IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received February 18, 2019, accepted March 27, 2019, date of publication April 3, 2019, date of current version April 19, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2909113

A Practical Collision-Based Power Analysis on RSA Prime Generation and Its Countermeasure

SANGYUB LEE[1,2], SUNG MIN CHO[1,2], HEESEOK KIM[3], AND SEOKHIE HONG[1,2]
[1]Graduate School of Information Security, Korea University, Seoul 02841, South Korea
[2]Institute of Cyber Security & Privacy, Korea University, Seoul 02841, South Korea
[3]Department of Cyber Security, College of Science and Technology, Korea University, Sejong 30019, South Korea

Corresponding author: Seokhie Hong (shhong@korea.ac.kr)

This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT under Grant NRF-2017R1C1B2004583.

ABSTRACT We analyze the security of RSA prime generation implemented on embedded devices by a practical power analysis attack. Unlike previous differential power analysis-based attack on primality tests of RSA prime generation exploiting the deterministic relationship among multiple prime candidates manipulated by consecutive primality tests, we propose a collision-based power analysis attack on the Miller–Rabin test for a single prime candidate which can recover the secret prime with a single attempt by exploiting collision characteristics of simple power analysis resistant modular exponentiation algorithms. Hence, our attack does not require the incremental prime search assumption and is applicable when countermeasures against previous attacks are deployed since it also does not require the assumption of trial divisions with small primes on prime candidates. For a realistic setting, where five 512-bit modular exponentiations are operated on an ARM Cortex-M4 microcontroller as recommended by FIPS 186-4 standard, we successfully recover the secret exponent to an extent that a feasible exhaustive search is needed for the full recovery of the secret prime. This is a first practical result of recovering a full secret of modular exponentiation which manipulates 512-bit RSA primitives with collision-based power analysis in a single attempt, where the previous attack demonstrates the result for 192-bit ECC primitive implementations. We also present a countermeasure against our attack which requires only one additional modular subtraction for the loop of square-and-multiply-always exponentiation algorithm. An experimental result for the effectiveness of our proposed countermeasure is presented.

INDEX TERMS Cryptography, digital signatures, public key, side-channel attacks.

I. INTRODUCTION

Security of RSA implementations against Side-Channel Analysis (SCA) have been well studied until now. Most of such researches focus on signature generation or decryption process [1]–[13]. Resistance against SCAs of such processes must be considered because they manipulate private key directly by modular exponentiation and immediate forgery can be caused if this information is exposed. Hence countermeasures securing the exponent on modular exponentiations have been proposed [14]–[17] and especially resistance against single trace attacks such as Simple Power Analysis (SPA) or Timing Attacks [1] is being a minimum requirement for secure implementation on embedded devices.

The associate editor coordinating the review of this manuscript and approving it for publication was Yinghui Zhang.

However advanced single trace attacks which can be applicable in the presence of SPA countermeasures have been proposed [3], [5]–[8]. Walter [3] proposed Big Mac attack which can distinguish squaring and multiplication in a modular exponentiation by Euclidean distance with a single trace. Inspired by the Walter's work, Clavier *et al.* [5] proposed Horizontal Correlation Analysis (HCA) which can exploit correlation between intermediate data and power consumption in a single trace. Recovery of Secret Exponent by Triangular Trace Analysis (ROSETTA) [6] distinguish squaring operations by exploiting inner-collisions of a long integer multiplication (LIM) caused by the same input single precision operations. Horizontal Collison Correlation Attack (HCCA) [7] exploits collisions originated by the same operand inputted in two LIMs. Hanley *et al.* [8] enhanced HCCA by detecting collisions between input and output operand of two LIMs. These attacks, except HCA, can be categorized as collision-based side-channel attacks.

Besides, on the practical perspective, only simulated results of the collision-based attacks are presented except the work of Hanley *et al.* [8] which demonstrates experimental results targeting 192-bit implementations of scalar multiplication on a 32-bit microcontroller and a FPGA. More recently, Danger *et al.* [10] extended HCCA by exploiting collisions of multiple multiplications in scalar multiplication targeting a 384-bit implementation on a 64-bit architecture. References [11], [12] exhibit attack experiments applying HCCA [7] and ROSETTA [6] on specific elliptic curves of 192-bit and 256-bit implementations, respectively. Luo *et al.* [13] demonstrate side-channel vulnerabilities of a state-of-the-art ECC library by exploiting collisions of power and electromagnetic traces acquired from an 8-bit and a 32-bit microcontrollers. Among these attacks only the work of Hanley *et al.* [8] demonstrates practical results targeting recovery of full secret, that is, a 192-bit secret scalar, with a single attempt. Although [10]–[13] present experimental results to validate the principles of their attack, practical effectiveness for the recovery of full secret with actual single trace are unknown.

Despite the notation, so-called *single trace attacks*, recovering a full secret of modular exponentiation or scalar multiplication with a single trace in real world accompanies several difficulties. Because of the large computational costs of scalar multiplication or modular exponentiation, acquired number of samples of side-channel information, for example power consumption, from a measuring device such as an oscilloscope is large where the memory of the measuring device is limited. This causes low sample Points Per Clock (PPC) of target device and consequently exploiting collision becomes difficult because determining points of interest (POIs) on which targeting collision leakage exists in unit operations of scalar multiplication or modular exponentiation is difficult.

On the other hand, it can be said that studies on security against SCAs of RSA key generation relatively less highlighted until recently, since this have been considered to be one-time process such as the device personalization before a device is released to a user. However, a need for key generations during the device life-time is rising in today's mobile and Internet of Things environments.

An RSA key generation accompanies generation of primes p and q to calculate an RSA modulus $n = p \cdot q$ and standard such as NIST FIPS PUB 186-4 [18] presents recommendations of provable and probable prime generation processes. Compared to provable prime generation, the latter one is preferred for its efficiency in time and memory usage [19]. Probable prime generation is composed of generation of random prime candidate and primality test, that is, the Miller-Rabin test, for the candidate and these are repeated until the candidate is determined to be a prime. However, both processes are expensive to operate iteratively on embedded devices. Hence, to reduce the cost of iterative access to the Random Number Generator (RBG), Brandt and Damgård proposed the incremental prime search technique which is composed of generating one odd random number from RBG as a seed for prime candidates and incrementing the seed by a fixed constant value repeatedly to acquire other prime candidates [20]. On the other hand, for the efficiency of primality test process, trial division with small primes or prime sieve methodology proposed [19], [21] to minimize the number of expensive modular exponentiation in the Miller-Rabin test for composite candidates.

If at least one of primes p and q, say it is p here, is exposed, an adversary can calculate the secret private key $d_{priv} = e^{-1} \bmod ((p-1)(q-1))$ because e is a public key and q can be calculated by factorizing $n = p \cdot q$ where n is also a public key. Hence security against SCAs on implementations of prime generation must be considered and relevant literature have been published. Finke *et al.* [22] and Bauer *et al.* [23] focus on trial division or prime sieve for prime candidates assuming basic SPA and Timing Attack countermeasures on primality tests are applied. Vuillaume *et al.* [24] proposed attacks on the primality tests and pointed out such countermeasures are insufficient but they assumed an incremental prime search of [19], [20].

Our contribution is threefold. First, we propose a practical collision-based power analysis on the Miller-Rabin test of RSA prime generation which can recover a secret prime with a single attempt by exploiting collision characteristics of SPA-resistant modular exponentiation algorithms. Unlike previous work [24] attacking the primality test by a Differential Power Analysis (DPA) depending on the deterministic relationship among prime candidates manipulated by the consecutive primality tests, that is, multiple Fermat tests or multiple Miller-Rabin tests with different prime candidate inputs, our attack targets a single prime candidate manipulated by a single Miller-Rabin test. Hence our attack does not require the incremental prime search assumption and neither additional template attacks nor fault attacks to compensate the small number of recovered bits caused by the limitation of the DPA as shown in [24]. Also, our attack is applicable when countermeasures against previous attacks of [22]–[24] are deployed since it does not require assuming trial divisions with small primes on prime candidates for the Miller-Rabin test. Secondly, we demonstrate a first practical result of recovering a full secret of 512-bit modular exponentiation operating 512-bit long integer multiplications while previous work [8] recovered a 192-bit secret with a single attempt. Comparing with other works [9]–[13], where practical results of attacking at maximum 384-bit ECC primitives are presented, attacking 512-bit RSA primitives has more difficulties such as low PPC problems and issues on extracting POIs since it requires more power consumption samples to acquire from a measuring device with limited resources. We present detailed methodology addressing such difficulties for a realistic setting where only five 512-bit modular exponentiations for the Miller-Rabin test on a single prime candidate are operated on an ARM Cortex-M4 based STM32F405 microcontroller [25] as recommended by FIPS 186-4 standard.

As result, we recovered a 512-bit prime with only three-bit failure and for the remained secret, a feasible exhaustive search is needed while previous attacks [22]–[24] require the lattice-reduction technique [26] since it is unaffordable by the exhaustive search to recover the remained bits. Finally, we propose a new countermeasure for the square-and-multiply-always exponentiation algorithm against our attack which requires only one additional modular subtraction for each loop. An experimental result is also presented to show the effectiveness of our countermeasure.

This paper is organized as follows. In Section 2, we briefly introduce about RSA prime generation and the Miller-Rabin test and previous attacks. Next, we analyze and perform experiments exploiting collisions on exponentiations in the Miller-Rabin test. In Section 4, we propose a simple countermeasure against our attack and present discussions. We conclude this paper in Section 5.

II. PRELIMINARIES

A. RSA PRIME GENERATION AND MILLER-RABIN PRIMALITY TEST

We briefly describe here about RSA prime generation on the basis of NIST FIPS 186-4 [18] standard. An RSA key consists of public key (n, e) and private key (n, d_{priv}), where n is a modulus which is a product of two primes p and q, that is $n = p \cdot q$, e is a public key exponent and d_{priv} is a private key exponent which satisfy $d_{priv} = e^{-1} \bmod \phi(n)$, where $\phi(\cdot)$ is Euler's totient function. Therefore, to establish an RSA key pair, generation of two primes p and q with the same length must be preceded. Recommendation of the standard consists of provable and probable prime generations corresponding to Appendices B.3.2 and B3.4 and Appendices B.3.3, B.3.5, and B.3.6, respectively. In this paper we focus only on the security of probable prime generations basically composed of generation of prime candidates and probable primality tests.

Since the cost of access to random number generator on embedded devices is high, a method of generating a random number and incrementing it by deterministic way [19], [20] for prime candidates is proposed. On the other hand, primality tests on the prime candidates are also expensive, hence trial divisions with small prime factors [19] are done for each prime candidate to reduce the number of primality tests on composite candidates. Then, for a prime candidate passed trial division process, a primality test, that is, the Miller-Rabin primality test is performed.

The Miller-Rabin test determines whether an input odd integer w is prime. During its operation, as described in Alg. 1 [18], d and r is computed such that $w - 1 = 2^r \cdot d$ then modular exponentiation $a^d \bmod w$ is performed with different random bases a, k times at maximum in the case w is a probable prime. If this exponentiation is naively implemented an adversary can recover the exponent d by performing SPA [1]. After d is recovered, the adversary can guess r exhaustively since with high probability r is small value and then calculate $x = 2^{\tilde{r}} \cdot \tilde{d} + 1$ where $\tilde{r}$ and $\tilde{d}$ denote candidate values of the correct r and d respectively. By calculating $GCD(x, n)$ for every guess of r and finding the case that results in $GCD(x, n) = x$, the adversary can recover the one secret prime p (or q) as for the case $x = p$ (or q) then recover the another secret prime q (or p) by factorizing $n = p \cdot q$ where n is known because it is public key of RSA cryptosystem. Hence security against SPA [1] revealing the exponent, which can be led to exposure of secret primes during the modular exponentiation should be considered basically when RSA prime generation is implemented on embedded devices.

Algorithm 1 Miller-Rabin Probabilistic Primality Test

Input: odd integer to be tested for primality w, number of iterations of the test to be performed k
Output: PROBABLY PRIME or COMPOSITE

```
 1: Let $r$ be the largest integer such that $2^r$ divides $w - 1$
 2: $d = (w - 1)/2^r$
 3: for $i = 1$ to $k$ do
 4:     Obtain a random integer $a$ of length(w) from an RBG, such that $2 \le a \le w - 2$
 5:     $z = a^d \bmod w$
 6:     if (($z = 1$) or ($z = w - 1$)) then go to Step 13
 7:     for $j = 1$ to $r - 1$ do
 8:         $z = z^2 \bmod w$
 9:         if ($z = w - 1$) then go to Step 13
10:         if ($z = 1$) then go to Step 12
11:     end for
12:     Return COMPOSITE
13:     continue
14: end for
15: Return PROBABLY PRIME
```

B. SIDE-CHANNEL ATTACKS ON RSA PRIME GENERATION

Besides, SCAs on RSA prime generation have been studied until recently. Finke *et al.* [22] proposed an attack on trial division process exploiting the facts that the process is terminated immediately when the candidate is divided by a small prime factor and prime candidates are incremented by a constant value. By observing prime factors which terminate the process by SPA and formulating relations between prime factors and prime candidates they can recover partial information of a secret prime. Bauer *et al.* [23] attacked on a regular trial division process which can thwart the attack of [22] but their attack still depends on the relations between incrementing prime candidates and small prime factors of trial division process. Unlike [22], [23], Vuillaume *et al.* target primality tests [24] but their attack still requires the incremental prime candidate search assumption to exploit their formula to reveal the secret prime. Therefore, all previous attacks can be thwarted by making the generation of prime candidates undeterministic and for all cases recovered information is limited since side-channel information is restricted by nature of prime generation process (to loosen this restriction, Vuillaume *et al.* [24] employ a fault attack [9] but we do not focus on fault attacks in this paper), thus they should associate the lattice-reduction technique by Coppersmith [26] to compensate the unrecovered secrets. Our proposed attack only targets the Miller-Rabin primality test itself not assuming the incremental prime candidate search or the trial divisions and recovers the secret information to the level of requiring a feasible simple exhaustive search for the remaining secret. Thus previous attacks can be incorporated with our attack when an RSA prime generation is naively implemented and even when countermeasures applied to thwart previous ones, our attack is still applicable since the Miller-Rabin primality test on prime candidate is mandatory for any probable prime generations.

C. SPA-RESISTANT EXPONENTIATION ALGORITHMS AND SINGLE TRACE ATTACKS

We focus on the security of the Miller-Rabin test hereafter. As noted in Subsection A, for secure implementation of the Miller-Rabin test, SPA-resistant exponentiation algorithms are employed such as square-and-multiply-always [14], Montgomery ladder [15], [16] and side-channel atomic exponentiation [17]. The former two algorithms operate exponentiation by executing two multiplications per one loop whereas the rest iterates one multiplication regularly. Therefore, for all three cases where such exponentiation algorithm is applied respectively, an adversary cannot distinguish between squaring and multiplication by observing the power consumption of each operation.

In this situation, the adversary can consider more advanced single trace attacks such as Big Mac attack [3], Horizontal Correlation Analysis [5], Horizontal Collison Correlation Attack [7], Recovery of Secret Exponent by Triangular Trace Analysis [6] or *Improved* Horizontal Collision Correlation Attack (IHCCA) [8]. Modular exponentiations operated in the Miller-Rabin test are inherently immune to HCA [5] because the adversary cannot compute the intermediate values of exponentiations where their bases are random for each operation. Eventually, so-called collision-based attacks of [3], [6]–[8] can be considered. However, their experimental results are introduced by simulations [6], [7] or for Elliptic Curve Cryptography (ECC) implementations [8], where bit length to be recovered is smaller compared to modular exponentiations of the Miller-Rabin test, since these attacks require precise extraction of POIs where power consumption leaks information for determining a collision. For such extraction of POIs, adequate PPC which can be achieved by low clock frequency of target device given limited maximum sampling rate by a measuring device, that is, an oscilloscope, is preferable. Unlike attacking ECC case [8], modular exponentiation algorithm takes much more time, moreover it manipulates longer bit length, that is, 512-bit, 1024-bit, or 1536-bit for the Miller-Rabin test, such that requires operating in higher clock frequencies. Besides, minimum number of rounds for the Miller-Rabin test recommended by FIPS 186-4 [18] also hinders the adversary because exploitable power consumption traces to compensate low PPC problem by utilizing multiple traces, for example, reducing the noise by averaging traces, are limited. In the next section, we describe detailed methodology of addressing these problems and extract relevant POIs then recover the secret prime manipulated in the Miller-Rabin test.

III. PRACTICAL COLLISION-BASED POWER ANALYSIS ON THE MILLER-RABIN TEST

In this section, we present a collision power analysis on modular exponentiations in the Miller-Rabin test. As described in Section II.A, the exponent d should not be disclosed to an adversary. As bases are random and an adversary cannot choose them, DPA-like attack [2], [5] is not possible. Thus, for the Miller-Rabin test SPA-resistant modular exponentiation is required since SPA can reveal the exponent from a single power trace.

In the following, we analyze vulnerabilities of SPA-resistant exponentiation algorithms which can cause collisions in power consumption traces. Next, we illustrate details of our attack with practical experiments. As described in the previous section, we have low PPC problem and limited number of power consumption traces of modular exponentiations. We present a sequence of techniques to extract points of interest defeating the low PPC problem and perform collision power analysis on modular exponentiations in the Miller-Rabin test for a single prime candidate. Moreover, we propose a further analysis technique utilizing the secret exponent candidates recovered from the latter attack and enhancing our attack result, then finally, recover 512-bit secret prime with only three-bit failure.

A. COLLISION CHARACTERISTICS OF SPA-RESISTANT EXPONENTIATION ALGORITHMS

A single power trace of modular exponentiation consists of multiple subtraces corresponding to field multiplications. For example, in Alg. 2, two field multiplications are executed each loop, n times. Assuming an adversary can detect when two field multiplications have the same input in (at least) one operand, namely when collision occurs, it can recover the secret exponent bit by bit exploiting following collision characteristics. Note that the adversary does not need to know the input value of field multiplications.

Square-and-multiply-always exponentiation can be described as Alg. 2. Suppose the adversary is to recover a bit of secret exponent d, for example, when $i = s$ and R_0 has an intermediate value of x_s, two inputs of field multiplication in Step 3 in Alg. 2 can be represented as $[x_s, x_s]$. Then we can write down the next input sequence of field multiplications as:

- Step 5 (or 7): $[x_s^2, x]$
- Step 3 if $(d_s = 1)$: $[x_s^2 \cdot x, x_s^2 \cdot x]$
- Step 3 if $(d_s = 0)$: $[x_s^2, x_s^2]$

As shown above, only in the case of $d_s = 0$, the first operand inputs of field multiplications of Step 7 and Step 3 of the next loop have collision.

Algorithm 2 Square-and-Multiply-Always

```
Input: $x$, $d = (d_{n-1}, ..., d_0)_2$
Output: $y = x^d$
1: $R_0 \leftarrow 1; R_1 \leftarrow x; R_2 \leftarrow 1$
2: for $i = n-1$ down to 0 do
3:     $R_0 \leftarrow R_0R_0$
4:     if $(d_i = 0)$ then
5:         $R_2 \leftarrow R_0R_1$
6:     else [if $(d_i = 1)$]
7:         $R_0 \leftarrow R_0R_1$
8: end for
9: Return $R_0$
```

We can find the collision characteristic of Montgomery Ladder of [15] described in Alg. 3. Depending on the secret bit value, field multiplications of Step 4–5 or Step 7–8 are operated. In the former case, the first operand of multiplications has collision unconditionally for R_0 is updated after processing of Step 5 is completed. On the other hand, if $d_i = 1$, only the second operand has collision in the similar way. Hence, after investigating the position of collision of two subtraces of field multiplications, the adversary can detect whether Step 4–5 or Step 7–8 have operated and also deduce the secret exponent bit (Hanley *et al.* proposed a simple countermeasure to avoid this kind of attack in [8] but we describe this for the completeness).

Algorithm 3 Montgomery Ladder

```
Input: $x$, $d = (d_{n-1}, ..., d_0)_2$
Output: $y = x^d$
 1: $R_0 \leftarrow 1; R_1 \leftarrow x$
 2: for $i = n-1$ down to 0 do
 3:     if $(d_i = 0)$ then
 4:         $R_1 \leftarrow R_0R_1$
 5:         $R_0 \leftarrow R_0R_0$
 6:     else [if $(d_i = 1)$]
 7:         $R_0 \leftarrow R_0R_1$
 8:         $R_1 \leftarrow R_1R_1$
 9: end for
10: Return $R_0$
```

Side-channel atomic exponentiation is similar to simple square-and-multiply exponentiation [27] except it has no conditional branches, as shown in Alg. 4, for the sake of SPA-resistance. Unlike the former two algorithms, it does not have regular sequence of two field multiplications, that is, squaring and (dummy) multiplication. Instead, collision characteristic is simple. If and only if when $d_i = 1$, field multiplication of R_0R_1 occurs and the input value of R_1 is always x. Therefore, if the adversary can acquire the subtrace of field multiplication R_0R_1, it can apply a collision power analysis on side-channel atomic exponentiation.

Next, we present some detailed methods for exploiting these vulnerabilities in a practical experiment. As described in Section II.C, determining collision of power consumption trace requires precise extraction of POIs. In attacking exponentiations of the Miller-Rabin test context, this is challenging process because we have no information to utilize for extracting POIs but power consumption traces. Moreover, we suffer from low PPC problem comparing with attacking ECC implementations. Our strategy is, first, to extract POIs with noise. For this, we reconstructed power consumption traces and removed operational dependent leakages by subtracting mean traces to minimize the noise. Then we recovered partial information of the secret exponent exploiting collision characteristics we described. Finally, we combine partial information on the same secret and utilize the result to strengthen the POI extraction. In the following, we describe the detail of our experiment.

Algorithm 4 Side-Channel Atomic Exponentiation

```
Input: $x$, $d = (d_{n-1}, ..., d_0)_2$
Output: $y = x^d$
1: $R_0 \leftarrow 1 ; R_1 \leftarrow x ; i \leftarrow n-1$
2: $k \leftarrow 0$
3: while $(i \geq 0)$ do
4:     $R_1 \leftarrow R_0R_k$
5:     $k \leftarrow k \oplus d_i ; i \leftarrow i - \neg k$
6: end while
7: Return $R_0$
```

B. EXPERIMENTAL RESULT

1) EXPERIMENTAL SETUP

We conduct a practical experiment of our attack on a square-and-multiply-always based Miller-Rabin test for prime generation of RSA-1024, that is, testing a 512-bit prime, which is implemented on ARM Cortex-M4 based STM32F405 microcontroller [25] which is embedded on ChipWhisperer [29] CW308T-STM32F [30] target board. Alg. 2 is implemented in C and ARM assembly language and operated at 100MHz clock frequency. The power consumption signal is amplified ten times by CW501 differential probe [31] and then low-pass filtered with 150MHz cutoff frequency and captured at a 500MS/s sampling rate by LeCroy HDO6104A oscilloscope [32] (We tried to perform our attack using ChipWhisperer's capture device. However, we failed to capture the whole exponentiation power trace even though we used the ‘‘Streaming Mode’’ which is its best feature for acquiring power traces of a long-time operation. Moreover, our attack might not be successful even if we could collect the whole trace with the capture device because its maximum sampling rate is 10MS/s while our attacks result is based on 500MS/s).

2) RECONSTRUCTING POWER CONSUMPTION TRACES

We collect five power consumption traces of Alg. 2 with the same exponent d and different random bases x. Note that the minimum number of rounds of the Miller-Rabin test for 512-bit primes recommended by FIPS 186-4 is five [18]. Field multiplications of Step 3, Step 5, and

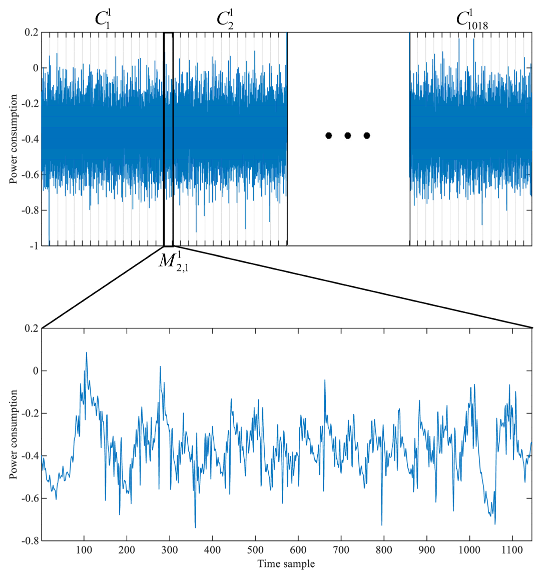


FIGURE 1. **Example of reconstructed power consumption trace.**

Step 7 in Alg. 2 are implemented as a sequence of school-book multiple-precision multiplication and modular reduction [27]. To exploit the collision characteristic of Alg. 2, as analyzed in Section III.A, we only focus on the multiple-precision multiplication part. In our case, it is implemented with two types of functions whose power consumption is different from each other, that is, a multiple-precision multiplication is composed of one (16 words)×(1 word) multiplication and fifteen (16 words)×(1 word) multiply-and-accumulate functions. Since we target to determine manipulation of the same input in the first operand of two multiple-precision multiplications, we reconstruct power consumption traces C_i^k with only multiply-and-accumulate functions where i indicates the index of multiple-precision multiplications in an exponentiation and k indicates the number of round of the Miller-Rabin test. First, we find a power consumption trace of a single multiply-and-accumulate as a reference and calculate correlation coefficient point by point with the whole power consumption traces of five exponentiations. Then from the point where the peak of correlation coefficient exists, we cut the power consumption samples with the same length of the reference. After repeating this, we can find all multiply-and-accumulate functions and obtain $C_i^k = [M_{i,1}^k \parallel M_{i,2}^k \parallel \cdots \parallel M_{i,15}^k]$ where $M_{i,j}^k$ represents the power consumption of a multiply-and-accumulate function. Fig. 1 shows an example of reconstructed power consumption trace.

3) POST-PROCESSING POWER CONSUMPTION TRACES: REMOVING OPERATION-DEPENDENT LEAKAGE

Before further analysis, we process power consumption traces to have mostly data-dependent leakage [28] since we have to determine manipulated data of two multi-precision multiplications, more precisely, for recovering the j-th bit of the exponent, $C_{2\times j}^k$ and $C_{2\times j+1}^k$, is same or not. To this end, we compute mean traces of C_i^k for two separate cases where the index i is even or odd, then subtract them from each C_i^k accordingly ($2 \le i \le 2(n_d-1)+1$ and n_d denotes the number of bits of the secret exponent d). Alg. 5 illustrates this process. Note that, unlike Alg. 2, the implementation assumes the most significant bit of d is always one for practical reason, so R_0 in Step 1 starts with x and i in Step 2 starts with $(n-2)$. Therefore we can recover from $(n-1)$-th to the second bit, that is, (n_d-2) bits in total, of the secret exponent.

Algorithm 5 Post-Processing Traces

```
Input: $C_i^k$
Output: post-processed $C_i^k$
1: for $k = 1$ to 5 do
2:     $\overline{C}_{even}^k = (C_2^k + C_4^k + \cdots + C_{2\times n_d}^k)/(n_d-2)$
3:     $\overline{C}_{odd}^k = (C_3^k + C_5^k + \cdots + C_{2\times n_d+1}^k)/(n_d-2)$
4:     for $j = 1$ to $(n_d-2)$ do
5:         $C_{2\times j}^k = C_{2\times j}^k - \overline{C}_{even}^k$
6:         $C_{2\times j+1}^k = C_{2\times j+1}^k - \overline{C}_{odd}^k$
7:     end for
8: end for
9: Return $C_i^k$
```

4) FINDING POINTS OF INTEREST: PHASE ONE

To determine points of interest which are to be used for our attack, we calculate a correlation coefficient vector *CI* using Alg. 6. Since we have no information about the secret exponent, we utilize all $C_{2\times j}^k$ and $C_{2\times j+1}^k$ for all k. By the collision characteristic of Alg. 2, some $C_{2\times j}^k$ and $C_{2\times j+1}^k$ have collision for the cases where the j-th bit of the exponent is zero. For the opposite cases, using $C_{2\times j}^k$ and $C_{2\times j+1}^k$ for finding the points having collision characteristic has no effect, moreover it results in adding noise. To minimize this noise and maximize the effect of collision in given situation we reconstruct C_i^k by stacking the power consumption traces of $M_{i,j}^k$ vertically because utilizing as many number of traces as possible helps reduce the noise. Fig. 2 presents the result of Alg. 6 (l_M is the number of samples of a single $M_{i,j}^k$ trace).

5) RECOVERING SECRET EXPONENT CANDIDATES WITH COLLISION POWER ANALYSIS

As shown in Fig. 2, there are correlation coefficient peaks corresponding to the collision characteristic for each of $M_{i,j}^k$ despite the noise described in the previous section. By selecting indices from *CI* as points of interest on which correlation coefficient value is larger or equal than some *threshold* and extracting samples from each $M_{i,j}^k$ on the basis of the points of interest, we can reconstruct C_i^k as $\tilde{C}_i^k = [\tilde{M}_{i,1}^k \parallel \tilde{M}_{i,2}^k \parallel \cdots \parallel \tilde{M}_{i,15}^k]$ where $\tilde{M}_{i,j}^k$ represents the group of samples extracted from $M_{i,j}^k$. Now we can perform a collision power analysis by calculating correlation coefficient δ_j between $\tilde{C}_{2\times j}$ and $\tilde{C}_{2\times j+1}$ for recovering j-th bit of the secret exponent and

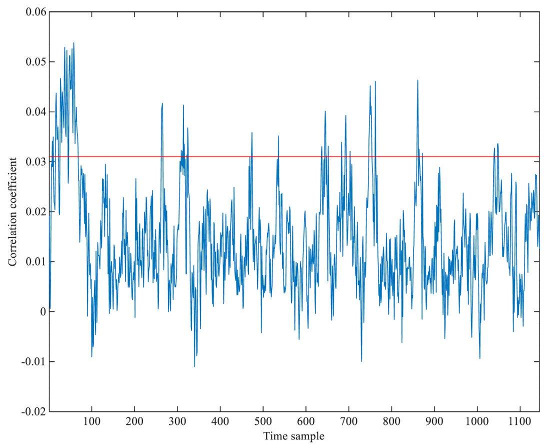


FIGURE 2. Correlation coefficient trace CI. Straight line represents the threshold of 0.031 used for choosing points of interest.

Algorithm 6 Calculating Correlation Coefficient Trace to Find POIs

```
Input: $C_i^k = [M_{i,1}^k \parallel M_{i,2}^k \parallel \cdots \parallel M_{i,15}^k]$
Output: $(1 \times l_M)$ correlation coefficient vector $CI$
 1: for $k = 1$ to 5 do
 2:     for $j = 1$ to $(n_d - 2)$ do
 3:         for $m = 1$ to 15 do
 4:             $M_{even} = [M_{even}; M_{2\times j,m}^k]$
 5:             $M_{odd} = [M_{odd}; M_{2\times j+1,m}^k]$
 6:         end for
 7:     end for
 8: end for
 9: $CI = CorrT(M_{even}, M_{odd})$
10: Return $CI$
```

determining whether it is zero (or one) by comparing each δ_j with the mean value of all δ_j. If δ_j is larger than the mean, j-th bit is considered to be zero, if not, this bit is considered to be one. This process is described in Alg. 8 (we can perform this for all $k = \{1, ..., 5\}$ independently using the same *CI*, so we omit k in the algorithm description). Fig. 3 is an example of Δ for the case of $k = 1$. And Table 1 represents the result of performing our attacks for all five number of rounds of the Miller-Rabin test and combining them (this is described in the next section).

6) FURTHER ANALYSIS AND RECOVERING SECRET PRIME

We perform further analysis exploiting the result of the previous section. As we have five different candidates derived for one secret exponent, we can combine them by applying majority rule bit by bit. Using this combined exponent, denoted by $\tilde{d}_{total}$ in Alg. 9, making correlation coefficient trace introduced in Subsection 4) can be more sophisticated by calculating *CI*s separately for only (probably) collision occurring case and the opposite case. This results in reducing the noise described in Subsection 4), therefore we can acquire *CI*s having an order of magnitude larger and more clear peaks as shown in Fig. 4(a). Then we calculate a differential vector of two *CI*s, denoted CI_D, to maximize the collision characteristic. Replacing *CI* with CI_D in Alg. 8 and performing the attack again, we recover one more bit for combined result with 0.1007 threshold value for extracting points of interest. Table 2 represents this result.

Now we estimate the computational complexity for full recovery of the secret exponent. Cost of exhaustive search for

TABLE 1. Success Rate of Each Attack.

Iteration(k)	Number of Bits Revealed	Success Rate of Revealing $(n_d - 2) = 508$ Bits
1	473	93.11%
2	439	86.42%
3	453	89.17%
4	487	95.87%
5	474	93.31%
Combined	504	99.21%

Algorithm 7 *CorrT* Function

```
Input: Two matrices $A = (a_{i,j})$ and $B = (b_{i,j})$ of the same size $(1 \le i \le M, 1 \le j \le N)$
Output: $(\rho_1, \rho_2, \ldots, \rho_N)$
1: for $i = 1$ to $N$ do
2:     $X = [a_{1,i}, a_{2,i}, \ldots, a_{M,i}]^T$
3:     $Y = [b_{1,i}, b_{2,i}, \ldots, b_{M,i}]^T$
4:     $\rho_i = corr(X, Y)$
5: end for
6: Return $(\rho_1, \rho_2, \ldots, \rho_N)$
```

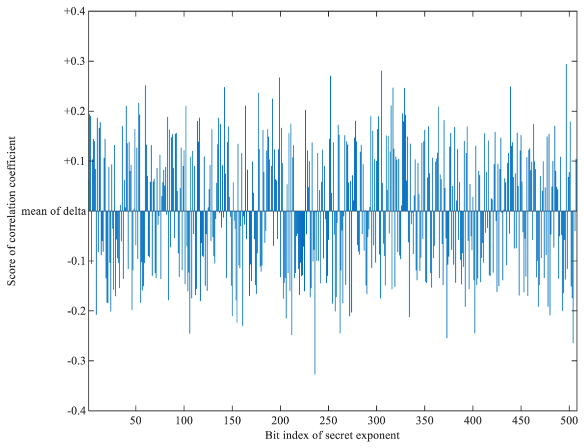


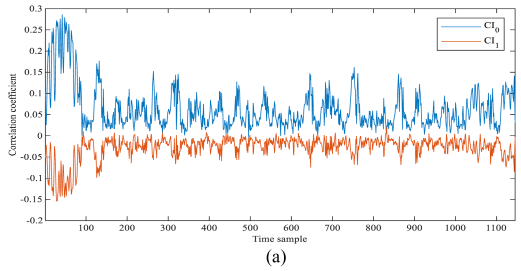
FIGURE 3. Correlation coefficient score for k=1. Bits having values above mean(Δ) =0.0331 considered to be zero bits otherwise one bits.

TABLE 2. Success Rate of Each Attack Using CI_D.

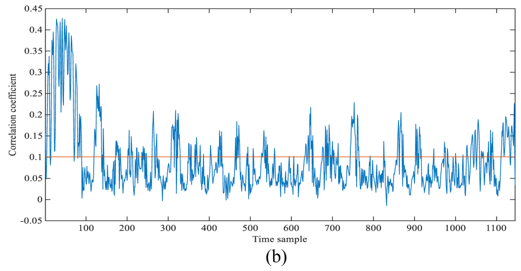
Iteration(k)	Number of Bits Revealed	Success Rate of Revealing $(n_d - 2) = 508$ Bits
1	469	92.32%
2	444	87.40%
3	452	88.98%
4	485	95.47%
5	468	92.13%
Combined	505	99.41%

Algorithm 8 Recovering the Secret Exponent With Collision Power Analysis

```
Input: $C_i = [M_{i,1} \| M_{i,2} \| \cdots \| M_{i,15}]$, $CI$
Output: Partial secret exponent $\tilde{d} = (d_{n_d-2}, \ldots, d_1)_2$ (where full secret exponent is $d = (d_{n_d-1}, \ldots, d_0)_2$)
1: Prepare a score vector $\Delta = [\delta_{n_d-2}, \ldots, \delta_1]$
2: // Select POIs having values larger or equal than arbitrary threshold then store indices in a vector $I_P = [i_1, \ldots, i_{nP}]$
3: for $i = 1$ to $l_M$ do
4:   if $(CI(i) \geq threshold)$ then
5:     $I_P = [I_p, i]$
6: end for
7: // Reconstruct $\tilde{C}_i$ with only POIs extracted $\tilde{M}_{i,j} = M_{i,j}(I_P)$
8: for $m = 1$ to $(n_d - 2)$ do
9:   $\tilde{C}_{2\times m} = [\tilde{M}_{2\times m,1} \| \tilde{M}_{2\times m,2} \| \cdots \| \tilde{M}_{2\times m,15}]$
10:  $\tilde{C}_{2\times m+1} = [\tilde{M}_{2\times m+1,1} \| \tilde{M}_{2\times m+1,2} \| \cdots \| \tilde{M}_{2\times m+1,15}]$
11:  $\delta_m = corr(\tilde{C}_{2\times m}, \tilde{C}_{2\times m+1})$
12: end for
13: for $m = 1$ to $(n_d - 2)$ do
14:   if $(\delta_m > mean(\Delta))$ then
15:     $d_m = 0$
16:   else
17:     $d_m = 1$
18: end for
19: Return $\tilde{d} = (d_{n_d-2}, \ldots, d_1)_2$
```



(a)



(b)

FIGURE 4. Correlation coefficient traces (a) calculated separately corresponding to bit value of secret exponent candidate and (b) their differential trace. Straight line represents the threshold of 0.1007 used for choosing points of interest.

the remaining three bits is ${}_{508}C_3 \times 2^3 < 2^{29}$. In addition, we need an exhaustive search for LSB assuming MSB is one, finally we have 2^{30} computational complexity for full recovery of the secret exponent. This is feasible even for off-the-shelf personal computers.

Algorithm 9 Calculating Correlation Coefficient Differential Trace Exploiting Recovered Secret Candidates to Find POIs

```
Input: $C_i^k = [M_{i,1}^k \| M_{i,2}^k \| \cdots \| M_{i,15}^k]$, $\tilde{d}^k$
Output: $(1 \times l_M)$ correlation coefficient differential vector $CI_D$
1: // Combine five results bit by bit using majority rule
2: for $j = 1$ to $(n_d - 2)$ do
3:   if $(\sum_{k=1}^{5} \tilde{d}^k(j) > 2.5)$ then
4:     $\tilde{d}_{total}(j) = 1$
5:   else
6:     $\tilde{d}_{total}(j) = 0$
7: end for
8: // Make correlation coefficient vectors separately according to the bit value
9: for $k = 1$ to 5 do
10:   for $j = 1$ to $(n_d - 2)$ do
11:     if $(\tilde{d}_{total}(j) = 0)$ then
12:       for $m = 1$ to 15 do
13:         $M_{even}^0 = [M_{even}^0; M_{2\times j,m}^k]$
14:         $M_{odd}^0 = [M_{odd}^0; M_{2\times j+1,m}^k]$
15:       end for
16:     else
17:       for $m = 1$ to 15 do
18:         $M_{even}^1 = [M_{even}^1; M_{2\times j,m}^k]$
19:         $M_{odd}^1 = [M_{odd}^1; M_{2\times j+1,m}^k]$
20:       end for
21:   end for
22: end for
23: // Make correlation coefficient differential trace
24: $CI_0 = CorrT(M_{even}^0, M_{odd}^0)$
25: $CI_1 = CorrT(M_{even}^1, M_{odd}^1)$
26: $CI_D = CI_0 - CI_1$
27: Return $CI_D$
```

IV. COUNTERMEASURES

In this section, we propose a possible countermeasure against our attack and discuss about other countermeasures. To reduce the effect of collision characteristic of Alg. 2, we take negative value of R_o at Step 8 of Alg. 10 so that for the next field multiplication of Step 3 squaring with negative value of R_o is operated. Making negative value can be implemented as modular subtraction. As a result, inputs of the first operand at Step 7 and Step 3 have no collision.

We present practical results of our countermeasure by conducting an experiment for an implementation of Alg. 10. All other experimental settings are identical to ones of Section III. Fig. 5 represents correlation coefficient trace resulted from Alg. 6. Unlike the result of Subsection 4) of Section III.B, shown as Fig. 2, correlation coefficient peaks are negative. By the way, negative correlation peaks also can be exploitable for extracting points of interest. Hence we perform the attack

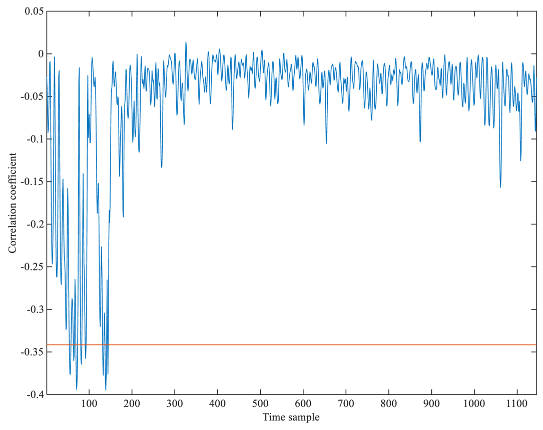


FIGURE 5. Correlation coefficient trace CI. Straight line represents the threshold of -0.3416 used for choosing points of interest.

TABLE 3. Success Rate of Each Attack on Implementation with Countermeasure using CI.

Iteration(k)	Number of Bits Revealed	Success Rate of Revealing $(n_d - 2) = 508$ Bits
1	286	56.30%
2	316	62.20%
3	295	58.07%
4	292	57.48%
5	293	57.68%
Combined	327	64.37%

of Alg. 8 to confirm the efficacy of our countermeasure (The sign of inequality at Step 4 become negative since peaks of *CI* is negative). As shown in Table 3 the success rate of the attack is decreased drastically even for the combined result.

Algorithm 10 Square-and-Multiply-Always With Countermeasure Against Our Attack

```
Input: $x$, $d = (d_{n-1}, \ldots, d_0)_2$
Output: $y = x^d$
 1: $R_0 \leftarrow 1$; $R_1 \leftarrow x$; $R_2 \leftarrow 1$
 2: for $i = n - 1$ down to 0 do
 3:     $R_0 \leftarrow R_0 R_0$
 4:     if $(d_i = 0)$ then
 5:         $R_2 \leftarrow R_0 R_1$
 6:     else [if $(d_i = 1)$]
 7:         $R_0 \leftarrow R_0 R_1$
 8:     $R_0 \leftarrow -R_0$        //$(p - R_0)$ assuming modulus is $p$
 9: end for
10: Return $R_0$
```

We conducted another experiment similar to Subsection 6) of Section III.B. For this time, we use actual secret exponent d as input of Alg. 9 to evaluate the optimum result for an adversary. As shown in Fig. 6, both peaks of *CI*s are negative for collision occurring and non-collision cases and their differential vector has almost an order of magnitude smaller peaks. Table 4 presents the result of the attack, although CI_D of Fig. 6(b) seems like useful for extracting points of

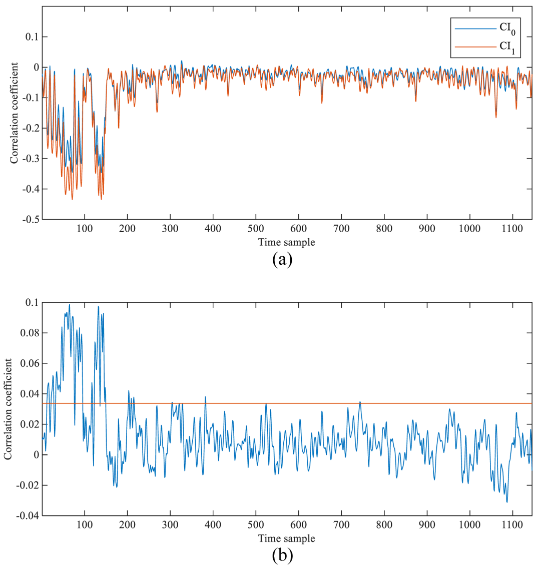


FIGURE 6. Correlation coefficient traces for implementation with countermeasure (a) calculated separately corresponding to bit value of secret exponent and (b) their differential trace. Straight line represents the threshold of 0.0338 used for choosing points of interest.

TABLE 4. Success Rate of Each Attack on Implementation with Countermeasure using CI_D.

Iteration(k)	Number of Bits Revealed	Success Rate of Revealing $(n_d - 2) = 508$ Bits
1	289	56.89%
2	323	63.58%
3	300	59.06%
4	298	58.66%
5	305	60.04%
Combined	341	67.13%

interest, the best success rate of the attack is still sufficiently low compared to the results of Table 1 and 2 because the computational cost for recovering the remaining bits become infeasible as we analyzed in Subsection 6) of Section III.B.

A countermeasure for Montgomery Ladder against our attack is presented by Hanley *et al.* in [8]. It is simply replacing from Step 3 to Step 8 of Alg. 3 with $R_{1-d_i} \leftarrow R_{1-d_i} R_{d_i}$. By this way, collision characteristic of Section III.A is removed as positions of operands are switched. Nevertheless, IHCCA defeats this countermeasure. To the best of our knowledge, countermeasures against IHCCA do not exist yet. This can be our future work including countermeasures for side-channel atomic exponentiation.

In addition to countermeasures on exponentiation algorithm, a method of interleaving exponentiations for multiple prime candidates is considerable in primality test level to prevent an adversary from acquiring power consumption traces for the same exponent.

V. CONCLUSION

We performed a practical collision-based power analysis attack on the Miller-Rabin test of RSA prime generation and recovered a 512-bit secret prime with only three-bit failure with a single attempt for a realistic setting where five 512-bit modular exponentiations are operated on an ARM Cortex-M4 microcontroller as recommended by FIPS 186-4 standard. Our attack is applicable even when countermeasures against previous attacks [22], [23] are deployed, also it does not require the incremental prime search assumption on which the attack of [24] depends, thus, countermeasures proposed to defeat the attack of [24] are not effective against our attack. Although we proposed a countermeasure on modular exponentiation algorithm level to reduce the collision characteristics which we exploited, it seems that changing in the primality test procedure, that is, not sequentially exponentiating for the same secret exponent is required for the higher security of RSA prime generation.

REFERENCES

[1] P. C. Kocher, "Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems," in *Proc. CRYPTO*, Santa Barbara, CA, USA, 1996, pp. 104–113.

[2] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Proc. CRYPTO*, Santa Barbara, CA, USA, 1999, pp. 388–397.

[3] C. D. Walter, "Sliding windows succumbs to big Mac attack," in *Proc. CHES*, Paris, France, 2001, pp. 286–299.

[4] H. Kim, T. H. Kim, J. C. Yoon, and S. Hong, "Practical second-order correlation power analysis on the message blinding method and its novel countermeasure for RSA," *ETRI J.*, vol. 32, no. 1, pp. 102–111, Feb. 2010.

[5] C. Clavier, B. Feix, G. Gagnerot, M. Roussellet, and V. Verneuil, "Horizontal correlation analysis on exponentiation," in *Proc. ICICS*, Barcelona, Spain, 2010, pp. 46–61.

[6] C. Clavier, B. Feix, G. Gagnerot, C. Giraud, M. Roussellet, and V. Verneuil, "ROSETTA for single trace analysis," in *Proc. INDOCRYPT*, Kolkata, India, 2012, pp. 140–155.

[7] A. Bauer, E. Jaulmes, E. Prouff, J. R. Reinhard, and J. Wild, "Horizontal collision correlation attack on elliptic curves," *Cryptogr. Commun.*, vol. 7, no. 1, pp. 91–119, Mar. 2015.

[8] N. Hanley, H. Kim, and M. Tunstall, "Exploiting collisions in addition chain-based exponentiation algorithms using a single trace," in *Proc. CT-RSA*, San Francisco, CA, USA, 2015, pp. 431–448.

[9] D. Boneh, R. A. DeMillo, and R. J. Lipton, "On the importance of checking cryptographic protocols for faults," in *Proc. EUROCRYPT*, Konstanz, Germany, 1997, pp. 37–51.

[10] J. L. Danger, S. Guilley, P. Hoogvorst, C. Murdica, and D. Naccache, "Improving the big Mac attack on elliptic curve cryptography," in *The New Codebreakers*. Heidelberg, Germany: Springer, 2016, pp. 374–386.

[11] P. Das, D. B. Roy, H. Boyapally, and D. Mukhopadhyay, "Inner collisions in ECC: Vulnerabilities of complete addition formulas for NIST curves," in *Proc. AsianHOST*, Yilan, Taiwan, 2016, pp. 1–6.

[12] S. M. Cho, S. Jin, and H. S. Kim, "Side-channel vulnerabilities of unified point addition on binary huff curve and its Countermeasure," *Appl. Sci.*, vol. 8, p. 10, Oct. 2018. Accessed: Jan. 25, 2019. doi: 10.3390/app8102002.

[13] C. Luo, Y. Fei, and D. Kaeli, "Effective simple-power analysis attacks of elliptic curve cryptography on embedded systems," in *Proc. ICCAD*, San Diego, CA, USA, 2018, pp. 1–7.

[14] J.-S. Coron, "Resistance against differential power analysis for elliptic curve cryptosystems," in *Proc. CHES*, Worcester, MA, USA, 1999, pp. 292–302.

[15] P. L. Montgomery, "Speeding the Pollard and elliptic curve methods of factorization," *Math. Comput.*, vol. 48, no. 177, pp. 243–264, Jan. 1987.

[16] M. Joye and S.-M. Yen, "The Montgomery powering ladder," in *Proc. CHES*, Redwood Shores, CA, USA, 2002, pp. 291–302.

[17] B. Chevallier-Mames, M. Ciet, and M. Joye, "Low-cost solutions for preventing simple side-channel analysis: Side-channel atomicity," *IEEE Trans. Comput.*, vol. 53, no. 6, pp. 760–768, Jun. 2004.

[18] *Digital Signature Standard (DSS)*, Standard FIPS PUB 186-4, National Institute of Standards and Technology, 2013.

[19] J. Brandt, I. Damgård, and P. Landrock, "Speeding up prime number generation," in *Proc. ASIACRYPT*, Fujiyoshida, Japan, 1991, pp. 440–449.

[20] J. Brandt and I. Damgård, "On generation of probable primes by incremental search," in *Proc. CRYPTO*, Santa Barbara, CA, USA, 1992, pp. 358–370.

[21] J. Gordon, "Strong primes are easy to find," in *Proc. EUROCRYPT*, Paris, France, 1984, pp. 216–223.

[22] T. Finke, M. Gebhardt, and W. Schindler, "A new side-channel attack on RSA prime generation," in *Proc. CHES*, Lausanne, Switzerland, 2009, pp. 141–155.

[23] A. Bauer, E. Jaulmes, V. Lomné, E. Prouff, and T. Roche, "Side-channel attack against RSA key generation algorithms," in *Proc. CHES*, Busan, South Korea, 2014, pp. 223–241.

[24] C. Vuillaume, T. Endo, and P. Wooderson, "RSA key generation: New attacks," in *Proc. COSADE*, Darmstadt, Germany, 2012, pp. 105–119.

[25] STMicroelectronics, Amsterdam, The Netherlands. *STM32F405/415*. Accessed: Dec. 17, 2018. [Online]. Available: https://www.st.com/en/microcontrollers/stm32f405-415.html

[26] D. Coppersmith, "Small solutions to polynomial equations, and low exponent RSA vulnerabilities," *J. Cryptol.*, vol. 10, no. 4, pp. 233–260, Sep. 1997.

[27] J. Katz, A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, FL, USA: CRC Press, 1996, pp. 595–616.

[28] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Berlin, Germany: Springer, 2007, pp. 61–70.

[29] C. O'Flynn and Z. D. Chen, "ChipWhisperer: An open-source platform for hardware embedded security research," in *Proc. COSADE*, Paris, France, 2014, pp. 243–260.

[30] NewAE Technology, Halifax, NS, Canada. *CW308T-STM32F*. Accessed: Dec. 17, 2018. [Online]. Available: http://wiki.newae.com/CW308T-STM32F

[31] NewAE Technology, Halifax, NS, Canada. *CW501 Differential Probe*. Accessed: Dec. 17, 2018. [Online]. Available: http://wiki.newae.com/CW501_Differential_Probe

[32] Teledyne LeCroy, Thousand Oaks, CA, USA. *HDO6104A*. Accessed: Dec. 17, 2018. [Online]. Available: http://teledynelecroy.com/oscilloscope/hdo6000a-high-definition-oscilloscopes/hdo6104a

SANGYUB LEE received the B.S. degree in telecommunications and the M.E. degree in information security from Korea University, Seoul, South Korea, in 2011 and 2015, respectively, where he is currently pursuing the Ph.D. degree in information security with the Graduate School of Information and Security. His research interests include side-channel attacks, public-key cryptosystems, and embedded system security.

SUNG MIN CHO received the M.A. degree in information security from Korea University, Seoul, South Korea, in 2011, where he is currently pursuing the Ph.D. degree with the Graduate School of Information Security. His specialty lies in the area of information security, and his research interests include algorithms for fast implementation and the side-channel attacks of public-key cryptosystems.

HEESEOK KIM received the B.S. degree in mathematics from Yonsei University, Seoul, South Korea, in 2006, and the M.S. and Ph.D. degrees in engineering and information security from Korea University, Seoul, South Korea, in 2008 and 2011, respectively. He was a Postdoctoral Researcher with the University of Bristol, U.K., from 2011 to 2012. From 2013 to 2016, he was a Senior Researcher with the Korea Institute of Science and Technology Information (KISTI). Since 2016, he has been with Korea University. His research interests include side-channel attacks, cryptography, and network security.

SEOKHIE HONG received the M.A. and Ph.D. degrees in mathematics from Korea University, Seoul, South Korea, in 1997 and 2001, respectively. He was with SECURITY Technologies Inc., from 2000 to 2004. From 2004 to 2005, he was a Postdoctoral Researcher with COSIC, KU Leuven, Belgium. Since 2005, he has been with Korea University, where he is currently with the Graduate School of Information Security. His specialty lies in the area of information security, and his research interests include the design and analysis of symmetric-key cryptosystems, public-key cryptosystems, and forensic systems.