# A Scan-based Attack on Elliptic Curve Cryptosystems in presence of Industrial Design-for-Testability Structures

Jean Da Rolt[1], Amitabh Das[2], Giorgio Di Natale[1], Marie-Lise Flottes[1], Bruno Rouzeyre[1], Ingrid Verbauwhede[2]
[1]LIRMM (Université Montpellier II /CNRS UMR 5506), Montpellier, France
{darolt, dinatale, flottes, rouzeyre}@lirmm.fr
[2]KU Leuven, ESAT/SCD-COSIC and IBBT, Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium
{amitabh.das, ingrid.verbauwhede}@esat.kuleuven.be

*Abstract*— **This paper presents a scan-based attack on hardware implementations of Elliptic Curve Cryptosystems (ECC). Several up-to-date Design-for-Testability (DfT) features are considered, including response compaction, X-Masking and partial scan. Practical aspects of the proposed scan-based attack are described, namely timing and leakage analysis that allows finding out data related to the secret key among the bits observed through the DfT structures. We use an experimental setup which allows full automation of the proposed scan attack on designs including DfT configurations. We require around 8 chosen points to implement the attack for retrieving a 192-bit scalar.**

***Keywords: Scan-based attacks, Elliptic Curve Cryptography, Design-for-Testability, Montgomery Ladder, Test compression***

## I. INTRODUCTION

Scan chain side-channel attacks exploit the scan chain DfT infrastructure inserted for testing a security circuit. Even if cryptographic algorithms are proven to be secure, accessing intermediate states via the test facilities compromises their strength. Testing ensures the product quality and is essential in secure ICs since design bugs may compromise the security. However, scan chains open a backdoor or side-channel through which secret information inside the chip leaks.

One of the scan-based attacks proposed in the literature, targets the ECC algorithm [1]. This attack relies on the assumption that the circuit contains a single scan chain. In actual designs this assumption is not realistic, since more complex DfT methods are required for meeting the design requirements and reducing the test cost. Techniques such as multiple scan chains, pattern decompression, response compaction and filters to increase tolerance to unknowns are nowadays current practices in the test infrastructure. These structures are often supposed to behave as countermeasures against scan attacks, due to the apparent reduction on the observability of internal states, as proposed in the Mentor Graphics Whitepaper [2] and by Liu et al. [3].

Breaking the continuity of the scan chains after manufacturing test, known as unbounding, is also not suitable. This is so since the in-field debug and test capability of the device, required for code or firmware upgrades during the lifetime of a product, is lost. Unbounding requires the use of fuses in the scan path which can be both expensive in terms of cost and area. Moreover, even if the scan chains are unbounded, probing attacks can be mounted on parts of the scan chains [4]. Additionally, for satisfying the higher levels of Common Criteria for Information Technology Security Evaluation [5], the device must be testable, which can be guaranteed through scan testing.

As presented in [6], scan-based attacks are effective against RSA implementations. In this paper, we target ECC implementations reusing the same attack algorithm adapted for the purpose. We describe all the issues in carrying out the attack in the presence of test compression, X-Masking and other advanced DfT methods and how to overcome them. Additionally, we prove its feasibility by empirically performing the attack on an ECC design in presence of such DfT structures. Moreover, the attack may be applied without knowledge of the DfT structures, which makes the attack more realistic.

The rest of the paper is structured as follows. Section II gives an overview of scan attacks and countermeasures previously described in the literature. General aspects of ECC are presented in Section III, with a description of the ECC hardware components targeted by the scan attack. It also contains our attacker model. Section IV describes the assumptions taken by the proposed attack as well as the scan attack principles and the differential attack mode. The new scan attack on DfT implementations and details of the practical aspects of the attack are presented in Section V. The tool developed for practical implementation of the proposed attack is described in Section VI. The experimental results of the scan attack on different DfT configurations are reported in Section VII. Finally, we conclude the paper with future work in Section VIII.

## II. RELATED WORK

### A. Scan Attacks

Scan attacks first appeared for symmetric-key implementations. The first work [7] in this domain was directed against a Data Encryption Standard (DES) block cipher. Yang et al. presented a two-step process starting with finding the position of the intermediate registers in the scan chain, and then extracting the DES first round key using only three chosen plaintexts. This was followed by a scan attack on the Advanced Encryption Standard (AES) [8] employing differential analysis. The attack proposed in [8] has the advantage that it does not need finding out the position of the intermediate registers. In [9] authors propose a scan attack based on differential analysis to cope with multiple scan chains, linear compaction and mask decoders.

Some scan attacks have been proposed against stream ciphers [10]. The authors demonstrate that the scan chain

structure of Linear Feedback Shift Registers used as stream ciphers may be determined and then it may compromise the device security.

Public-key ciphers have also been proven to be vulnerable to scan attacks. The scan attack on RSA in [11] targets the Binary exponentiation algorithm. Similarly, the scan attack on ECC in [1] is directed at the Montgomery multiplication method. Additionally to these works, in [6] authors proposed a scan attack on RSA that deals with DfT structures such as response compaction.

### B. Comparision with previous works

As far as we know the only scan attack that targets ECC is described in [1]. It considers a single scan path and it does not take into account the effect of other secret dependent FFs on the scan chain that may complicate the attack. In our proposed attack, we perform a detailed leakage analysis and work in differential mode. Our scan attack is successful even if there are other FFs that change their values along with the intermediate register of interest. Additionally, we require less number of messages to find the ECC secret key compared to [1]. The attack in [1] requires 29 points on an average and 35 points in the worst case to find a 163-bit ECC secret scalar. Our attack, on the other hand, requires 8 points on an average to retrieve a 192-bit secret scalar.

In comparison to [9], which also considers advanced DfT structures on symmetric-key ciphers, our work focus on public-key primitives. Since the structure of public-key algorithms is different than symmetric key ones, we need to apply a different approach. For instance, public-key algorithms usually perform an initial operation that involves the register that stores secret data, but this initial operation is not related to the key. This can be used to identify the position of the secret data in the scan chains. The same approach cannot be applied for symmetric key ciphers.

The principle of the attack presented here is similar to the one presented in [6]. It has been adapted for ECC instead of RSA implementations. The target algorithm is a Simple Power Analysis (SPA) resistant Montgomery Ladder, instead of the Montgomery exponentiation (not SPA-resistant) targeted in [6]. Additionally, in this paper we provide more results depending on different compaction ratios.

### C. Scan-based attack Countermeasures

Scan-based attack countermeasures may be classified into three different categories: (1) methods to control the access to the test facilities through the use of secure test wrappers [12]; (2) methods to detect unauthorized scan operations [13] such as probing and other invasive attacks; (3) methods that provide confusion of the stream shifted out from the scan outputs [13]. Additionally, it was suggested in [2] and [3] that advanced industrial DfT methods such as response compaction are enough to impede any attack. However authors [3] do not consider differential scan-based attacks, and therefore they suppose that identifying registers that store secret data is not feasible. Advanced attacks [9] have been conceived to hack AES ciphers even in presence of linear compactors.

## III. Elliptic Curve Cryptosystems

For public-key cryptographic implementations, ECC provides equivalent security to RSA with much smaller key sizes (160-bit ECC equivalent to 1024-bit RSA). An elliptic curve E over prime fields may be defined as the set of points P(x, y) satisfying an elliptic curve equation of the form:

$y^2 \bmod p = x^3 + ax + b \bmod p$, where x, y, a and b belong to a finite field $F_p$ and p is a prime number.

In this paper, we are using the 192-bit NIST ECC curve and work in prime fields ($F_p$), where the curve parameters used in our ECC implementation are obtained from [14].

ALGORITHM I. Montgomery Ladder for Elliptic Curve Cryptography

*Inputs*: $k = (1, k_{m-2}, \ldots, k_1, k_0)_2$, $P \in E(F_p)$
*Output*: $Q_0 = k.P$

1: $Q_0 \Leftarrow P$
2: $Q_1 \Leftarrow 2P$
3: **for** i = m − 2 to 0 **do**
  **if** ki =0
    $Q_1 = Q_0 + Q_1$, $Q_0 = 2\,Q_0$
  **else**
    $Q_0 = Q_0 + Q_1$, $Q_1 = 2\,Q_1$
  **end if**
4: **end for**
5: **return** $Q_0$

The scalar or point multiplication of P with k is generally performed using the Montgomery Powering Ladder [15] (shown in Algorithm I). This is one of the most efficient methods of performing scalar multiplication. This is so since it does not require any extra storage and has fast calculation times compared to other methods, thus allowing its efficient implementation. Also in this algorithm, a point addition and doubling operation are performed in each iteration of the main loop, making it resistant to simple power analysis attacks.

### A. Attacker Scenario

The attack proposed in this paper can be classified as chosen-point attack, where the attacker knows the point P. We consider the case that the attacker can observe the result of each ECC point multiplication with help of the DfT structure. The base point is assumed to be stored in non-volatile memory (ROM) while the scalar for the point multiplication is stored in RAM, as it keeps on changing. This is the usual case with smart-card implementations.

We consider various scenarios where the ECC point multiplication result can be observed or controlled. In these cases, we perform our scan attack on the ECC point multiplication at a certain point in the execution of one of the cryptographic protocols employing ECC. For instance, in the signature generation phase of the Elliptic Curve Digital Signature Algorithm (ECDSA) algorithm, knowing scalar ‘k’ in the generation of (x1, y1) = k.G (G is the base point) which is used to derive the signature, r = x1 mod n, we can deduce part of the secret. Another possible case is Elliptic curve based static Diffie-Hellman (ECDH) Key Exchange which is normally used to setup the session key for communication employing symmetric-key encryption. Here, the target of our attack will be generation of the public-key for the two communicating parties ($x_A$.G or $x_B$.G), which is actually an ECC point multiplication of the scalar secret keys with the generator base-point.

### B. ECC Implementation

As seen in Algorithm I, a point multiplication is performed through a series of point addition ($Q_0 + Q_1$) and point doubling ($2\,Q_0$ or $2\,Q_1$) steps. We perform both point addition and point doubling in projective coordinates for avoiding the costly modular inversion operation. In

projective coordinates, $Q_0$ and $Q_1$ cost each three 192-bit register ($Q_0[X]$, $Q_0[Y]$ and $Q_0[Z]$ and so on). There are various methods of performing point addition and doubling. We have used the 1998 Cohen–Miyaji–Ono mixed coordinates [16] for point addition, and 2007 Bernstein–Lange formulae [17] for point doubling from the Explicit Formulae database [18].

Irrespective of the implementation method, intermediate values $Q_0$ and $Q_1$ are temporarily stored in flip-flops (FFs) to be available at the next iteration step. These FFs are referred to as Secret Flip-Flops (SFFs) in the rest of this paper and will be the target of our scan attack.

## IV. Principles of Scan Attacks

### A. Assumptions of Scan Attacks

The scan attack proposed in this paper relies on some assumptions:

- the cryptographic algorithm is known;
- the scan chain structure is not known to the attacker. However, the input/output test pins are controllable;
- it is possible to control the scan enable pin and also to switch from functional mode to test mode, which allows the cipher operation to be interrupted at any moment;
- it is possible to control the cipher inputs and to observe the values related to the intermediate states by means of scan out;

### B. Differential Scan Attack

Contrary to the previous ECC scan attack, our approach works in the presence of test compression and X-masking. We use the differential mode [8][9] to overcome the obscurity caused by response compactors inserted by most of the industrial DfT tools. Fig. 1 shows a crypto core, its cipher plaintext, and the intermediate register that is inserted in a multiple scan chain circuit with response compaction. The rest of the circuit is not shown to give a more clear description. Real scenarios where other circuit registers are present are discussed in Section V.

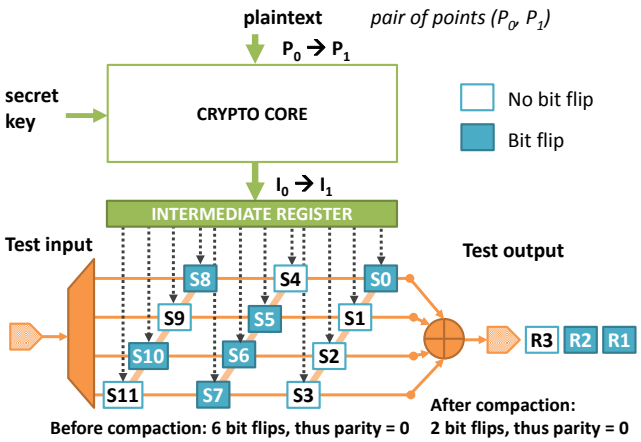


**Fig. 1.** Hamming Differences in the intermediate register observable at the test responses

Boxes S0 to S11 represent the 12 SFFs in the intermediate register. Due to the response compaction, the test response R1 stores the parity of S0 to S3 (slice 1), R2 stores the parity of S4 to S7 (slice 2) and R3 stores the parity of S8 to S11 (slice 3). It means that the actual value stored in any of these SFFs is not directly observable at the test responses. However, a difference in one of these SFFs implies a difference on the slice parity and hence at the response. This can be exploited by the attacker.

In the differential mode, a pair of plaintexts (points in the case of ECC) is applied, for example ($P_0$, $P_1$). The circuit is first reset and the message $P_0$ is loaded. Then after N clock cycles of mission mode (N depending on the cryptographic implementation), the circuit is halted and the intermediate register state $I_0$ is shifted out. The same procedure is repeated for the message $P_1$ for which $I_1$ is obtained.

Let the Hamming distance between $I_0$ and $I_1$ be equal to 6, as shown in Fig. 1. In this example highlighted boxes represent a bit flip between $I_0$ and $I_1$. We also suppose that the bits S0, S5, S6, S7, S8 and S10 are flipping. The parity of the differences at the intermediate register is equal to 0, since the Hamming distance is even. The flip at S0 is sensed at the test response R1. R2 flips due to odd number of flips at slice 2, while R3 does not flip since slice 3 has an even number of flips.

The parity of flips in the intermediate register matches with the parity of flips at the output of the response compactor. This comes from a basic property of this kind of response compactors: the parity of the Hamming Distance at the test output is equal to the parity of the Hamming Distance at the intermediate register. This property is valid for any possible configuration of scan chains (number of scan chains versus slices). Additionally it is also valid for compactors with multiple outputs. In this case, the measured parity should consider all compactor outputs. Thus using the differential mode the attacker may observe differences in the intermediate register, leading to a security breach.

## V. Distinguishing Attack

In this section, we show how the parity may be used as a distinguisher in order to retrieve the secret key using a chosen point attack. We first consider that the SFFs are inserted in the scan chain. Then, in Section V.A the practical aspects of leakage analysis are shown, for instance, when there are other FFs changing with the SFFs in the scan chains, or when not all of the SFFs are on the scan chains.

As presented in Section III, the Montgomery Powering Ladder method consists of repeating point addition and point doubling several times. For each iteration, the value stored in $Q_0$ is the result of the point doubling if the key bit is 0; otherwise $Q_0$ stores the result of the point addition. Thus observing $Q_0$ allows the detection of the current value of the key bit. This procedure must be repeated for the entire key length (192 bits in our example).

In order to detect if $Q_0$ stores the value of the point addition or the value of the point doubling, $Q_0$ must be scanned out. Additionally, as explained in the Section V.B, differential attacks use pairs of plaintexts to overcome the obscurity due to response compaction. In the case of ECC, plaintexts are actually the input points. This pair must be properly chosen so that a difference on the parity of $Q_0$ would lead to the key bit. The process to derive 'good' pairs of points is shown in Fig. 2:

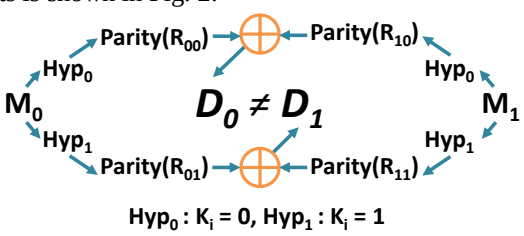


**Fig. 2.** Hypothesis Decision

These useful pairs are software generated. First, a random pair of 192-bit points is generated using a software pseudo-random number generator. We denote them here as ($P_0$, $P_1$). Then, the corresponding output responses (after one iteration of the multiplication algorithm) are computed for each of these points assuming the current key bit to be '0' (Hypothesis $Hyp_0$) or '1' (Hypothesis $Hyp_1$). Let $R_{ij}$ be the response to point $P_i$ with a current key bit j. For instance, ($R_{00}$, $R_{01}$, $R_{10}$, $R_{11}$) are the responses to points $P_0$ and $P_1$ for key bit '0' and '1' respectively. Let Parity($R_{00}$), Parity($R_{01}$), Parity($R_{10}$) and Parity($R_{11}$) be the parities of these responses.

Let $D_0$ be the Hamming distance between Parity($R_{00}$) and Parity($R_{10}$) and $D_1$ be the Hamming distance between Parity($R_{01}$) and Parity($R_{11}$). If $D_0 \neq D_1$, then the points ($P_0$, $P_1$) are taken to be useful, otherwise they are rejected because the value of the current key bit cannot be deduced from the responses to ($P_0$, $P_1$). This process is thus repeated till a pair of 'good' points is obtained.

After a good pair of points is calculated, it may be applied to the actual circuit. For both elements of the pair, the application is executed in mission mode for the number of clock cycles corresponding to the targeted step (key bit). Then, switching to test mode, the scan contents are shifted out and the difference of parities at the test output bitstream is measured. If it is equal to $D_0$, then the hypothesis 0 is correct and the secret key bit is 0. If it is equal to $D_1$, then the secret key bit is 1. This procedure is repeated for all the bits of the secret scalar.

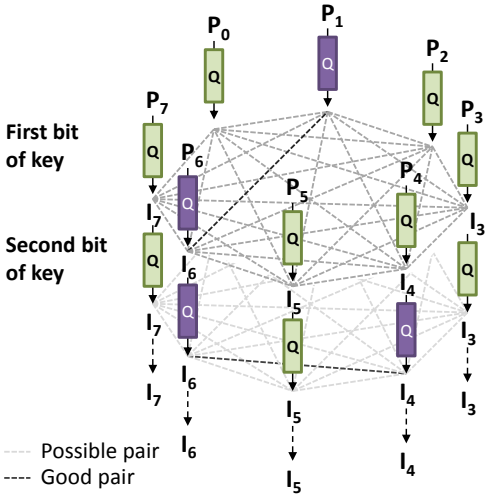


**Fig. 3:** Finding good set of points

Fig. 3 depicts an example of good pair choice. The points $P_0$ to $P_7$ represent a set of points which contains at least one pair that verifies the good pair property for each bit of the secret key. $I_0$ to $I_6$ represents the intermediate register $Q_0$ (or $Q_1$ depending on the attack target). As it can be seen, for retrieving the first bit of the key, the attacker needs a single pair of points ($P_1$, $P_6$). Then for retrieving the second bit of the private key, if the pair ($P_1$, $P_6$) does not satisfy the good property, then a third point must be added to the set of points, that satisfies the good property when combined with at least one of the previous pairs ($P_1$ or $P_6$). In this case $P_4$ verifies the good property with $P_6$. Verifying the good property of a new point against all the previous points reduces the number of required points drastically. Theoretically, the probability of verifying the good property for each bit of the secret key given a set with n points is:

$$Pgp = \frac{\frac{\binom{n}{2}}{2} - 1}{\frac{\binom{n}{2}}{2}}$$

Suppose n equal to 9, the probability of finding the good property is 94.44%.

### A. *Practical Aspects: Leakage Analysis*

In the procedure above we assumed that the entire state of $Q_0$ is inserted at the scan chain ($Q_1$ can be targeted as well without any additional issue) and that there are no other flip-flops in the scan chain. However, in presence of partial scan or X-Masking logic, which is used to prevent unpredictable internal states that can corrupt the test output [19], part of the $Q_0$ register may be filtered and thus not totally observable. In addition, scan chains may include scan FFs not related to the crypto core. We detail here the proposed attack in such cases.

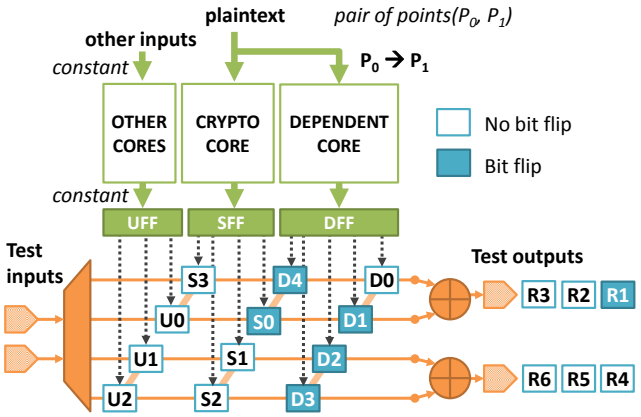


**Fig. 4.** Leakage Analysis

Fig. 4 shows a design containing three types of FFs, depending on the value they store. UFFs correspond to the other cores in the design that store data unrelated to the secret. SFFs belong to the registers directly related to the intermediate register, which store information related to the secret key. DFFs store data related to the cipher but not the intermediate registers themselves (such as input/output buffers or other cipher registers). The leakage, if it exists, concerns the SFFs. Fig. 4 shows examples of different scenarios of test responses that are classified in three main cases:

#### 1) *Useless responses*

Useless responses are the ones that are not related to the secret. In other words, the response depends only on UFFs (it is the case of R6).

#### 2) *Exploitable responses*

Exploitable responses are the ones that depend only on SFFs or on SFFs plus UFFs (like R3 and R5). The attacker can observe SFF flips at these response bits. It must be noticed that UFFs present no difference since the "other inputs" are kept constant during all the differential procedure.

#### 3) *Non-exploitable responses*

Non-exploitable responses are the ones that depend at least on one DFF (like R2, R4 and R1). The attacker cannot observe SFF flips due to the presence of DFFs that are affected by the plaintext input whose logic is not supposed to be known. Section 5.B describes the reasons for not considering this as an inherent countermeasure.

The goal of the leakage analysis is to find out if a particular bit of the intermediate register (SFF) can be observed at one of the test response bits. Thus the analysis is focused on a single SFF at a time, looking for an eventual bit flip. In our example, we start with S0. We denote $S0_i^N$ as the value stored in S0 after N clock cycles while the design is running in mission mode from the point $P_i$ (the first event in mission mode is a reset). Similar definitions hold for the other scan FFs. In order to find out which test response bit is related to S0, we use the differential procedure described in Section IV.B with a special set of pairs $(P_i, P_j)_0$ with the following property:

$(P_i, P_j)_0 : S0_i^N \neq S0_j^N$, and $SX_i^N = SX_j^N$ for X = 1 to 3.

In other words the pairs $(P_i, P_j)_0$ cause a single bit flip on the SFFs (in this case: S0), as highlighted in Fig. 4. Unfortunately, S0 is compressed with D4 that may eventually flip for the pair $(P_i, P_j)_0$ and thus mask the S0 flip. This is a case where R3 is not exploitable and thus differences on S0 are not observable.

If we repeat the same procedure for S1, S2 and S3, with respective set of pairs $(P_i, P_j)_1$ $(P_i, P_j)_2$ $(P_i, P_j)_3$, we will notice that the only test response bits that always change are R3 and R5 (they are not masked by any DFF).

In the case of ECC this method must be repeated for all the intermediate register bits (192 times), until one leakage is found. It must be noticed that all the 192 bits of the intermediate register depend on the secret scalar. Thus observing at least one of them through the test response allows the attacker to retrieve the secret (using the good pairs described in Section V).

Finding set of pairs $(P_i, P_j)_X$ is rather easy since the first two operations to be executed in Algorithm I are independent of the key, but the results are stored in the intermediate register.

It must be noted that in SoC designs, each IP is probably compliant with the IEEE 1500 wrapper. It allows access to single IPs, and thus it may help the attacker to focus only on the cryptographic IP.

### B. *Inherent Countermeasure*

It must be noticed that the case "Non-exploitable responses" described in Section V.A may be eventually considered as an inherent countermeasure against scan attacks: if each SFF is in the same slice as a DFF. However, this solution contains two drawbacks. The first one is that it is not easily implementable because the position of the scan flip-flops is set by place and route (P&R) tools, and choosing the position at will requires internal modification of that design step. The second and most important issue concerns the security of this solution. In Section V.A we supposed that the attacker does not known the relationship between DFFs and the plaintext, in order to propose an attack that does not depend on deep knowledge of design details. There may be attackers that have this kind of information, which would compromise this countermeasure; however this is a case of security by obscurity.

### C. *Timing Estimative*

The scan-based attack on ECC is targeted at finding the secret scalar (192-bits in our case). It is crucial to find the exact time to scan out the contents of the intermediate registers using the scan chains.

Since the same hardware is commonly used for both encryption and decryption, we can run a second hardware with a known scalar in order to get the timing estimations. For instance, the attacker must find out the number of clock cycles that a pair of point operations takes. For our target Montgomery Powering Ladder, it is the sequence of point addition and doubling operations. With a known scalar, we know the number of point operations required for the addition and doubling operations of the ECC point multiplication (Algorithm I). Dividing the total time of execution for this multiplication by the number of pairs of operations gives the approximate time required for one combined point addition and doubling operation. Then using repeated trial-and-error steps of comparing the actual output with the expected result after one pair of point operations, it may be possible to find out the exact number of clock cycles required.

## VI. SCAN ATTACK EXPERIMENTAL SET-UP

In order to perform the attack in actual designs, we developed a scan attack tool. It has two main features: to find the possible leakage points (as explained in Section V), and to automatically apply the attack method to a given gate-level netlist of a design. With the help of this tool, the security of several DfT configurations may be analyzed. Besides the attack on ECC explained in this paper, this tool can also comprise of methods to attack RSA [6], as well as AES. More details on the attack tool can be found in [6].

## VII. EXPERIMENTAL RESULTS AND DISCUSSION

In order to test the effectiveness of the attack, we implemented a 192-bit ECC algorithm in hardware using the Gezel environment [20], and then synthesized the gate-level design using Synopsys DfT Compiler (v2010.12) with a TSMC 130nm library. The total number of FFs in the design is 3355. Out of these, 855 belong to the UFF type. $Q_0$ consists of 576 FFs, from which each projective coordinate ($Q_0[X]$, $Q_0[Y]$ and $Q_0[Z]$) has 192 FFs (SFF type). $Q_1$ contains 576 FFs as well (also SFF type). And finally 1348 belong to the DFF type (see Section V A). For all the test cases, Synopsys DfT Compiler was used to insert the DfT structures, including decompression/compaction and X-Masking logic. All the runs were performed on a 4 GB Intel Xeon CPU X5460 with four processors.

#### 1) *Scan attack timing & number of points*

Concerning the attack timing, it consists in two phases: identification of leakage points, as described in Section V A; and the use of these observable points to perform the attack. The first phase takes approximately 49.37 seconds per bit (in average) and the second one takes 49 seconds. 89% of both the timings are due to the design simulation in ModelSIM SE, the remaining time comes from the C++ code execution.

For our test case, we required around 8 points over the elliptic curve to find out the secret multiplier *k*. In other words, all the necessary good pairs $(P_0, P_1)$ are created from these 8 points.

#### 2) *In Presence of DfT Methods*

A set of representative DfT configurations for which the scan attack was proven to be effective are shown in Table I. Config. 1 and Config. 2 have no compression structures. Config. 3 and Config. 4 have 120 scan chains and they have compression rate equal to 12 and 24 respectively. Config. 5 shows a test case where X-Masking is activated. In Table I, 'attack successful' means that all the bits of the secret scalar were retrieved.

TABLE I. DFT CONFIGURATIONS

| | # of scan chains | Compress rate | X-Masking | # of points | Attack successful* |
|---|---|---|---|---|---|
| **Config. 1** | 1 | No compression | No | 8 | Yes |
| **Config. 2** | 120 | No compression | No | 8 | Yes |
| **Config. 3** | 120 | 12 | No | 9 | Yes |
| **Config. 4** | 120 | 24 | No | 8 | Yes |
| **Config. 5** | 120 | 12 | Yes | 9 | Yes |

*All secret scalar bits successfully retrieved in the attack

*3) In presence of proposed countermeasures*

*a) In presence of inverters*

One of the countermeasures proposed in the literature is the insertion of dummy inverters after some SFFs of the scan chain [21]. This technique aims at confusing the attacker, since the sensitive data observed at the scan chain may be inverted. However, these inverters are placed always at the same location in the scan chain and thus they are completely transparent to differential scan attacks.

The effectiveness of the attack against this countermeasure was validated on the ECC design with Config. 3 of Table I. Two implementations were considered with 1677 and 2516 inverters (50% and 75 % of the overall FFs in the design respectively) randomly inserted in the scan chains. For both cases, the tool was able to find leakage points and then to retrieve the secret scalar.

*b) In presence of partial scan*

Depending on the design, not all the FFs need to be inserted in the scan chain in order to achieve high testability. As proposed in [22], partial scan may be used for increasing the security against scan attacks. However, the authors suppose that the attacker needs the whole sensitive register to retrieve the secret key. As was described in Section V, the leakage analysis feature can be used to find out which bits of the sensitive register are inserted in the scan chain. Once these bits are identified, the attack can proceed with only partial information, since each bit of the sensitive register is related to the key.

For evaluating the effectiveness of the scan attack tool in the presence of partial scan, we configured the DfT tool in such a way so as not to insert some of the sensitive register FFs in the scan chain. In the first case, half of the SFFs were inserted in the chain. The tool was able to correctly identify all the leaking bits and then to retrieve the secret scalar. Also in the worst case situation, i.e., where only one secret bit was inserted in a scan chain of a design with multiple scan chains and test compression, the tool was still able to find out the correct secret scalar.

## VIII. CONCLUSION

In this paper, we show that the scan-based attack that is useful against RSA implementation can be adapted to work against ECC. Actual gate-level designs with advanced DfT configurations were evaluated against the proposed attack. With the help of our new scan attack setup we retrieved the secret in ECC circuits containing multiple scan chains with linear response compaction and X-Masking. We also experimentally proved the effectiveness of our new attack against some countermeasures proposed in the literature.


## ACKNOWLEDGEMENT

This work was supported in part by the Research Council K.U.Leuven: GOA TENSE (GOA/11/007) and by the European Commission through the ICT programme under contract ICT-2007-216676 ECRYPT II. In addition, this work is supported in part by the Flemish Government, FWO G.0550.12N, by the Hercules Foundation AKUL/11/19.


## REFERENCES


[1] Nara, R.; Togawa, N.; Yanagisawa, M.; Ohtsuki, T.; , "Scan-based attack against elliptic curve cryptosystems," 15th Asia and South Pacific Design Automation Conference (ASP-DAC) 2010.

[2] "High Quality Test Solutions for Secure Applications", Mentor Graphics, Silicon Test and Yield Analysis Whitepaper, April 2010.

[3] Liu, C.; Huang, Y.; , "Effects of Embedded Decompression and Compaction Architectures on Side-Channel Attack Resistance," VLSI Test Symposium, 2007. 25th IEEE, pp.461-468, 6-10 May 2007.

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

[5] Common Criteria for Smart Cards, http://www.commoncriteriaportal.org/

[6] Da Rolt, J.; Das A.; Di Natale, G.; Flottes, M.-L.; Rouzeyre, B.; Verbauwhede, I.; , "A New Scan Attack on RSA in Presence of Industrial Countermeasures", COSADE 2012, Lecture Notes in Computer Science Volume 7275, 2012, pp 89-104.

[7] Yang, B.; Wu, K.; Karri, R.; , "Scan based side channel attack on dedicated hardware implementations of Data Encryption Standard," International Test Conference, 2004.

[8] Yang, B.; Wu, K.; Karri, R.; , "Secure Scan: A Design-for-Test Architecture for Crypto Chips," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2006.

[9] Da Rolt, J.; Di Natale, G.; Flottes, M.-L.; Rouzeyre, B.; , "Are advanced DfT structures sufficient for preventing scan-attacks?," VLSI Test Symposium (VTS), 2011 IEEE, pp.246-251, June 2012.

[10] Liu, Y., Wu, K., Karri, R.; , "Scan-based Attacks on Linear Feedback Shift Register Based Stream Ciphers," ACM Transactions on Design Automation of Electronic Systems (TODAES) 2011.

[11] Nara R.; Satoh, K.; Yanagisawa, M.; Ohtsuki, T.; Togawa, N; , "Scan-based side-channel attack against RSA cryptosystems using scan signatures," IEICE Transaction Fundamentals, 2010.

[12] Das A.; Knezevic, M.; Seys, S.; Verbauwhede, I.; , "Challenge-response based secure test wrapper for testing cryptographic circuits Test Symposium, 2011. European, May 2011.

[13] Hely, D.; Bancel, F.; Flottes, M.L.; Rouzeyre, B.; , "Secure scan techniques: a comparison," Test Symposium, 2005. European, pp. 190- 195, 22-25 May 2005.

[14] Hankerson D., Menezes A., and Vanstone S.: Guide to Elliptic Curve Cryptography, pp. 262, Sample parameters.

[15] Montgomery, P.; , "Speeding the pollard and elliptic curve methods for factorizations," Mathematics of Computation, vol. 48, pp. 243-264, 1987.

[16] Cohen H.; Miyaji, A.; Ono, T.; , "Efficient Elliptic Curve Exponentiation Using Mixed Coordinates," ASIACRYPT'1998.

[17] J. Bernstein, D.; Lange, T.; , "Analysis and optimization of elliptic-curve single-scalar multiplication," 2000 Mathematics Subject Classification.

[18] Explicit Formula Database for Jacobian coordinates with a4=-3 for short Weierstrass curves, http://www.hyperelliptic.org/EFD/g1p/auto-shortw-jacobian-3.html/

[19] Mitra, S.; Kim, K.; , "X-compact: an efficient response compaction technique for test cost reduction," Test Conference, 2002. Proceedings. International, pp. 311- 320, 2002.

[20] Gezel Hardware/Software Codesign Environment, http://rijndael.ece.vt.edu/gezel2/

[21] Sengar, G.; Mukhopadhayay, D.; Chowdhury, D.R.; , "An Efficient Approach to Develop Secure Scan Tree for Crypto-Hardware," ADCOM 2007.

[22] Inoue, M.; Yoneda, T.; Hasegawa, M.; Fujiwara, H.; , "Partial Scan Approach for Secret Information Protection," Test Symposium, 2009 14th IEEE European, pp.143-148, 2009.