Hardware Security and Trust: A New Battlefield of Information

Gang Qu(✉)

University of Maryland, College Park, MD 20742, USA
gangqu@umd.edu

Abstract. Hardware security and trust has received a lot of attention in the past 25 years. The purpose of this paper is to introduce the fundamental problems related to hardware security and trust to audiences who do not necessarily have hardware design background. In order to do that, we first discuss the evolving roles of hardware in security from an *enable* to an *enhancer* and now an *enforcer* as it get involves more and more in system security. Then we review the following key problems in hardware security, physical attacks, side channel analysis, intellectual property protection, hardware Trojan, hardware security primitives, and applications in security and trust. We provide a novel view of these problems and the corresponding solutions from the perspective of information battle between the attackers and designers, where we consider three types of information: data collected, processed, and stored by the hardware; information hidden in the design as watermark, fingerprint, and Trojans; and the chip fabrication variations that can be extracted and utilized. It is interesting to see how the hardware security and trust problems can be unified under this framework of information battle (stealing and protection). Unfortunately, there are more unknowns and challenges than what we have discovered on this framework as we illustrated in the section of open problems. However, the emerging Internet of Things and cyber physical systems have provided a large application field for researchers and practitioners to work on hardware based lightweight security.

Keywords: Hardware security · Trusted IC · Intellectual property protection · Reverse engineering · Side channel analysis · Physical unclonable function · Hardware trojan · Logic obfuscation · Hardware security primitives · Information hiding · Lightweight authentication

1 Introduction

The year of 1996 saw two important events in what we now know as hardware security and trust. First, timing attack was reported as a computationally inexpensive method to break cryptosystems including Diffie-Hellman, RSA, and DSS [1]. This leads to the discovery of various side channel analysis (SCA) attacks, which take advantage of system's different execution time, power consumption, electromagnetic emission or other physically measureable characteristics while running same operations with different values (such as bit '0' and bit '1') to reveal the cryptographic keys. Second, the Virtual

Q. Zhu et al. (Eds.): GameSec 2020, LNCS 12513, pp. 486–501, 2020.
https://doi.org/10.1007/978-3-030-64793-3_27

Socket Interface Alliance (VSIA) was founded to enhance semiconductor industry's design productivity by establishing standards for the adoption of intellectual property (IP). The alliance attracted more than 200 companies worldwide and was dissolved in 2008 after accomplishing it mission. VSIA identified six challenges and built development and working groups (DWG) for each of them in 1997. IP protection was one of the most technically challenging with the goals to 1) enable IP providers to protect their IPs against unauthorized use, 2) protect all types of design data used to produce and deliver IPs, 3) detect use of IPs, and 4) trace use of IPs [2]. Most reported research efforts were on side channel attacks and IP protection for about a decade before several other important discoveries in hardware security and trust.

The problems of trusted integrated circuit (IC) design and hardware Trojan detection were proposed around 2005 and 2007, respectively. One of the most notable efforts on these problems is a sequence of DARPA programs: trusted integrated circuits (TRUST), integrity and reliability of integrated circuits (IRIS), supply chain hardware integrity for electronics defense (SHIELD), and Automatic Implementation of Secure SoCs (AISS). Trusted IC was defined as *doing exactly what it is asked, no more and no less* [3] and was recommended to be re-defined more precisely as "*no less and no malicious more*" [4]. One way to make ICs untrusted is to embed hardware Trojans (HT), which is a piece of circuit that is added to the design or modified from the original design for malicious purposes. HT was first reported in [5]. Also in 2007, silicon physical unclonable function (PUF) got a great deal of attention. PUF is a device or sub-circuit embedded on chip to capture the fabrication variations in the forms of path delay, device voltage transfer, or other characteristics. Such variations exist in the silicon manufacturing process and are considered to be unpredictable and uncontrollable. PUF can generate and store secret that can be used as keys or seeds to generate random numbers; and create challenge-response pairs that can be used for chip authentications [6].

These are just a few key topics in the emerging field of hardware security and trust. In 1999, the Cryptographic Hardware and Embedded Systems (CHES) conference was founded "for research on both design and analysis of cryptographic hardware and software implementations". In 2008, the International Symposium on Hardware Oriented Security and Trust (HOST) was established "for researchers and practitioners to advance knowledge and technologies related to hardware security and assurance". Nowadays, all the major conferences on hardware, architecture, and system design cover the topics of hardware security and trust, which is also listed in the leading security conferences. For instance, Crypto solicits submissions on "secure implementation and optimization in hardware". USENIX Security has an area of "hardware security" with topics on secure computer architecture, embedded system security, malicious and counterfeit hardware detection, and side channels.

In this paper, we will discuss the evolving role of hardware in security and cybersecurity in Sect. 2. We will introduce the key problems in hardware security and trust and the state-of-the-art approaches in Sects. 3. The synergy of these problems and solutions will be analyzed in Sect. 4. Unlike a comprehensive survey for hardware engineers, we will provide the researchers with little hardware design background the perspectives of information battle between the attackers and defenders. We discuss the open problems in hardware security and conclude the paper in Sect. 5.

2 The Role of Hardware in Cybersecurity

In 2009, I was invited to give a talk to a group of audience who are not computer engineers about hardware's role in security and trustworthy computing. I used the terms *enabler, enhancer*, and *enforcer* to describe this evolving role and I also questioned whether hardware has become the weakest link in security and trust.

Security starts with cryptography which is built on sound mathematics foundations and implemented either in software or hardware. But ultimately it is the computer hardware that enables us to realize all the security protocols. Consider the extremely high computational complexity of the modern cryptography schemes, it is impossible for human to do the computation manually without the help of computing devices. For example, in the modular exponentiation operation which computes $a^e (modn)$, all the values are huge number with the exponent e suggested to be 1024-bit or longer. In such cases, hardware is absolutely needed as an enabler.

It is well known that many applications, security related or not, have better performance when implemented in hardware comparing to their software based implementation. Dedicated hardware, sometimes called accelerators, are built for the purpose of performance enhancement. Ironically, hardware is also used to break the security protocols (e.g. through brute-force attacks). Indeed, it was the increasing computation power that made data encryption standards (DES) unsecure and motivated the establishment of the advanced encryption standard (AES). In 2001, Rijndael ciphers was selected from many AES candidates in part because of its efficiency and implementation details [7].

Then computer hardware becomes more actively involved in security and trustworthy computing. The first line of defense is built in hardware to protect the CPU, memory and data. For example, a biometric coprocessor checks user's biological features such as fingerprint, iris, and pulse to authenticate the user before giving user the permission to access the computer or the network. Another example is the trusted platform module (TPM) chips that are embedded to all the laptops and smart phones. A TPM chip helps system to manage all the security and trusted computing functions.

However, the high involvement of hardware in security also introduces the new attacking surface in hardware implementation of the cryptographic systems. Hardware engineers are traditionally trained to optimize performance and security is not considered when hardware is designed and built. This gives attackers another target, in particular when there is no flaw in the crypto algorithms, software vulnerabilities have been patched, and network communication becomes secure. The side channel attack we mentioned earlier is one example. More security and trust vulnerabilities in hardware will be discussed later, which prompts me to ask the question whether hardware is the weakest link (after human) in cybersecurity (Fig. 1).

From the information standpoint, an enabler is a passive information processor; an enhancer is a dedicated processor for specific information process; and an enhancer is one that collects information, processes information, and makes decisions (such as authentication and access control) accordingly. Information may leak during the process on hardware, causing security vulnerabilities that we will elaborate next.

Security and Hardware

- Enabler
- Enhancer
- Enforcer

Weakest Link?

Fig. 1. A slide that illustrates the role of hardware in security and trust (2011).

3 Key Problems in Hardware Security and Trust

In this section, we review the key research problems and practices in hardware security and trust. We will also discuss the existing countermeasures. More detailed and conclusive list of topics can be found in the call for papers of the recent conferences focusing on hardware security in the Appendix.

3.1 Physical Attacks

As the name suggests, physical attacks refer to the attacks where an attacker has physical access to a system or is within its proximity to collet certain physical information. The goal of physical attacks is to break or "learn" the system without authorization. Unlike cryptanalysis which uses mathematical analysis on the cryptographic algorithms to find flaws, physical attacks attempt to exploit the vulnerabilities in the implementation of the system. Based on whether the target system will be damaged during and after the attack, physical attacks can be classified in three groups: invasive attacks where the attacker "breaks" the system physically to learn, the system will be damaged and there will be tampering evidence left; non-invasive attacks where the attacker learns by "using" the system without causing any damage or leaving any trace of tampering; and semi-invasive attacks where the attacker needs to access the surface of the system without "breaking" or "damaging" it, there will be no or very little tamper evidence.

In hardware, invasive physical attacks are also known as reverse engineering where an attacker will depackage the chip or device to expose the silicon die to learn the inner structure and the functionality of the chip. For modern multi-layer chips, the attacker will remove layer by layer to study features in each layer. Reverse engineering is legal and very common in industry as companies use it to learn from their competitors and legacy systems where the detailed design information is unavailable. There are commercial advanced reverse engineering tools available, which makes reverse engineering based attacks possible. Reverse engineering will cause damage to the chip or device and thus cannot be repeated on the same device. More about reverse engineering and the countermeasures will be discussed in the section of design IP protection.

Common non-invasive attacks include side channel analysis (which we will elaborate in details in the next subsection), fault based attacks, data remanence, and brute force. The idea of fault based attacks is to put the system into abnormal and unexpected execution state in the hope that such states are not well protected by design. This can be achieved by injecting faulty data or unexpected instructions, or changing the execution environment such as lowing the voltage. Data remanence are attacks on the data stored in the SRAM, EEPROM, or flash memory. Because of certain physical features of these memory, data stored for a long time may leave some trace even after it is removed or powered down, protected data may also become readable at extreme environment such as low temperature or frequently changed voltage. When the search space is not sufficiently huge, brute force search for the cryptographic keys or backdoor access to a system becomes possible with the help of today's powerful computers.

Semi-invasive attacks to hardware normally require depackaging the chip but will not need the reverse engineering steps to learn and will not make physical contacts with the internal wires. This normally helps to launch more powerful attacks such as fault injection or side channel analysis because now the silicon die is decapsulated and exposed to the attackers.

3.2 Side Channel Analysis

An attacker can observe a system's physical characteristics from side channels during execution and uses such characteristics to reveal the system's secret information such as the cryptographic keys. These physical features can be power, current, timing or delay, electromagnetic emission, acoustic and optical information, and even the system's output values. Side channel analysis (SCA) attacks have two phases: measuring and data analysis. During the first phase, the attacker will monitor the system's execution and collect the physical features of interest. Then, the attacker will perform data analysis on the collected side channel information to determine the on-chip secret information.

SCA attacks are perhaps the most successful attacks to modern cryptographic systems for two main reasons. First, they target the weakness of the implementation of the crypto algorithms, not the algorithms themselves. Therefore, a mathematically sound algorithm can become vulnerable against SCA attacks. Second, these attacks are non-invasive, passive, and will not leave any trace of attack. They use the signals leaked from side channels during system's normal execution and thus it will be hard to detect and catch such attacks.

SCA attacks rely on the fact that the execution of the same operations with different input values will generate different trace on the side channels. For example, in the popular square and multiply implementation of modular exponentiation, the computation will be performed iteratively on each of the key bit with the multiplication being carried out only when the key bit is '1'. This will create asymmetric information in terms of the execution time and power consumption for different key bit values, enabling the timing and power analysis attacks.

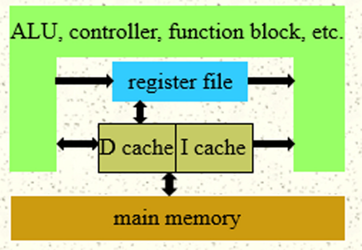


Fig. 2. Side channels in a simplified microprocessor.

Figure 2 illustrates a simplified architecture of a standard microprocessor or a computing device. It has its memory hierarchy of the main memory, data cache, instruction cache, and register files. This is the central processing unit, or CPU, with control logics, function blocks, and the arithmetic and logic unit (ALU). In a typical flow of the execution of a software or program, instructions and data will be loaded from the main memory to I-cache and D-cache. The registers are the closest storage to the CPU and thus have the fastest access time. The CPU will take instructions and data from these memory units and process them accordingly. The result will be written back to the memory, either cache or the main memory.

Now assume that we store some secret data in the registers during the execution and revisit the typical execution blow to identify the side channel vulnerabilities. First, memory load operation will get data from D cache to the register file. This needs the memory address of the data. If the address is determined by or related to the secret data, the secret might leak from the memory address. When the secret data is overwritten by the data from the memory, there will also be information leak. For example, when the register file is reset to be all 0s, it requires power to overwrite all the 1s, but there is almost zero power consumption on the bit that was previously 0. Similarly, when a memory store operation is performance, information might leak from either the memory address or the data to be written to the memory. During arithmetic and logic operations, particularly when the operation is performed at bit-level, the secret data may be exposed through side channel. For example, the execution of a multiplication with two random large numbers and the same multiplication with one operand equal to one will have quite different behaviors that can be observed through power or timing side channels. Finally, data might leak from the control flow of the execution as we have seen from the example of modular exponentiation where whether the multiplication is executed will depend on the key bit values.

Common countermeasures to SCA attacks either try to hide side channel information or remove its dependency on the secret data. Crypto algorithms can be modified, typically by randomization, to remove the correlation between the cryptographic key values and the side channel information generated while running the crypto algorithm using the key. Second, physical security can be used to keep the attackers away from the proximity, access, and possession of the system under attack. For example, acoustic shielding can protect acoustic emission and the secure construction zoning is common to prevent potential EM emission attacks. Third, design partitioning, in particular the emerging 2.5D and 3D fabrication and split manufacturing, can help to mitigate SCA attacks. Separating on-chip infrastructures such as power supply rails, clock networks, and testing facilities from crypto operations and other applications will make it more challenging for side channel information collection. Masking and blinding is another approach to remove the correlation between the secret data and the side channel signals. For instance, XORing the output of a logic unit with some pre-selected data will mask the real output, which can be retrieved only when the pre-selected data is known. Finally, hiding is the most common methods to increase the difficulty for the SCA attackers to gather side channel data. This can be achieved by careful design that will leak identical information on different key values, by using asynchronous logic, or by generating random noise.

3.3 Intellectual Property Protection

Design intellectual properties (IP) are the components or units that can be considered as stand-alone for being reused or integrated into a larger design with efforts much lower than redesigning the component. IPs are the most valuable for the company who designs, manufactures, and owns them. However, an adversary can steal or misuse IPs by forging, tampering, counterfeiting, overbuilding, and so on. Most of these IP infringements require reserve engineering to some extent.

The VISA has identified three major IP protection methods: **Deterrent** methods enable an IP owner to deter the infringer from contemplating IP theft by using proper legal means including patents, copyrights, contracts, trademarks, and trade secrets. **Protection** mechanisms use means such as encryption, licensing agreements, obfuscation, dedicated hardware, or chemicals to prevent unauthorized access to the IP. **Detection** approaches such as digital watermarking, fingerprinting, and metering, help the IP owners to detect and trace both legal and illegal use of their IPs.

Most protection mechanisms are mature and could be effective, but they incur additional design cost such as the computational expensive encryption/decryption, the integration and packaging of chemicals and dedicated hardware ware. Deterrent methods do not directly prevent IP piracy from happening, but rather discourage the misuse of IPs because the attackers, once being caught, may face lawsuits and severe penalty to recover the financial loss of the IP owner. However, all of the aforementioned means except trade secrets are affirmative rights, which means that it is the IP owner's responsibility to identify IP infringement and catch the IP infringer. Therefore, majority of efforts in IP protection in the past couple of decades are on the detection approaches.

Digital watermarking embeds IP owner's signature into the IP during its design, integration, and testing phases. The watermark, if needed, can be retrieved from the IP to prove the authorship. Digital fingerprinting incorporates IP buyer's unique information into the IP in order to identify the traitor should any IP infringement happens. Metering is a means to create/insert tags into each copy of the IP or chip, making them unique to facilitate the trace chip. Recently developed IP protection methods have made the distinction between protection and detection approached vague. For example, active metering and logic locking techniques are protection mechanisms that provide chip owners post-fabrication control of the fabricated chips (because they can disable the normal usage of the chip). Meanwhile, circuit obfuscation method intentionally introduces ambiguity to chip design to confuse reverse engineering attackers and should be considered as detection approach. Split manufacturing and 3D integration technologies also facilitate IP protection by giving designers the option of fabricating chips in multiple foundries and serve both the purposes of protection and detection.

3.4 Hardware Trojan

A hardware Trojan is any modification or addition to a circuit for malicious purpose. Common malicious goals of hardware Trojan include leaking sensitive information, changing or controlling the functionality of the circuit, and reducing circuit reliability. Based on different criteria, hardware Trojans can be categorized in many different ways. We list a few below as we discuss more features of hardware Trojan.

First, hardware Trojans can be as small as only a few logic gates (e.g. malicious on-chip sensors and the killer switch) and can be as large as a functional block such as a powerful antenna which is capable of sending out sensitive information. An external disable/enable signal combined with a simple 2-input logic AND gate can be used to control any functional units (e.g. the encryption engine) on the chip.

These small and large hardware Trojans can be inserted almost all over the system. Trojans in the on-chip clock (or power) network can change system's clock frequency (or operating voltage) to launch fault attacks or timing (or power) side channel information leaking. Trojans in the system's memory structure can maliciously change data or leak information. Trojans in the CPU or functional units can create faulty output or disable the functional units. Trojans in the input-output periphery can facilitate fault injection attacks or provide misleading results.

Hardware Trojans normally are triggered by specific signals or events that are rare to occur under normal execution mode (in order to minimize the chance of being detected or accidentally activated). These triggers can be from inside the chip such as whether the counter has reached a specific value (time bomb) or certain on-chip temperature has been sensed by the temperature sensor. They can also be controlled externally by triggers hidden behind user input or environmental conditions.

Hardware Trojans can be embedded during any untrusted phase in the chip design, fabrication, and testing process, or any stage of the IC supply chain. This makes hardware Trojan detection and prevention a very challenging task. It is important to mention that the goal of hardware Trojan detection is determine whether a chip or system contains any hardware Trojan. If a Trojan is found, we can conclude that the chip or system cannot be trusted. But unfortunately, one can never claim a chip is Trojan free because a system's functionality can never be completely specified and thus there are always unspecified functionality being implemented in the chip [10].

Hardware Trojan detection can be done at chip test time followed by run-time monitoring. At test time, one can use logic test-based approaches to run different input and verify the corresponding output generated by the chip while monitoring chip's execution behavior. Since we have mentioned earlier that Trojans are triggered by rare signals or events, such logic test detection methods could fail. Monitoring side channels during the test can help to capture some Trojans but it may have high false alarm rate due to measurement errors, noise, and hardware fabrication variations. Such SCA based monitoring should be kept during the normal execution of the chip because test time detection methods can never discover all the Trojans.

3.5 Hardware Security Primitives

It is well know that chips, even identical designed and fabricated with the same mask on the same wafer, exhibit the intrinsic manufacturing variations from the complicated semiconductor fabrication process. Such variations are believed to be random and cause the unpredictable chip performance. However, it is extremely difficult, if not impossible, to model or control the manufacturing variations as the semiconductor industry has failed to do so in the past half century. The concept of physical unclonable function (PUF) takes advantage of the randomness, unclonability, and uncontrollability of such intrinsic variations to deliver security primitives. The most popular applications of PUF are to create and store cryptographic keys, to facilitate random number generation, and to generate challenge-response pairs for authentication.

Figure 3 shows the basic ring oscillator (RO) PUF structure. As we can see, N identical ROs are implemented but the fabrication variations will make them have different delays. The two multiplexers (MUX) will select two ROs and compare their delay through the readings of the two counters. For example, one can define a bit ‘0’ is the RO on the top is faster and a bit ‘1’ if the one at the bottom is faster. There are numerous reports on how to create PUF bits, make them robust against operation environments, optimize the amount of bits generated from a given hardware resource, and how to use PUF for security applications.

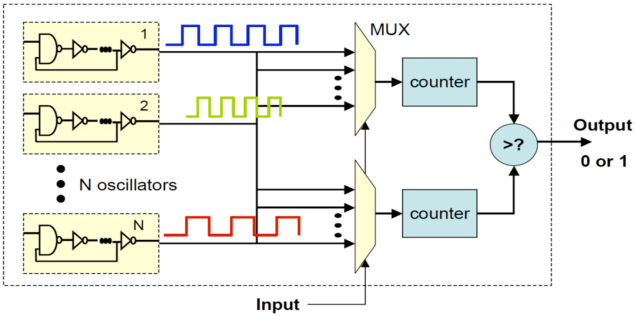


Fig. 3. The architecture of ring oscillator PUF [6].

3.6 Applications in Security and Trust

There are many research efforts to connect circuit level hardware security with lower level such as memory and new materials (resistive RAM, phase change memory, Spin-transfer torque magnetic RAM, etc.), and with upper level at architecture, software, communication and physical layer. We give two examples on how hardware can help security at system or device level.

As hardware is the root of all systems ranging from the sensors, smart portable devices, and medical implants to vehicle components, smart home appliances, smart grid, cloud computing servers, and the general Internet of Things (IoT) and cyber physical systems (CPS), it is not surprising to see that they are used not only to implement system security protocols, but also contribute to improve system security. This is particularly true for the recent IoT and CPS applications where the devices may be resource constrained and cannot afford to the computational and resource expensive cryptographic solutions. For example, many hardware based lightweight authentication schemes have been proposed to authenticate device, user, and computation.

Figure 4 shows how silicon PUF can be used to enhance the entropy of a random variable [8]. The bell-look curve at the bottom is the Shannon entropy when a random input bit is generated with a given percentage of 1’s. For example, in the middle when there are 50% 1’s and 50% 0’s, perfect entropy is reached. But on both ends when there are very few 1's or a lot of 1's, the entropy will be very low. With the assist of RO PUF, we can see that the entropy is improved significantly. The top curve is the case when PUF is combined with XOR. This indicates a cost effective way to convert data from a poor entropy source to high entropy bit-stream.

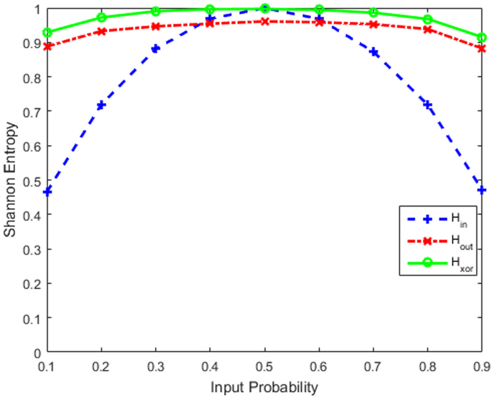


Fig. 4. Bit entropy enhanced by silicon PUF [8].

As another example, Fig. 5 shows how the benchmark images "snowflakes" and "trees" can be superimposed to generate the image of "snowfall", where each pixel value of the "snowfall" is obtained by adding the pixel values of "snowflakes" and "trees" at the same pixel position [9]. We design an adder in hardware to perform this operation. When the same adder design is implemented on two different FPGA boards, we reduce the operating voltage for both FPGA to force addition errors as shown in Fig. 6. Clearly we can see the visual difference between these erroneous images and the original one. More importantly, different adders create different errors, making it possible to use such fabrication variation induced errors for device authentication [9].

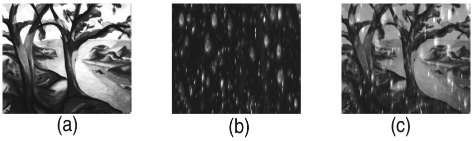


Fig. 5. Creation of "snowfall" (c) by superimposing "snowflakes" (b) on top of "trees" (a) [9].

4 Information Battle Perspective of Hardware Security

We analyze the key problems in hardware security and trust listed in the previous section from the perspective of information.

Consider the role that hardware plays as an enabler and an enhancer, it may process and store sensitive information of the attacker's interest. Through different types of

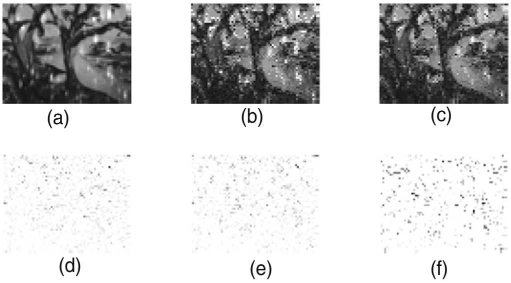


Fig. 6. An example of the effect of fabrication variations in voltage over-scaling based computation. In (a) the gray scale image *Snowfall* is computed using *trees* and *snowflakes* without voltage over-scaling; in (b) and (c) the image is computed under voltage over-scaling with two adders which are identical in every aspect, except the process variation of the transistors; (d) and (e) shows the error pattern found in the figure (b) and (c). This error pattern shows the deviations for each adder from the correct image. Subfigure (f) shows the difference between the two error pattern (d) and (e). The source images were downsized to 52 × 40 pixels for reducing computation time.

attacks that include physical attacks and side channel analysis, the attacker attempts to obtain the desired information directly or indirectly. For example, invasive attack to a smartcard can reveal the contents stored in the memory, analyzing timing or power side channel information can help the attacker to reveal the cryptographic key used in the crypto algorithm. Consequently, all the countermeasures against such attacks try to hide the sensitive information, to disable the attacker's access to side channels, or to remove the correlation between the sensitive information and the side channel information.

For design IP protection, the design and implementation details are the value of the IP and take the forms of hardware device, IP cores, gate layout, FPGA configurable bit streams, Verilog code, optimization algorithms, and so on. Reverse engineering is one vehicle for the attackers to retrieve these information from a IP product. IP protection methods either protect such information, making them inaccessible, or embed more information into the IP as watermark, fingerprint, and tags for detection purpose. So here we see a new dimension of information protection by adding more proof-carrying information into the design and implementation of IPs. This can be done as encryption (where the information to be protected is encrypted and the encryption key becomes the vital information for decryption) or obfuscation (where the original design information is hidden behind the camouflaged logic gates). Figure 7 illustrates the basic idea of digital watermarking as a steganography system [11]. Digital fingerprint and metering tags can be embedded in the same way.

Unlike watermarking multimedia contents, adding digital watermark into hardware IP has a fundamentally different challenge: the contents of the multimedia artifacts can change as long as the end consumer, which is human, cannot tell or can tolerate the difference between the original and the watermarked copy. However, for hardware IP, changes to the design and implementation of the IP are normally unacceptable because they may cause malfunctions. As depicted in Fig. 6, we view the creation of IP as solving a constraint optimization problem, where system's specifications and design requirements are

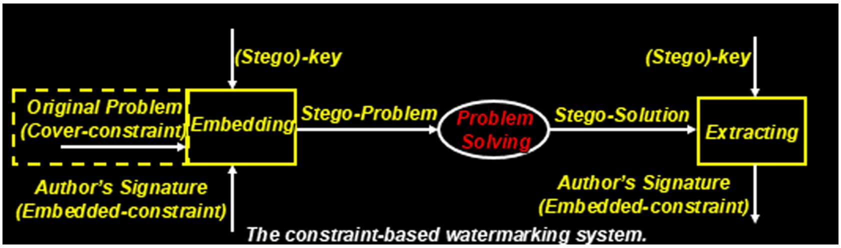


Fig. 7. Digital watermarking for IP protection as a steganography system [11].

considered to be the original constraints. We convert IP author's signature into constraints that will not cause any conflicts or violation to the original constraints. The proposed digital watermarking system is a steganography system where the original constraints serve as the cover-constraint and the author's signature is the embedded-constraint. The stego-problem to solve is both sets of constraints. Now we solve this stego-problem, that is, design and implement the IP to satisfy both the original and the embedded constraints. The solution, or the developed IP, will have the property to meet not only the original constraints, but also a set of seemingly random constraints that we embedded as watermark, or the proof of author's signature. It is crucial to extract the watermarking information from the stego-solution.

Interestingly, hardware Trojan also can be considered as information embedded into the system, but for malicious purposes. Trojan detection becomes the process of finding such hidden information and evidently attacker's physical attack and SCA methods has be utilized.

The intrinsic fabrication variation information carried by the chip is another type of interesting information. PUF is the circuitry that collects such information and converts it to data that can be used for security such as cryptographic keys. As we have discussed earlier, one of the biggest challenge in PUF is its usability as most of the fabrication variations are very sensitive to the chip's operating environment including power, voltage, and temperature. If we consider PUF as the noise introduced during the manufacturing process, PUF information's sensitivity to environmental factors is the "measurement" error/noise when the PUF circuitry collects the fabrication variation. When PUF is used in the system, various security concerns, such as how to steal PUF information or forge the challenge-response pair, share the same core of how to protect the PUF information from unauthorized access or usage. This brings us back to the start of this section where we discussed securing data and data processing against attacks.

5 Open Problems and Conclusions

Security becomes one vital concern for almost all systems. Hardware, as the player to collect, process, store, and transmit information, not only causes various security vulnerabilities, largely due to the lack of consideration of security and trust in hardware design flow, but can also makes the system more security at lower cost. So the first and most important challenge for hardware security is how to convince the users that *the hardware system is secure and trusted*. For example, when a sensor collects data and sends the sensor readings to the network, how to check whether the right and accurate data is collected? Are the sensor's data calibration and other pre-processing schemes executed in a secure and trusted execution environment? Is the data storage secure and uncompromised? Does the sensor have any Trojan or side channel that might leak the data?

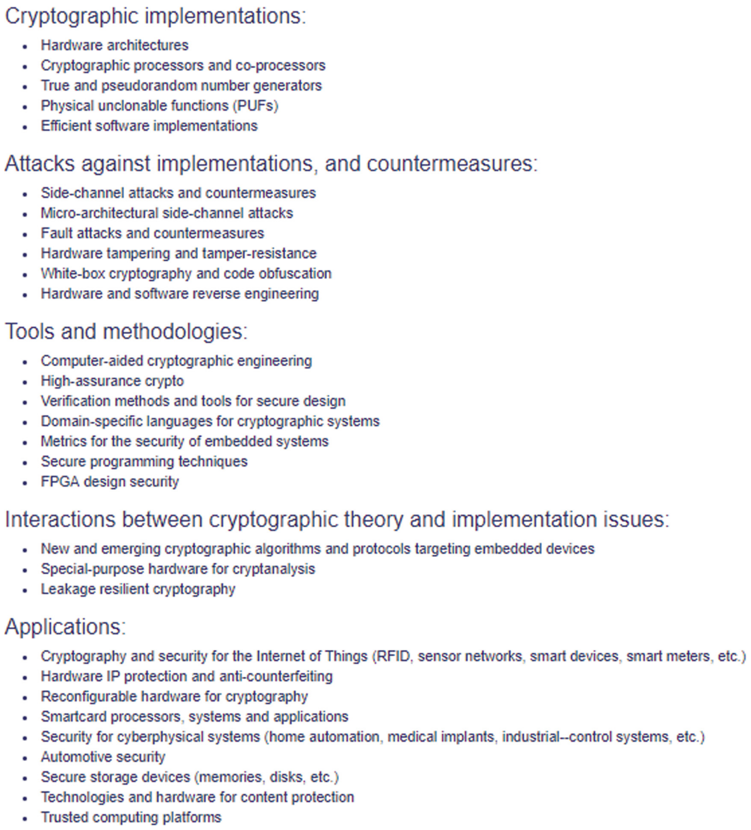
Secondly, *how to utilize hardware to build and enhance system security and trust?* We already know hardware security primitives have this potential with the advantage of low cost and sometimes better security (e.g. unclonability of the fabrication variations and requirement of the physical presence of or close proximity to the hardware). Such hardware based lightweight schemes are good for applications such as authentication when the security level is low. How to establish a formal foundation for the hardware base lightweight security protocols? This seems to be a very challenging task as the semiconductor industry still does not have any accurate models for various fabrication variations and it will be impossible to conduct any quantitative study on the security protocols built on such variations. For example, the randomness and unclonability of variations are just the general belief without any proof or validation. Nevertheless, finding new security applications based on hardware is still of great interest.

Finally, from the perspective of hardware designer, *IP protection is still an important yet open problem.* As one of the earliest challenges from the industry, IP protection is a real problem and still has not received the attention it deserves. In part, this is due to the complexity of IP validation and integration as well as other challenges for IP reuse. Before IP reuse becomes a common design practice, we cannot see the true value of IPs and how serve IP infringement could be. Fortunately, the incidents of tampering, reverse engineering based IP stealing, and counterfeiting reported in the recent years have raised the global awareness of IP protection. The existing IP protection techniques are not adequate. For example, two the most well-studied active IP protection methods, logic locking and circuit obfuscation are vulnerable to SAT-based attacks. Digital watermarking and fingerprinting methods are relatively mature and there is the ongoing efforts to integrate them into the hardware design flow. However, the impact to the system performance caused by embedding watermark and fingerprint is still unknown.

Acknowledgement. This work is supported in part by the DARPA project entitled "INDEPENDENT VERIFICATION &VALIDATION (IV&V) OF THE AISS PROGRAM".

Appendix

CHES 2021 list of topics in the call for paper (https://ches.iacr.org/2021/callforpapers.php)



Cryptographic implementations:

- Hardware architectures
- Cryptographic processors and co-processors
- True and pseudorandom number generators
- Physical unclonable functions (PUFs)
- Efficient software implementations

Attacks against implementations, and countermeasures:

- Side-channel attacks and countermeasures
- Micro-architectural side-channel attacks
- Fault attacks and countermeasures
- Hardware tampering and tamper-resistance
- White-box cryptography and code obfuscation
- Hardware and software reverse engineering

Tools and methodologies:

- Computer-aided cryptographic engineering
- High-assurance crypto
- Verification methods and tools for secure design
- Domain-specific languages for cryptographic systems
- Metrics for the security of embedded systems
- Secure programming techniques
- FPGA design security

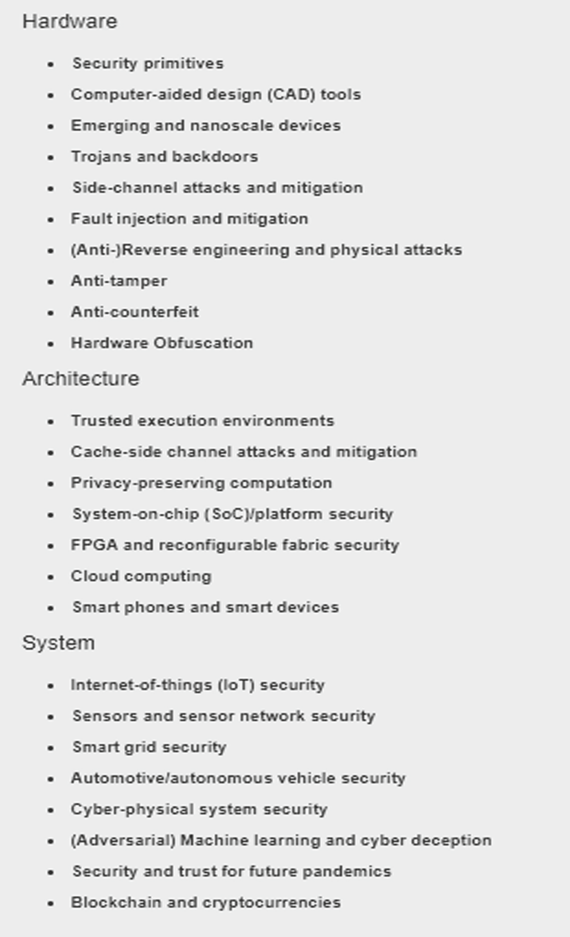
Interactions between cryptographic theory and implementation issues:

- New and emerging cryptographic algorithms and protocols targeting embedded devices
- Special-purpose hardware for cryptanalysis
- Leakage resilient cryptography

Applications:

- Cryptography and security for the Internet of Things (RFID, sensor networks, smart devices, smart meters, etc.)
- Hardware IP protection and anti-counterfeiting
- Reconfigurable hardware for cryptography
- Smartcard processors, systems and applications
- Security for cyberphysical systems (home automation, medical implants, industrial--control systems, etc.)
- Automotive security
- Secure storage devices (memories, disks, etc.)
- Technologies and hardware for content protection
- Trusted computing platforms

HOST 2021 list of topics in the call for paper (http://www.hostsymposium.org/call-for-paper.php).



Hardware

- Security primitives
- Computer-aided design (CAD) tools
- Emerging and nanoscale devices
- Trojans and backdoors
- Side-channel attacks and mitigation
- Fault injection and mitigation
- (Anti-)Reverse engineering and physical attacks
- Anti-tamper
- Anti-counterfeit
- Hardware Obfuscation

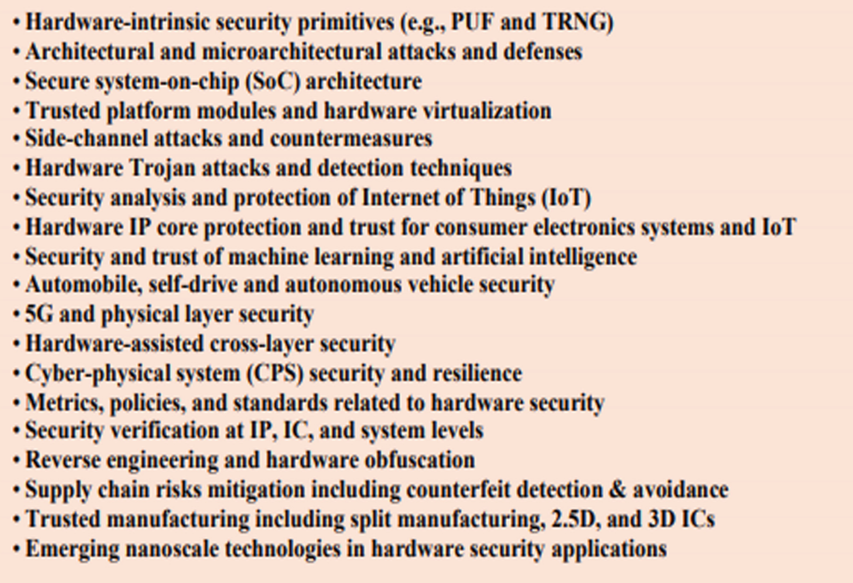
Architecture

- Trusted execution environments
- Cache-side channel attacks and mitigation
- Privacy-preserving computation
- System-on-chip (SoC)/platform security
- FPGA and reconfigurable fabric security
- Cloud computing
- Smart phones and smart devices

System

- Internet-of-things (IoT) security
- Sensors and sensor network security
- Smart grid security
- Automotive/autonomous vehicle security
- Cyber-physical system security
- (Adversarial) Machine learning and cyber deception
- Security and trust for future pandemics
- Blockchain and cryptocurrencies

AsianHOST 2020 list of topics in the call for paper (http://asianhost.org/2020/authors.htm#cfp)



- Hardware-intrinsic security primitives (e.g., PUF and TRNG)
- Architectural and microarchitectural attacks and defenses
- Secure system-on-chip (SoC) architecture
- Trusted platform modules and hardware virtualization
- Side-channel attacks and countermeasures
- Hardware Trojan attacks and detection techniques
- Security analysis and protection of Internet of Things (IoT)
- Hardware IP core protection and trust for consumer electronics systems and IoT
- Security and trust of machine learning and artificial intelligence
- Automobile, self-drive and autonomous vehicle security
- 5G and physical layer security
- Hardware-assisted cross-layer security
- Cyber-physical system (CPS) security and resilience
- Metrics, policies, and standards related to hardware security
- Security verification at IP, IC, and system levels
- Reverse engineering and hardware obfuscation
- Supply chain risks mitigation including counterfeit detection & avoidance
- Trusted manufacturing including split manufacturing, 2.5D, and 3D ICs
- Emerging nanoscale technologies in hardware security applications

References

1. Kocher, P.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Crypto 1996, pp. 104–113 (1996)
2. Virtual Socket Interface Alliance: Intellectual Property Protection White Paper: Schemes, Alternatives and Discussion, Version 1.1, January 2001
3. Report of the Defense Science Board Task Force on High Performance Microchip Supply, February 2005
4. Qu, G., Yuan, L.: Design THINGS for the internet of things – an EDA perspective. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design, pp. 411–416, November 2014
5. Agrawal, D., Baktir, S., Karakoyunlu, D., Rohatgi, P., Sunar, B.: Trojan Detection using IC Fingerprint. In: IEEE Symposium on Security and Privacy, pp. 296–310, May 2007
6. Suh, G.E., Devadas, S.: Physical unclonable functions for device authentication and secret key generation. In: Proceedings of 44th ACM/IEEE Design Automation Conference, pp. 9–14, June 2007
7. United States National Institute of Standards and Technology (NIST): Announcing the ADVANCED ENCRYPTION STANDARD (AES), Federal Information Processing Standards Publication 197, 26 November 2001
8. Wang, Q., Qu, G.: A silicon PUF based entropy pump. IEEE Trans. Dependable Secure Comput. **16**(3), 402–414 (2018)
9. Arafin, M., Gao, M., Qu, G.: VOLtA: voltage over-scaling based lightweight authentication for IoT applications. In: Proceedings of 22nd Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 336–341, January 2017
10. Gu, J., Qu, G., Zho, Q.: Information hiding for trusted system design. In: Proceedings of the 46th ACM/IEEE Design Automation Conference, pp. 698–701, June 2009
11. Qu, G., Potkonjak, M.: Intellectual Property Protection in VLSI Design: Theory and Practice. Springer Science and Business Media, Berlin, May 2007. https://orcid.org/10.1007/b105846