AUTOMATYKA/ AUTOMATICS • 2016 • Vol. 20 • No. 2

http://dx.doi.org/10.7494/automat.2016.20.2.39

Natalia Krzyworzeka*

Asymmetric cryptography and trapdoor one-way functions

1. Introduction

The asymmetric encryption scheme was invented in 1969 by James H. Ellis (a British engineer and mathematician) and is now considered to be the starting point for modern cryptography. The first successful implementation of the public-key algorithm was achieved in 1973 by Clifford C. Cocks (another British mathematician and cryptographer); however, his discovery was not published until 1997 due to its classified nature. Surprisingly, the same encryption mechanism was independently re-discovered by another group of cryptographers: Ron Rivest, Adi Shamir, and Len Adleman (from whom the RSA algorithm took its name).

As opposed to symmetric cryptography, public-key encryption allows us to send and receive messages without the need of exchanging a secret key with the other party. Instead, the encryption algorithm generates a pair of two complementary cryptographic keys: the encryption (public) and decryption (private) keys. Though the private key must be highly protected by the owner, the public key can be widely disseminated to anyone for the purpose of secret data transfer. The asymmetric-key scheme, presented in Figure 1, is based on the assumption that the public key will neither provide any information about the private key nor will it allow others to recreate it and that the message encrypted according to the public key will not be possible to decrypt by any known cryptographic attack method. In reality, the public and private keys are used as a metaphor to describe the operations of a chosen trapdoor one-way function and its inverse. The usefulness of an encryption algorithm depends on the difficulty of the mathematical problem; that is, the inversion of a TOWF as well as on the fluidity of the corresponding trapdoor.

* AGH University of Science and Technology, Faculty of Electrical Engineering, Automatics, Computer Science and Biomedical Engineering, Krakow, Poland

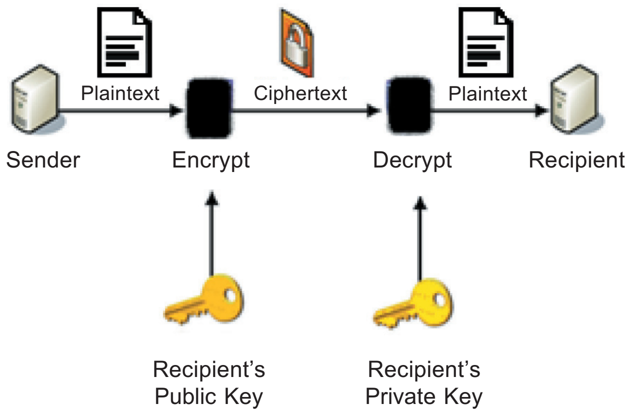

Fig. 1. Asymmetric encryption scheme

To ensure the safety of the asymmetric encryption, additional requirements concerning the generated keys are very strict. For example, in the RSA algorithm, the two prime numbers used as function parameters must be at least 150 digits long. In order to fully explain the RSA algorithm (and, thus, the public-key-encryption concept), a few basic mathematical theorems must be presented.

1.1. Algebraic one-way function

Algebraic one-way function: $F.gen(t)$: Z+→{0,1} is a type of function that is easy to compute in one direction but impossible to reverse. It should be understood that every function: $y = f(x)$, on domain $(0,\infty)$ is relatively easily performed, whereas its inverse: $x = f^{-1}(y)$ cannot be calculated, implying that the formula for such an operation either simply does not exist or that the chances of predicting the correct solution are negligible.

Additionally, for the function to be considered one-way, the following properties must be satisfied [1]:

- homomorphism: it is possible to perform algebraic *operations on such functions*, e.g., multiplying: $F(x) \cdot F(y) = F(x \cdot y)$, or adding two functions together: $F(x) + F(y) = F(x + y)$;
- ring-homomorphism: if R and S are rings, then a ring homomorphism is function $f : R \to S$ such that:
  - $f(a + b) = f(a) + f(b)$ for all a and b in R,
  - $f(ab) = f(a)\, f(b)$ for all a and b in R,
  - $f(1_R) = 1_S$.

Due to the lack of mathematical proof, the actual existence of one-way functions is still questioned. We know several functions that capture the properties of the complexity--theoretic one-way function and are classified as one, but only under the condition that an algorithm capable of efficiently computing its inverse has yet to be discovered.

The existence of a number-theoretic one-way function would prove that complexity classes P and NP are not equal ($P \neq NP$), which would resolve one of the famous questions of theoretical computer science. It would also imply the existence of many other important concepts, such as:

- pseudorandom generators,
- pseudorandom function families,
- bit commitment schemes,
- private-key encryption schemes secure against adaptive chosen-ciphertext attack,
- message authentication codes,
- digital signature schemes (secure against adaptive chosen-message attack).

1.2. Trapdoor one-way function

A trapdoor function is a special case of one-way function ($F : X \to Y$) that contains a "trapdoor." The term "trapdoor" refers to specific information usually obtained during the process of creating a TOWF that can provide an easily computable inverse of a said function. In other words, whereas $y = f^{-1}(x)$ is considered to be unsolvable by modern computers, equation: $y = f^{-1}(x, k)$ (with k being additional knowledge) is easily calculated. Although there is no evidence that a TOWF can be constructed from any one-way function, a number of methods creating the so-called one-time trapdoor one--way function that can be used in certain transaction-related data-encryption processes have been published [2]. The general idea of a trapdoor one-way function is presented in Figure 2.

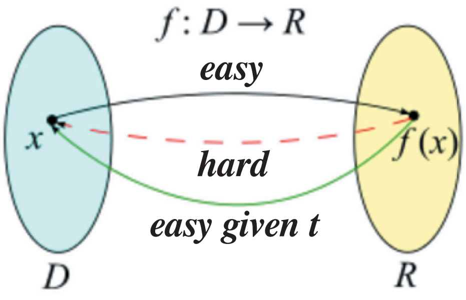

Fig. 2. Trapdoor one-way function [3]

From the definition: function $f : \{0,1\}^{l(n)} \cdot \{0,1\}^n \rightarrow \{0,1\}^{m(n)}$ is a trapdoor one-way function if:

- it is a one-way function;
- for fixed public key $9xy \in : \{0,1\}^{l(n)}$, $f(x, y)$ is viewed as function $f_y(x)$ of x that maps n bits to $m(n)$ bits; then, there is an efficient algorithm that, on input $\langle y, f_y(x), z\rangle$ produces x' such that $f_y(x') = f_y(x)$, for some trapdoor key $z \in \{0,1\}^{k(n)}$;
- F is a trapdoor one-way hash function if f is also a one-way hash function;
- given M and $f(M)$, it is hard to find message $M' \neq M$ such that $f(M') = f(M)$.

It is not known if a trapdoor one-way function can be constructed from any one-way function [4].

In asymmetric cryptography, the trapdoor one-way function formula used for message encryption becomes the public key, and the secret information (trapdoor) used to compute its inverse becomes the private key. As of this day we know of only two algorithms (both based on large prime factorization) that capture the trapdoor one-way function properties: RSA and Rabin.

2. The RSA algorithm

The trapdoor one-way function used as the encryption mechanism in the RSA algorithm is modular exponentiation:

$$m^e \bmod N = c,$$

where:

m – secret message,
e – public exponent,
N – public value,
c – encrypted message.

In order to completely understand the TOWF mechanism behind this algorithm, knowledge of the following theorems is required.

Prime factorization

Prime factorization of a positive integer is a list of prime numbers (prime factors) together with their multiplicities that divide that integer exactly. The process of finding these numbers is called integer factorization. The fundamental theorem of arithmetic states that each positive integer has a single unique prime factorization. For example:

the prime factors of positive integer 864 are: $864 = 2^5 \cdot 3^3$, where 2 and 3 are the prime factors.

For 1092: $1092 = 3 \cdot 2^2 \cdot 7 \cdot 13$, where 2, 3, 7 and 13 are the prime factors.

The asymmetric-key encryption relies on the fact that there is no efficient way of finding the prime factorization of large integers. The RSA algorithm generates two giant prime numbers p and q (each about 150 digits long) and sets $N = p \cdot q$. The only thing required to break the code is to find the prime factorization of N; however, due to the complexity of the calculations, it is impossible for modern computers to find the correct answer. The estimated time required to break the RSA algorithm is measured in millions of years.

2.1. Euler's totient function

Euler's totient function (Euler's phi function), which is a special case of Fermat's Little Theorem, for a given number n, $\varphi(n)$, outputs how many integers k in the range $1 \leq k \leq n$ are relatively prime to n, meaning that they share a common divisor (gcd $(n, k) = 1$). For example:

$$n = 12,$$

$$\varphi(n) \equiv 5, 7, 11,$$

$$\varphi(n) = 3.$$

The phi function is considered to be a special case of the trapdoor function, because it is very hard and incredibly time-consuming to evaluate $\varphi(n)$ for large integers unless we know that n is a prime number (Fig. 3). Knowing that a prime number can only be divided by itself and 1, the phi function of a prime integer is represented as:

$$\varphi(P) = P - 1,$$

where P – the prime number.

For example: $P = 21377$, $\varphi(P) = 21376$.

Euler's totient function is also multiplicative:

$$\varphi(a \cdot b) = \varphi(a) \cdot \varphi(b)$$

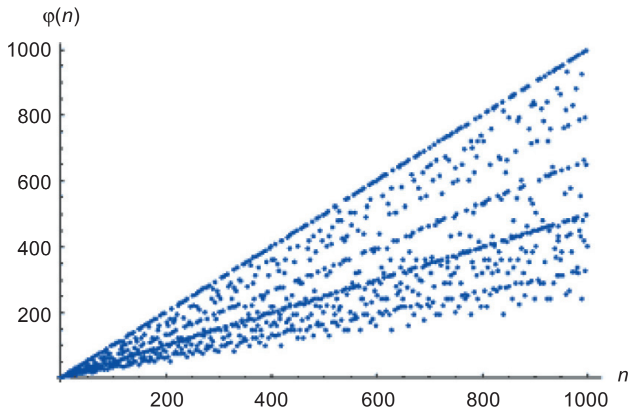

Fig. 3. Euler's totient function for values 1–1000 [5]

2.2. Euler's (totient) theorem

Euler's totient theorem (also known as the Fermat–Euler theorem) presents the connection between the modulus and exponential calculation. From the definition: if p and a are coprime positive integers, then:

$$a^{\varphi(n)}\, mod\; n \equiv 1,$$

where $\varphi(n)$ is Euler's totient function.

Euler's totient theorem (which is a generalization of Fermat's little theorem) was published in 1736 by Leonhard Euler. It may be used to reduce large powers modulo k. For instance: if $a^{\varphi(n)}\, mod\; n \equiv 1$, then $(a^{\varphi(n)})^k\, mod\; n \equiv 1$.

I. e.g., $7^4 \equiv 1\ (mod\ 10)$, and we get $7^{222} \equiv 7^{4 \cdot 55 + 2} \equiv (7^4)^{55} \cdot 7^2 \equiv 1^{55} \cdot 7^2 \equiv 49 \equiv 9\ (mod\ 10)$.

All of these theorems are crucial for the RSA algorithm on the grounds that they connect the modular exponentiation (which is a one-way function) with Euler's PHI function (which is a trapdoor one-way function) and prime factorization. In 1997, Clifford Cox combined all of these theorems to perform the following calculations.

Knowing that, from equation:

$$m^e\, mod\, N = c,$$

which is the encryption formula, we can define the inverse formula: $c^{e \cdot d} mod\, N = m$.

$$m^{\varphi(n)}\,mod\,N \equiv 1,$$

$$m^{\varphi(n)\cdot d}\,mod\,N \equiv 1,$$

$$m \cdot m^{\varphi(n)\cdot d}\,mod\,N \equiv m,$$

$$m^{\;1+d\cdot\varphi(n)}\,mod\,N \equiv m,$$

and finally:

$$c^{e\cdot d} mod N = m,$$

$$d = \frac{k\cdot\varphi(n)+1}{e},$$

where:

m	–	secret message,
E	–	public exponent,
N	–	public value,
C	–	encrypted message,
d	–	private key.

Here is an example of the RSA encryption process (the numbers are intentionally small for clarity):

$$\mathrm{gcd}(e,\ \varphi\ (n)) = 1,$$

where e – random, rather small odd number,

$$p = 53, g = 59, n = 3127.$$

There are several known trapdoor functions used in cryptography. Despite the early claims, one-way trapdoor functions are very rare and hard to find. This limits the possibilities for asymmetric cryptography and narrows the field of encryption schemes. As of this day, we only know two families of the TOWF that are useful for cryptographic purposes.

3. Rabin trapdoor function

Many efforts have been put forth to find an poly(n)-time algorithm capable of factoring an n-digit numbers. This implies the conclusion that such an algorithm does not exist. In cryptography, to ensure security against a brute-force attack, integers used for prime factorization must not only be very long but also relatively far apart.

The Rabin algorithm relies on the difficulty of factoring Blum integers. In mathematics, a natural number n is a Blum integer if $n = p \cdot q$ is a semiprime for which p and q are distinct prime numbers congruent to 3 *mod* 4 [6]. This means that p and q must be of the form $4t + 3$:

$$q \; mod \; 4 = 3,$$

$$p \; mod \; 4 = 3.$$

Which also indicates that the factors of a Blum integer are Gaussian primes with no imaginary part. The first few Blum integers are:

21, 33, 57, 69, 77, 93, 129, 133, 141, 161, 177…

The public key in Rabin's encryption algorithm is Blum integer $n = p \cdot q$, and primes q and p constitute the private key. The security of this cryptosystem against a chosen plaintext attack is provided by the fact that the modulus of n cannot be efficiently factored.

Rabin's encryption scheme:

- Keys: choose random large primes P and Q of length n: with $P, Q = 3 \; (mod \; 4)$, compute $N = P \cdot Q$.

Note that Euler's totient function for N can be expressed as:

$$\varphi(N) \; (mod \; 4) = (P - 1)(Q - 1) \; (mod \; 4) = 2 \cdot 2 \; (mod \; 4) = 0 \; (mod \; 4).$$

- Public key: N.
- Private key: P, Q.
- The encryption: if $X \in Z \cdot N$ let $Y = RABIN_N\,(X) = X^2 \; (mod \; ()N)$.
- Decryption: compute $X' = \sqrt{Y} \; (mod \; N)$. There are four square roots of the Y modulo n: X_1, X_2, X_3, X_4.

One of these four messages will be the encrypted data.

3.1. Trapdoor in Rabin's algorithm

When integer $P = 3$ (mod 4), there is a simple formula to compute the square root of Y in mod P.

$$\begin{aligned}(\pm c^{(p+1)/4})^2 &\equiv c^{(p+1)/2}\, mod\ P\\ &\equiv c^{(p-1)/2}\ c\ mod\ P\\ &\equiv c\, mod\ P\end{aligned}$$

Form Fermat's Little Theorem, we know that:

$$c^{(p-1)/2} = 1\, mod\, P.$$

Hence, the two square roots of c mod p are:

$$\pm c^{(p+1)/4}.$$

In a similar fashion, the two square roots of c mod q are:

$$\pm c^{(q+1)/4}.$$

Then, we can obtain the four square roots of c mod n using the Chinese Remainder Theorem.

Example:

$p = 7, q = 11, N = 77, m = 10$, where m is the secret message:

$$Y = X_2\ (mod\ 77),$$

$$Y \equiv 10^2 \equiv 23\ mod\ 77,$$

$$23^{(7+1)/4} \equiv 2^2 \equiv 4\ mod\ 7,$$

$$23^{(11+1)/4} \equiv 1^3 \equiv 1\ mod\ 11.$$

Using the Chinese Remainder Theorem to compute the square root of 23 mod 77 to be ± 10, ± 32 mod 77, we get: $m_1 = 10, m_2 = 67, m_3 = 32, m_4 = 45$.

The Rabin cryptosystem is more efficient and secure than the RSA algorithm. It was published in 1979 by Michael O. Rabin. The fact that we have to "manually" choose the right answer in Rabin's asymmetric encryption prevents it from finding widespread practical use.

4. One-time trapdoor one-way function

It is important to mention the one-time trapdoor one-way function (OTTOWF). Despite the fact that it is highly desirable that trapdoors remain secret even after their use in classical applications, there are some exceptions that require that the trapdoor does not remain secret after its use [7]. They are mostly based on the factoring assumption or generic one-way functions and require the new generation of all parameters after each use. Applications that can benefit from the OTTOWF are:

- electronic cash system (to ensure that coins are only spent once),
- digital right management system supporting delegation (to trace the fact that the proxy has used its trapdoor),
- ensuring fairness between various parties in a protocol.

5. Usage of public-key encryption

5.1. Asymmetric-key encryption in practice

Although asymmetric cryptography is superior to symmetric cryptography in many ways, there are certain disadvantages that did not allow public-key encryption from completely eliminating other algorithms. The biggest disadvantages of asymmetric-key algorithms is the fact that they are often very slow and the encrypted message has to be of a fixed size. For example, comparing RSA to the AES cryptosystem, one can conclude that AES is not only much faster and safer but that it can also be easily implemented on various operating systems, whereas the RSA algorithm cannot.

A solution to these problems is the so-called hybrid cryptosystem, which combines the convenience of a public-key cryptosystem with the efficiency of a symmetric-key cryptosystem. It uses two separate encrypting methods:

- key encapsulation: which is a public-key encryption scheme,
- data encapsulation: which is a symmetric-key scheme.

Typical usage of asymmetric-key encryption is for securing electronic communication over an open-networked environment (HTTPS protocol) and for obtaining authentication as well as digital signatures. It is also often used to secure the transmission of pay television programs to be accessed only by authorized subscribers.

There are also many other RSA-based security systems, including dual fingerprints fusion for cryptographic key generation [8], presented in Figure 4. It is an all real--time application that establishes a secured information exchange between the users. A cryptographic key is generated using dual fingerprint fusion. The key can only be obtained by using the fingerprint biometric trait of the receiver. Even if an attacker obtains the fingerprint stored in the database, it is impossible to decrypt the message because the algorithm requires dual fingerprints for the generation of a cryptographic key.

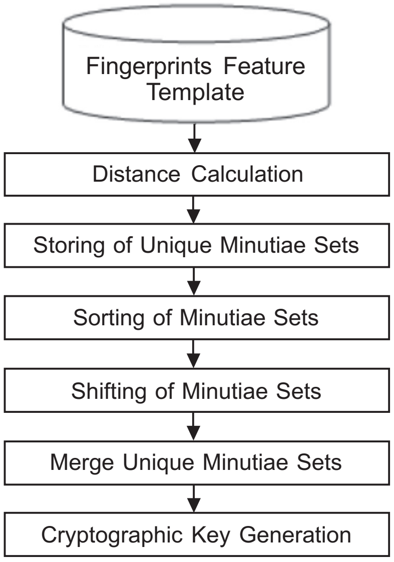
Fingerprints Feature Template
Distance Calculation
Storing of Unique Minutiae Sets
Sorting of Minutiae Sets
Shifting of Minutiae Sets
Merge Unique Minutiae Sets
Cryptographic Key Generation

Fig. 4. Cryptographic key generation in dual fingerprints fusion algorithm [8]

5.2. RSA algorithm implementation in hardware

Although it is more difficult to implement RSA algorithms in an integrated circuit (hardware) rather than in software, there are a few successful examples that prove that this can be achieved.

The solution of an FPGA architecture proposed by G. Ananth, Karthikeyan, and Dr. V. Kamakoti [9] provides high flexibility with the speed and physical security of traditional hardware. The design was created implementing a multi-board architecture using two Altera® boards, with one constantly generating random numbers using a data encryption standard (DES) random-bit generator and the second containing the RSA-implemented design that incorporates the reception of random numbers from hardware inputs.

Other RSA algorithm implementation on the FPGA architecture is a single chip for the Digital Envelope Scheme designed by R. Srinivasan, Dr. V. Vaidehi, J. Balaji, and S. Heema [10]. This system also enables the use of the DES algorithm by exploiting the parallelism present in DES and RSA operations. It offers a throughput of 3.5 Gb/s at a system clock rate of 54.7 MHz, and it is built with the use of an Altera Apex 20KE chip.

Another single-chip implementation of the RSA cryptosystem [11] proposed by Nikolaus Langecan was design for the purpose of financial application security. A "GCD – General Crypto Device" hardware architecture was proven to not only be more time efficient and secure, but it also decreased the power consumption of the sensor nodes in the case of strong cryptography use.

7. Conclusion

Recent improvements in frequency performance as well as core design may question whether customized IP blocks are still needed for secure algorithms. As proven before [12], the use of a core design supporting the RSA algorithm can save space in and the cost of the hardware. The implementation of cryptographic algorithms on programmable devices like FPGAs run noticeably faster than on software while preserving the physical security of the hardware solutions [13]. Although the flexibility of ASIC implementations still remains the biggest disadvantage, the FPGA architecture may provide the needed adaptability as well as the high speed of custom hardware.

References

[1] Catalano D., Fiore D., Gennaro R., Vamvourellis K., *Algebraic (Trapdoor) One-Way Functions and their Applications*, in: A. Sahai (ed.) *Theory of Cryptography. Lecture Notes in Computer Science*, Vol. 7785, Berlin – Heidelberg 2013, pp. 680–699.

[2] Pandey O., Pass R., Vaikuntanathan V., *Adaptive One-way Functions and Applications*, in: D. Wagner (ed.), *CRYPTO 2008 Proceedings of the 28th Annual conference on Cryptology: Advances in Cryptology*, Santa Barbara 2008, pp. 57–74.

[3] Ostrovsky R., *CS-282A / MATH-209A Foundations of Cryptography*, "Draft Lecture Notes", Winter 2010.

[4] Gardner M., *"Trapdoor Ciphers" and "Trapdoor Ciphers II." Chs. 13–14*, in: *Penrose Tiles and Trapdoor Ciphers and the Return of Dr. Matrix*, New York 1989, pp. 183–204.

[5] Khan Academy, *Euler's totient function*, https://pl.khanacademy.org [26.02.2016].

[6] Hurd J., *Blum Integers*, Trinity College, 20.01.1997, www.gilith.com/research/talks/cambridge 1997.pdf [26.02.2016].

[7] Cathalo J., Petit Smals Ch., *One-Time Trapdoor One-Way Functions*, in: M. Burmester, G. Tsudik, S. Magliveras, I. Ilić (ed.), *Information Security. ISC 2010. Lecture Notes in Computer Science*, Vol. 6531, Berlin – Heidelberg 2010, pp. 283–298.

[8] Vincenzo Conti S.V., *Fingerprint Traits and RSA Algorithm Fusion Technique*, in: *Sixth International Conference on Complex, Intelligent, and Software Intensive Systems*, Palermo 2012, pp. 351–356.

[9] Ananth G., Karthikeyan U.S. [participants], Kamakoti V. [instructor], *Cryptographic Algorithm Using a MultiBoard FPGA Architecture*, Nios II Embedded Processor Design Contest – Outstanding Designs, 2005.

[10] Srinivasan R., Vaidehi V., Balaji J., Heema S., *Single Chip Efficient FPGA implementation of RSA and DES for Digital Envelope Scheme*, “WSEAS Transactions on Communication”, Iss. 2, Vol. 3, April 2004, pp. 664–669, https://www.researchgate.net/publication/242093457_A_Single_Chip_Efficient_FPGA_implementation_of_RSA_and_DES_for_Digital_Envelope_Scheme [26.02.2016].

[11] Nikolaus L., *Single-chip implementation of a cryptosystem for financial applications*, in: R. Hirschfeld (ed.), *Financial Cryptography*. *First International Conference, FC ’97. Proceedings*, Berlin – Heidelberg – New York 1997.

[12] Alkalbani A.S., Mantoro T., Md Tap A.O., *Comparison between RSA Hardware and Software Implementation for WSNs Security Schemes*, in: *International Conference on Information and Communication Technology for the Muslim World (ICT4M)*, Jakarta 2010.

[13] Maxim Integrated Products © 2013 Inc., *Cryptography in software or hardware – it depends on the need*. *Tutorial*, 25.05.2012, pdfserv.maximintegrated.com/en/an/TUT5716.pdf [26.02.2016].