New Single-Trace Side-Channel Attacks on a Specific Class of Elgamal Cryptosystem

N. Mahdion, Hadi Soleimany, Pouya Habibi, Farokhlagha Moazami

Cyberspace Research Institute, Shahid Beheshti University

Abstract. In 2005, Yen et al. proposed the first $N-1$ attack on the modular exponentiation algorithms such as BRIP and square-and-multiply-always methods. This attack makes use of the ciphertext $N-1$ as a distinguisher to obtain a strong relation between side-channel leakages and secret exponent. The so-called $N-1$ attack is one of the most important attacks, as it requires a non-adaptive chosen ciphertext which is considered as a more realistic attack model compared to adaptive chosen ciphertext scenario. To protect the implementation against $N-1$ attack, several literatures propose the simplest solution, i.e. "block the special message $N-1$". In this paper, we conduct an in-depth research on the $N-1$ attack based on the square-and-multiply-always (SMA) and Montgomery Ladder (ML) algorithms. We show that despite the unaccepted ciphertext $N-1$ countermeasure, other types of $N-1$ attacks is applicable to specific classes of Elgamal cryptosystems. We propose new chosen-message power-analysis attacks which utilize a chosen ciphertext c such that $c^2 = -1 \bmod p$ where p is the prime number used as a modulus in Elgamal. Such a ciphertext can be found simply when $p \equiv 1 \bmod 4$. We demonstrate that ML and SMA algorithms are subjected to our new $N-1$-type attack by utilizing a different ciphertext. We implement the proposed attacks on the TARGET Board of the ChipWhisperer CW1173 and our experiments validate the feasibility and effectiveness of the attacks by using only a single power trace.

Key words: Elgamal cryptosystem, Side-channel attacks, Montgomery Ladder, Square and Multiply Always, $N-1$ attack

1 Introduction

The implementation of cryptographic algorithms is a complicated and important process. Although most of the cryptographic primitives are mathematically secure, their unprotected implementations on embedded devices can pose serious threats by exploiting side-channel information leakages. This information includes power consumption, electromagnetic radiation, system run times, acoustic and etc. which are correlated with the secret values during data processing. Power-analysis attacks are a powerful type of side-channel attack originally described by Kocher. This class of attacks has been applied successfully against the implementations of popular public-key cryptosystems RSA and Elgamal [14]

which make use of exponentiation algorithms. The primary side-channel attacks against modular exponentiation algorithms rely on certain physical phenomena, which allows one to distinguish between multiplication and squaring operations [13]. Messerges et al. proposed three types of power-analysis attacks against RSA with multiple random plaintexts[15]. To mitigate these attacks, the implementations of modular exponentiation utilize a same sequence of instructions for multiplication and squaring operations, which makes it challenging to differentiate between these two operations for random input messages in practice [12]. In response, various methods have been proposed that use the leak of sensitive information during the decryption process of chosen messages [19, 21, 17]. In particular, several chosen-message attacks have been applied on public key encryption in [2, 3, 7, 8, 10, 11, 12, 16, 20].

Simple power analysis (SPA) attacks on RSA implemented based on CRT mode with adaptively chosen messages is proposed by Novak[17]. Furthermore, a differential power analysis (DPA) attack based on the Hamming weight of an intermediate value was applied to RSA in the CRT mode [4]. "Doubling attack" is another type of the chosen-message attack in which the attacker can retrieve the secret exponent by generating a collision between the corresponding power traces of the two related messages X and X^2 [6].

However, the DPA attack proposed in [4] cannot be applied to the Montgomery Ladder algorithm. In addition, methods like inserting dummy operations may defeat the aforementioned attacks, since these attacks require more than one power trace. Single-trace side-channel attacks, which utilize a single measurement are more devastating than other types of power-analysis attacks which make use of several measurements and can be mitigated by methods like utilizing dummy operations. As another direction, Yen et al. described a simple yet effective attack based on a particular input data $N-1$ where N is a modulus in RSA or Elgamal cryptosystems [22]. The input $N-1$ generates internal collisions in the state of the modular exponentiation which make it possible to distinguish between squaring and multiplying operations in one power trace. The $N-1$ attack can be applied to the left-to-right square-multiply-always and BRIP algorithms. Further research on the $N-1$ attacks was performed by Ding et al. who proposed to utilize new input messages that are applicable to Boscher's right-to-left exponentiation algorithm [5]. Furthermore, they found out that the Montgomery Powering Ladder is subjected to the $N-1$ attack.

Several researches have been dedicated to protect the implementation of exponentiation functions against side-channel attacks. However, a strong countermeasure cannot be established for free. It has an impact on the latency, power consumption and size of the implementation. Consequently, simple countermeasures in real-world applications are preferred as they utilize a small overhead, rather than complicated solutions with notable extra cost. In this direction, to protect the implementation against the $N-1$ attack, several articles propose the simplest solution, i.e. "block the special message $N-1$".

In this paper, we propose a new type of the $N-1$ attack on a specific class of Elgamal which is implemented based on the square-and-multiply-always (SMA)

or Montgomery Ladder (ML) algorithms. In our attack, we utilize a chosen ciphertext c such that $c^2 = -1 \bmod p$ where p is the prime number used as a modulus in Elgamal. Such a ciphertext can be found simply when $p \equiv 1 \bmod 4$. We demonstrate that SMA and ML algorithms cannot be defeated by $N-1$-type attacks even if the special message $N-1$ is blocked by the user. To evaluate the theoretical model, we implement the SMA and ML algorithms on an Atmel ATXMEGA128D4 8-bit microcontroller that is located on the TARGET Board of the ChipWhisperer CW1173 and applied the proposed attack on these algorithms. The experimental results show that our attack is applicable to these two algorithms by using only a single power trace.
The remainder of this paper is organized as follows: Section 2 gives an overview of previous works. Section 3 describes our new attack and how it works on SMA and ML algorithms. Section 4 presents our experimental results. Finally, Section 5 summarizes the paper and suggests further research.

2 Previous Comparative Power Analysis Attacks

In order to recover secret information of RSA or Elgamal cryptosystems from exponentiation algorithms that are assumed secure against side-channel attacks, several power-analysis attacks have been proposed based on (adaptive) chosen messages during the last decade. Several proposed attacks use a common power analysis technique and fall under the same umbrella. The main idea of the variety of approaches is based on the fact that, dual execution of the same operation for the same values is detectable in the power traces. In particular, by performing two multiplications $a \times b$ and $c \times d$, one can expect the corresponding power consumption traces to be similar if the pair values (a, b) and (c, d) are (almost) equal $(a = c, b = d)$. Similarly, it is to be expected to have dissimilar power consumption traces if unequal values with different size are used. In this section we briefly review the previous comparative power analysis attacks.

2.1 Doubling attack

In Doubling attack, it is assumed that the attacker has access to the two power consumption traces corresponding to the decryption of two related messages X_1 and X_2. The related messages should be selected smartly to cause collisions between their power traces at some time frames. The exact locations of such collisions in the power traces can be used to retrieve the secret exponent. Doubling attacks can be applied to different exponentiation algorithms. In what follows, we describe the basic idea of the attack presented on the left-to-right SMA exponentiation algorithm that is described in Algorithm 1. To reveal the secret exponent, the adversary uses two related inputs $X \mod N$ and $X^2 \mod N$ to make a collision between the corresponding power consumption traces. According to Algorithm 1, when the $i-th$ bit of the exponent equals to 0 a collision is generated between the squaring operation at the $(i-1)-th$ and $i-th$ cycle in the power trace of X and X^2, respectively. Figure 1 shows this verity for the secret exponent of "101001...".

Algorithm 1 Left-to-Right Square and Multiply Exponentiation

Require: c, $D = (d_{l-1}, ..., d_1, d_0)_2$, p.
Ensure: $m = c^D \bmod p$.
1: $m \leftarrow 1$
2: **for** $i = l - 1$ downto 0 **do**
3: $\quad m \leftarrow m \times m \bmod p \quad \triangleright S$
4: $\quad$ **if** $d_i = 1$ **then**
5: $\quad\quad m \leftarrow m \times c \bmod p \quad \triangleright M$
6: $\quad$ **end if**
7: **end for**
8: **return** m

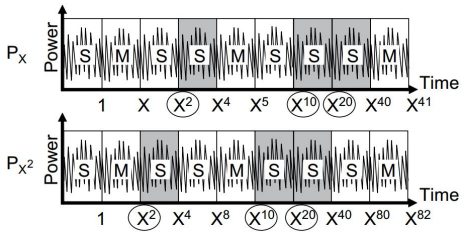

Fig. 1: Doubling attack[12].

2.2 $N - 1$ Attack on SMA Algorithm

In this section, we describe a side-channel attack that makes use of a specific ciphertext $N - 1$ to generate internal collisions in a single trace. This attack is based on the observation that $(n-1)^j \mod n$ equals to $n-1$ and 1 when j is odd and even, respectively. This observation is used in [22] to attack the square-and-multiply-always exponentiation algorithm which is described in Algorithm 2. Two cases are possible in the i-th iteration which depend on the value of d_{i-1}:

- With the assumption that $d_{i-1} = 0$, the input intermediated value m equals to 1. Consequently, the square operation is computed as $S_i : m = 1^2 = 1 \mod n$ in the i-th iteration. Then the multiply operation is performed as $M_i : t = 1 \times (N - 1)$.
- With the assumption that $d_{i-1} = 1$, the input intermediated value m equals to $N - 1$. Consequently, the square operation is computed as $S_i : m = (N-1)^2 = 1 \mod n$ in the i-th iteration. After that the multiply operation is performed as $M_i : t = 1 \times (N - 1)$.

As a result, the SM-sequence resulting from the computation of the SMA exponentiation contains two types of squaring operations, which creates a type of side-channel leakage that depends on the (secret) exponent. The squaring operation in each iteration is either 1^2 (when $d_{i-1} = 0$) or $(N-1)^2$ (when $d_{i-1} = 0$), in which it is expected the corresponding power consumption traces to be different. This fact allows the attacker to distinguish between performing 1^2 and $(N-1)^2$

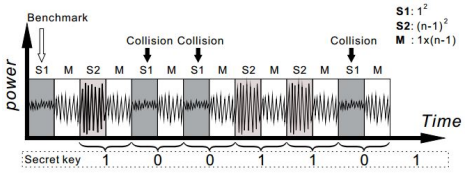

Fig. 2: (N-1) attack on Square and Multiply Always[5].

and consequently it is possible to recover the secret key. Figure 2 demonstrates an attack on SMA exponentiation.

Algorithm 2 Square-and-multiply-always (SMA) Exponentiation

Require: c, $d = (d_{l-1}...d_0)_2$, p .
Ensure: $m = c^d \bmod p$.
1: $m \leftarrow 1$
2: **for** $i = l - 1$ downto 0 **do**
3: $m \leftarrow m^2 \bmod p$ ▷ S_i
4: $t \leftarrow m \times c \bmod p$ ▷ M_i
5: **if** $d_i = 1$ **then**
6: $m \leftarrow t$
7: **end if**
8: **end for**
9: **return** m

2.3 Further research on $N - 1$ attack

Ding et al.[5] presented an in-depth study on the $N - 1$ attack. They carried out new types of $N - 1$ attacks on Boschers right-to-left binary exponentiation algorithm (Algorithm 3) by utilizing two new chosen ciphertexts 1 and $N + 1$. They also investigated the $N - 1$ attack on the Montgomery Powering Ladder algorithm (Algorithm 4). In the following, we describe a summary of their work on both of the algorithms.

2.3.1 Boscher's Right-to-left Exponentiation Algorithm

In this section we describe a variant of $N-1$ attack on the Boschers right-to-left binary exponentiation algorithm. With the assumption that $R_2 = 1$ in the the first or second iteration, the values of register R_0 and R_1 remain unchanged during the execution of Algorithm 3. Therefore, the multiply operation (M) in line 6, performs as follows:

- If $d_i = 0$: $R_1 = R_1 \times 1 \bmod n = r^{-1} \times 1 \bmod n$.
- If $d_i = 1$: $R_0 = R_0 \times 1 \bmod n = r \times 1 \bmod n$.

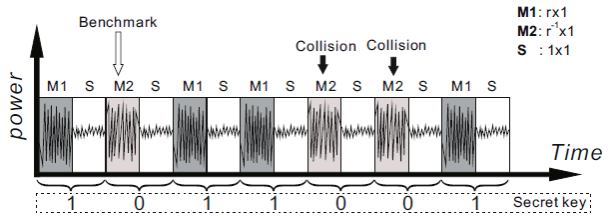

Fig. 3: New attacks on Boscher's Right-to-left [5].

By distinguishing between two M operations $(r^{-1} \times 1) \bmod n$ and $(r \times 1) \bmod n$ via the side-channel leakage, the adversary can recover the secret key bit by bit. Considering the fact that register R_2 in Algorithm 3 is the input message (ciphertext), there exist two methods for setting the value of register R_2 to 1. The first method is to set the ciphertext to a value of 1. The second method is to select the value of R_2 such that $R_2^2 \mod n = 1$. Therefore, if the attacker considers $(n-1)$ or $(n+1)$ as the ciphertext, the value of register R_2 becomes equal to 1 ($(n-1)^2 \bmod n = 1$ and $(n+1)^2 \bmod n = 1$) after the first iteration.

Algorithm 3 Boscher's Right-to-left Exponentiation (Boscher)

Require: c, n, $d_{l-1}...d_0)_2$.
Ensure: $m = c^d \bmod n$ or "$Error$".
1: chose a random integer r
2: $R_0 \leftarrow r$
3: $R_1 \leftarrow r^{-1} \bmod n$
4: $R_2 \leftarrow c$
5: **for** $i = 0$ to $l - 1$ **do**
6: $\quad R_{1-d_i} \leftarrow R_{1-d_i} \times R_2 \bmod n \quad \triangleright M_i$
7: $\quad R_2 \leftarrow R_2 \times R_2 \bmod n \quad \triangleright S_i$
8: **end for**
9: **if** $R_0 \times R_1 \times c = R_2$ **then**
10: $\quad$ **return** $r^{-1} \times R_0 \bmod n$
11: **else**
12: $\quad$ **return** "$Error$"
13: **end if**

Figure 3 illustrates attack on Algorithm 3, when the ciphertext is 1.

2.3.2 Montgomery Powering Ladder Algorithm

In the Montgomery Powering Ladder (MPL) algorithm, the square operation in the $(i-1)$-th iteration (S_{i-1}) depends directly on the relation between two consecutive secret exponent bits d_i and d_{i-1}. For $d_i = d_{i-1}$, the output of S_i and the input of S_{i-1} are the same ($OS_i = IS_{i-1}$). Conversely, for $d_i \neq d_{i-1}$, the output of M_i and the input of S_{i-1} are equal ($OM_i = IS_{i-1}$). For the ciphertext

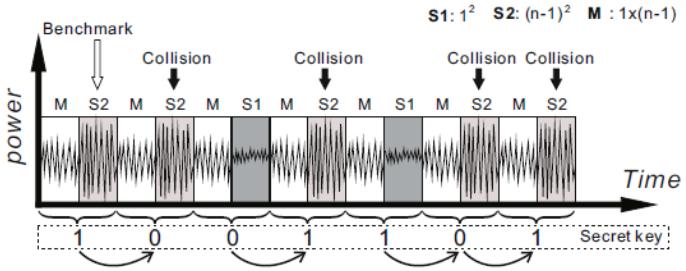

Fig. 4: $N-1$ attack on Montgomery Powering Ladder [5].

$(n-1)$ as the input of MPL algorithm, the outputs of M and S operations remain always $(n-1)$ and 1, respectively. Therefore, the attacker can find the relation between the two consecutive secret exponent bits d_i and d_{i-1}, as follows:

- If $d_i = d_{i-1}$ ($OS_i = IS_{i-1} = 1 \mod n$): the squaring operation S_{i-1} performs $(1 \times 1) \mod n$.
- If $d_i \neq d_{i-1}$ ($OM_i = IS_{i-1} = (n-1) \mod n$): the squaring operation S_{i-1} performs $((n-1) \times (n-1)) \bmod n$.

Figure 4 illustrates an example of the described attack on MPL exponentiation algorithm.

Algorithm 4 Montgomery Powering Ladder (MPL)

Require: c, n, $d = (d_l d_{l-1} ... d_0)_2$ $(d_l = 1)$.
Ensure: $m = c^d \bmod n$.
1: $R_0 \leftarrow c$
2: $R_1 \leftarrow R_0 \times R_0 \bmod n$
3: **for** $i = l-1$ downto 0 **do**
4: $\quad R_{1-d_i} \leftarrow R_{d_i} \times R_{1-d_i} \bmod n \quad \triangleright M_i$
5: $\quad R_{d_i} \leftarrow R_{d_i} \times R_{d_i} \bmod n \quad \triangleright S_i$
6: **end for**
7: **return** R_0

3 New Attacks

An efficient and simple countermeasure to prevent the $N-1$ attack on RSA and Elgamal is to block the ciphertext $N-1$ for the decryption [11]. In this section, by demonstrating attacks based on another ciphertext we show this simple countermeasure cannot defeat a specific class of an Elgamal cryptosystem. Our basic idea is similar to the original $N-1$ attack. We select a ciphertext

that enhances the differences between operations performed during the modular exponentiation algorithms according to the bit pattern of the secret key. We utilize a chosen message c such that $c^2 = -1 \bmod p$. In what follows, we first explain how a ciphertext c can be found such that $c^2 = -1 \bmod p$. After that, we show how the desired ciphertext c can be utilized o apply a single-trace side-channel attack on Square and Multiply Always (SMA) and Montgomery Ladder (ML) algorithms.

3.1 Generating the Desired Ciphertext

Our chosen-message side-channel attacks are based on the decryption of a chosen ciphertext c such that $c^2 = -1 \bmod p$ where p is the prime number used as a modulus in Elgamal. In this section, we show that such a ciphertext can be found in practice when $p \equiv 1 \mod 4$ by using Little Fermat Theorem.
Little Fermat Theorem: If p is a prime number and a is any number not divisible by p, then

$$a^{(p-1)} \bmod p = 1.$$

Based on the Little Fermat Theorem, we can conclude $a^{\frac{p-1}{2}} = \pm 1 \bmod p$ where a is an integer not divisible by p. In mathematics, if $a^{\frac{p-1}{2}} = 1 \bmod p$, a is called a quadratic residue mod p and if $a^{\frac{p-1}{2}} = -1 \bmod p$, a is called a nonresidue mod p. Let a be a nonresidue mod p so $a^{\frac{p-1}{2}} = p - 1 = -1 \bmod p$, when $p = 4k + 1$, we have

$$a^{\frac{p-1}{2}} = a^{\frac{4k+1-1}{2}} = a^{\frac{4k}{2}} = a^{2k} = (a^k)^2 \bmod p.$$

Let $a^k = c$, therefore we have $c^2 = -1 = p - 1 \bmod p$ and c is the appropriate number to choose as a distinguisher. In the next two subsections, we explain how we can use this number as a distingusher to obtain a relation between side-channel leakages and the secret exponent.

3.2 Attack on Square and Multiply Always Algorithm

One of the efficient algorithms for modular exponentiation is the Square and Multiply Always algorithm (SMA). Algorithm 2 illustrates an implementation of the SMA algorithm. In this section, we describe an attack that uses only one chosen ciphertext and is able to extract the secret key by one power consumption trace. By crafting a suitable ciphertext c such that $c^2 = p - 1 \bmod p$, the output of squaring (line 3 of Algorithm 2) is always 1 or $p - 1$. These values inter as input of the multiplication operation (line 4). Consequently, the output of multiplication operation (line 4) is either c or $p - c$. Hence, when the bit of the secret key is 1 then the input of the squaring operation is either c or $p - c$ and when the bit of the secret key is 0 then the the input of squaring operation is either 1 or $p-1$. By these observations, the bits of the secret key can be retrieved as follows.

Table 1: Summery of our attack on SMA Algorithm

d_i	Multiplication in $(i-1)$-th iteration
0	$(1 \times c) \mod p$ (has short trace)
1	$(p-1 \times c) \mod p$ (has long trace)

- If $d_i = 1$, the input of the multiplication operation at the $(i-1)$-th iteration is either c or $p - c$. In both cases, the multiplication operation performs $((p-1) \times c) \bmod p$.
- If $d_i = 0$, the input of the multiplication operation at the $(i-1)$-th iteration is either 1 or $p - 1$. In both cases, the multiplication operation performs $(1 \times c) \bmod p$.

In summary, as described above, the results of running Algorithm 2 on input c can be represented as Table 1.
Since $(p-1)$ is a large random-looking number in comparison to the value of 1, we expect that the differences in the patterns of two multiply operations $((p-1) \times c) \bmod p$ and $(1 \times c) \bmod p$. These differences can be easily perceived in a single power consumption trace by visual observation.

3.3 Attack on Montgomery Ladder Algorithm

Another popular algorithm for modular exponentiation is the Montgomery Ladder algorithm (ML) which is demonstrated in Algorithm 5. In the i-th iteration of the algorithm, both squaring and multiply operations perform independent of the value d_i.

Algorithm 5 Montgomery Ladder (ML) Exponentiation

Require: c, n, $d = (d_l d_{l-1} ... d_0)_2$ $(d_l = 1)$.
Ensure: $m = c^d \bmod n$.
1: $R_0 \leftarrow c$
2: $R_1 \leftarrow R_0 \times R_0 \bmod n$
3: **for** $i = l - 1$ downto 0 **do**
4: $\quad R_{1-d_i} \leftarrow R_0 \times R_1 \bmod n \quad \triangleright M_i$
5: $\quad R_{d_i} \leftarrow R_{d_i} \times R_{d_i} \bmod n \quad \triangleright S_i$
6: **end for**
7: **return** R_0

Let us assume that the ciphertext c is given as an input to the ML algorithm such that $c^2 = -1 \bmod p$. We show that how decryption of c can reveal a relation between two consecutive secret exponent bits d_i and d_{i-1}. To describe this process in more detail, we analyze the values of intermediate variables separately as follows:

Table 2: The executed operations for all of the modes $d_i \longrightarrow d_{i-1}$

$d_i \longrightarrow d_{i-1}$	$\begin{cases} M_{i-1} \\ S_{i-1} \end{cases}$
$1 \longrightarrow 1$	$\begin{cases} p-c \times 1 \\ 1 \times 1 \end{cases}$ or $\begin{cases} c \times p-1 \\ p-1 \times p-1 \end{cases}$
$0 \longrightarrow 0$	$\begin{cases} p-1 \times p-c \\ p-1 \times p-1 \end{cases}$ or $\begin{cases} 1 \times c \\ 1 \times 1 \end{cases}$
$1 \longrightarrow 0$	$\begin{cases} p-c \times 1 \\ p-c \times p-c \end{cases}$ or $\begin{cases} c \times p-1 \\ c \times c \end{cases}$
$0 \longrightarrow 1$	$\begin{cases} p-1 \times p-c \\ p-c \times p-c \end{cases}$ or $\begin{cases} 1 \times c \\ c \times c \end{cases}$

Observation 1 *During the execution of the modular exponentiation ML algorithm with the base c where $c^2 = p-1 \mod p$, the intermediate variables R_0 and R_1 always get one of the values c, $p-c$, 1 and $p-1$.*

Observation 1 follows from the fact that the the multiplicative order of $c \mod p$ is 4. Consequently, the intermediate variables R_0 and R_1 always get one of the values c, $c^2 = -1 = p-1 \mod p$, $c^3 = -c = p-c \mod p$, $c^4 = (-1)^2 = 1 \mod p$.

Observation 2 *During the execution of the modular exponentiation ML algorithm with the base c where $c^2 = p-1 \mod p$, only four cases are possible to be performed in the multiply operation $M_i : R_0 \times R_1$ (line 4 in Algorithm 5). The possible squaring operations which can be performed are: $(1 \times c)$, $(p-1 \times p-c)$, $(c \times p-1)$ and $(p-c \times 1)$.*

Observation 2 follows from the possible values for the variables R_0 and R_1 presented in Observation 1.

Observation 3 *During the execution of the modular exponentiation ML algorithm with the base c where $c^2 = p-1 \mod p$, only four cases are possible to be performed in the squaring operation $S_i : R_{d_i} \times R_{d_i}$ (line 5 in Algorithm 5). The possible multiplications which can be performed are: (1×1), $(p-1 \times p-1 = 1 \mod p)$, $(c \times c)$ and $((p-c) \times (p-c))$.*

Similarly, Observation 3 follows from the possible values for the variables R_0 and R_1 presented in Observation 1.
By considering Observation 2 and Observation 3, we categorize the executed operations for all of the modes $d_i \longrightarrow d_{i-1}$ $(0 \leq i \leq l-1)$ as Table 2 where M_{i-1} and S_{i-1} denote multiplication and squaring operations at the $(i-1)$-th iteration, respectively.

Table 3: Distinguishing the secret exponent bits on ML Algorithm

$d_i \longrightarrow d_{i-1}$	S_{i-1}
$d_i = d_{i-1}$	(1×1) or $(p-1 \times p-1) \mod p$
$d_i \neq d_{i-1}$	$(c \times c)$ or $(p-c \times p-c) \mod p$

While the above relation can be utilized for mounting a side-channel attack, a more precise analysis of information presented in Table 2 reveals that the modular squaring operation can be exploited solely to identify the secret exponent bits. When two consecutive secret exponent bits d_i and d_{i-1} are equal to each other, the performed squaring operation is either 1×1 or $(p-1) \times (p-1)$. Conversely, when two consecutive secret exponent bits d_i and d_{i-1} are not equal, the performed squaring operation is either $c \times c$ or $(p-c) \times (p-c)$. This point is illustrated in Table 3.

4 Experimental Verification

In this section, we discuss the practicality of the proposed attacks in Section 3. To verify the theoretical model, we implemented SMA and ML algorithms on an Atmel ATXMEGA128D4 8-bit micro-controller that was located on the TARGET Board of the ChipWhisperer CW1173 [18]. In order to conduct our attacks, we carried out two phases. In the first phase, we measured and save a single power consumption trace via the CAPTURE Board of the ChipWhisperer, while the TARGET Board is executing the decryption algorithm for the ciphertext c such that $c^2 = -1 \mod p$. In the second phase, we exploited the simple analysis of the power consumption trace to recover the secret exponent bits.
In the remainder of this section, we first present a preliminary experiment that we have performed to validate the proposed attack on an SMA algorithm. After that, we present a concrete experiment to attack the ML algorithm through a single power consumption trace captured from a real device.

4.1 Experimental results on Square and Multiply Always Algorithm

We sent the ciphertext c such that $c^2 = -1 \mod p$ to Algorithm 2 for the decryption process and measure the power consumption. Figure 5 illustrates the captured trace during the execution of the SMA exponentiation algorithm with the base c (such that $c^2 = -1 \bmod p$) and the exponent with the first bits of $(1001101)_2$. As it can be observable from Figure 5, it is simply possible to distinguish between the multiplications and the squaring operations. In order to obtain a clean and more readable figure, we illustrate the measured power consumption trace T by a sequence of subtraces $T = \{(T_S, T_M)_{l-1}, (T_S, T_M)_{l-2}, ..., (T_S, T_M)_1, (T_S, T_M)_0\}$ in which T_S and T_M correspond to squaring and multiply operations, respectively. The squaring operation is marked by S and the multiplication operation is marked by M which are also separated by vertical dot lines in Figure 5.

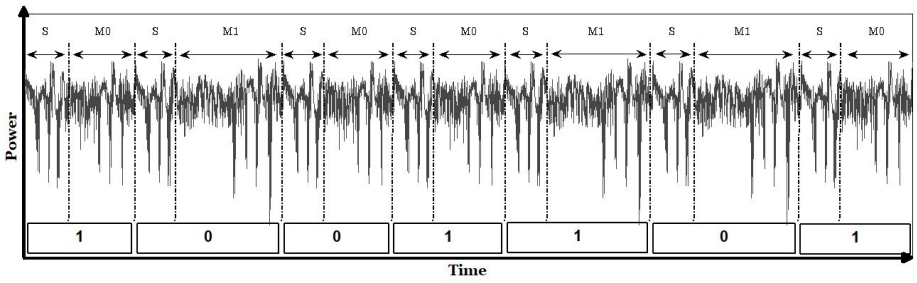

Fig. 5: Experimental results on SMA algorithm

The proposed attack in Section 3.2 is based on the expectation of an observable difference between two multiply operations $((p-1)\times c) \bmod p$ and $(1\times c) \bmod p$ in a single power consumption trace. Consequently, we consider only T_M subtraces that correspond to the mutliply operations: $T_M = \{T_{M_{l-1}}, T_{M_{l-2}}, ..., T_{M_1}, T_{M_0}\}$. The differences in the patterns of power consumption between the multiplications $(1 \times c) \bmod p$ and $((p-1) \times c) \bmod p$ is extremely robust and can be easily distinguished by visual observation. In Figure 5, the corresponding pattern of the multiply operation $(1\times c) \bmod p$ denoted by $M0$ has a significantly lower and shorter power consumption subtrace in comparison to the corresponding pattern of the multiply operation $((p-1) \times c) \bmod p$ denoted by $M1$. This observation conforms to our expectation, since $p-1$ is a large random-looking number in comparison to the value of 1. Finally, we can recover each exponent bit directly by visual observation of the power consumption trace.

4.2 Experimental results on Montgomery Ladder Algorithm

We first briefly recall the main observation that is used in the proposed attack on ML algorithm in Section 3.3. In the process of decrypting the ciphertext c such that $c^2 = p-1 \bmod p$ the performed squaring operation in the i-th iteration can be classified into two main classes depending on the two consecutive secret exponent bits d_i and d_{i-1} as follows:

- $d_i = d_{i-1}$: the performed squaring operation is either 1×1 or $(p-1)\times(p-1)$.
- $d_i \neq d_{i-1}$: the performed squaring operation is either $c\times c$ or $(p-c)\times(p-c)$.

To assess the practicability of exploiting the leakage that depends on the (secret) exponent in a realistic case, we decrypted the ciphertext c such that $c^2 = -1 \bmod p$ by using Algorithm 5 in which the first bits of the key are $(111010)_2$ and measured the corresponding power consumption. Figure 6 illustrates the captured trace.
Similar to the previous experiment presented in Section 4.1, a simple comparison is sufficient to distinguish between the multiplication and squaring operations. We illustrate the measured power consumption trace by a sequence of subtraces $T = \{(T_M, T_S)_{l-1}, (T_M, T_S)_{l-2}, ..., (T_M, T_S)_1, (T_M, T_S)_0\}$ in which T_M and T_S

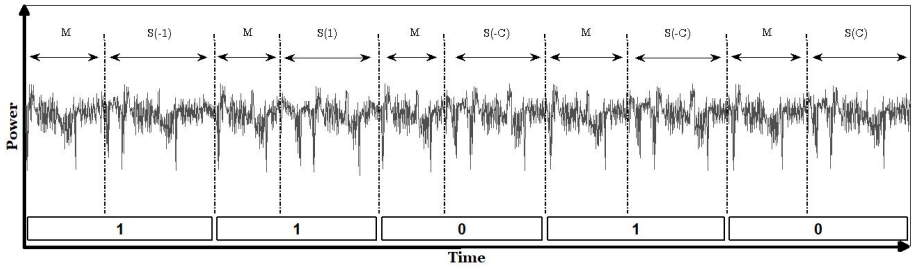

Fig. 6: Attack on ML algorithm by utilizing the ciphertext c such that $c^2 = -1 \bmod p$

correspond to multiply and squaring operations, respectively. Multiplication operation is marked by M and squaring operation is marked by S which are also separated by vertical dot lines in Figure 6. However, in order to perform the proposed attack in Section 3.3 on ML algorithm, one should also be able to distinguish between the four aforementioned squaring operations. Unlike SMA Algorithm, it is difficult to determine the actual squaring operation based on the simple observation of the corresponding subtrace. Instead, we can use the template technique to overcome this challenge. A template attack consists of two phases. In the first phase, the characterization takes place. In the second phase, we perform a template matching to determine the actual input values of the squaring operations in the decryption process.

In our experience, the squaring operations $(1 \times 1) \bmod p$, $(p-1 \times p-1) \bmod p$, $(c \times c) \bmod p$ and $(p-c \times p-c) \bmod p$ have different patterns. These patterns can be simply used in the matching phase to recognize the actual value used in the squaring operations. In Figure 6, we denote the squaring operations $(1 \times 1) \bmod p$, $(p-1 \times p-1) \bmod p$, $(c \times c) \bmod p$ and $(p-c \times p-c) \bmod p$ by $S(1)$, $S(-1)$, $S(c)$ and $S(-c)$, respectively. As it is observable from Figure 6 the squaring operations can reveal the secret exponent value. The squaring operations $(1 \times 1) \bmod p$ and $(p-1 \times p-1) \bmod p$ are executed when two consecutive secret exponent bits are the same. Similarly, the squaring operations $(c \times c) \bmod p$ and $(p-c \times p-c) \bmod p$ are executed when two consecutive secret exponent bits are different. Therefore, we can recover all of the bits of the key.

5 Conclusion

In this paper, we proposed a new chosen ciphertext attack on Elgamal implementations using Square and Multiply Always (SMA) and Montgomery Ladder (ML) algorithms. We exploited the leakage of power consumption during the decryption execution of a specific ciphertext c such that $c^2 = p-1 \bmod p$, where p is the prime module in Elgamal cryptosystem. We show that such a ciphertext exists for a special case of p where $p \mod 4 = 1$.

To verify the existence of the leakage that depends on the secret exponent, we provided a concrete experimentation of our attack. The experimental results confirm the practicability of the proposed attack on both SMA and ML algorithms by utilizing the power measurement of only one single decryption.
One natural direction for further research is to develop a similar attack for applying on the other modular exponentiation algorithms. As another direction, one can attempt to pursue reusing the presented idea for elliptic-curve cryptosystems.
In this paper, we take another step towards the generalization of the so-called $N-1$ attack by utilizing a specific ciphertext c such that $c^2 \mod p = -1$. It remains to be studied whether or not other types of ciphertext with property $c^\ell \mod p = -1$ with small values of ℓ can be utilized to apply new attacks against Elgamal cryptosystem in practice.
The experimental results presented in this paper show that the proposed attacks are applicable on small devices like an Atmel ATXMEGA128D4 8-bit microcontroller. Recently several attacks on public key cryptosystems have been applied on the personal computers and laptops[9, 1], . It remains to be studied if the proposed attacks are applicable to more complicated devices like laptops.

Bibliography

[1] Pierre Belgarric, Pierre-Alain Fouque, Gilles Macario-Rat, and Mehdi Tibouchi. Side-channel analysis of weierstrass and koblitz curve ECDSA on android smartphones. In *CT-RSA 2016*, pages 236–252. Springer, 2016.

[2] Christophe Clavier and Benoit Feix. Updated recommendations for blinded exponentiation vs. single trace analysis. In *COSADE 2013*, pages 80–98. Springer, 2013.

[3] Jean-Christophe Courrège, Benoit Feix, and Mylène Roussellet. Simple power analysis on exponentiation revisited. In *CARDIS 2010*, pages 65–79. Springer, 2010.

[4] Bert den Boer, Kerstin Lemke, and Guntram Wicke. A DPA attack against the modular reduction within a CRT implementation of RSA. In *CHES 2002*, pages 228–243. Springer, 2002.

[5] Zhaojing Ding, Wei Guo, Liangjian Su, Jizeng Wei, and Haihua Gu. Further research on N-1 attack against exponentiation algorithms. In *ACISP 2014*, pages 162–175. Springer, 2014.

[6] Pierre-Alain Fouque and Frédéric Valette. The doubling attack - *Why Upwards Is Better than Downwards*. In *CHES 2003*, pages 269–280. Springer, 2003.

[7] Daniel Genkin, Lev Pachmanov, Itamar Pipman, and Eran Tromer. Stealing keys from pcs using a radio: Cheap electromagnetic attacks on windowed exponentiation. In *CHES 2015*, pages 207–228. Springer, 2015.

[8] Daniel Genkin, Lev Pachmanov, Itamar Pipman, and Eran Tromer. ECDH key-extraction via low-bandwidth electromagnetic attacks on pcs. In *CT-RSA 2016*, pages 219–235. Springer, 2016.

[9] Daniel Genkin, Itamar Pipman, and Eran Tromer. Get your hands off my laptop: Physical side-channel key-extraction attacks on pcs. In *CHES 2014*, pages 242–260. Springer, 2014.

[10] Daniel Genkin, Adi Shamir, and Eran Tromer. RSA key extraction via low-bandwidth acoustic cryptanalysis. *IACR Cryptology ePrint Archive*, 2013:857, 2013.

[11] Naofumi Homma, Atsushi Miyamoto, Takafumi Aoki, Akashi Satoh, and Adi Shamir. Collision-based power analysis of modular exponentiation using chosen-message pairs. In *CHES 2008*, pages 15–29. Springer, 2008.

[12] Naofumi Homma, Atsushi Miyamoto, Takafumi Aoki, Akashi Satoh, and Adi Shamir. Comparative power analysis of modular exponentiation algorithms. *IEEE Trans. Computers*, 59(6):795–807, 2010.

[13] Paul C. Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *CRYPTO 1996*, pages 104–113. Springer, 1996.

[14] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *CRYPTO 1999*, pages 388–397. Springer, 1999.

[15] Thomas S. Messerges, Ezzy A. Dabbish, and Robert H. Sloan. Power analysis attacks of modular exponentiation in smartcards. In *CHES 1999*, pages 144–157. Springer, 1999.

[16] Atsushi Miyamoto, Naofumi Homma, Takafumi Aoki, and Akashi Satoh. Enhanced power analysis attack using chosen message against RSA hardware implementations. In *ISCAS 2008*, pages 3282–3285. Springer, 2008.

[17] Roman Novak. Spa-based adaptive chosen-ciphertext attack on RSA implementation. In *PKC 2002*, pages 252–262. Springer, 2002.

[18] Colin O'Flynn and Zhizhang (David) Chen. Chipwhisperer: An opensource platform for hardware embedded security research. In *COSADE 2015*, pages 243–260. Springer, 2015.

[19] Werner Schindler. A timing attack against RSA with the chinese remainder theorem. In *CHES 2000*, pages 109–124. Springer, 2000.

[20] Werner Schindler. Exclusive exponent blinding may not suffice to prevent timing attacks on RSA. In *CHES 2015*, pages 229–247. Springer, 2015.

[21] Colin D. Walter and Susan Thompson. Distinguishing exponent digits by observing modular subtractions. In *CT-RSA 2001*, pages 192–207. Springer, 2001.

[22] Sung-Ming Yen, Wei-Chih Lien, Sang-Jae Moon, and JaeCheol Ha. Power analysis by exploiting chosen message and internal collisions - vulnerability of checking mechanism for rsa-decryption. In *Mycrypt 2005*, pages 183–195. Springer, 2005.