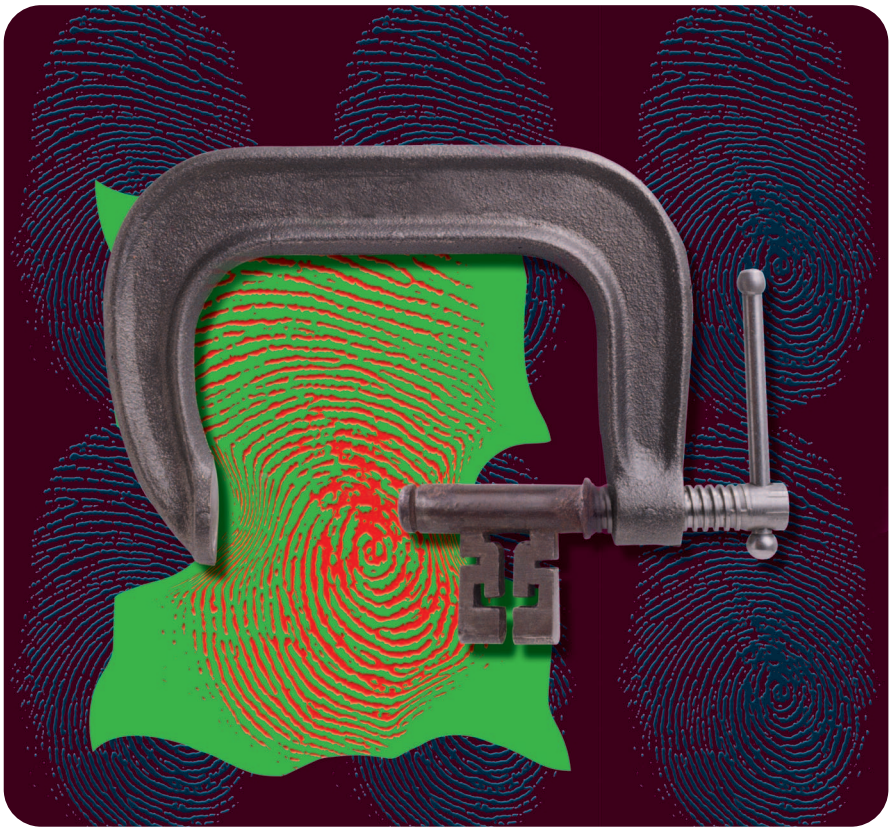
Maximizing the hash function of authentication codes

IOANNIS I. YIAKOUMIS, MARKOS E. PAPADONIKOLAKIS, HARRIS E. MICHAIL, ATHANASIOS P. KAKAROUNTAS, AND COSTAS E. GOUTIS

A DESIGN APPROACH to create small-sized, high-speed implementations of the keyed-hash message authentication code (HMAC) is the focus of this article. The goal of this approach is to increase the HMAC throughput to a level that can be used in modern telecommunication applications such as virtual private networks (VPNs) and the oncoming 802.11n. We focus on increasing the maximum operating frequency that, compared to commercially available IP cores, ranges from 30% to 390%. The proposed implementation doesn't introduce significant area penalty. More specifically, the overall increase in lookup tables required by our implementation is less than 10% compared to that of other implementations.



Implementing hash functions

Hash functions are common and critical cryptographic primitives. Their primary application is combined use with public-key cryptosystems in digital signature schemes. Hash functions compress a string of arbitrary length to a string of fixed length. Their main purpose is to produce a fingerprint of a message or some other block of data that will provide a high level of security for communication protocols.

Implementing a hash function on hardware presents numerous advantages. Hardware implementations present higher throughput than software, thus being more adaptable for high-speed applications. They operate without interuption, contrary to software implementations in a multitask environment. Hardware provides higher level security than software in cases of hacking attempts.

The most widespread functions are secure hash algorithm-1 (SHA-1) and message digest (MD5). These two hash functions are widely known for being used in the HMAC, which is used in numerous communication applications to address authentication issues.

The SHA-1 hash function is selected for the digital signature algorithm (DSA) as specified in the digital signature standard whenever a secure hash algorithm is required for federal applications. The SHA and MD-family hash functions are used widely in the field of communications, where, until recently, throughput of the cryptographic systems was not required to be high. However, since the use of the HMAC in the IPSec, e-payment, and VPN applications, the throughput of the cryptographic system has to reach the highest degree of throughput, especially for the server. In applications with high transmission and reception rates, any latency or delay in calculating the digital signature of the data packet decreases the network's quality of service. Software implementations are presenting unacceptable performance for high-speed applications such as e-commerce, e-health, and video conferences. Poor performance and bulk implementations of HMAC IP cores are currently occurring in the market; Intron and Ocean Logic implementations are one example.

The latter facts were a strong motivation to propose a novel hardware implementation of the HMAC. We aim to provide a low-cost design approach, compared to the solutions proposed by both academia and industry, to satisfy the requirements of the new communication applications. It introduces a negligible area penalty, increasing the throughput and keeping the area small enough for most portable communication devices. The main contribution of this work is the design approach to optimize performance without introducing extra area.

The HMAC algorithm

The HMAC standard defines a special mechanism that guarantees message authentication for transmission through a nonsecure communication channel. The

0278-6648/06/$20.00 © 2006 IEEE

main design approach for this mechanism is to use a cryptographic hash function (usually the MD5 or SHA-1). The purpose of the HMAC is to authenticate both the source of a message and its integrity. The main parameters of the HMAC are the message input and the secret key, which is known only to the message originator and the intended receiver. The main function of the HMAC is the generation of a value (the MAC), formed by condensing the message input and the secret key. The MAC value is sent along with the message, and the receiver has to evaluate that the received message generates the received MAC value using the secret key, which is agreed upon by the message originator and the receiver. The final MAC value is given by the expression shown in (1), where text is the plain text of the message, K is the secret key and K_0 is K appended with zeros to form a $\mathrm{mod}_{32}(n)$ byte key, i_{pad} and o_{pad} are predefined constants, and $\oplus$ is bitwise XOR.

$$\mathrm{HMAC}(K, \mathrm{text}) = H((K_0 \oplus \mathrm{ipad}) \| H(K_0 \oplus \mathrm{opad}) \| \mathrm{text}) \quad (1)$$

Proposed HMAC implementation

The architecture of the proposed HMAC offers a significant benefit concerning the maximum achieved operation frequency. The critical path is observed to the hash core block, where the hash functions are implemented. This allows the design effort to be focused on the hash core and the optimization of the hash functions' critical path. The two hash functions are then presented along with some critical optimizations on the critical path. Solutions are offered for applications that require either HMAC-MD5, SHA-1, or a combined of the HMAC-MD5-SHA-1 function.

SHA-1 hash function

The SHA-1 hash function is an iterative algorithm that requires 80 transformation steps to generate the final hash value or message digest (MD). In each transformation step, a hash operation is performed that takes as inputs five 32-b variables (a, b, c, d, e), and two extra 32-b words (one is the message schedule W_t that is provided by the padding unit, and the other word is a constant K_t predefined by the standard). The calculations taking place in each operation (clock cycle t) are described in (2), where $\mathrm{ROTL}_x(y)$ represents rotation of word y to the left by x b and $f_t(z, w, v)$ represents the nonlinear function associated to clock cycle t

$$\begin{aligned} e_t &= d_{t-1} \\ d_t &= c_{t-1} \\ c_t &= \mathrm{ROTL}_{30}(b_{t-1}) \\ b_t &= a_{t-1} \\ a_t &= \mathrm{ROTL}_5(a_{t-1}) \\ &\quad + f_t(b_{t-1}, c_{t-1}, d_{t-1}) \\ &\quad + K_t + W_t \end{aligned} \quad (2)$$

The linear function f_t changes every 20 cycles. Thus, the SHA-1 is divided in four rounds of 20 identical operations, based on the used nonlinear function. The hash value resulted from the 80 iterations is a 160-b MD.

From (2), since the first four operations are hardwire logic (thus introducing no delay), it is easy to determine that the critical path is located in the calculation of at, which is equal to the delay of three carry-propagate adders (CPA). However, there is a design approach that tries to exploit the characteristics of the carry save adder (CAS) to minimize the critical path.

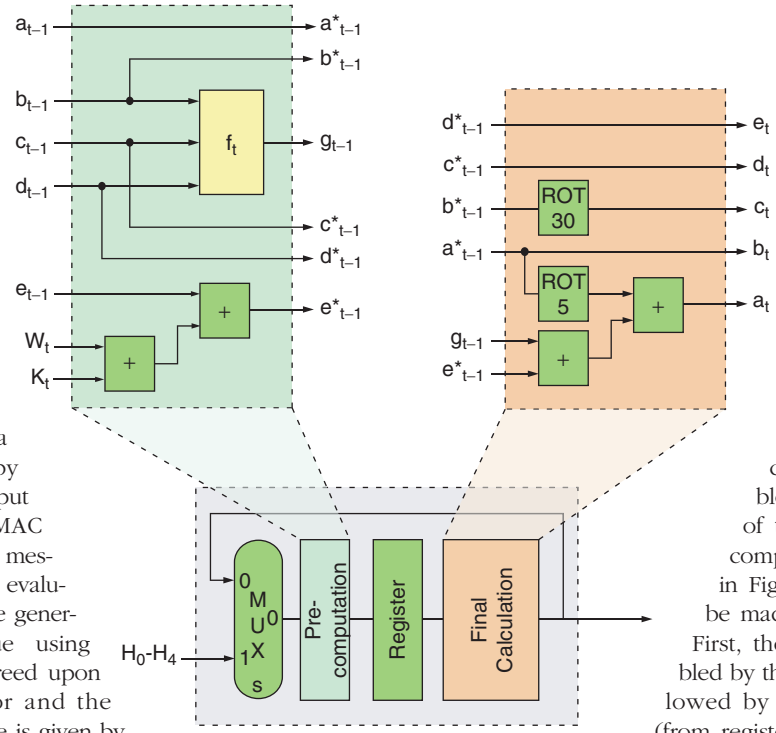


Fig. 1 The modified SHA-1 operation block, separated in two calculation phases

The proposed design approach to optimize the critical path exploits the fact that a_t is calculated using the inputs of cycle $t-1$. Thus, we can precompute some intermediate values, store them in a register, and use them without introducing any delay. So, we transform (2) in (3) to reduce the critical path. The new operation block of the SHA-1, as a result of the application of the precomputation stage, is illustrated in Fig. 1. Some observations can be made analyzing (3) and Fig. 1. First, the new data path is assembled by the final calculation block followed by the precomputation stage (from register output to register input). The critical path is observed in the calculation of a_t (or $e*_{t-l}$) and presents a delay of two adders, synthesized as a CSA and a CPA. Second, the introduced area penalty is a single register that stores the intermediate value g_{t-1}. Additionally, power dissipation is kept low and almost the same as that of the initial implementation. The extra power dissipation is that of the read/write operations of the introduced register. On the other hand, the paths are shortened and balanced, reducing the glitches and the dynamic power dissipation on the circuit's wires The introduction of this precomputational stage is a novel design approach.

$$\begin{aligned} e'_{t-1} &= e_{t-1} + K_t + W, \; e_t = d'_{t-1} \\ d'_{t-1} &= d_{t-1}, \quad d_t = c'_{t-1} \\ c'_{t-1} &= c_{t-1}, \quad c_t = ROTL_{30}(b'_{t-1}) \\ b'_{t-1} &= b_{t-1}, \quad b_t = d'_{t-1} \\ d'_{t-1} &= a_{t-1}, \quad a_t = ROTL_5(d'_{t-1}) \\ &\qquad + e'_{t-1} + g_{t-1} \\ g_{t-1} &= f_t(b_{t-1}, c_{t-1}, d_{t-1}). \end{aligned} \quad (3)$$

MD5 hash function

MD5 is an improved version of MD4, which addresses several known successful attacks on MD4. As in SHA-1, MD5 focuses on the transformation of an initial input through iterative operations. MD5 produces a 128-b MD instead of the 160-b

hash value of SHA-1. Additionally, there are still four rounds consisting of 16 operations each. There are four 32-b (a, b, c, d) inputs and two extra 32-b values that are transformed iteratively to produce the final MD. One is the message schedule M_t that is provided by the padding unit, and the other is a constant L_t predefined by the standard. The calculations that take place in each operation (clock cycle t) are described in (4), where $fn_t(z, w, v)$ represents the nonlinear function associated to clock cycle t. Rotation in (4) is performed for s positions, which vary from cycle to cycle and are predefined by the standard

$$\begin{aligned} d_t &= c_{t-1} \\ c_t &= b_{t-1} \\ b_t &= b_{t-1} + \mathrm{ROTL}_s(a_{t-1} + fn_t \\ &\quad \times (b_{t-1}, c_{t-1}, d_{t-1}) + M_t + L_t) \\ a_t &= d_{t-1}, \end{aligned} \tag{4}$$

The critical path is located on the calculation of a sole output b_t. Identically to SHA-1, a precomputational stage can be applied to this hash function to reduce the critical path.

HMAC implementation scenarios

As already mentioned, HMAC can be implemented using one hash function or two hash functions combined to operate when selected. Also, both SHA-1 and MD5 hash functions have an identical parameter; they both have four discreet rounds. The above offer a wide range of characteristics of the HMAC implementation that, if exploited wisely, can give solutions depending on the nature of the application.

If the critical parameter is a small area, a rolling loop technique can be applied. As illustrated in Fig. 1, the output of the operational block is fed back to the input through precomputation stage. Notice that the main benefit of the insertion of the precomputation stage is that a_t, which is the output of the final calculation block, enters the precomputation stage as the new a_{t-1}, which is a wire directly connected to the register. This technique allows small-sized implementations through reuse of the same configurable operation block. Configurability issues have to address correct selection of the nonlinear function for both hash functions and the rotate positions in the case of MD5.

If the critical design parameter is performance, with a more relaxed area constraint, then pipeline can be applied. As already mentioned, a common characteristic of the two hash functions is the four rounds. Thus, applying a pipeline stage to every round results in a quadruplicating of the achieved throughput. This technique exploits small-sized implementations based on rolling loop and the characteristic of the four rounds to result in relatively small sized implementations, achieving throughput four times higher than the limit imposed by the design of the operation block of the hash function.

In the case of implementing HMAC-MD5 or HMAC-SHA1, the throughput is directly associated to the maximum operating frequency of the hash function's operation block. The proposed modifications of the two hash functions significantly reduce making the critical path. The implementation of HMAC-MD5 or HMAC-SHA1, using the precomputational stage, scores a 30% increase of throughput if no pipeline is applied.

In many applications, there is a need for the selective use of SHA-1 or MD5. There are two design approaches for coexistence of the two hash functions. The first is the implementation of the two hash functions as separate cores and selection through a multiplexer. Although this approach presents low design complexity, it is not optimal for small-area requirements. Power dissipation is also considerably high. The second design approach is the exploration of the two hash functions to locate resources that can be used by both functions. In this case, area requirements are reduced and extra power dissipation is a factor of the latter approach only.

Table 1. Characteristics of the proposed HMAC implementations for the targeted FPGA technologies.

HMAC	Slices	Op.Frequency (MHz)	Throughput (Mb/s)
	Xilinx Virtex-II (−6)		
SHA-1	854	162	1024.0
MD5	797	96	756.0
SHA-1 MD5 (perf.)	1357	96	606.8 756.0
SHA-1 MD5 (area)	982	81	512.0 638.0
	Xilinx Virtex-E (−8)		
SHA-1	686	111	701.6
MD5	612	65	512.0
SHA-1 MD5 (perf.)	1100	65	410.8 512.0
SHA-1 MD5 (area)	780	61	385.5 480.4

Implementation and results

Considering the afrementioned implementation scenarios, we implemented several HMAC designs to verify and evaluate the value of the presented design approach. The designs were captured in VHDL and were fully simulated and verified using commercial tools. The XILINX Virtex field programmable gate array technologies were selected as the targeted technologies synthesizing the designs for the Virtex-II and Virtex-E device families. We used these device families to exploit the different characteristics offered by each. More specifically, Virtex-E is appropriate for area-optimized designs offering compact and area efficiency, while Virtex-II presents performance efficiency.

Results of the implementations

In Table 1, the characteristics of the proposed HMAC implementations are offered. Only implementations of the Virtex-E FPGA family were fully verified, and numbers reflect experimental results. The results of the FPGA technologies are reported from the synthesis tool. The implementation of the combined hash functions is considered for two target design parameters: performance optimized, which uses implementation of two separate cores and selection through a multiplexer, and area optimized, which exploits commonly reused primitives. The reported throughput corresponds to a design approach with rolling loop technique applied but without pipeline. If the pipeline technique is applied, throughput is quadrupled, and the area is increased an average of 3.21 times. This is the first time that an implementation without pipeline exceeds 1 Gb/s in Virtex-II FPGA

technology. As illustrated, synthesizing the area-optimized design for Virtex-E results in even smaller area requirements, while the performance-optimized design takes further advantage of the Virtex-II device family. It can be observed that HMAC-SHA1 is more efficient, in terms of performance, than HMAC-MD5. The combined HMAC-SHA1-MD5 doesn't present the performance benefits of SHA1, because the maximum operating frequency is limited to that of the lower one, which in this case is MD5. However, this solution offers reduced area requirements by reusing the same resources for both SHA1 and MD5.

In Table 2, the characteristics of the commercial HMAC IP cores are reported to make comparisons. Every single SHA-1 implementation (not an HMAC-SHA1) is marked with a * at the start. These designs are offered as a reference due to the explicit dependency of the maximum operating frequency of the HMAC from the critical path of the used hash function. As illustrated in Table 2, MD5 is not exploiting the high-speed performance efficiencies of Virtex-II device families, focusing on a optimum area design, while much better performance could be obtained with only a small area tradeoff. Moreover, MD5 presents even worse characteristics, in terms of both performance and area, while it doesn't either exploit the Virtex-II high performance efficiency or make the effort for an optimum area design. Analyzing the performance of the implementations presented in Table 2, it can be observed that throughput of the proposed HMAC implementations exceeds those of the available commercial IP cores by up to 390%.

Conclusions

A novel design approach for the development of small sized and high-speed HMACs was presented in this article. The approach showed that the critical path can be further reduced by exploiting special properties of the included hash functions. A significant design effort was made to keep the area low. The experimental results showed that a negligible area penalty was introduced for achieving an increase in throughput up to 390% compared to the competing implementations. Finally, the design was fully tested and verified for the Xilinx Virtex-E FPGA family using a prototype board.

Read more about it

- *Secure Hash Standard*, NIST, FIPS Pub. 180-2, 2002.
- *IETF Network Working Group*, RFC 1321, 1992.
- The Keyed-Hash Message Authentication Code (HMAC) (Standard). NIST, FIPS Pub 198 Standard, 2002.
- *Digital Signature*, NIST, FIPS Pub 186-2, 2000.
- IP Security Protocol [Online] Charter, Internet Drafts for IPSec. Available: http://www.ietf.org/html charters/ipsec-charter.html
- S. Dominikus, "A hardware implementation of MD-4 family hash algorithms," in *Proc. IEEE Int. Conf. Electronics, Circuits and Systems*, 2002, pp. 1143–1146.
- N. Sklavos, G. Dimitroulakos, and O. Koufopavlou, "An ultra high speed architecture for VLSI implementation of hash functions," in *Proc. IEEE Int. Conf. Electronics, Circuits and Systems*, 2003, pp. 990–993.
- T. Grembowski, R. Lien, K. Gaj, N. Nguyen, P. Bellows, J. Flidr, T. Lehman, and B. Schott, "Comparative analysis of the hardware implementations of hash functions SHA-1 and SHA-512," in *Proc. Information Security Conf.*, Springer-Verlag, Berlin, Germany, 2002, pp. 75–89.
- B. den Boer and A. Bosselaers, "An attack on the last two rounds of MD4," in *Proc. CRYPTO '91, Advances in Cryptology*, Springer Verlag, Berlin, Germany, 1992, pp. 194–203.
- ALMA Technologies. Available: http://www.alma-tech.com
- Bisquare Systems Private Ltd. Available: http://www.bisquare.com
- Helion Technology Ltd. Available: http://www.heliontech.com
- Intron, Ltd. Available: http://www.lviv.uar.net/~intron/
- Ocean Logic Ltd. Available: http://www.ocean-logic.com
- Amphion. Available: http://www.amphion.com/index.html

Table 2. Characteristics of other HMAC implementations for the targeted FPGA technologies.

HMAC	Slices	Op.Frequency (MHz)	Throughput (Mb/s)
	Xilinx Virtex-II (−6)		
*SHA-1 [12]	573	140	874.0
*SHA-1 [14]	612	79	498.1
*MD5 [12]	613	96	744.0
*MD5 [14]	614	62	488.3
*MD5 [15]	844	60	472.0
SHA1 & MD5 [12]	888	95	593.0 736.0
	Xilinx Virtex-E (−8)		
*SHA-1 [13]	716	71	449.0
*SHA-1 [14]	612	72	451.9
*MD5 [14]	605	50	393.8
SHA-1 [11]	579	66	422.4
MD5 [11]	324	50	400.0

The designs that are marked with a '*' are indicating implementations of the described hash functions, not HMAC implementations.

About the authors

Ioannis I. Yiakoumis is a student of electrical and computer engineering at the University of Patras, Greece. He is a Student Member of IEEE. His research interests include hardware design, computer security, wireless networks, and embedded systems programming.

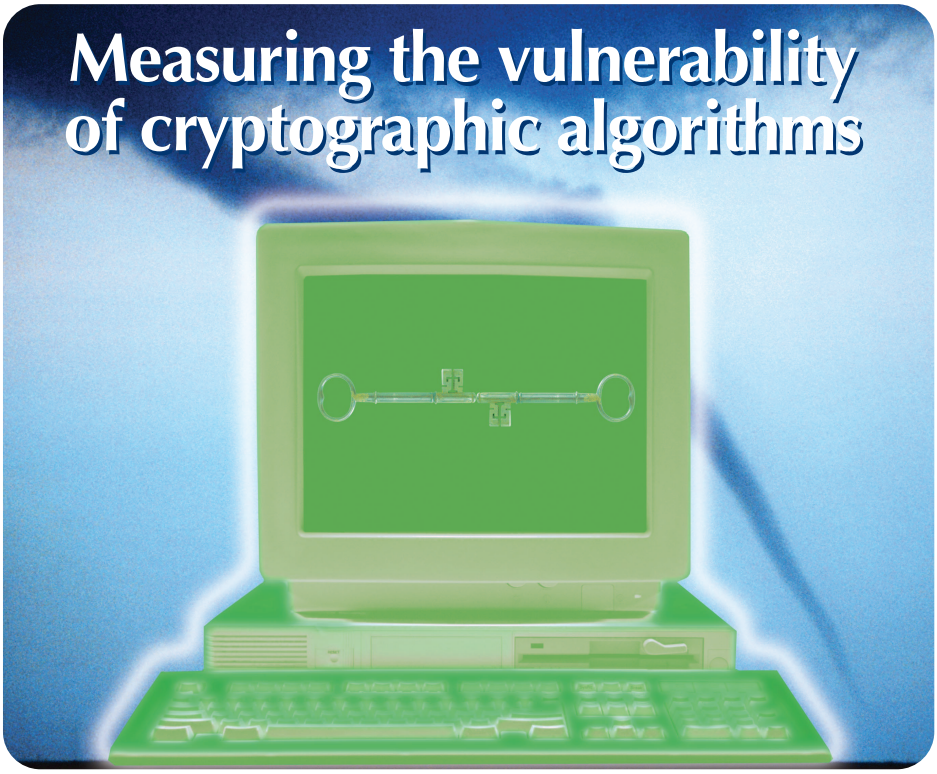
Markos E. Papadonikolakis is a student of electrical and computer engineering at the University of Patras, Greece. He is a Student Member of the IEEE. His research includes computer security, hardware design, and image encoding.

Harris E. Michail is a researcher of electrical and computer engineering at the University of Patras, Greece. He is a Member of the IEEE, the Technical Chamber of Greece, and the Greek Electrical Engineering Society. His research includes computer security, hardware design, and reconfigurable architectures.

Athanasios P. Kakarountas is with the electrical and computer engineering department, University of Patras, Greece. He is a Member of the IEEE.

Costas E. Goutis is with the electrical and computer engineering department, University of Patras, Greece. He is a Member of the IEEE.

Measuring the vulnerability of cryptographic algorithms



ELKE DE MULDER,
PIETER BUYSSCHAERT,
SIDDIKA B. ÖRS,
PETER DELMOTTE,
BART PRENEEL,
GUY VANDENBOSCH,
AND
INGRID VERBAUWHEDE

ELLIPTIC CURVE CRYPTOGRAPHY is a public key cryptosystem that is becoming increasingly popular. Implementations of cryptographic algorithms should not only be fast, compact, and power efficient, they should also resist side channel attacks. One of the side channels is the electromagnetic radiation out of an integrated circuit. Hence, it is very important to assess the vulnerability of implementations of cryptosystems against these attacks. A simple electromagnetic analysis (SEMA) attack on an unprotected implementation can find all the key bits with only one measurement. We also describe a differential electromagnetic analysis (DEMA) attack on an improved implementation and demonstrate that a correlation analysis requires 1,000 measurements to find the key bits.

Cryptographic algorithms and protocols hold the key

Keeping information secret and authentic is a very old concern, but the exponential growth of technology exacerbates the need for secure communication. Cryptographic algorithms and protocols are essential in protecting the confidentiality and authentication of data; they replace the problem of protecting information by protecting short cryptographic keys.

Ironically, the very same technology that forms the basis for the higher demand in security has a few annoying side effects. The use of side channels to break a cryptosystem was introduced by P. Kocher. In this context, a side channel is a physical property that can be measured externally during the execution of a cryptographic algorithm to derive information on secret keys. Examples are the execution time of the algorithm on the chip or the power consumption of implementations of cryptosystems. With this idea, cryptanalysis no longer focuses exclusively on the mathematical aspects but also evaluates weaknesses of implementations. The three main physical properties of cryptographic modules can be exploited in side channel attacks: power consumption, timing, and electromagnetic radiation. Others such as sound and heat are currently being explored but see less promising.

Elliptic curve cryptography (ECC) was proposed independently by Miller and Koblitz in the 1980s. Since then a considerable amount of research has been performed on secure and efficient ECC implementations.

This article reports on the first implementation of an electromagnetic analysis (EMA) attack on a hardware implementation of an elliptic curve (EC) processor with a key length of 160 b. Earlier work is either theoretical or presents attacks on software implementations for 8-b smart cards. The main difference between our implementation of an EC processor and these software implementations is that, in our hardware, all operations are done in parallel. Hence, the number of bit transitions during every clock cycle can be up to 160, compared to eight for a smart card. This implies that the predictions of the transitions are much harder. To detect the effect of any bit changes, we have to increase the number of measurements by a factor of 20 or more.

The U.S. government has been aware of electromagnetic leakage since the 1950s. The resulting standards are called TEMPEST and are partially available at <http://cryptome.org/nsa-tempest.htm>. The first published papers are the work of J. Quisquater and D. Samyde and the Gemplus team. According to D. Agrawal, there are two types of radiations: intentional and unintentional. Later on, information of different side channels was combined in so-called multichannel attacks in which the side channels are not necessarily of a different kind.

Until now, most papers on EMA applied similar techniques such as power analysis while apparently much more information is available to be

0278-6648/06/$20.00 © 2006 IEEE

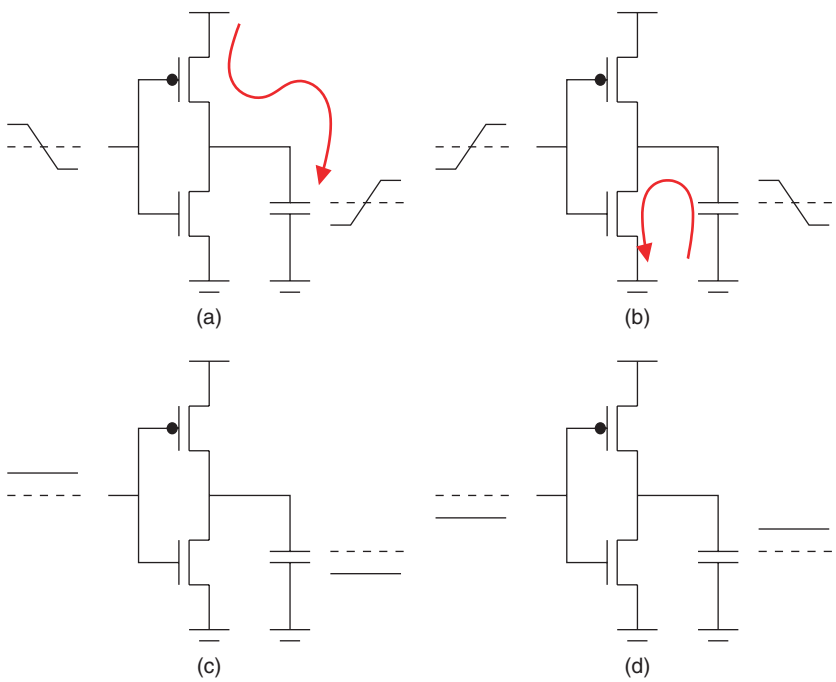


Fig. 1 Possible switching events for a CMOS invertor (From left to right, from top to bottom (a)–(b), (c)–(d)

explored. It is likely that future work will also deal with combinations of EMA with other side channel attacks.

Elliptic curves over GF(p)

The public key cryptosystem implemented on the field programmable gate array (FPGA) is the elliptic curve cryptosystem. An elliptic curve E is expressed in terms of the Weierstrass equation: $y^2 = x^3 + ax + b$, where $a, b \in GF(p)$. The points on this curve can be added to each other, and the resulting point is again a point on the same curve. The point at infinity zero plays a role analogous to that of the number 0 in ordinary addition. Thus, $P + O = P$ and $P + (-P) = O$ for all points P. With these properties, it is straight forward to introduce the point or scalar multiplication as the main operation for ECC, i.e., $kP = P + P + \ldots P\,(k\ times)$. This operation can be calculated by using the double-and-add algorithm as shown in Algorithm 1.

Algorithm 1: Elliptic Curve Point Multiplication

Require: EC point $P = (x, y)$, integer k, $0 < k < M$, $k = (k_{l-1}, k_{l-2}, \ldots, k_0)_2$, $k_{l-1} = 1$

Ensure: $Q = (x', y') = [k]P$

1. $Q \leftarrow P$
2. **for** i from l-2 downto 0 **do**
3. $\quad Q \leftarrow 2Q$
4. $\quad$ **if** $k_i = 1$ **then**
5. $\quad\quad Q \leftarrow Q + P$
6. $\quad$ **end if**
7. **end for**

The goal is to guess the key bits k_i because by finding them, the algorithm is broken.

Electromagnetic analysis attack

Nowadays, complementary metal-oxide semiconductor (CMOS) is by far the most commonly used technology to implement digital integrated circuits. A CMOS gate consists of a pull-up network with C-MOS transistors and a pull-down network with n-MOS transistors. Those networks are complementary. When the input is stable, only one of the two networks conducts. The most simple logic gate is an inverter; its power consumption is representative for all logic ports and gives a general image of the power consumption in a CMOS circuit. During the functioning of the inverter, three types of power consumption can be distinguished: the leakage current, the current that flows from the power source to the ground during the switching from 0 to 1 (short-circuit current), and the current used to charge and discharge the different capacitors in a digital network (dynamic power consumption). The last one causes the biggest power consumption in present designs because of all the wiring in an FPGA on which this algorithm is implemented. These capacitors, however, are necessary to maintain the two different logic levels. The capacitor, when charging or discharging differs so to switch an invertor from 0 to 1 or from 1 to 0, consumes a different amount of power. In addition, all capacitors for each gate differ, which results in a different power consumption of the different gates according to the data being processed. This is partially illustrated in Fig. 1: (a) shows charging the load capacitor of the invertor, (b) shows the discharging phase, and (c) and (d) do not show any change in charge because the input voltage is not altered.

The sudden current pulse that occurs during the transition of the output of a CMOS gate causes a variation of the electromagnetic field surrounding the chip. This can be monitored, for example, by inductive probes that are particularly sensitive to the related impulsion. When using a loop antenna, the voltage induced by the current equals

$$V = -\frac{d\phi}{dt}$$

$$\phi = \int \vec{B}.\vec{A}$$

where V is the probe's output voltage, ϕ the magnetic flux sensed by probe, t is the time, B is the magnetic field, and A is the area that it penetrates.

Two types of electromagnetic analysis attacks are distinguished. In a SEMA attack, an attacker uses the information from one electromagnetic radiation measurement directly to determine (parts of) the secret key. In a DEMA attack, many measurements are used to filter out noise and the key is derived using a statistical analysis. A SEMA attack is typically used when there is a conditional branch in the algorithm that results in a different radiation pattern whenever the branch is taken. A DEMA attack uses the property that processing different data needs a distinct amount of power and radiates a different field.

A simple analysis

In a simple power analysis attack, an attacker uses the side-channel information from one measurement directly to determine parts of the secret key. These attacks are possible because of differences between executed instructions. They need to have a simple relationship with the key, like the *f.e.* key dependent

branches. The algorithm under attack in this article can be implemented with key dependent branches as in shown in line four of Algorithm 1. If the instruction in this branch has a power consumption graph that is distinguishable from other instructions in the algorithm, it will be very easy to gain some information about the key. An attacker measures the power trace of the cryptodevice and tries to relate his knowledge about the implementation on the power trace.

A differential analysis

A differential analysis works according to the following procedure and is shown in the flowchart. In the first step, a number of side-channel measurements are taken while the cryptographic device encrypts the same algorithm with the same key but different inputs. In the second step, the attacker chooses a point of attack where the intermediate results at that time only depend on a part of the key and calculates the hypothetical side-channel value for every input based on all guesses for the subkey according to a model of the side channel. In the next step, the attacker uses statistical techniques to verify which hypothesis about the key is correct by looking for a relationship between the hypothetical side channel values and the real ones.

As explained earlier, when there is some switching from 0 to 1 or 1 to 0, more power is used than when there is no switching activity. This relates the power consumption with the amount of bit toggles, giving us a model to mimic power as a side channel. When an attacker chooses a point of attack for which he can calculate the intermediate values of the algorithm with knowledge of the input and a partial guess from the key, he can relate the power consumptions of the measurements with different inputs to the hypothetical values of the amount of bit toggles. If this statistical analysis makes sense, the key guess was correct. If not, the key guess was probably incorrect and a new one sould be made.

Correlation analysis

In DEMA, an attacker uses a hypothetical model of the attacked device. The quality of this model is dependent on the knowledge of the attacker. The model is used to predict several values for the electromagnetic radiation of a device, which are compared to the real, measured electromagnetic radiation of the device. Comparisons are performed by applying statistical methods on the data. Among others, the most popular are the distance-of-mean test and the correlation analysis. For the correlation analysis, the model predicts the amount of side channel leakage at a certain moment of time in the execution. These predictions are correlated to the real electromagnetic radiation. The correlation can be measured using the Pearson correlation coefficient. Let t_i denote the ith measurement data (i.e., the ith trace) and T the set of traces. Let p_i denote the prediction of the model for the ith trace and P the set of such predictions. Then we calculate

$$C(T, P) = \frac{E(T.P) - E(T).E(P)}{\sqrt{\mathrm{Var}(T).\mathrm{Var}(P)}}$$
$$-1 \leq C(T, P) \leq 1$$

where $E(T)$ denotes the expectation (average) trace of the set of traces T and $Var(T)$ denotes the variance of a set of traces T. T and P are said to be uncorrelated, if $C(T, P)$ equals zero. Otherwise, they are said to be correlated. If their correlation is high, i.e., if $C(T, P)$ is close to $+1$ or -1, it is usually assumed that the prediction of the model, and thus the key hypothesis, is correct.

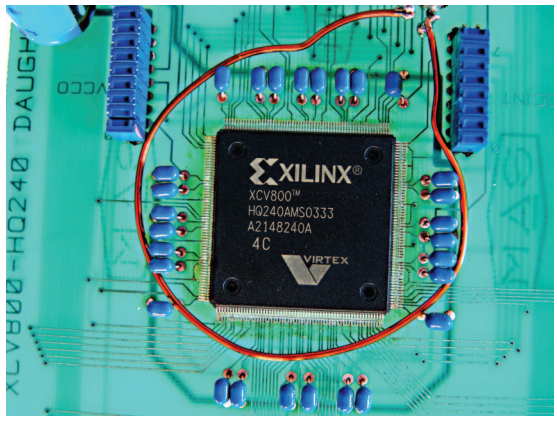


Fig. 2 The FPGA and the handmade antenna

Measurement setup

Figure 2 shows the most important part of our measurement setup: the VIRTEX FPGA that is under attack. Because the field surrounding the chip is mainly a magnetic field in the near field, a loop antenna is used to pick up the variations of the field. Our setup consists of essentially two boards. The main board is responsible for interfacing to the PC via the parallel port. It is connected with the XILINX parallel cable to program the VIRTEX FPGA and provides some LEDs, switches, and buttons for testing purposes. The daughter board itself carries the VIRTEX FPGA. It allows to access some pins for triggering and to measure the power consumption of the VIRTEX FPGA in a convenient way.

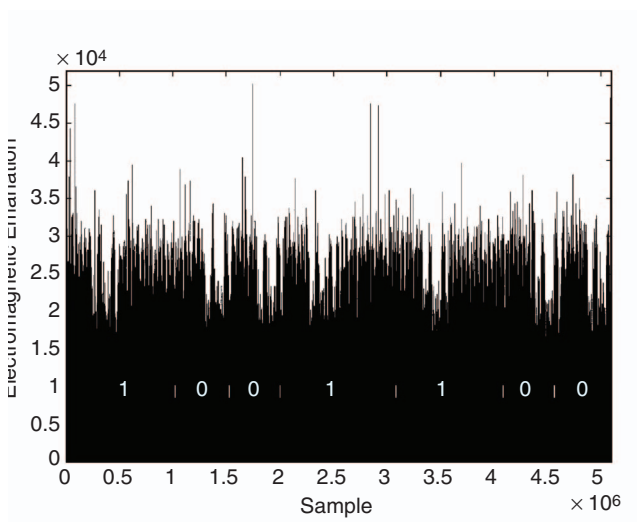


Fig. 3 Electromagnetic radiation trace of a 160-b EC point multiplication with double-and-add algorithm

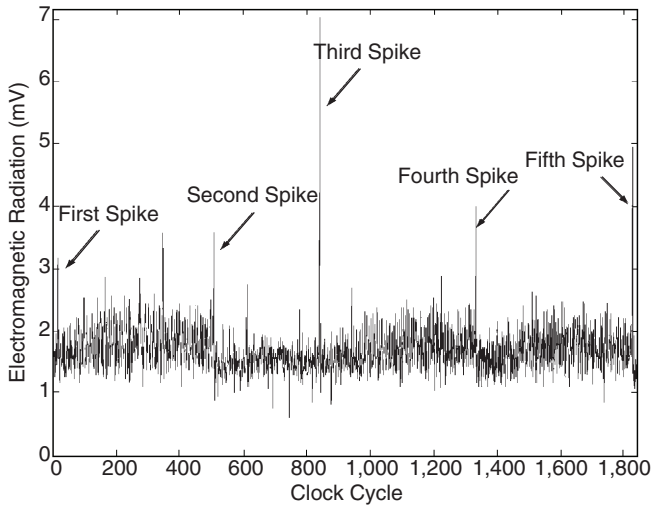


Fig. 4 The EM trace of the first measurement after taking the maximum value in every clock cycle

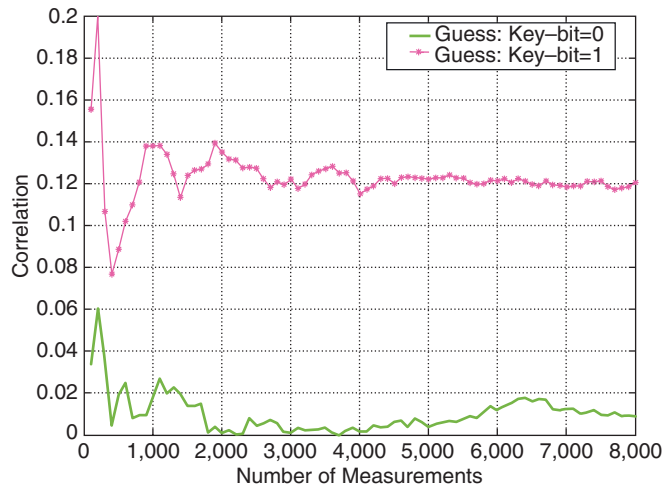


Fig. 5 Correlation in function of the number of measurements for the fifth peak

SEMA attack on an FPGA implementation of an EC processor

The EM radiation trace of a 160-b EC point multiplication is shown in Fig. 3. The SEMA attack is implemented on the EC processor published in S.B. Ors et al, which uses Algorithm 1 for EC point multiplication. It can be derived from Fig. 3 that the key used during this measurement is 11001100, because there is difference between the EM radiation traces of the EC point addition and doubling. The SEMA attack was successful because of the conditional branch in Step 4 of Algorithm 1.

As a countermeasure to this attack, we implemented the EC point multiplication by using the always double and add algorithm published in J.S. Coron. Algorithm 2 shows that the EC point addition is executed independently from the value of the key bits. One EM radiation measurement will not reveal the key bits.

Algorithm 2: Elliptic Curve Point Multiplication

Require: EC point $P = (x, y)$, integer k, $0 < k < M$, $k = (k_{l-1}, k_{l-2}, \ldots, k_0)_2$, $k_{l-1} = 1$

Ensure: $Q = (x', y') = [k]P$

1. $Q \leftarrow P$
2. **for** i from l-2 downto 0 **do**
3. $\quad Q_1 \leftarrow 2Q$
4. $\quad Q_2 \leftarrow Q_1 + P$
5. $\quad$ **if** $k_i = 1$ **then**
6. $\quad\quad Q \leftarrow Q_2$
7. $\quad$ **else**
8. $\quad\quad Q \leftarrow Q_1$
9. $\quad$ **end if**
10. **end for**

DEMA attack on an FPGA implementation of an EC processor

The target for our DEMA attack is the second most significant bit (MSB) of the key k_{l-2} in Algorithm 2. If $k_{l-2} = 0$, then Q will be updated by 2P, otherwise by 3P at step 5 in Algorithm 2.

In the first step of our attack, we have produced a so-called EM radiation file. For this purpose, N random points were chosen on the EC and one fixed but random key. We have let the FPGA execute N point multiplications of N EC points, P_i, $i = 1 \ldots N$ with the same key, k as $Q_i = [k]P_i$. We will attack the circuit at the time the coordinates of Q_1 is updated for the second time at step 3 of Algorithm 2. With these measurements, an N x 2 000 000 matrix, M_1 is produced. We have applied a preprocessing technique to reduce the amount of measurement data in every clock cycle. We have found the maximum value of the measurement data in each clock cycle and stored them in matrix M_2. Because the clock frequency of the function generator we have used for our experiments was slightly differing during the measurements, the number of points in one clock cycle D_i has to be found. To compute D_i, we have to know the exact clock frequency. For this, we have calculated the discrete fourier transform of each measurement. Figure 4 shows the first measurement after taking the maximum value in every clock cycle.

We have implemented the EC point multiplication with Algorithm 2 in the C programming language. The C program computes N EC point multiplications with N EC points and the key. The EC points and the key are the same as the ones given to the FPGA. During the execution of the EC point multiplications, the C program computes the number of bits that change from 0 to 1 and from 1 to 0 in some registers at the corresponding steps to the five spikes shown in Fig. 4. The number of transitions is used as the EM radiation prediction.

We have predicted the EM radiation of the events that correspond to the five spikes shown in Fig. 4 for $k_{l-2} = 0$ and $k_{l-2} = 1$ for each measurement and stored them in M_3. Now, we can learn the right value of k_{l-2} by finding the correlations between M_3 and M_2. There will be two values for each spike: one for the guess that the key bit is 0, one for the guess that the key bit is 1. The correlations for spike five give us the correct key bit by using only 1000 measurements. The correlation for the guess that the key bit is 1 is much higher than the correlation for the other guess as shown in Fig. 5.

After 1,000 measurements, the correlation for the $k_{l-2} = 1$ guess starts to differ from the correlation for the $k_{l-2} = 0$ guess. The correlation for the $k_{l-2} = 1$ guess starts to rise, for the $k_{l-2} = 0$ guess the correlation stays around 0.

Conclusions

In this article, we have presented a SEMA and DEMA electromagnetic analysis attack on an FPGA implementation of an elliptic curve processor. As a result of

a SEMA attack on an unprotected implementation, we can find all the key bits using just one measurement. We have conducted a DEMA attack on the always double and add implementation and have shown that it is possible to find the key bits by making more measurements and using correlation analysis. Our attacks show that electromagnetic attacks form a realistic threat for a broad range of cryptographic hardware implementations. Further work is necessary to optimize these attacks using more sophisticated antennas and signal processing techniques. On the other hand, system designers and cryptographers should jointly develop, implement, and evaluate additional countermeasures against side channel attacks. These can consist of frequent key updates and various masking and decorrelation approaches.

Read more about it

- P. Kocher, "Timing attacks on implementations of Diffie-Hellman, RSA, DSS and other systems," in *Advances in Cryptology: Proce. CRYPTO'96*, Springer-Verlag, vol. 1109, 1996, pp. 104–113.
- P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology: Proc. CRYPTO'99*, vol. 1666, 1999, pp. 388–397.
- S.-M. Kang and Y. Leblebici, *CMOS Digital Integrated Circuits: Analysis and Design*. New York: McGraw Hill, 2002.
- NSA, "NSA TEMPEST Documents," [Online] Available: http://www.cryptome.org/nsa-tempest. htm.
- J.J. Quisquater and D. Samyde, "Electromagnetic analysis (EMA): Measures and counter-measures for smart cards," in *Proc. Int. Conf. Research Smart Cards: Smart Card Programming and Security (E-smart)*, I. Attali and T. Jensen, Eds., 2001, pp. 200–210.
- D. Agrawal, B. Archambeault, J.R. Rao, and P. Rohatgi, "The EM side-channel(s): Attacks and assessment methodologies," in *Proc. 4th Int. Workshop Cryptographic Hardware and Embedded Systems (CHES)*, B.S. Kaliski Jr., Ç.K. Koç, and C. Paar, Eds., 2002, pp. 29–45.
- D. Agrawal, B. Archambeault, S. Chari, J.R. Rao, and P. Rohatgi, "Advances in side-channel cryptanalysis," *RSA Lab. Cryptobytes*, vol. 6, no. 1, pp. 20–32, 2003.
- D. Agrawal, J.R. Rao, and P. Rohatgi, "Multi-channel attacks," in *Proc. 5th Int. Workshop Cryptographic Hardware and Embedded Systems (CHES)*, C. Walter, Ç.K. Koç, and C. Paar, Eds., 2003, pp. 2–16, .
- S.B. Örs, E. Oswald, and B. Preneel, "Power-analysis attacks on an FPGA, First experimental results," in *Proc. 5th Int. Workshop Cryptographic Hardware and Embedded Systems (CHES)*, C. Walter, Ç.K. Koç, and C. Paar, Eds., 2003, pp. 35–50.
- S.B. Örs, L. Batina, B. Preneel, and J. Vandewalle, "Hardware implementation of an elliptic curve processor over GF(p)," in *Proc. IEEE 14th Int. Conf. Application-Specific Systems, Architectures and Processors, 2003, pp. 433-443.*
- J.S. Coron, "Resistance against differential power analysis for elliptic curve cryptosystems," in *Proc. 1st Int. Workshop Cryptographic Hardware and Embedded Systems (CHES)*, Ç.K. Koç and C. Paar, Eds., 1999 vol. 1717, pp. 292–302.

About the authors

Elke De Mulder is a Ph.D. student at the Katholieke Universiteit Leuven, Belgium.

Pieter Buysschaert is a project leader at Belgocontrol, Belgium.

Saddika B. Örs is a research assistant at the Istanbul Technical University.

Peter Delmotte is working at Proximus, Belgium.

Bart Preneel is professor in the Electrical Engineering Department of the Katholieke Universiteit Leuven in Belgium.

Guy Vandenbosch is professor in the Electrical Engineering Department of the Katholieke Universiteit Leuven in Belgium.

Ingrid Verbauwhede is professor in the Electrical Engineering Department of the Katholieke Universiteit Leuven in Belgium.