Trojan Side-Channels: Lightweight Hardware Trojans through Side-Channel Engineering

Lang Lin[§], Markus Kasper[†], Tim Güneysu[†],
Christof Paar[†§], and Wayne Burleson[§]

[§]Department of Electrical and Computer Engineering
University of Massachusetts, Amherst, USA
[†]Horst Görtz Institute for IT Security
Ruhr University Bochum, Germany
{llin,burleson}@ecs.umass.edu
{gueneysu,mkasper,cpaar}@crypto.rub.de

Abstract. The general trend in semiconductor industry to separate design from fabrication leads to potential threats from untrusted integrated circuit foundries. In particular, malicious hardware components can be covertly inserted at the foundry to implement hidden backdoors for unauthorized exposure of secret information. This paper proposes a new class of hardware Trojans which intentionally induce physical side-channels to convey secret information. We demonstrate power side-channels engineered to leak information below the effective noise power level of the device. Two concepts of very small implementations of *Trojan side-channels* (TSC) are introduced and evaluated with respect to their feasibility on Xilinx FPGAs. Their lightweight implementations indicate a high resistance to detection by conventional test and inspection methods. Furthermore, the proposed TSCs come with a physical encryption property, so that even a successful detection of the artificially introduced side-channel will not allow unhindered access to the secret information.

Key words: Trojan Hardware, Side-Channel Analysis, Covert Channel, Trojan Side-Channel, Hardware Trojan Detection

1 Introduction

Historically the choice to implement cryptographic routines in hardware was mainly driven by high-security applications such as banking or government systems. Nowadays, this has changed since the trend towards system-on-a-chip solutions has facilitated the integration of high-performance cryptography also in commercial off-the-shelf silicon devices. For example, the majority of current PCs and laptops are sold with built-in trusted platform module (TPM) chips. Another pertinent example is the trend towards pervasive embedded computing, bringing hardware cryptography to products such as smartphones, payment cards, RFID-equipped goods or medical devices. Since these personal devices can

be physically accessed by their owners, their security often relies on hardware-based security modules. In this context, security modules implemented in silicon are generally considered more trustworthy than software solutions. This is mainly due to the need for expensive tools to modify and probe circuits on the submicron scale, which imposes a significant barrier to attackers and thus implicitly provides a basic level of protection against key extractions and algorithm manipulations.

Due to the recent fabless trends in semiconductor industry, malicious circuit manipulations such as "hardware Trojans" can be furtively implanted into the genuine integrated circuits (IC) to compromise their security. Such attacks on the hardware design of chips lead to serious consequences, as (1) a very large number of devices will be affected, e.g., millions of e-banking authentication tokens or TPM chips, (2) the attack might not be noticed for a very long time and perhaps more importantly, (3) security breaks of this kind are almost impossible to fix because there is no practical hardware equivalent for software updates. Even though there is no solid evidence on malicious manipulations of commercial devices at manufacturing time up to now, "Trojan hardware" is considered a serious threat for security modules of all kinds [1]. Given the complexity of the current semiconductor supply chain including fabless semiconductor companies, there is an urgent need to implement organizational measures [2] to enable trusted IC manufacturing.

Several recent academic works highlighted potential threats by demonstrating concepts of possible hardware Trojans. In [3] a malicious core embedded into a central processing unit (CPU) is proposed. This work rates the detectability of such Trojans as low mainly due to the small hardware overhead (a total of 2300 gates for the Trojan circuits) and timing perturbations. In [5], the register-transfer-level (RTL) netlist of a cryptographic application on reconfigurable hardware is manipulated with additional logic to implement malicious hardware. This Trojan has a complicated triggering pattern that will most likely remain undetected in conventional function tests. Another work highlights the possibilities to use hardware Trojans to covertly leak secret information through wireless channels such as thermal, optical and radio channels [6]. Nevertheless this work still requires trigger circuitry and most of the proposed channels are realized by signals on output pins. This might be a drawback when it comes to the detectability of the malicious circuitry.

Well-designed cryptographic hardware modules are very difficult to be analyzed or modified. However, for the same reasons it is also difficult for chip-designers to detect malicious manipulations that were introduced to their circuitry during the manufacturing process. Modern ICs often contain large blocks of unused circuits, which may be left from previous versions of the design or for temporary testing purposes. Malicious hardware can be hidden in these unused chip-areas. As long as the TSC circuits are tiny, for example less than 100 gates, they cannot be easily distinguished from other hundreds of thousands of gates by basic chip layout inspection.

To detect hardware Trojans, three general approaches [7] have been proposed. The failure analysis community employs sophisticated techniques for visual in-

spection such as optical and scanning electron microscopy (SEM) or even picosecond imaging circuit analysis [8]. These methods are very dependent on laboratory instrumentation and are often not feasible to be applied to production-run ICs. Other approaches generate test patterns using the standard VLSI fault detection tools to find unexpected device behavior generated by malicious hardware [9, 10]. However, this method may not detect Trojans with complicated triggering patterns or carefully hidden channels that are leaking information. The third approach profiles an IC by various analog measurements of, e.g., power traces or internal delays. Then the profile of a trusted Trojan-free IC is used as reference for analyzing suspicious ICs. In [11], the power consumption of ICs is profiled to detect Trojans by means of a subspace projection analysis [12]. Other works based on side-channel and path delay profiles are described in [13, 14]. Evaluators following this approach can only detect Trojan hardware circuits if at least 0.1-0.01% of the pristine ICs circuit area is modified [11]. This translates into hundreds of extra gates for most modern devices. To summarize, there are no feasible evaluation techniques to our knowledge that can detect hardware manipulations with very small gate counts.

In general, IC function tests do not include side-channel analysis (SCA). SCA allows to extract information from physical channels inherently existing in electronic devices. The most commonly analyzed side-channels are the power consumption and the electromagnetic radiation [15] of running ICs. In addition, many other physical properties like timing behavior or sound waves [16, 17] have been shown to leak exploitable information. During the last decade many different side-channel analyses have been demonstrated, that exploit physical information leakage to compromise the security of cryptographic routines especially in embedded devices.

So far to the best of our knowledge, most research works examine side-channels as undesired phenomena that require special attention to protect devices from sophisticated attacks. In this paper we change this perspective and use intentionally introduced side-channel leakage as a building block for Trojan circuitry. We propose and demonstrate hardware Trojans which are much more subtle than introducing complex logic blocks, but still can completely compromise otherwise secure hardware by leaking exploitable information. In particular, we design Trojans using less than 100 gates to generate artificial power side-channels suitable to covertly leak secret information. We refer to these covert channels as “Trojan side-channels” (TSC).

This paper is organized as follows: Section 2 introduces the basic concepts of TSC. Section 3 presents two different approaches for TSCs that are built upon spread-spectrum theory and artificial leakage functions induced on the key schedule of cryptographic algorithms. In Section 4, we discuss further work and research related to the concept of TSCs, before we finally draw conclusions in Section 5.

2 Introducing the Trojan Side-Channel

Before discussing the concept of Trojan side-channels, we introduce the parties involved in a Trojan side-channel scenario. We refer to the party implanting the Trojan hardware into the genuine ICs as *attacker*, and the party attempting to detect infected ICs as *evaluator*. An attacker could be for example an untrusted IC foundry, and an evaluator, who attempts to verify correctness and integrity of an IC, a Common Criteria Testing Laboratory (CCTL).
Furthermore we assume that the designers implemented all cryptographic schemes on their device with state-of-the-art countermeasures against SCA attacks, such as the use of side-channel resistant logic styles. Hence, the initial implementation on the chips should be considered as side-channel resistant and not exposing any secret information by side-channel leakage. Note that a device protected at this level is likely to be evaluated according to its side-channel resistance. Thus, a Trojan implanted in such a device needs to be designed to evade detection even during evaluation of the pristine IC's side-channel resistance by sophisticated methods such as higher-order power analysis or template attacks.

Based on these requirements we define the following design goals for a circuit-level implementation of a TSC:

- Detectability:
  - Size: The required amount of logic gates has to be minimized to evade detection of the TSC by evaluators.
  - Side-Channel Leakage: The TSC must not be detected when performing power analyses targeting the pristine ICs functionality. As a minimum requirement, the relative power consumption of the TSC circuit with respect to the overall power should be negligible so that it cannot be obtained from the power traces just by visual inspection.
  - Trigger: The TSC must not effect functionality of the device in any way to avoid detection by extensive functionality testing. This also prohibits triggering and communication using external pins.
- Usability:
  - The TSC must not be exploited by anyone else than the attacker, who knows all details of the modification. We call this "encryption property".

The principle of a TSC is visualized in Figure 1. The compromised device is modeled by an IC having an embedded crypto core. Without TSC, the secret key K cannot be recovered by means of SCA. During the IC manufacturing process, the attacker covertly inserts a TSC circuit that encodes K into physical leakage. We model this encoding by an encryption function[1] $e(K)$, which is designed to reserve usage and detection of the side-channel only to the implementing attacker. Once the IC is deployed in the field, an evaluator must not be able to detect the TSC. Furthermore, the encryption property of the Trojan is designed to avoid usage of the Trojan even by an evaluator that is aware of the existence of the TSC.

[1] Note that the notion of encryption in this context does not necessarily imply the strong security properties as commonly assumed for cryptographic schemes.

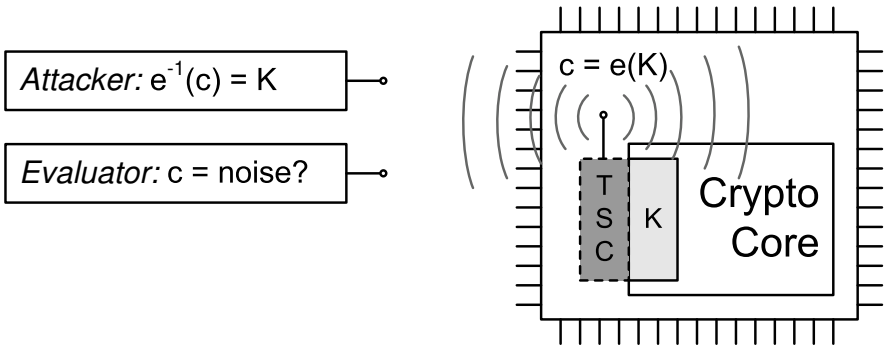


Fig. 1. Principle of Trojan side-channels

TSCs that incorporate such an encryption property require special attention during IC security evaluation. A good encryption property requires an evaluator to overcome an infeasible computational or experimental effort, e.g., 2^{80} calculations or 2^{40} measurements, to access the secret information that is exposed. On the other side the attacker (who designed the TSC) needs to have some advantageous knowledge, allowing him to make use of the TSC within feasible efforts.

The class of TSCs introduced in this work use an internal state (artificially introduced as part of the TSC or an inherently existing internal state) to encode the secret information to be leaked by means of a logic combination function. The output of this logic is then connected to a building block acting as an artificial leakage source. For the side-channel based on power consumption, such a leakage circuit can be realized, for example, using big capacitances, toggling logic or pseudo-NMOS gates. Note that the amount of generated leakage is part of the TSC design space and can be engineered to take any desired signal-to-noise ratio (SNR). This choice of a SNR affects both, attacker and evaluator, as it determines the amount of measurement samples required to detect and use the TSC. In addition, it might also affect the size of the TSC.

3 Implementations of TSCs

In the following sections, we demonstrate two very different ways to implement TSCs. These examples shall provide a first impression of the flexibility of the introduced concept of TSCs and highlight the wide range of design options available to attackers. Although TSCs aim to be implemented in ASICs, we preliminary demonstrate their capability by practical experiments on FPGA implementations. Note that these FPGA implementations should be regarded as proof-of-concept implementations only, because FPGAs are (due to their course-grain logic elements) not able to fully mimic the design options and structural characteristics that apply for ASICs.

3.1 TSCs based on spread-spectrum theory

Our first TSC design is a side-channel adapting the concepts from spread-spectrum communications (also known as code-division multiple access (CDMA)) to distribute the leakage of single bits over many clock cycles. The basics of the CDMA encoding are very similar to conventional stream-ciphers. A bit sequence (the code) is used to modulate information bits using XOR. Contrary to the stream-cipher concept, CDMA uses many code bits to transfer single bits of information, i.e., the code bits are changing much faster than the information bits. This strategy spreads the information contained in a single bit along a longer bit (or code) sequence which allows transmission and recovery of information in subliminal channels even below noise level. This property makes CDMA the method of choice to implement hidden military communication channels. The demodulation used to decode CDMA channels helps to understand how CDMA can establish channels in this sub-noise domain. The process of decoding using a correlation demodulator is very close to what the community of cryptographers knows as correlation power analysis. The demodulator uses subsequent power measurements and correlates them to the synchronized code sequence. If the correct code has been used, this leads to a positive correlation coefficient for encoded zeros and a negative correlation coefficient for encoded ones. The more power traces the demodulator analyzes, the more "process gain" (which is the ratio of code sequence length to the bit information length in spread-spectrum encoding) is available to overcome a low demodulation SNR. Note that the CDMA channel can only be demodulated using the correct code sequence and demodulation with different code sequences will not lead to any significant correlation. Therefore it is possible to transfer bits on multiple CDMA channels simultaneously, as each CDMA channel is indistinguishable from noise for all other channels.

Our TSC employs this method by using a pseudo-random number generator (PRNG) to create a CDMA code sequence. This sequence is then used to XOR-modulate the secret information bits. The modulated sequence is forwarded to a leakage circuit (LC) to set up a covert CDMA channel in the power side-channel. In this model, the advantage of the attacker is the knowledge about the exact setup of the code generator (more precisely, the initialization vector and feedback coefficients of the implemented PRNG) which are required to predict the code sequence. Knowing all details of the PRNG used gives the attacker the essential advantage over evaluators who cannot distinguish the covert channel from noise. For decoding this channel, the attacker performs a correlation demodulation on measurement points of subsequent clock cycles as described above. When evaluating side-channel leakage of the compromised device, the leakage due to the TSC will not be detected during attacks on the pristine IC core. Note that depending on the leakage generating circuit, it might be necessary to consider a mapping of the used code with respect to a suitable power model prior to correlation-based demodulation. Note further that to amplify the quality of results, the attacker can repeat measurements several times and average the corresponding traces to reduce the overall impact of noise.

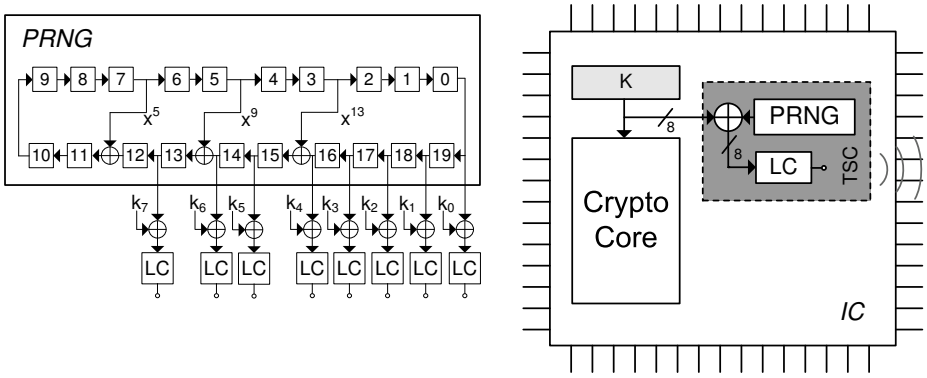


Fig. 2. Diagram of a spread-spectrum TSC circuit based on a pseudo-random number generator (PRNG) and a separate leakage circuit (LC).

Experimental Results To demonstrate the applicability of the proposed TSC, we implement an entire AES cryptosystem and a TSC using the linear feedback shift register (LFSR) shown in Figure 2 on a Xilinx Spartan-3E FPGA running at 50 MHz. We use a 1 Ω serial current-sensing resistor to probe the power consumption of the FPGA core, similar to the experimental setups described in [22, 23]. The transient power traces are measured by an Agilent Infiniium 54832D oscilloscope. In this experiment, we used a single CDMA channel to transmit a single key bit. The leakage generating circuit (LC) was realized by connecting eight identical flip-flop elements to the single output of the XOR gate to mimic a large capacitance. To demodulate the CDMA channel, the attacker has to take into account that the flip-flops do not leak a good Hamming weight signal. Therefore, the power consumption behavior of the leakage circuit has to be modeled to map the CDMA code to the leakage circuit. In this case the flip-flops will cause short circuit currents when toggling, and additional currents from discharging their load capacitances on one-to-zero transitions. The latter can be used to distinguish the transmitted data during demodulation: For encoded bits which are one, the circuit will have a higher leakage on all zero-to-one transitions of the code, while for encoded zeros the circuit will generate more leakage currents on one-to-zero transitions of the code. The attacker uses the resulting two different code sequences for demodulation. The higher correlation coefficient will identify the actually transferred bit.

In our setup, the PRNG is based on a simple LFSR using the primitive polynomial $x^{20}+ x^{13}+ x^9+ x^5+ 1$. This PRNG thus generates a code sequence with a maximum order of 2^{20}-1. We successfully extracted the secret key bit with code lengths of at least 1030 bits. In Figure 3, the correlation coefficient for the code sequence identifying the transmitted bit is with 0.04 considerably larger than the correlation coefficients resulting from wrong choices of code sequences and false guesses of the transmitted bit. The FPGA implementation of this TSC circuit requires 42 flip-flops (FF) and 6 look-up tables (LUT) occupying

23 slices (49 gates would be required for the corresponding ASIC circuit). Note that these resource consumption is nearly negligible with respect to our iterative AES implementation that requires 531 slices utilizing 442 FFs and 825 LUTs (this does even not take additional resources for control or I/O into account).

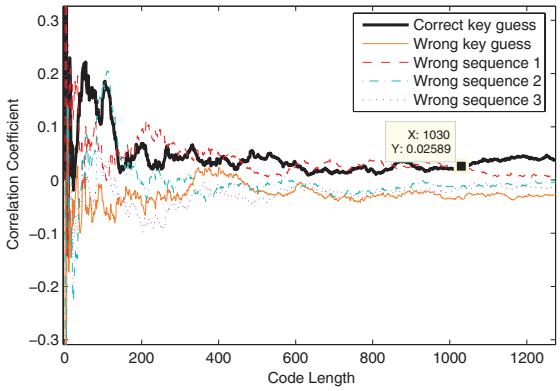


Fig. 3. Required number of power traces to extract key bits from TSC based on spread-spectrum technique

Detectability The additional leakage of this spread-spectrum inspired TSC only depends on the key and a fixed code-generator. An evaluator can exploit this, by first averaging many traces using a fixed input and then, in a second step, averaging over another set of traces, where only a single key bit has been flipped. By calculating the difference of means of those traces he might get enough information to recover the code sequence that has been used to leak the toggled key bit. To harden the proposed TSC against this method we suggest to transfer only combination of bits, such that further interpretation by means of algebraic equations is required to understand and use the TSC. An alternative method to prevent unauthorized use of our TSC by an evaluator is to create an interdependence between plaintext input and initialization vector of the PRNG, i.e., either we generate the PRNG's initialization value based on a combinatorial circuit from the plaintext or we introduce a previous initialization phase and clock all plaintext bits through the PRNG using a separate shift register and an additional XOR gate (similar to the initialization of the A5/1 streamcipher). Although these approach requires additional logic and/or clock cycles, this involves a significantly higher effort for the evaluator to get access to the Trojan's communication channel.

3.2 TSCs using known input values

Our second proposal is a subtle TSC that leaks secret information obtained during the run of a block-cipher's key schedule. More precisely, we demonstrate an attack on the AES-128 block-cipher and its corresponding key schedule in this section. While the TSC following spread-spectrum theory used subsequent leakages to implement a CDMA channel, this design was inspired by conventional side-channel analysis. The idea is to artificially introduce leaking intermediate states in the key schedule that depend on known input bits and key bits, but that naturally would not occur during regular processing of the cipher. These values can then be exploited by differential power analysis attacks. The TSC proposed uses an artificial intermediate state consisting of only a single bit of generated leakage. We evaluated several functions for combining input bits with key bits to find methods that

1. theoretically allow a good discrimination of wrong and right key bit guesses in a differential power analysis using the correlation coefficient,
2. use only a few input bits (≤ 16),
3. require a few logic gates (< 100),
4. do not lead to any exploitable correlation in case intermediate values the crypto-core processes during his regular operation are attacked.

For our demonstration TSC, we selected a very straightforward AND-XOR combination that is based on up to 16 input bits, but promises easily detectable results. This function uses AND conjunctions to pairwise combine each key bit with another input bit. The output of the AND gates are then combined to the leaked intermediate value by XORing all of them.

The encryption property of this TSC results from the attackers choice to select which of the 128 input bits of the AES cipher are combined with which key bits. Note that the attacker's secret to access the TSC is based on a permuted choice; in other words it can be implemented solely by wiring. For the sake of simplicity, we propose to leak 1-2 bytes of the AES round key for each round of the key schedule. This could be, for example, the first and the third byte of each round key. Note that if only one byte is leaked, the key space is already reduced sufficiently ($128 - 10 \cdot 8 = 48$ bit) for an attacker to mount a brute-force attack on a known plaintext-ciphertext pair, e.g., by using special-purpose hardware as a COPACOBANA [4]. On the other hand, designing the TSC to extract two bytes simultaneously enables the attacker to reveal the whole key without the need to apply brute-force to any unknown bits.

Analyzing the available key space, we consider an intermediate value generated as described above using 8 key bits and 8 input bits. In AES-128, we have 16 possible key bytes that can be leaked by a TSC[2]. There are $128!/120!$ different choices for selecting a sequence of 8 different bits from 128 bits. Therefore, we estimate the effective key space to $128!/120! \cdot 16 \approx 9.22 \cdot 10^{17}$ possible keys, which

[2] It makes sense to use complete bytes instead of unrelated bits, due to the SBOX inside the g-function of the AES key schedule. This facilitates reconstruction of the AES key from the leaked round key snippets.

corresponds to a key length of approximately 59 bits. For combination functions using 16 bits the keyspace is significantly larger.

Experimental Results We evaluated this TSC by implementing an AES key schedule connected to the proposed TSC on a SASEBO standard side-channel evaluation board [18]. The target FPGA on this board is a Xilinx Virtex-2 PRO XC2VP7-5 FPGA. The implemented TSC is sketched in Figure 4 and is based only on 8 plaintext bits and 8 bits of the round key register within the AES-128 key schedule. Note that many real-world implementations store the plaintext input in the state which is overwritten in subsequent rounds. Hence, the TSC might require an additional register to store the used plaintext bits for the entire runtime of the attack.

In this setup, we additionally employed a leakage circuit (LC) that can be implemented very efficiently with FPGAs (although our main focus are TSCs on ASICs, this could be a solution for the case that an FPGA is the target device). We configured a single LUT of the FPGA as 16-bit shift register (SRL16 feature) and loaded it with an initial alternating sequence of zeros and ones. The shift register is only clocked in case the input to the leakage circuit is one, which results in an additional dynamic power consumption. Including the logic from the combinatorial circuit, the overall size of the implemented TSC results to only 14 LUTs occupying a total of 7 slices (equivalent to 29 gates when implemented as ASIC).

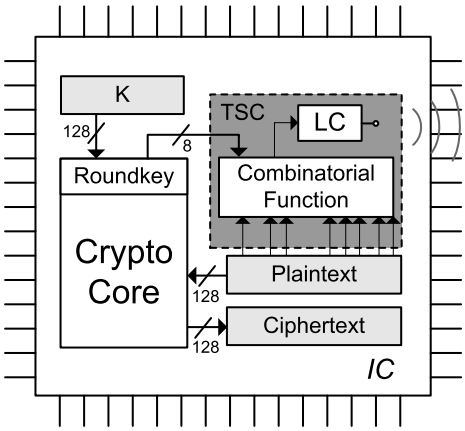
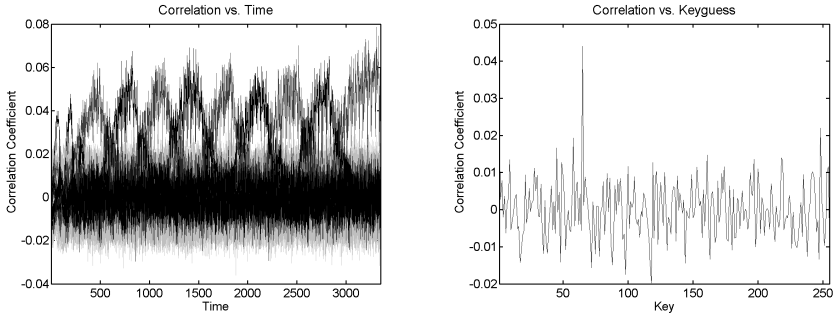


Fig. 4. Diagram showing an alternative TSC circuit attacking 8 bits of an AES-128 key schedule. It consists of a combinatorial circuit taking a set of plaintext and round key bits as inputs that are combined into a single output bit, finally leaked into the power signature using a leakage circuit (LC).

Our experimental results demonstrate that a recovery of the TSC information can easily be achieved by means of a correlation power analysis. The plots of the correlation coefficient show very distinct peaks for correct key guesses. The example detection given in Figure 5 shows 10 peaks each indicating a covert transmission of another key byte. The figure to the right shows the correlation coefficients used to identify the content of the fourth transmitted byte as the value 65.



(a) Subsequent exposure of 10 key bytes (b) Discrimination of the key byte for the 4th transmission

Fig. 5. Recovery of the TSC Information by means of the correlation coefficient.

Detectability This TSC approach can also be detected and exploited by diligent evaluation of the device. In this case an evaluator again uses variations of traces that differ only in single plaintext or key bits. This allows him to distinguish which key bits and plaintext bits were used in the combination function. While flipping combinations of bits, analysis of the corresponding behavior of the TSC will finally allow to reverse-engineer the used combination function.

To make our TSC less susceptible for these types of detection strategies, we suggest to extend the applied combination function by XORing additional plaintext bits, logic combinations of plaintext bits, key bits or even combinations of key bits. By introducing such complex linear or non-linear equations, the interpretation of the observed differences by the evaluator can be complicated to an unfeasible level. Such an improvement also assists the attacker to detect a key that only consists of zeros. Up to now, the detection of the zero key is based on the absence of any correlation for all other key guesses, which might be an undesired property.

4 Further work

In this work, we introduce a general concept of Trojan side-channels and still let room for many improvements for both the implementation and the detec-

tion perspective. In this section we discuss aspects beyond the scope of this paper that are subject to (our) current research: the detectability and availability of TSCs by third parties and evaluators has not been sufficiently verified experimentally yet. Upcoming designs of TSCs in ASICs have to be tested with special attentions to these properties, so that their threats can be better understood and judged. Improvements to the TSCs include smaller and more subtle combination functions, better leakage circuits and more elegant methods for CDMA code generation. For example, the shift register of the input dependent TSC could be omitted by shifting the contents of the register containing the plaintext bits instead. The leakage circuit could even completely be omitted by using a combination function with well-designed leakage behavior. For a first idea on TSC performance in ASICs, we have already performed experiments using circuit-level simulations of the CDMA-based TSC implemented in a 45nm predictive transistor model. These experiments also indicate feasibility of our concept in real hardware implementations beyond the limited scope of FPGAs with its very coarse-grain logic elements. Therefore, the ASIC implementations of the TSCs require much less gates than the logic required for our preliminary FPGA experiments.

5 Conclusions

Trojan side-channels form a subtle class of hardware Trojans that are very promising to evade detection strategies applied by evaluators. The known methods for detection of "conventional" Trojan hardware circuits, such as optical inspection of metal layers or fingerprinting of circuits, will most likely fail on TSCs due to their extremely small size. TSCs also do not require a direct connection to I/O pins and do not effect the functional behavior of the pristine IC. Since TSCs are only dependent on inputs of the cryptographic implementation under attack, the design space of TSCs allows for a multitude of potential TSC properties. The two types of TSCs demonstrated in this work show how considerable these differences might be, resulting in completely different TSC applications and detection schemes. Moreover, the degrees of freedom include the selection of

- physical channels (power, EM radiation, timing, heat, etc.),
- combination functions,
- internal states determining also time and/or input dependence of the leakage
- leakage circuits depending on their input or on transitions of their input

TSCs implemented during the manufacturing process in untrusted semiconductor foundries pose a very serious threat to all kinds of security modules. We provided a short discussion on the detectability of our TSCs by side-channel evaluators. Given the flexibility of TSCs with better hiding methods, further research is required for evaluators to develop more practical detection schemes to recognize the next generation TSCs.

TSCs based on CDMA are an universal tool for leaking information independent of the cryptographic algorithm used. Moreover, information other than the mere key can be leaked through the CDMA side-channel. We would like to stress that CDMA-based TSCs can potentially also find applications in constructive uses.

- Since TSC can be viewed as a form of physical encryption, one can imagine other cryptographic protocols and applications using TSC as building blocks.
- TSC can be used for anti-counterfeiting in a straightforward manner: authentic ICs can identify themselves via the TSC by sending an ID, whereas illegal but functionally correct copies lack this capability.
- TSC can be used for conveying internal status information about a circuit to facilitate the testability of a circuit.

Acknowledgements

The work described in this paper has been supported in part by the European Commission through the ICT programme under contract ICT-2007-216676 ECRYPT II.

References

1. High Performance Microchip Supply, Annual Report by the Defense Science Board, `http://www.acq.osd.mil/dsb/`, 2008.
2. R. McCormack, It's Like Putting A Band-Aid On A Bullet Hole, *Manufacturing & Technology News*, `http://www.manufacturingnews.com/news/08/0228/art1.html`, 2008.
3. S. T. King, J. Tucek, A. Cozzie, C. Grier, W. Jiang, Y. Zhou: Designing and implementing malicious hardware. In: *Proceedings of the 1st USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, pp. 1–8, 2008.
4. T. Güneysu, T. Kasper, M. Novotný, C. Paar, and A. Rupp. Cryptanalysis with COPACOBANA. *IEEE Transactions on Computers*, 57(11):1498–1513, November 2008.
5. Z. Chen, X. Guo, R. Nagesh, A. Reddy, M. Gora, A. Maiti: Hardware Trojan Designs on BASYS FPGA Board. In: *Embedded System Challenge Contest in Cyber Security Awareness Week (CSAW)*, 2008.
6. F. Kiamilev, R. Hoover: Demonstration of Hardware Trojans. In: *DEFCON 16*, Las Vegas, 2008.
7. X. Wang , M. Tehranipoor , J. Plusquellic: Detecting Malicious Inclusions in Secure Hardware: Challenges and Solutions. In: *1st IEEE International Workshop on Hardware-Oriented Security and Trust (HOST)*, pp. 15–19, 2008.
8. J. M. Soden, R. E. Anderson, C. L. Henderson: IC Failure Analysis: Magic, Mystery, and Science. In: *IEEE Design & Test of Computers*, Vol. 14, pp. 59–69, 1997.
9. M. Banga, M. S. Hsiao: A Region Based Approach for the Identification of Hardware Trojans. In: *1st IEEE International Workshop on Hardware-Oriented Security and Trust (HOST)*, pp. 40–47, 2008.

10. R. Chakraborty, S. Paul, S. Bhunia: On-Demand Transparency for Improving Hardware Trojan Detectability. In: *1st IEEE International Workshop on Hardware-Oriented Security and Trust (HOST)*, pp. 48–50, 2008.
11. D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, B. Sunar: Trojan Detection using IC Fingerprinting. In: *IEEE Symposium on Security and Privacy*, pp. 296–310, 2007.
12. K. Fukunaga: Introduction to Statistical Pattern Recognition, Second Edition, Computer Science and Scientific Computing Series, Academic Press, September, 1990.
13. R. M. Rad, X. Wang, M. Tehranipoor, J. Plusquellic: Power supply signal calibration techniques for improving detection resolution to hardware Trojans. In: *International Conference on Computer-Aided Design (ICCAD)*, pp. 632–639, 2008.
14. Y. Jin, Y. Makris: Hardware Trojan Detection Using Path Delay Fingerprint. In: *1st IEEE International Workshop on Hardware-Oriented Security and Trust (HOST)*, pp. 51–57, 2008.
15. P. Kocher, J. Jaffe, B. Jun: Differential Power Analysis. In: *Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO)*, Lecture Notes in Computer Science, Vol. 1666, pp. 388–397, 1999.
16. P. Kocher: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: *Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO)*, Lecture Notes in Computer Science, Vol. 1109, pp. 104–113, 1996.
17. A. Shamir, E. Tromer: Acoustic cryptanalysis. `http://people.csail.mit.edu/tromer/acoustic/`
18. Resarch Center for Information Security (RCIS): Side-channel Attack Standard Evaluation Board (SASEBO). `http://www.rcis.aist.go.jp/special/SASEBO/index-en.html`
19. J. Proakis: Digital communications. Fourth edition, McGraw-Hill, 2000.
20. B. Gierlichs, K. Lemke-Rust, C. Paar: Templates vs. Stochastic Methods. In: *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, Lecture Notes in Computer Science, Vol. 4249, pp. 15–29, 2006.
21. J. Rajski, J. Tyszer: Primitive polynomials over GF(2) of degree up to 660 with uniformly distributed coefficients. In: *Journal of Electronic Testing: theory and applications*, pp. 645–657, 2003.
22. F. Standaert, L. Oldenzeel, D. Samyde, J. Quisquater: Power analysis of FPGAs: how practical is the attack? In: *International Conference on Field Programmable Logic and Applications (FPL)*, pp. 701–711, 2003.
23. S. Ors, E. Oswald, B. Preneel: Power-analysis attacks on an FPGA - first experimental results. In: *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, Lecture Notes in Computer Science, Vol. 2779 , pp. 35–50, 2003.