FOOTAGE FIRM, INC.

////////////////////
Ingrid Verbauwhede

Security Adds an Extra Dimension to IC Design

Future IC design must focus on security in addition to low power and energy

Internet of Things (IoT) devices, including medical implants, sensor nodes, wearables, automotive elements, and more, are typically immersed in the environment and extremely resource constrained. They operate using batteries

or scavenged energy: thus, optimizing for power and/or energy is of the utmost importance. However, to protect privacy, authenticate data or sources of information, and provide resistance to physical manipulation, we must add security, cryptographic capabilities, and other countermeasures to IC design.

The implementation of these cryptographic algorithms and protocols must be compact and energy

efficient but, at the same time, resistant to physical attacks and leakage of sensitive information from the device during computations. Adding countermeasures increases the overall cost and energy budget and adds an extra design dimension. Attackers will always go for the weakest link. Thus, security needs to be considered at every design step and requires its own testing strategies.

Digital Object Identifier 10.1109/MSSC.2017.2745799
Date of publication: 16 November 2017

The implementation of cryptographic algorithms and protocols must be compact and energy efficient but resistant to physical attacks and leakage of sensitive information.

This short article was motivated by a recognition of the need for security in embedded devices. It begins with the attacker model, discusses associated design and test methods, and offers several illustrative examples. The main focus is on digital hardware implementations for cryptographic algorithms, protocols, and security devices; the article does not address hardware solutions to protect against software attacks.

It is difficult to measure security as there are no commonly agreed upon units for such measurement. Therefore, security is based on an attacker model and evaluated against the assumptions made in this attacker model. For typical digital circuit design, we distinguish among the following broad categories: the black box model (used mostly in the past), the gray box model of current designs, and the immersed model of the future. Also, white box cryptography exists.

Black Box Attacker Model: The Past

Modern cryptography began in the 1970s with the first U.S. standard for symmetric key encryption, the Data Encryption Standard (DES); the invention of public key cryptography; and the development of the public key Rivest–Shamir–

Adleman (RSA) algorithm. These developments appeared in conjunction with novel electronic communication means and were spurred by the electronics revolution.

The DES algorithm was developed with efficient hardware implementation in mind. Indeed, it performs poorly in software. Cryptography, in this case, is applied to protect the information flow between two communicating parties. For the attacker model, we assume that the devices owned by Alice and Bob (the two main “characters” in every cryptographic protocol) operate in black boxes, meaning that only the cryptographic inputs and outputs of the devices can be observed by Eve, the attacker. This is illustrated in Figure 1, where each device has a root of trust, indicated by the green dots. Examples are computers and servers in computer rooms or offices.

Typical attacks are performed on the network connections between devices, e.g., a local area network, the Internet, or any wireless link between devices. In the black box model, Eve collects input/output pairs (i.e., plaintext/ciphertext pairs), which she uses to guess the secret key. She is also allowed to modify input data or adaptively supply her own input data. The security strength in this

model is based purely on the computational complexity of the underlying cryptographic algorithms. If Eve succeeds in guessing the key more quickly than brute force (i.e., faster than trying all possible keys), then the cryptographic community considers the algorithm broken.

In this context, Moore’s law is important for the algorithm designer. Indeed, Moore’s law helps the attacker, as it gives him/her more computational power for brute-forcing cryptographic algorithms. The consequence is that key lengths for cryptographic algorithms keep growing. First, the European network ECRYPT and later the European Union Agency for Network and Information Security published documents regarding required key lengths for near- and long-term security [3]. For long-term security, the suggested key lengths are 256 for the symmetric key size, 512 for the hash output size, 15,360 for the RSA modulus size, and 512 for bit elliptic curves.

As a result, the digital hardware designer often has to design for very large word lengths in combination with unusual arithmetic. The most critical component of secret key algorithms such as DES or the Advanced Encryption Standard (AES) are the “substitution boxes,” or Sboxes. Special effort is expended to make these either compact or very fast [11]. The most critical components of public key algorithms such as RSA or elliptic curve-based cryptography rely on the implementation of finite-field arithmetic and, more specifically exponentiation algorithms and finite-field multipliers. The focus of digital hardware designers in the black box context is thus *efficiency*: small area, high throughput, low power, and low energy. In this sense, the IC design process is no different from optimizations in other fields, such as image, video, or communications applications.

Testing of black box security circuits focuses, therefore, on functional correctness, as well as on measuring throughput, power, and energy. Care must be taken to ensure that

FIGURE 1: An illustration of the black box attacker model, with the green dots representing the devices’ roots of trust.

sensitive registers (such as key registers) are not included on scan chains or that these scan chains are disconnected before shipment. Fortunately, because cryptographic algorithms aim at maximum confusion and diffusion, fault coverage is typically very high for testing cryptographic algorithms.

Gray Box Attacker Model: The Present

Today, electronic devices and computations are more dispersed and distributed. In the IoT context, we assume that electronics are in the hands of the user, e.g., mobile devices, smart cards, game consoles, and other gadgets. Sensor nodes are distributed in the environment, and medical devices are implanted into our bodies. Electronics are everywhere in our cars: to monitor sensors and drive actuators, provide an entertainment system, enable autonomous driving, and more. Security, reliability, and safety are of the utmost importance.

In this context, we assume an attacker model in which the attacker has access to both the communication channel and the devices themselves, as illustrated in Figure 2. Instead of using only input/output plaintext/ciphertext pairs, the attacker obtains extra information by observing (a passive attack) or manipulating (an active attack) one IC *while* it is performing the calculations on the embedded device. This extra side-channel information can take many forms.

Timing Attacks

In a timing attack, the attacker will observe differences in execution time, depending on the values of the key or other sensitive data. Cache attacks are a well-known example. They are effective for table-based implementations of the Sboxes of symmetric key algorithms: if data in the cache depend on the key, then timing differences leak information [1].

In reaction, native AES instructions were added to high-end x86 processors: these both improve performance and run in data-independent time [10].

For typical digital circuit design, we distinguish among the black box model (used mostly in the past), the gray box model of current designs, and the immersed model of the future.

Public key algorithms, such as RSA or elliptic-curve public key schemes, typically scan the key bits in a bit-serial fashion. A sequential implementation that contains key-dependent if-then-else structures will leak information: when the if branch takes a different execution time than the else branch, the difference in execution time again leaks information concerning the key. Therefore, huge effort is spent to make sure that a cryptographic implementation in hardware or software runs in constant time.

For instance, it is crucial that finite-state machines consume exactly the same amount of cycles, independently of the data being processed. Unfortunately, most hardware optimization and synthesis tools (and also software compilers) will remove the dummy code or logic that was added to make a circuit or implementation constant-time, because it is considered redundant or dead code. Hence, there is a need for tools that check or create constant-time implementations [7]. Note that timing attacks can also be executed remotely; thus, they are also a concern for cloud and server implementations.

Side-Channel Attacks: Simple and Differential

Similarly, data-dependent variations in power consumption or electromagnetic (EM) radiation can reveal sensitive data. Collectively, these are called *side-channel attacks*. These require that the attacker is in close proximity to the device and can monitor the power consumption or pick up the EM radiation. Such attacks are passive: they monitor the device without disturbing its normal operation. The danger of passive attacks is that the device itself might not be aware that it is being monitored, which is further evidence of the need to develop circuits that can detect observation [4].

Simple power attacks (SPAs) rely on one or, perhaps, a few power or EM measurements to obtain the secret key. Template attacks are one example of an SPA; here, a template of the device under attack is available. Creating the template could take a huge number of measurements, but the attack on the actual device might take only one or, at most, a few such measurements. (The term *SPA* is somewhat misleading because, in

FIGURE 2: An example of the gray box attacker model, with the green dot representing the device's root of trust.

In the future, we expect that the capabilities of attackers will improve and that we will evolve from a gray testing box model toward an immersed model.

general, SPA attacks are not that simple to mount.)

Differential power attacks (DPAs) and EM attacks typically require multiple (in the hundreds or, sometimes, millions) power or EM traces to guess cryptographic secrets. In a typical setup, the attacker creates a model for the power consumption behavior of the device. For example, the attacker assumes that the power consumption is related to the Hamming distance between current and previous data values stored in flip-flops or registers. Indeed, this closely corresponds to the dynamic power consumption profile in standard complementary-metal-oxide-semiconductor designs, where power is consumed only when the state of the flip-flop changes. Up until now, we have typically ignored leakage current and other noisy effects because dynamic power is still dominant. DPA attacks are quite robust and resilient to noise: noisy uncorrelated

effects can be reduced by increasing the number of measurements.

To protect symmetric key algorithms such as AES, two main classes of countermeasures have been developed. One is a set of circuit styles that make the switching activity of gates independent of the data being processed: it is based on dynamic differential logic styles [12]. Masking techniques are a second major class of countermeasures: these are based on randomly splitting sensitive data into multiple shares and obtaining a subset of shares that do not disclose those actual data [8]. Public key algorithms are usually protected at the algorithm level (by adding randomness to the calculations) with techniques such as key or scalar blinding or the use of projective coordinates, making it difficult for the attacker to obtain multiple power or EM measurements.

In practice, side-channel information is used to reduce the computational

complexity of a brute-force attack. As an example, the popular correlation power attack reduces the effort of attacking an AES key from 2^{128} to the size of the AES Sbox, 2^8 (the input data to the Sbox being correlated to the key). The attacker makes 256 possible Hamming distance power models and looks for the model that results in a maximum correlation with the measured power traces. This is then repeated for every key byte. Hence, a correlation attack is a divide-and-conquer attack: it combines side-channel information with brute-forcing 256 possible key byte guesses.

Fault Attacks

Fault attacks are an active attack but not necessarily an invasive one. Faults can be triggered by playing with the power supply, introducing clock glitches, freezing or heating the device (typically also applied to random number generators), or any other means that triggers faulty results [5].

IC designers have developed many test strategies to address random faults in ICs, including those based on scan chains, built-in-self-tests, and more. Many countermeasures, such as adding redundancy, could be reused to detect fault attacks. There are, however, two main differences between an attacker aiming to inject faults and a designer testing for random faults. The first is the simple fact that knowing whether a fault occurred (yes or no) could in itself disclose sensitive information and so needs to be avoided [2]. The second is that the designer must avoid having key registers and other sensitive data sitting on the scan chains. This would otherwise provide a very nice back door.

Side Note on White Box Cryptography

White box cryptography is used in the context of secure software distribution, where a cryptographic algorithm with an associated key is converted into a key-specific,

FIGURE 3: A graphic depicting an immersed model.

tamper-resistant piece of software. It is used when no hardware cryptographic module or trusted element is available. Mathematically, most white box schemes are broken, but specific applications such as digital rights management still make use of white box cryptography. However, this topic will not be further discussed here.

Future Attacker Models

In the future, we expect that the capabilities of attackers will improve and that we will evolve from a gray box testing model toward an immersed model, as shown in Figure. 3. The IoT is a complex system with lightweight, resource-constrained devices at the edge. These devices communicate with each other and with mobile portable devices, such as phones, personal computers, lap tops, etc. At the center, cloud computing and servers provide extraordinary compute power to all.

So far, in the gray box model, we assume that an attacker observes one device or one communication link at a time. In the immersed model, we assume that multiple nodes supported by malicious servers or botnets can conspire to attack one poor IoT device. Thus, besides providing protection with strong cryptographic algorithms and strong physical defenses against side-channel attacks, we also need strong cryptographic protocols that protect a system even if individual nodes fail. In this context, we see many interesting research challenges.

Because electronics are immersed into the environment, we need strong roots of trust. These roots of trust will be in hardware, they need to be minimal in size, and there should be some explicit proof that one can trust them. Included in this set will be physically unclonable functions to derive device-specific keys and embedded true-random-number generators that can resist a wide range of attacks. Second, there is a need to develop more hardware circuits to support software security and provide, for instance,

In the immersed model, we assume that multiple nodes supported by malicious servers or botnets can conspire to attack one poor IoT device.

control flow integrity, software integrity, remote attestation, and so on. A third challenge is the development of quantum computers. Major classes of current public key algorithms based on RSA and elliptic curves will be broken if/when quantum computers appear. Mathematicians are developing a new generation of algorithms, but the challenge for the digital hardware designer is to implement them, including all current software, side-channel, and fault attack countermeasures. Also lacking currently is support for security by design methods and tools.

To conclude, the future of secure digital hardware design will require a complex interaction among digital circuit design, computer arithmetic, cryptography, and evolving technology. And, while in the past, we worried only about energy and power minimization, we now have to face the additional constraint that our circuits need to be resistant to a wide range of attacks.

References

- [1] D. J. Bernstein. (2004). Cache-timing attacks on AES. [Online]. Available: <https://cr.yp.to/papers.html#cachetiming>
- [2] D. Boneh, R. DeMillo, and R. Lipton, "On the importance of eliminating errors in cryptographic computations," *J. Cryptol.*, vol. 14, no. 2, pp. 101–119, Mar. 2001.
- [3] N. P. Smart, Ed. ENISA. (2014). Algorithms, key size and parameters report. [Online]. Available: <https://www.enisa.europa.eu/publications/algorithms-key-size-and-parameters-report-2014>
- [4] N. Homma, Y. Hayashi, N. Miura, D. Fujimoto, M. Nagata, and T. Aoki, "Design methodology and validity verification for a reactive countermeasure against em attacks," *J. Cryptol.*, vol. 30, no. 2, pp. 373–391, 2017.
- [5] D. Karaklajic, J. Schmidt, and I. Verbauwhede, "Hardware designer's guide to fault attacks," *IEEE Trans. Very Large Scale Integration Syst.*, vol. 21, no. 12, pp. 2295–2306, 2013.
- [6] P. Maene, J. Goetzfried, R. de Clercq, T. Mueller, F. Freiling, and I. Verbauwhede, "Hardware-based trusted computing architectures for isolation and attestation," *IEEE Trans. Computers*, 2017, to be published.

- [7] O. Reparaz, J. Balasch, and I. Verbauwhede, "Dude, is my code constant time?" in *Proc. Conf. Design, Automation and Test in Europe*, 2017, p. 6.
- [8] O. Reparaz, B. Bilgin, B. Gierlichs, S. Nikola, and I. Verbauwhede, "Consolidating masking schemes," in *Advances in Cryptology, Lecture Notes in Computer Science*, vol. 9215, R. Gennaro, and M. J. Robshaw, Eds. New York: Springer-Verlag, 2015, pp. 764–783.
- [9] V. Rozic, O. Reparaz, and I. Verbauwhede, "A 5.1μJ per point-multiplication elliptic curve cryptographic processor," *Int. J. Circuit Theory Applicat.* vol. 45, no. 2, pp. 170–187, 2016.
- [10] S. Gueron. (2012, Aug. 2). Intel advanced encryption standard (Intel AES) instructions set—Rev 3.01. [Online]. Available: <https://software.intel.com/en-us/articles/intel-advanced-encryption-standard-aes-instructions-set>
- [11] S. Satpathy, S. Mathew, V. Suresh, M. Anders, H. Kaul, A. Agarwal, S. Hsu, G. K. Chen, and R. Krishnamurthy, "250mV-950mV 1.1Tbps/W double-affine mapped Sbox based composite-field SMS4 encrypt/decrypt accelerator in 14nm tri-gate CMOS," in *Proc. VLSI Circuits*, 2016.
- [12] K. Tiri and I. Verbauwhede, "A digital design flow for secure integrated circuits," *IEEE Trans. Computer-Aided Design Integrated Circuits Syst.*, vol. 25, no. 7, pp. 1197–1208, 2006.

About the Author

Ingrid Verbauwhede (ingrid.verbauwhede@esat.kuleuven.be) heads the embedded systems and hardware team of the research group COSIC at the KU Leuven in Belgium. She is an adjunct professor in the Electrical Engineering Department at the University of California, Los Angeles, and a member of the Royal Academy of Belgium for Science and the Arts. She received a European Research Council Advanced grant in 2016, as well as the 2017 IEEE Computer Society Technical Achievement Award. She is a pioneer in the field of efficient and secure implementations of cryptographic algorithms in embedded contexts on application-specific ICs, field-programmable gate arrays, and embedded software.