International Journal of Cryptology Research 5(2): 45-55 (2015)

A Secure Cryptographic Algorithm against Side Channel Attacks

Nur Azman Abu[1] and Amir Hamzah Abd Ghafar[2]

*[1]Faculty of Information and Communication Technology,
Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya,
76100 Durian Tunggal,
Melaka, Malaysia.*

*[2]Al-Kindi Cryptography Research Laboratory,
Institute for Mathematical Research,
Universiti Putra Malaysia,
43400 Serdang, Selangor, Malaysia*

ABSTRACT

Historically, a computing resource is scarce and expensive. In the last few decades, considerable efforts have been made to design efficient codes in terms of the storage space and running time. Due to the progress on computing resources and low cost of memory, an efficient algorithm has ironically become a vulnerable threat to cryptographic operations. An efficient unbalanced code opens another room for side channel attacks on the private key of public key infrastructure (PKI). This paper shall highlight and propose balanced secure algorithms for cryptographic operations to avoid feasible side channel attacks in the immediate future.

Keywords: Side channel attacks, secure programming.

1. INTRODUCTION

Electronic computers have evolved from exiguous experimental enterprises (Denning, 1982) in the 1950s to prolific data processing systems in the twenty first century. There are two computing resources which computer scientists have been paying attention in the last century. They are space and time. The constraint on the space has been discounted from the last century due to an electronic progress and advancement from silicon chip industry. At the same time, the running time of computational operations is still flexibly upgraded by a more efficient code. Even though, computer science textbooks have clearly outlined the efficiency of running time in terms of asymptotic big-oh notation, the real applications still quest for more efficient average running time with regards to its constant factors.

This quest for efficient running time has become meaningless in the realm of microseconds per transaction. Still there is an argument for batch processing such as check clearance which runs few millions per hour. The same quest has been welcomed in the field of cryptography. An efficient algorithm has always been accepted as an economic progress. Theoretically, a more efficient algorithm, which saves even 50% of running time such as the ones proposed by (Othman, 2008) and (Koval, 2010) or by a constant factor, is not a progress at all especially in cryptography. Unless it is asymptotically faster such as a reduction from $O(n^3)$ down to $O(n^2 \log n)$, the saving by a constant factor such as from $O(5n^3)$ down to $O(2n^3)$ is not a considerable progress at all. The savings by a constant factor may easily come from the omission of unnecessary redundancy to keep the code running with balanced operations. The speed is not an essence here. It is crucial to keep a balanced computational operation in cryptography between bit 0 and bit 1 during a decryption process while using the private secret key.

This vulnerable phenomenon is due to an attack called side-channel attack by Paul (Kocher, 1996) which uses the timing difference observed from modular exponentiation process to determine the secret exponent being used. Modular exponentiation algorithm is commonly used in various cryptosystems including Diffie-Hellman, DSS and RSA. A readout from a physical indicator, in this case, on computational time; makes this attack quickly expanding. Next, Paul (Kocher and et al,. 1999) orchestrated an attack concentrated on the computational power observed during a decryption process. It used the same weakness in unbalanced modular exponentiation algorithm which will be discussed in the next few sections. Other than computational time and power, other external factors such as transmission methodology by cryptographic hardware have also been used in side channel attack such as cache response (Osvik et. al., 2006) and electromagnetic radiation (Gandolfi et. al., 2001).

Recently, Daniel (Genkin and et. al., 2014) focuses on a source of noise such as the vibration of electronic components in the computer. During a decryption process, the noise sometimes heard as a faint high-pitched tone or hiss often generated by capacitors and correlated with system activity since CPUs drastically change their power draw according to different type of operations they perform. The key extraction attack finds the secret key bits sequentially one by one. It is sufficient to differentiate between the pitch during the decryption process when the bit is 0 and 1. Since the CPU is running different operation during the short period of bit 0 from bit 1, naturally, the computer will emit different sound signals. Theoretically, this attack is feasible due to an imbalance algorithm deep in the internal details of GnuPG's modular exponentiation algorithm implementation of RSA (Rivest et al, 1978).

2. FREQUENCY SAMPLING

According to Nyquist Theorem, 2 samples per cycle of the input signals should properly define the target signals. Unlike in symmetric cipher, a signing or decryption process in PKI is typically running sequentially one bit private key at a time. During the decryption process when the bit is 1, the computing device produce higher pitch sounds. If this pitch sound can be accurately measured and identified at least at the level of twice the frequency signals, the attacker will be able to differentiate the pitch during the decryption process when the bit is 0 or 1. Daniel (Genkin and et. al., 2014) gives a sample set of a strong component at 35.2 kHz and 38.1 kHz when the secret bits are 0 and 1 respectively. A minimum of sampling frequency at 96 kHz is required to observe such signal components. It should be noted here that a sampling frequency comes in a multiple of 44,100 Hz from Audio CD or 48,000 Hz the standard audio sampling rate used by most professional digital video equipment. The next available high sampling rate is 192 kHz. It is sufficient to analyse the residue details of computing devices during the decryption or digital signing process. The sampling frequency goes up to 5,644,800 Hz for the Super Audio CD (Reefman and Nuijten, 2001).

3. AN EFFICIENT UNBALANCED ALGORITHM

An efficient algorithm which produces an accurate output does not imply the algorithm is secure. In order to be secure the algorithm must be balanced on the different critical input especially on the secret element the algorithm supposed to protect at all computational cost such as the private key. An efficient classic algorithm which runs right-to-left on the exponent of power modulo is pseudo coded in Algorithm−***A*** below. The Algorithm–***A*** is widely used in implementation of RSA due to its efficiency.

```
Algorithm−A: PowerMod ($a$, $b$, $N$)
Output: $a^b \bmod N$

Let $b = b_0 b_1 b_2 \cdots b_{n-1} = \sum_{i=0}^{n-1} b_i \cdot 2^i$ written in big-endian
$L = a^0 = 1$, $R = a^1 = a$.
for $i = 0$ to $n-1$ do
        if $b_i = 1$ then
                $L = L \cdot R \bmod N$
        end( if-then )
        $R = R^2 \bmod N$
end( for )
return($L$).
```

This is an efficient textbook algorithm. It is necessary to have square mod operation on each bit of the exponent. However, it is only sufficient to have multiply mod operation whenever the exponent bit is one. This condition tends to cause the algorithm to become unbalanced. Nevertheless, Algorithm−$\boldsymbol{A}_1$ will be complemented with Montgomery reduction (Montgomery, 1985) in most of RSA implementations. Montgomery happens every time there is a multiplication of $L{\cdot}R$ mod N and $R = R^2$ mod N. This complementary algorithm is shown below.

```
Algorithm – $\boldsymbol{A}_1$: Montgomery ($a$, $b$, $N$)
Output: $a^b$ mod $N$

$a' = a{\cdot}K$ mod $N$, $b' = b{\cdot}K$ mod
$K{\cdot}K' - N{\cdot}N' = 1$
$z = a'{\cdot} b'$
$r$ = ($z$ mod $R$ )( $N'$ mod $R$ )
$s$ = ($z$+$r{\cdot}N$) = $R$
if $s \geq N$ then
        $s = s - N$  //extra reduction process
end ( if-then )
return ($s$).
```

Usage of Montgomery reduction will result in multiplication of two numbers that are about the same number of bits. This triggers implementers to use Karatsuba (Karatsuba and Ofman, 1962) multiplication as this type of multiplication is effective on these numbers. Unfortunately, the difference in the processing power required to have or not to have multiply mod operation that depends on the nature of the exponent bit can certainly be acoustically detected.

4. A BALANCED SECURE ALGORITHM

In this section we will discuss the following algorithm:

```
Algorithm–$\boldsymbol{B}$: Target Sum($b$)

Let $b = b_0 b_1 b_2 \cdots b_{n-1} = \sum_{i=0}^{n-1} b_i \cdot 2^i$ written in big-endian
$L$= 0, $R$ = 1.
```

```
for $i = n-1$ down to 0 do
        if $b_i = 0$ then
                $R = L+R$
                $L = 2L$.
        end( if-then )
        if $b_i = 1$ then
                $L = L+R$
                $R = 2R$.
        end( if-then )
end( for )
return($L$).
```

The security of the cryptographic system should not depend on the secrecy of the algorithms nor apparatus being used. They shall be made public and open to all users (Abu and Sahib, 2010). The algorithms operating on secret private keys must be securely balanced in order to operate in open environment. It should be noted that the Algorithm−***A*** is a popular cryptographic codings in practice. It is not balanced hence rendering them insecure and vulnerable to side channel attack in an open environment. Alternatively, an open balanced algorithm is called for here. The Algorithm–***B***, however, is the basic left-to-right algorithm (Levitin, 2012) which has been neglected and discounted on its importance. The algorithm presented above carry the same operation regardless of the binary of the secret key whether it is zero or one.

4.1 Algorithm–*B* and Elliptic Curve Cryptosystem

It is very natural to adjust Algorithm–***B*** to compute point projection in an elliptic curve cryptosystem. Neal Koblitz and Victor S. Miller have independently first proposed the use of elliptic curves for cryptography at about the same time in 1986. A sample on point projection is written in Algorithm–***C*** below. The initial value zero has been replaced by an identity point zero at infinity. The addition L+R and doubling have been replaced by point addition and point doubling respectively.

Algorithm–***C***: Point Projection(λ, P)
Output: λP

```
Let $\lambda = b_0b_1b_2\cdots b_{n-1} = \sum_{i=0}^{n-1} b_i \cdot 2^i$ written in big-endian
$L$= $\underline{\mathbf{0}}$, $R = P$,   where $\underline{\mathbf{0}}$ is an identity point zero at infinity
for $i = n-1$ down to  0 do
        if $b_i = 0$ then
                $R$ = AddPoint($L$, $R$)
                $L$ = DoublePoint($L$)
        end( if-then )
        if $b_i = 1$ then
                $L$ = AddPoint($L$, $R$)
                $R$ = DoublePoint($R$)
        end( if-then )
end( for )
return($L$).
```

4.2 Algorithm–*B* and RSA

Next, Algorithm–***B*** shall be used to do power mod operation in RSA encryption or decryption (Rivest et. al., 1978). In this case, however the exponent shall be the binary sequence $b = b_0b_1b_2\cdots b_{n-1}$. As tailored in Algorithm–***D***, the initial value zero has been replaced by one. The addition operation $L+R$ and doubling have been replaced by multiplication modulo N and square modulo N respectively.

Algorithm–***D***: PowerMod (a, b, N)
Output: a^b mod N

```
Let $b = b_0b_1b_2\cdots b_{n-1} = \sum_{i=0}^{n-1} b_i \cdot 2^i$ written in big-endian
$L = a^0 = 1$, $R = a^1 = a$.
for $i = n-1$ down to  0 do
        if $b_i = 0$ then
                $R = L{\cdot}R$ mod $N$
                $L = L^2$ mod $N$
        end( if-then )
        if $b_i = 1$ then
                $L = L{\cdot}R$ mod $N$
                $R = R^2$ mod $N$
        end( if-then )
end( for )
return($L$).
```

4.3 Algorithm–*B* and the General Lucas Sequences

The LUC cryptosystem (Smith and Lennon, 1993) has been designed based on the general Lucas functions. Therefore, the security of this cryptosystem is analogous to the RSA cryptosystem. It relies on the difficulty of factoring *N* back into its prime factors *P* and *Q*. Similarly, an LUC cryptosystem is also vulnerable to side channel attacks. The encryption and decryption processes are similar to power modulo operation with minor adjustment.

Let (p, q) be nonzero integers, and let α and β be the two complex roots of the quadratic polynomial $x^2 - px + q$. Then the general Lucas sequences (U_n, V_n) satisfy the recurrence relations

$$U_0 = 0,\ U_1 = 1 \text{ and } U_{n+1} = p\,U_n - q\,U_{n-1},$$
$$V_0 = 2,\ U_1 = p \text{ and } V_{n+1} = p\,V_n - q\,V_{n-1}.$$

The Lucas sequences may be written in closed forms as

$$U_n = \frac{\alpha^n - \beta^n}{\alpha - \beta} \text{ and } V_n = \alpha^n + \beta^n$$

Typically, the parameter *q* is set to be one. There is only one parameter *p* in computing the Lucas sequences. Even though the sequence *V* is more popular for LUC cryptosystem, here in Algorithm–***E*** below, both sequences *U* and *V* are being written for better references. While the product of public and private exponents; $e{\cdot}d \equiv 1 \bmod (P-1)(Q-1)$ in RSA cryptosystem, it is $e{\cdot}d \equiv 1 \bmod (P^2-1)(Q^2-1)$ in the LUC cryptosystem. The encryption process of this system is the computations of V_e, while the decryption process is done by the computations of V_d. The V_e and V_d are both Lucas functions (Ali et. al. 2008).

Algorithm–***E***: General Lucas Sequences UV(b, p, N)
Output: U_b mod N and V_b mod N

```
Let $b$ = $b_0 b_1 b_2 \cdots b_{n-1} = \sum_{i=0}^{n-1} b_i \cdot 2^i$ written in big-endian
$UL$= 0, $UR$ = 1, $VL$= 2, $VR$ = $p$,

for $i = n-1$ down to 0 do
        if $b_i = 0$ then
                $UR = UR \cdot VL - 1 \bmod N$
                $UL = UL \cdot VL \quad \bmod N$
                $VR = VL \cdot VR - p \bmod N$
                $VL = VL^2 - 2 \quad \bmod N$
        end( if-then )

        if $b_i = 1$ then
                $UL = UR \cdot VL - 1 \bmod N$
                $UR = UR \cdot VR \quad \bmod N$
                $VL = VL \cdot VR - p \bmod N$
                $VR = VR^2 - 2 \quad \bmod N$
        end( if-then )
end( for )
return($UL$, $VL$).
```

The idea is as follows: Let $b = (b_{n-1} \ldots b_1\ b_0)_2$ in binary and the Lucas sequences begin with the ordered pairs $(U_0, U_1) = (0, 1)$ and $(V_0, V_1) = (2, p)$. Moving from left to right in the binary representation of b, having the pair (U_m, U_{m+1}) we compute either (U_{2m}, U_{2m+1}) and (V_{2m}, V_{2m+1}) (if the bit b_i is a 0) or (U_{2m+1}, U_{2m+2}) and (V_{2m+1}, V_{2m+2}) (if the bit b_i is a 1). This process will terminate with the ordered pair (U_b, U_{b+1}) and (V_b, V_{b+1}).

5. DISCUSSION

An individual CPU operation clocked at several Gigahertzes is too fast for a high fidelity microphone to sample and digest. However, long operations such as RSA modular exponentiation do create distinctive acoustic spectral characters over few milliseconds. In this paper, a set of algorithms have been highlighted for common public key cryptography such as RSA, ECC and LUC that carry the same operation regardless of the binary of the secret key whether it is zero or one. Popular cryptographic coding in practice, however, is not balanced hence rendering them unsecure and vulnerable to side channel attack in an open environment. It is just a matter of time before this onsite attack will happen since the available audio technology at a high frequency sampling is already sufficient to harness and harvest the private key.

One may argue that the side channel attack can easily be deterred by introducing a random element before decryption using private key is initiated. This method which is called blinding and implemented in the latest PKCS#1 (Jonsson and Kaliski, 2003), is not suitable to be utilized in devices with low memory space. This is because it needs to embed a pseudorandom generator algorithm to maintain its randomness. However, a balanced algorithm does not need to utilize such space because the additional operation in the algorithm (i.e modular addition and multiplication) can be cheaply calculated.

6. CONCLUSION

An adversary who is capable of capturing the moment during a decryption process may overcome the cryptosystem. An efficient unbalanced code opens this vulnerable ream to maneuver via side channel attacks on the private key of public key infrastructure. An inexpensive cheap high fidelity digital audio has been well developed about the same time of PKI. A present high frequency digital acoustic sampling has become a new threat to the unbalanced cryptographic codes running on small devices. A few hundred kHz frequencies sampling, using ultrasound microphones in several orders of magnitude relative to the GHz-scale clock rates of the attacked computers, is sufficient to capture reminiscent of decryption process. It is also just a matter of time, a more advanced audio processing operating on higher frequency becomes cheaply available poses serious threat to these unbalanced cryptographic codes running on regular computers. This paper has proposed a left-to-right balanced secure algorithms for cryptographic operations to avoid feasible side channel practical attacks in the immediate future.

REFERENCES

Abu, N. A. and Shahrin, S. 2010. Random Ambience Key Generation Live on Demand, *Proceedings 2nd International Conference on Signal Processing Systems* (*ICSPS* 2010). **1**:110-114.

Ali, Z. M., Othman, M., Said, M. R. M., and Sulaiman, M. N. (2008). An Efficient Computation Technique for Cryptosystems Based on Lucas Functions, *Proceedings of the International Conference on Computer and Communication Engineering*. 187-190.

Denning, D. E. (1982). *Cryptography and Data Security*. Addison-Wesley, preface page v.

Gandolfi, K., Mourtel, C. and Olivier, F. (2001). Electromagnetic analysis: Concrete results. *Procedings of CHES* 2001:251–261.

Genkin, D., Shamir, A. and Tromer, E. (2014). RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis, *Advances in Cryptology*–CRYPTO 2014:444–461.

Jonsson, J., and Kaliski, B. (2003). Public-key cryptography standards (PKCS)# 1: RSA cryptography specifications version 2.1.

Karatsuba, A. A. and Ofman, Y.P. (1962). Multiplication of Many-Digital Numbers by Automatic Computers. *Proceedings of the USSR Academy of Sciences*:293–294.

Koblitz, N. (1987). Elliptic Curve Cryptosystems. *Mathematics of Computation*. **44**(177): 203–209.

Kocher, P. C. (1996). Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. *Proceedings of CRYPTO* '96:104–113.

Kocher, P. C., Jaffe, J. and Jun, B. (1999). Differential Power Analysis, *Proceedings of CRYPTO* '99:388–397.

Koval, A. (2010). On Lucas Sequences Computation. *International Journal of Communications, Network and System Sciences*. **3**:943–944.

Levitin, A. (2012). *Introduction to the Design and Analysis of Algorithms*. Pearson.

Miller, V. S. (1985). Use of Elliptic Curves in Cryptography. *Proceedings Advances in Cryptology* (*CRYPTO* ’85) *Lecture Notes in Computer Science*. **218**:417–426.

Montgomery, P. L. (1985). Modular Multiplication without Trial Division. *Mathematics of Computation*. **44**(170):519–521.

Osvik, D. A., Shamir, A. and Tromer, E. (2006). Cache attacks and countermeasures: the case of AES. *Proceedings of CT-RSA* 2006:001–020.

Othman, M., Abulhirat, E. M., Ali, Z. M., Said, M. R. M. and Johari, R. (2008). A New Computation Algorithm for a Cryptosystem Based on Lucas Functions. Journal of Computational. **4**:1056–1060.

Reefman, D. and Nuijten, P. (2001). Why Direct Stream Digital is the Best Choice as A Digital Audio Format. 110*th* *Audio Engineering Society Convention*.

Rivest, R. L., Shamir, A. and Adelman, L. (1978). A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, *Communications of ACM*. **21**(2): 122–126.

Smith, P. J., Lennon, G. J. J. (1993). LUC: a new public key cryptosystem, 9*th* *IFIP Symposium on Computer Science Security*:103–117.