signature hence also provides an integrity check. Mind that both protections only hold if the verifier is assured that the public key he uses corresponds exactly to the private key used by the signer. Hence, the public key itself also needs to be authenticated. This is a problem in its own and will be further discussed in Section 3.2.5.
Since digital signatures can often only be placed on relatively short messages, they are generally combined with hash functions. A signature is then generated on the fixed-length hash output of the message instead.
For many asymmetric cryptosystems, public key encryption and digital signature generation are exactly dual, i.e., by interchanging the public and the private key one moves from encryption to signing and vice versa. However, it is highly recommended to use different key pairs for encryption and signing.

Parameters and Security

The most widely used security notion for digital signatures is *existential forgery resistance under adaptive chosen message attack*. In short, this means that an adversary cannot forge a signature on a given message, even if he can query a signing oracle for all but this message.

Important Instantiations

- RSA PKCS#1 v1.5, this is the digital signature part of the standard for which we already discussed the public key encryption part in Section 3.2.1. It is widely used, e.g., in TLS and S/MIME. The same cautions and recommendations with respect to the choice of parameters and parameter lengths hold as for the encryption part. It is recommended to use at least 160-bit hash functions, and 224-bit hash functions for future applications. The use of MD5 is strongly advised against. A public key exponent > 2^{16} is recommended. Since there is again no security reduction to the RSA problem, it is strongly recommended to use RSA-PSS instead.
- RSA-PSS (Probabilistic Signature Scheme) uses the same parameters as the signature scheme of PKCS#1 v1.5 and additionally a mask-generating function. The security of RSA-PSS is reduced to the RSA problem in the random oracle model. Recommendations for parameter choices and lengths are the same as for PKCS#1 v1.5.
- DSA (Digital Signature Algorithm) is a signature scheme based on the discrete log (ElGamal) problem instead of the RSA problem. The parameters are p, a

1024-bit prime and *q*, a 160-bit prime divisor of *p-1*. For future applications, larger lengths (at least 2432-bits and 224-bits) for respectively *p* and *q* are recommended. The standard specifies the use of SHA-1 as a hash function, but for new applications, a stronger hash function is advisable. The scheme in itself is considered to be secure, but care should be taken to prevent *misuse*, which could lead to weaknesses. DSA is widely deployed, e.g., in Internet Key Exchange (IKE), TLS and a number of other standards.

- ECDSA (Elliptic Curve DSA) is an elliptic curve variant of DSA. It is widely used, e.g., in SECG, IKE and TLS. ECDSA takes as parameters a subgroup over an elliptic curve defined on a prime or binary field and a key. The standard provides 4 curves on prime fields and 8 on binary fields, with key lengths of respectively 160-, 521-, 384- and 512-bits, but random curve generation is allowed. As for DSA, weaknesses can occur for badly chosen parameters, especially badly generated curves.

3.2.3 Public Key Authentication and Identification

Definition and Functionality

Authentication and/or identification is done through a protocol executed between a prover and a verifier. The goal of the prover is the convince any verifier that he is who he claims to be, usually by proving his knowledge of some secret data, which is in accordance with the verifier's public knowledge. The protocol is sound if nobody but the real prover can convince the verifier.

Parameters and Security

It is important to note that an authentication/identification protocol only assures the identity of the prover at the time of execution. To provide additional security, e.g., during a session, a trusted path should be in place.

Important Instantiations

- GQ (Guillou-Quisquater zero knowledge identification protocol).
- Fiat-Shamir.
- Schnorr.

3.2.4 Key Encapsulation Mechanisms

Definition and Functionality

As already mentioned earlier, public key encryption is in most cases only useful for the transport of a secret key, which is consequently used in symmetric primitives. This is called a *hybrid scheme*, and the public key encryption of the symmetric key is called a *key encapsulation mechanism* (KEM). In general, a KEM consists of a public key encryption scheme extended with a secure symmetric key generation procedure.

Parameters and Security

The security requirements and limitations of a KEM are basically the same as for public key encryption, extended with the security of the secret key derivation function (KDF).

Important Instantiations

- RSA-KEM a RSA based KEM. Tight reduction to RSA inversion in the random oracle model.
- ECIES-KEM an elliptic curve based KEM. Tight reduction to the decisional Diffie-Hellman problem in the random oracle model.
- PSEC-KEM elliptic curve based KEM. Tight reduction to the computational Diffie-Hellman problem in the random oracle model.
- ACE-KEM as PSEC-KEM extended with a hash function. Tight reduction to the decisional Diffie-Hellman problem in the standard model, with some assumptions on the hash function (2^{nd} pre-image resistance) and the KDF.

3.2.5 Key Agreement and Key Distribution

Definition and Functionality

Key agreement and distribution can be done with symmetric primitives (e.g., Kerberos), through a hybrid method (see KEM in Section 3.2.4) or in an asymmetric way, based on the Diffie-Hellman (DH) protocol. DH key agreement is widely used, e.g., in SSL/TLS.

Parameters and Security

It is very important to note that standard „opportunistic" DH key agreement is susceptible to active man-in-the-middle attacks. Hence, the exchanged messages in the DH-protocol need to be authenticated, e.g., using digital signatures. DH-key agreement offers perfect forward secrecy.

Important Instantiations

- Internet Key Exchange.

4 Hardware Building Blocks

This chapter covers hardware-level technologies and design techniques.

4.1 Memory Considerations

The selection of a memory technology for an application may have security implications. This section explores some of the issues and mitigations.

4.1.1 Mask ROMs

The contents of a mask read-only memory (ROM) are defined at the manufacturing stage by the mask set and are fixed. For high volume, low cost applications requiring on-chip ROM the superior bit density of mask ROM is attractive. As a typical example microcontroller manufacturers may offer a choice of flash or mask ROM for program memory, the latter being used when the software is production ready.
When considering invasive attacks on the mask ROM, optical read out of the memory must be considered. In this type of attack the contents of the ROM are inferred from the physical structure of the memory array. Resistance to optical read out ranges from very low for metal layer programming to high for implantation programming, where the contents are defined by the doping implants used during fabrication. In this case there is no visible difference between set and cleared bits in the memory structure. A middle ground exists with contact layer programmed mask ROMs where the absence or presence of a via defines the contents. Exposing the via for optical read out requires further die deprocessing for modern deep submicron processes using Certificate Management Protocol (CMP) planarization. This increases the difficultly level for the attacker [29].
Note that dopant-selective cryptographic etches exist [30] which can reveal the doping levels in an implantation programmed mask ROM, as well as a variant of scanning tunneling microscopy (STM) called scanning capacitance force microscopy (SCFM) which can perform the same function [31].

4.1.2 Antifuse ROMs

Antifuse technology such as that used to implement non-volatile configuration memory in some FPGAs uses a one-time programmable via to connect adjacent metal layers. In particular, the Vialink antifuse used by Quicklogic can be used to build one-time programmable (OTP) ROM using standard ASIC processes, with some additional processing steps [32]. For current process nodes the resistance of such a ROM to optical read out is good due to the difficulty of resolving the fused vias [33].

4.1.3 EEPROM / Flash

Electrically Erasable Programmable ROM (EEPROM) and Flash technologies use a floating gate, which makes optical readout difficult since the gate charge cannot be detected directly. In the context of an invasive attack it may possible to manipulate

the contents of these memories using targeted ultra-violet (UV) light. Such an attack would proceed in a similar fashion to the early attacks on UV EEPROMs [40]. To increase the difficulty level for the attacker a top metal layer covering the memory region should be used.

4.1.4 SRAM

Although a volatile memory technology, Static Random Access Memory (SRAMs) is subject low temperature data remanence [37], radiation imprinting [39] and aging related data remanence such as electro migration [36]. An additional high voltage imprinting mechanism using high voltage transients is referred to in [39]. These phenomena can allow data recovery from SRAM in the powered down state.
SRAM data remanance attacks are typically countered by monitoring the parameters of interest (temperature, radiation flux and/or dose, supply voltage) and removing SRAM power on detection of an excursion outside the acceptable limits, in order to achieve data erasure. SRAM power down is achieved by 'crowbarring' the supply with a low impedance path. Alternatively data zeroisation can be employed. Data remanence due to aging effects can be mitigated by relocating sensitive data periodically.
Battery backed SRAM supported by tamper detection circuits provide a secure non-volatile alternative to Flash or EEPROM when the cost, complexity and reliability implications can be justified.

4.1.5 Security Fuses

Many microcontrollers, non-volatile FPGAs and CPLDs prevent read back of internal non-volatile memory (Flash, EEPROM) by means of security fuses. The location of the security fuses on the die should be carefully considered. In order to make the fuses difficult to locate during an invasive attack, they should be embedded inside the memory array and ideally multiple fuses should be used to raise the difficulty level for the attacker. Monitoring the fuses constantly or each time a memory access is required is preferable to a one-time check at device power-up. An informative treatment of the evolution of security fuses in microcontrollers is to be found in [29].
UV attacks against security fuses need to be considered and a top metal layer shield covering of the fuses may be considered. The fuse logic should be inverted such that a UV attack disables the read back capability rather than enabling it.

4.1.6 Resistance to Energy Probes

As an alternative to mechanical micro probing a large number of probing techniques based on electron, ion or photon beams exist, which allow the state of a CMOS transistor to be read or manipulated. A low cost technique to read and manipulate the contents of a powered SRAM is presented in [41]. It sees reasonable to assume that non-volatile memories, registers and sequential elements can be similarly accessed assuming their positions can be located. It should be noted that top layer

metal shields might be transparent to attacks using wavelengths outside the optical range.

4.2 Tamper Detection

4.2.1 Top Layer Sensor Meshes

A sensor mesh of power, ground and sense lines implemented in the top metal layer can detect micro probing attacks as well as impede optical attacks. The assumption here is that the mesh monitoring circuitry is continuously powered and erases non-volatile memory when the mesh is triggered [49]. In many situations this is not practical and the mesh is only monitored when the device is operational. This opens up a significant window of opportunity for the attacker when the device is unpowered.

FIB attacks on the mesh itself or on the monitoring circuitry should be considered. The beam resolution of a FIB (a typical value is 5 nm) is sufficiently low to bypass meshes implemented in current technologies (ITRS 2010 ½ metal pitch is 45 nm [58]). Backside FIB attacks, where access to silicon structures is obtained through the die substrate, can render a top layer mesh ineffective.

4.2.2 Power Supply Monitors

Under- or overvoltage attacks can induce fault conditions within the device. To protect against these a voltage monitoring circuit may be used to issue a tamper event when the power supply is outside acceptable limits. Ideally the circuit should be sensitive to transients as well as longer lived events.

4.2.3 Temperature Monitors

Temperature monitors can detect attempts to induce faults in the device by issuing a tamper event when the operating temperature is outside acceptable limits. In particular SRAM low temperature data remanence can be mitigated by removing the SRAM power supply when a low temperature event is detected. High temperature should also trigger a tamper event since aging effects such as electro migration increase with temperature.

4.2.4 Clock Monitors

Clock monitors issue a tamper event when the clock frequency goes outside acceptable limits in order to detect attempts to stop the clock (to single step) or induce a fault by overclocking. Clock transients (glitches) should also be detected (if possible) as they have been used as the basis of successful attacks on smartcards [44].
Where possible an external clock should not be used directly. A derived clock is preferable for example by using a Phase-Locked Loop (PLL). The response of the

PLL or clock conditioning circuit to abnormal clock inputs, power supply and environmental inputs should be considered.

4.2.5 Light Sensors

Attempts to decapsulate the chip can be detected by a light sensor and erasing sensitive material by 'crowbarring' the SRAM supply or activating circuitry to catastrophically damage the device (for example by introducing a short on the power supplies and fusing the power grid).

4.2.6 Radiation Monitors

In order to defend against data remanence attacks on SRAM by radiation imprinting [39], some form of radiation sensor can be considered. Two radiation parameters are of interest, incident flux and total cumulative dose. The total cumulative dose is important since low levels of radiation over long time periods may be sufficient to imprint data.
Flux sensors based on phototransistors allow a low cost implementation. Dosage sensors of low cost and power are not currently available [39]. A dosimeter method based on 6T CMOS SRAM cells is presented in [38].

4.3 Design Techniques

4.3.1 Dynamic Logic

An attacker may have interest in memory contents, which are only present for brief instants during the operation of a device. In this case stopping or slowing the clock can aid an attack, for example reading SRAM contents using the optical probing technique discussed in [41]. Dynamic logic may avert such an attack since the state will be lost without a periodic refresh. The response of dynamic logic to low temperature events should be considered, as it is likely that it behaves in a similar fashion to Dynamic Random Access Memory (DRAM) which has been shown to exhibit low temperature data remanence.

4.3.2 Asynchronous Logic

Asynchronous logic may be less susceptible to power analysis attacks. Dual rail encoding techniques can make fault injection attacks difficult, as state changes require two storage elements to be simultaneously switched. An asynchronous design technique with fault injection resistant features is presented in [48].

4.3.3 Bus Encryption

Bus encryption has been used in an attempt to render micro probing attacks on on-chip buses ineffective. In such cases code and data is only present in plaintext form within the processor. The performance implications of such an approach imply that lightweight encryption algorithms (which may be proprietary) must be used. A

practical attack on bus encryption involving a secure microcontroller is demonstrated in [35]. Simpler attacks based on directly probing (mechanically or via energy beam) the processor registers can be envisaged and would render bus encryption ineffective.

4.3.4 Flat Chip Layouts

Hierarchical layouts give the attacker useful cues as to the location of functional units on the chip (for example the Arithmetic Logic Unit (ALU) or instruction decoder of a CPU). Using a flat approach makes finding particular nodes on the chip surface difficult for micro probing or fault induction.

4.4 Debug and Test Structures

The security implications of circuit structures intended for debug and test must be carefully considered. In a typical design scan chains will allow most if not all circuit nodes to be measured and manipulated for manufacturing test purposes. The scan chains are typically accessible on the device's IO pins. This makes for a potent side channel [47]. A case study demonstrating cryptographic key extraction over scan chains is presented in [45] and a DES attack in [47]. All debug and test port, modes and Build-In-Self-Test (BIST) functions pose security risks and require careful analysis. Consideration should be given to physically disabling such features after manufacturing test.

4.5 Process Technology

4.5.1 Targeted Technology Node

Smaller feature sizes and more metal layers make the attacker's job more difficult. As an example mechanical micro probing becomes more difficult at smaller feature sizes necessitating the use of deposited probe pads or more complex energy probing techniques. Additionally, the increased transistor budget allows more security features to be added.

4.5.2 Substrate Considerations

Optical inspection of the chip surface becomes difficult with deep submicron technology nodes due to decreasing feature sizes, CMP planarization and increased metal layer counts. As a result optical backside imaging techniques have been developed which exploit the fact that bulk silicon is almost transparent for IR wavelengths [42]. The choice of substrate can hinder optical backside imaging. Silicon on insulator technologies, which are opaque at the wavelengths of interest, as well as silicon substrates with high doping concentrations [43] are of interest. It should be noted that such defenses could be countered by substrate thinning using precision grinding techniques.

4.5.3 Protective Coatings and Passivation Layers

Coatings exist which when removed damage the die surface or damage the chip through a buildup of electrical charge when a FIB is used (silicon carbide or boron nitride) [46].

4.6 Higher Level Assemblies

In the context of the UNIQUE project, "higher level assemblies" refer to sub-systems that include integrated circuit(s). These can be smart cards, packaged ICs (including systems-in-package), PCBs, modules, casings and equipments.
In high security grade applications, the protection and the erasure of sensitive data often requires a second protection level above the ICs. For instance, in a point of sales terminal, an authorized opening of the device will surely provoke the erasure of some sensitive credentials stored in a chip.
Although the UNIQUE project is mostly IC-centric, this chapter provides an outlook of additional protection techniques commonly used in secure systems.

4.6.1 Power Supplies

Erasing data requires energy. Therefore permanent energy sources are necessary to protect a device when the main supply is disconnected. Disposable or rechargeable batteries or super-capacitors are used. Recent advances in the battery domain, such as thin batteries [50], fuel the innovation in this domain and bring new integration possibilities.

4.6.2 Systems-in-Packages

Systems-in-Packages (SiP) are packaged ICs that comprise several stacked dice. It is common to find a microprocessor, a Flash memory and a Random Access Memory (RAM) in a single IC on board mobile phones to achieve a higher integration degree. The R&D community is very active in this domain, mainly driven by the mobile consumer market.
Such SiP can also increase the security level, for instance by making the memory bus probing more difficult. Future developments could embed thin-film batteries for emergency erasing.
SiP open up the possibility of a system which is resistant to invasive backside attacks which are performed through the silicon substrate and can bypass many security features on the front side of the die. By orienting the stacked dice such that only the front sides are exposed on the top and bottom of the stack, backside attacks are prevented. The frontsides can be protected using the top metal shielding techniques discussed above.

4.6.3 Shielding and Casing

Similarly to the IC level, some high security grade applications will require shielding at the board level. These shields have several functions:

- Limit the eavesdropping possibilities.
- Embed sensors to detect eavesdropping and opening attempts.
- Limit the access to the inner circuitry, e.g., the voltage regulators, to render some attacks more difficult (e.g., side channel attacks and perturbation attacks).

The shield will also surely comprise an energy source for the emergency erasure.
There are many types of shields, from boxes geared with opening contacts to more sophisticated shields that detect the mere drilling attempts [51]. Some intermediate solutions can comprise parts of the shield embedded in the PCB inner layers.
Finally, these assemblies can be sealed with specific adhesives to make tampering attempts evident.

4.6.4 Sensors

Sensors are used to detect eavesdropping or openings and to trigger emergency erasures. They complement the IC-level sensors. They can be single electric switches, light detectors, temperature range detectors, voltage range detectors, X-ray and other kinds of radiation detectors.
As evocated above, these sensors can also be webs of conductive materials embedded in the shielding that correlate variations of electric characteristics to drilling and cutting attempts.

5 PUF Building Blocks

This section gives an overview of existing technologies and building blocks based on Physical Unclonable Functions. The scope is limited to electronic PUFs that can be used on ICs.

5.1 Different Types of PUFs

A Physical Unclonable Function is a physical structure embedded in an IC that is very hard to clone on the physical level due to its unique micro- or nano-scale properties that originate from inherent deep-submicron manufacturing process variations. Device unique characteristics of a PUF are measured by providing a challenge to the structure (i.e., a certain stimulus) and reading out the corresponding response. Different types of PUFs are known from literature. This section gives an overview of them.

5.1.1 SRAM PUF

- Description: In order to acquire a random string of bits for IP protection within ICs, uninitialized SRAM memories can be used. These uninitialized memories have unpredictable start-up behavior, during which the value of an SRAM cell can become either 0 or 1. Tests have shown that this behavior is random between different cells (due to manufacturing variation that cannot be controlled), but robust for a single cell. Combining these properties makes uninitialized SRAM suitable for use as a PUF. See [9].
- Platform: The SRAM PUF requires uninitialized SRAM, which makes it unsuitable for most types of FPGAs currently available. Suitable for use on ASICs. Available as a standard component in every ASIC technology node and therefore widely applicable.
- Maturity: Implemented and tested on FPGA platform, see [9]. Intrinsic-ID has performed measurements on various ASIC platforms (from different technology nodes and foundries) confirming its robustness and applicability in these nodes. Furthermore a security evaluation against invasive attacks has been performed. Publication of experimental results is pending.

5.1.2 Butterfly PUF

- Description: The butterfly PUF has been derived from the SRAM PUF. In case of butterfly PUFs, the SRAM has been replaced by cross-coupled latches to construct an unstable cell, which has a start-up value of either 0 or 1 based on variations in the production process. See [10].
- Platform: Suitable for both FPGA and ASIC platforms.
- Maturity: Implemented and tested on FPGA platform, see publication in [10].

5.1.3 D-flip-flop PUF

- Description: Flip-flop PUFs [11] are based on the power-up characteristic of (uninitialized) D-flip-flops. Due to uncontrolled process variations, each flip-flop will have a bias to switch its output to either the 0 state or the 1 state when the IC is powered up.
- Platform: Suitable for both FPGA and ASIC platforms. Available in standard cell libraries of most IC technology processes.
- Maturity: Implemented and tested on FPGA platform, see publication in [11].

5.1.4 Arbiter PUF

- Description: In an arbiter PUF two delay paths are excited simultaneously, which will make two transitions race against each other through their respective paths. At the end of both paths an arbiter awaits to determine which of the two rising edges arrives first. Based on which is first the arbiter will set its output to 0 or 1. Besides this 1 bit output, the circuit has an *n*-bit input (challenge) to configure the delay paths. See [1] and [8].
- Platform: Suitable for use on both FPGA and ASIC.
- Maturity: Implemented and tested on both platforms. Experimental results have been published in [1] and [8].

5.1.5 Ring Oscillator PUF

- Description: A PUF circuit comprised of many identically laid-out delay loops (ring oscillators), which oscillate with a particular frequency. Due to manufacturing variation each ring oscillates at a slightly different frequency. In order to generate an output bit, two rings are selected and their frequencies compared. A *k*-bit output can be created by selecting *k* different oscillator pairs. See [8].
- Platform: Suitable for use on both ASIC and FPGA platforms.
- Maturity: Implemented and tested on FPGA platform. Experimental results have been published in [8].

5.1.6 Crossbar Memory PUF

- Description: A crossbar memory [12] is a multi-layered (nano) structure that consists of: (1) a layer of parallel word lines, (2) a layer of high-k dielectric material, and (3) a layer of parallel bit lines that are directed perpendicular to the word lines of layer 1. This creates a uniquely addressable cell at each junction of word and bit lines. Effectively this structure implements an array of diodes with unique current-voltage characteristics.
- Platform: Suitable for implementation on ASICs.
- Maturity: TODO.

5.1.7 Coating PUF

- Description: A Coating PUF consists of a coating with random dielectric particles that forms a protective top layer of an IC [13][15]. The upper metal layer of the IC contains metal sensor structures for reading out local variations in coating capacitance. A signal processing algorithm on the IC transforms the capacitance readings into a binary PUF response.
- Platform: Suitable for use on ASICs.
- Maturity: Implemented and tested on ASIC, see publication in [15].

5.1.8 Optical PUF

- Description: Optical PUFs are one of the earliest forms of PUFs that are described in the literature [16]. An optical PUF consists of a transparent material with light scattering particles that is illuminated with a laser beam. The resulting speckle pattern is processed (usually with Gabor transforms and quantization algorithms) to form a binary PUF response [17]. Different challenges are obtained by changing the angle of incidence or the wavelength of the laser beam. A so-called "integrated PUF" is a miniaturized version of the optical PUF, which can be implemented on an IC [13].
- Platform: Integrated optical PUFs can be implemented on a chip, but require non-standard components.
- Maturity: No single-chip implementations known.

5.2 Constructions and Mechanisms Using PUFs

Physical Unclonable Functions can be used to construct various security mechanisms. This section gives an overview of known mechanisms.

5.2.1 Secure Key Storage

Cryptographic keys can be securely stored using PUFs [14][15][9][28]. Instead of storing a cryptographic key in non-volatile memory, the key is reconstructed from a measured PUF response and helper data stored in non-volatile memory. The device-unique helper data does not reveal information about the key (in an information theoretic sense), under the assumption that the PUF response is not known.

5.2.2 Device Authentication

Using the challenge response mechanism of PUFs, devices or ICs can be authenticated uniquely [8][16][15]. During an enrolment phase, a database of challenge-response pairs is stored for each of the devices that need to be authenticated. Later on, in the authentication phase, the authenticator sends a challenge to the device. The device challenges its PUF and sends back an answer that is based on the PUF response. The authenticator verifies the correctness of this answer by using the response that is stored in its database. Note that each challenge-response pair can only be used once to prevent replay attacks.

5.2.3 Secure Activation

With a similar mechanism as used for key reconstruction (see Section 5.2.1), a secure product activation system or feature activation system can be implemented. In this case helper data is used as an activation code [19][20][9]. The helper data is chosen in such a way that a certain pre-defined key is reconstructed on a specific device. A hardware or software module inside the IC of the device compares the reconstructed key (based on a PUF measurement) with the pre-defined key and enables the design or certain features of the design if both keys match. Since the activation code (helper data) is device unique, it will not lead to a successful activation when it is copied to a second device.

5.2.4 Reconfigurable PUF

A reconfigurable PUF is a PUF that has a reconfiguration mechanism, which makes it possible to change the PUF into a new one with unpredictable challenge-response behavior. Reconfigurable PUFs were introduced by Lim [22], however his implementation (arbiter PUF with additional floating gate transistors for reconfiguration) is not ideal since it does not exclude the possibility to return to a previous configuration. More secure implementations that guarantee only one-way reconfigurations are suggested in [21] by using optical PUFs or phase-change memory. Actual implementations of such reconfigurable PUFs do not exist.

5.2.5 Controlled PUF

Controlled PUFs (CPUFs) are PUFs that can only be accessed via an algorithm that is physically bound to the PUF in an inseparable way. This algorithm can restrict the challenges, which are presented to the PUF, and can also limit the information about responses, that is given to the outside world. The security of CPUFs relies on computational complexity. Among the types of attacks, that can be prevented using CPUFs, are "chosen challenge" and "man-in-the-middle" attacks. Furthermore, a CPUF can be used to create multiple personalities within a single CPUF in order to enhance privacy of the user by limiting external tracking possibilities. Examples of CPUF applications are certified execution (to make sure that a computation was carried out on a specific processor chip with certified results), smartcard authentication and possibly software binding. See [23].

5.2.6 Hardware Entangled Cryptography

Recently a new PUF based cryptographic primitive called the PUF-PRF (PUF-based pseudo random function) was introduced in [27]. This primitive is used to construct a block cipher that offers protection against both algorithmic and physical attackers.

5.2.7 Random Number Generator

Instead of using PUFs for "normal" operations, the random number generator focuses on meta-stable PUF challenges. These meta-stable challenges are

challenges, which do not return a consistent, but rather an unpredictable (perhaps even random), response. In a random number generator, these meta-stable responses (in combination with post-processing) are used to create random numbers. Analysis in [24] suggests that PUF-based random number generators are a cheap and viable alternative to more and complicated hardware random number generation. Examples of random number generator applications are randomized algorithms and cryptographic applications (e.g., for generating keys or padding purposes). See [24].

5.2.8 Software Binding

In [25], a flexible design flow is proposed for binding software intellectual property to specific hardware. This methodology consists of two parts, one before and one after delivery to the end-user. Before delivery to the end-user, an FPGA-unique key is extracted using the PUF in an enrolment process. This key is used to encrypt the software intellectual property. After delivery to the end-user, when the FPGA boots up, a security kernel extracts the PUF-based key from the FPGA in order to decrypt the encrypted software intellectual property. Unfortunately, according to [26] it is easy to show that this mechanism can be bypassed by running the software in a virtual machine that has been primed to behave as the PUF would.
Another approach to binding SW that is running on processors configured on an FPGA is presented in [9]. The described protocols make use of a trusted third party that keeps a database of challenge-response pairs for each device.

6 References

[1] http://www.sia-online.org/cs/anticounterfeiting
[2] http://www.cosic.esat.kuleuven.be/publications/article-1252.pdf
[3] http://www.ecrypt.eu.org/documents/D.SPA.7.pdf
[4] https://www.cosic.esat.kuleuven.be/nessie/deliverables/D20-v2.pdf
[5] http://www.ecrypt.eu.org/lightweight/index.php/Block_ciphers
[6] http://www.ecrypt.eu.org/lightweight/index.php/Block_ciphers
[7] J. W. Lee, D. Lim, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas, "A Technique to Build a Secret Key in Integrated Circuits for Identification and Authentication Applications", Proceedings of the IEEE VLSI Circuits Symposium, pp 176-179, 2004.
[8] G. E. Suh and S. Devadas, "Physical Unclonable Functions for Device Authentication and Secret Key Generation", Proceedings of the 44th Design Automation Conference, DAC, pp 9-14, 2007.
[9] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls, "FPGA Intrinsic PUFs and Their Use for IP Protection", Cryptographic Hardware and Embedded Systems – CHES, 2007, pp 63-80, 2007.
[10] S. S. Kumar, J. Guajardo, R. Maes, G.-J. Schrijen, and P. Tuyls, "The Butterfly PUF: Protecting IP on every FPGA", IEEE International Workshop on Hardware-Oriented Security and Trust - HOST, 2008, pages 67-70, 2008.
[11] R. Maes, P. Tuyls, and I. Verbauwhede, "Intrinsic PUFs from Flip-flops on Reconfigurable Devices", 3rd Benelux Workshop on Information and System Security - WISSec, 2008
[12] T. Jun, "Circuit Approaches to Physical Cryptography", Diploma Thesis Technische Universitaet Muenchen.
[13] B. Skoric, G.-J. Schrijen, W. Ophey, R. Wolters, N. Verhaegh, and J. van Geloven, "Experimental hardware for coating PUFs and optical PUFs", Security with Noisy Data - On Private Biometrics, Secure Key Storage and Anti-Counterfeiting, pp 255-268, 2007.
[14] P. Tuyls, G.J. Schrijen, F. Willems, T. Ignatenko, "Secure Key Storage with PUFs", Security with Noisy Data - On Private Biometrics, Secure Key Storage and Anti-Counterfeiting, pp 269-292, 2007.
[15] P. Tuyls, G.-J. Schrijen, B. Skoric, J. van Geloven, N. Verhaegh and R. Wolters, "Read-proof hardware from protective coatings", Cryptographic Hardware and Embedded Systems - CHES, p 369- 383, 2006.
[16] R.S. Pappu, "Physical one-way functions", PhD. Thesis, Massachusetts Institute of Technology, March 2001.
[17] B. Skoric, P. Tuyls, W. Ophey, "Robust key extraction from Physical Uncloneable Functions", Applied Cryptography and Network Security (ACNS) 2005, pp 407-422.
[18] P. Tuyls, B. Skoric, S. Stallinga, A.H.M. Akkermans, W. Ophey, "Information-theoretic security analysis of Physical Uncloneable Functions", Financial Cryptography and Data Security - FC, pp 141-155. Springer, 2005.

[19] J. Guajardo, S.S. Kumar, G.J. Schrijen, P. Tuyls, "Brand and IP protection with Physical Unclonable Functions", In IEEE International Symposium on Circuits and Systems – ISCAS, May 18-21, 2008.

[20] J. Guajardo, B. Skoric, S.S. Kumar, T. Bel, A.H.M. Blom, G.J. Schrijen, "Anti-counterfeiting, Key Distribution and Key Storage in an Ambient World via Physical Unclonable Functions", Information System Frontiers, pp 19-41, 2009.

[21] K. Kursawe, A.R. Sadeghi, D. Schellekens, B. Skoric and P. Tuyls, "Reconfigurable Physical Unclonable Functions – Enabling Technology for Tamper-Resistant Storage", IEEE International Workshop on Hardware-Oriented Security and Trust - HOST, 2009.

[22] D. Lim, "Extracting Secret Keys from Integrated Circuits," Master's thesis, Massachusetts Institute of Technology, May 2004.

[23] B. Gassend, D. E. Clarke, M. van Dijk, and S. Devadas, "Controlled Physical Random Functions", 18th Annual Computer Security Applications Conference - ACSAC, pp 149-160, 2002.

[24] C. W. O'Donnell, G. E. Suh, and S. Devadas, "PUF-Based Random Number Generation", *CSAIL CSG Technical Memo 481*, Massachusetts Institute of Technology, 2001.

[25] M. A. Gora, A. Maiti, and P. Schaumont, "A Flexible Design Flow for Software IP Binding in Commodity FPGA", IEEE Fourth International Symposium on Industrial Embedded Systems – SIES, pp 211-218, 2009.

[26] H. Busch, M. Sotakova, S. Katzenbeisser, R. Sion, "The PUF Promise", Trust and Trustworthy Computing, 2010.

[27] F. Armknecht, R. Maes, A.R. Sadeghi, B. Sunar, P. Tuyls, "Memory Leakage-Resilient Encryption Based on Physically Unclonable Functions", Asiacrypt, pp.685-702, 2009.

[28] P. Tuyls, B. Skoric, "Secret Key Generation from Classical Physics: Physical Unclonable Functions", Amiware: Hardware Technology Drivers of Ambient Intelligence, Philips Research Book Series, Volume 5, Springer 2006.

[29] S. Skorobogatov, "Semi-invasive attacks - a new approach to hardware security analysis," Technical Report UCAM-CL-TR-630, University of Cambridge, Computer Laboratory, pp. 28-30, April 2005.

[30] O. Kömmerling, M. G. Kuhn, "Design Principles for Tamper-Resistant Smartcard Processors", USENIX Workshop on Smartcard Technology, 1999.

[31] C.C. Williams, Two-Dimensional Dopant Profiling by Scanning Capacitance Microscopy, Annual Review of Material Science, 1999, Vol. 29, pp. 471–504

[32] V. J. Coli; J. Lipman, "pFSB technology enables field programmability in an ASIC environment," *Compcon Spring '93, Digest of Papers*, pp.385-389, 1993.

[33] S. Skorobogatov, "Semi-invasive attacks - a new approach to hardware security analysis," Technical Report UCAM-CL-TR-630, University of Cambridge, Computer Laboratory, p. 28, April 2005.

[34] S. Moore, R. Anderson, R. Mullins, G. Taylor, J. Fournier, "Balanced Self-Checking Asynchronous Logic for Smart Card Applications", Microprocessors and Microsystems Journal, 27(9), pp 421-430, 2003.
[35] M. Kuhn, "Cipher Instruction Search Attack on the Bus-Encryption Security Microcontroller DS5002FP," IEEE Trans. Computing, pp. 1153–1157, 1998.
[36] P. Gutmann, "Data Remanence in Semiconductor Devices", 10th USENIX Security Symposium, 2001.
[37] S. Skorobogatov, "Low Temperature Data Remanence in Static RAM", Technical Report UCAM-CL-TR-536, University of Cambridge, Computer Laboratory, 2002.
[38] L-Z. Scheick, G. M. Swift, "Dose and microdose measurement based on threshold shifts in MOSFET arrays in commercial SRAMs Nuclear Science", IEEE Transactions on , pp 2810 -2817, 2002.
[39] S. Weingart, "Physical Security Devices for Computer Subsystems: A Survey of Attacks and Defenses", CHES 2000, Springer LNCS 1965, pp. 302-317
[40] S. Skorobogatov, "Semi-invasive attacks - a new approach to hardware security analysis," Technical Report UCAM-CL-TR-630, University of Cambridge, Computer Laboratory, pp. 89, 2005.
[41] D. Samyde, S. Skorobogatov, R. Anderson, J.-J. Quisquater, "On a New Way to Read Data from Memory – SISW2002", First International IEEE Security in Storage Workshop, 2002.
[42] T. Eiles, G. Woods, V. Rao, "Optical probing of flip-chip-packaged microprocessors", Solid-State Circuits Conference, ISSCC, pp 220-221, 2000.
[43] R. A. Falk, "Near IR Absorption in Heavily Doped Silicon – An Empirical Approach", (ISTFA) International Symposium for Testing and Failure Analysis, 2000.
[44] Ross J. Anderson, Markus G. Kuhn, Tamper Resistance – a Cautionary Note, The Second USENIX Workshop on Electronic Commerce, Oakland, California, 1996.
[45] R. Torrance, "The state-of-the-art in Semiconductor Reverse Engineering at Chipworks", CHES, 2009.
[46] R. Anderson, "Security Engineering", Second Edition, p.509, 2008.
[47] B. Yang, K. Wu, R. Karri, "Scan based side channel attack on dedicated hardware implementations of Data Encryption Standard", Test Conference, 2004. Proceedings. ITC 2004. International, pp. 339-344, 26-28 Oct. 2004.
[48] S. W. Moore, R. J. Anderson, M. G. Kuhn, "Improving Smartcard Security using Self-Timed Circuit Technology", Fourth AciD-WG Workshop, 2000.
[49] O. Kömmerling, M. G. Kuhn, "Design principles for tamper-resistant smartcard processors", Proceedings of the USENIX Workshop on Smartcard Technology on USENIX Workshop on Smartcard Technology, p 2, 1999.
[50] "STMicroelectronics and Front Edge Technology to Bring Next-Generation Battery Technology to New Markets", http://www.st.com/stonline/stappl/cms/press/news/year2009/t2380.htm

[51] "GORE™ Tamper Respondent Surface Enclosure", http://www.gore.com/en_xx/products/electronic/anti-tamper/tamper-surface-enclosure.html
[52] http://www.autoidlabs.org/uploads/media/AUTOIDLABS-WP-SWNET-016.pdf
[53] B. Schneider, "Applied Cryptography", Second Edition, John Wiley & Sons, 1996
[54] http://ehash.iaik.tugraz.at/wiki/The_SHA-3_Zoo
[55] http://www.ecrypt.eu.org/stream
[56] http://www.i-hacked.com/content/view/18/90/
[57] http://www.yourtomtom.com/241/tomtom-map-security-hacked.html
[58] See p.4, Table B http://www.itrs.net/Links/2009ITRS/2009Chapters_2009Tables/2009_ExecSum.pdf

7 Glossary

A

AES	Advanced Encryption Standard
ALU	Arithmetic Logic Unit
ASIC	Application-Specific Integrated Circuit

B

BIST	Built-In-Self-Test

C

CBC	Cipher Block Chaining
CFB	Cipher Feedback Mode
CMOS	Complementary Metal Oxide Semiconductor
CMP	Certificate Management Protocol
COTS	Commercial Off-The-Shelf
CPUF	Controlled Physical Unclonable Function
CPLD	Complex Programmable Logic Device
CTR	Counter Mode

D

DES	Data Encryption Standard
DH	Diffie-Hellman
DPM	Direct Part Marking
DRAM	Dynamic Random Access Memory
DRM	Digital Right Management
DSA	Digital Signature Algorithm

E

ECB	Electronic Code Book Mode
ECDSA	Elliptic Curve DSA
ECRYPT	European Network of Excellence in Cryptology
EDA	Electronic Design Automation
EEPROM	Electrically Erasable Programmable ROM

F

FIB	Focused Ion Beam
FPGA	Field Programmable Gate Array

G

GQ	Guillou-Quisquater

I

IC	Integrated Circuit

IKE	Internet Key Exchange
IP	Intellectual Property
IPSec	Internet Protocol Security

K

KEM	Key Encapsulation Mechanism
KDF	Key Derivation Function

L

LTCC	Low Temperature Cofired Ceramics

M

MD5	Message Digest Algorithm 5

N

NESSIE	New European Schemes for Signatures, Integrity and Encryption
NIST	National Institute of Standard and Technology
NVRAM	Non-Volatile Random-Access Memory

O

OAEP	Optimal Asymmetric Encryption Padding
OEM	Original Equipment Manufacturer
OFB	Output Feedback Mode
OTP	One-Time Programmable

P

PCB	Printed Circuit Board
PLD	Programmable Logic Device
PLL	Phase-Locked Loop
PRF	Pseudo-Random Function
PSS	Probabilistic Signature Scheme
PUF	Physical Unclonable Function

R

RAM	Random Access Memory
RFID	Radio-Frequency Identification
ROM	Read-Only Memory

S

SCFM	Scanning Capacitance Force Microscopy
SHA-1	Secure Hash Algorithm-1
SIA	Semiconductor Industry Association
SiP	Systems-in-Package
SRAM	Static Random Access Memory

SSL	Secure Sockets Layer
STM	Scanning Tunneling Microscopy

T

TLS	Third Level Support
TRNG	True Random Number Generator
TSMC	Taiwan Semiconductor Manufacturing Company

U

UMTS	Universal Mobile Telecommunications System
UV	Ultra-Violet

W

WEP	Wired Equivalent Privacy