International Journal of Innovative Research in Information Security (IJIRIS) ISSN: 2349-7009(P)
Issue 01, Volume 4 (January 2017) www.ijiris.com ISSN: 2349-7017(O)

The RSA Algorithm Explored

Chuck Easttom
Collin College Professional Development
chuck@chuckeasttom.com

Abstract* — *This paper provides an overview of the RSA algorithm, exploring the foundations and mathematics in detail. It then uses this base information to explore issues with the RSA algorithm that detract from the algorithms security.

Keywords* — *RSA, Asymmetric Cryptography, Public Key Cryptography, Problems with RSA

I. INTRODUCTION

In this paper the basics of RSA will be explored, along with issues with the RSA algorithm. RSA is a commonly used asymmetric algorithm, perhaps the most widely used asymmetric algorithm (Yeh, Huang, Lin, & Chang, 2009; Ambedkar, Gupta, Gautam, & Bedi, 2011; Stallings, 2010; Mao, 2011). The algorithm was publicly described in 1977 by Ron Rivest, Adi Shamir, and Leonard Adleman at MIT. The letters RSA are the initials of their surnames. The algorithm is based on some interesting relationships with prime numbers. The security of RSA derives from the fact that it is difficult to factor a large integer into its prime factors

RSA is used extensively in e-commerce solutions (Ahmed, 2010; Constantinescu, Boldea, & Boboila, 2010; Aïmeur, Brassard, & Onana, 2006). It has been the asymmetric algorithm of choice for digital certificates, and thus an integral part of SSL and TLS.

As often happens in the history of cryptography, it turns out that there was a similar system developed earlier, but it was classified. In 1973, Clifford Cocks developed a similar system that remained classified until the late 1990's (Easttom, 2015). Mr. Cocks worked for the English government, specifically the Government Communications Headquarters (GCHQ). He did go on to develop other, non-classified, cryptographic innovations. In 2001 he invented one of the early identity based encryption methodologies, using aspects of number theory.

II. MATHEMATICAL FOUNDATIONS

The functionality of the RSA algorithm is based on aspects of number theory involving prime numbers and modulus operations (Rizvil &Wahdwa, 2010; Skurnick & Javadi, 2009; Goldston & Pintz, 2009; Hinek, 2009). The essential mathematics needed to understand RSA involve modulus operations and Euler's totient.

Leonard Euler was a pivotal figure in the history of mathematics. He made very significant contributions to number theory and is widely considered to be the father of graph theory. Some of his work in number theory forms part of the RSA algorithm. Specifically, Euler's totient is an important part of the RSA key generation algorithm. Euler's Totient or simply the totient is a term that denotes all the integers smaller than n that have no common factors with n. Another way of stating this is how many numbers smaller than n are co-prime with n. The symbol for the totient of a number is shown in figure 1.

$$\varphi(n)$$

Figure 1: Euler's Totient

For example, if n = 8 then you are asking how many numbers smaller than 8 have no common factors with 8. The answer would be 3, 5, and 7. 1 is also included as a special case, thus the totient of 8 would be 4. Another term for Euler's totient is the Euler phi function.

The next question Euler was the totient for prime numbers. If, for example, n = 7 then all positive integers smaller than n are co-prime to n. So, the totient of 7 is 6. Put more generally, the totient of any prime number p is p-1. Further, Euler proved that if you have two prime numbers (m and n) and multiplied them together to get k, the totient of that k, was simply the totient if m * the totient of n. Put another way, the totient of k is (m-1)(n-1).

In addition to Euler's totient, the modulus operation is a key mathematical component of the RSA algorithm. Modulus operations are important in cryptography, particularly in RSA. Let us begin with the most simple explanation, then later delve into more details. To use the modulus operator, simply divide A by N and return the remainder.

So, 5 mod 2 = 1
So, 12 mod 5 = 2

This explains how to use the modulus operator, and in many cases this is as far as many programming textbooks go. But this explanation only shows how one might use the modulus operation in programming, it does not explain what modulus arithmetic is. One way to think about modulus arithmetic is to imagine doing any integer math you might normally do, but bound your answers by some number. A classic example is the clock. It has numbers 1 through 12, so any arithmetic operation you do has to have an answer that is 12 or less. If it is currently 4 o'clock and I ask you to meet me in 10 hours, simple math would say I am asking you to meet me at 14 o'clock. But that is not possible, our clock is bounded by the number 12. The answer is simple, take your answer and use the mod operator with a modulus of 12 and look at the remainder.

14 mod 12 = 2

Thus far I have used the = symbol, however, this is not precisely accurate. An infinite number of mod operations would yield the answer 2. For examples:

26 mod 12 = 2
38 mod 12 = 2

In modular arithmetic, the equal symbol is not used. Rather the congruence symbol is. So, the mathematically correct way to write this is:

14 mod 12 ≡ 2
26 mod 12 ≡ 2
38 mod 12 ≡ 2

While the basic concepts of modular arithmetic dates back to Euclid who wrote about modular arithmetic in his book Elements, the modern approach to modular arithmetic was published by Carl Gauss in 1801.

III.THE ALGORITHM

Now that you understand the essentials of Euler's totient, and modular arithmetic, you are ready to examine the RSA algorithm. The Actual steps in the algorithm are not overly complicated. Let's look at them here:
Step 1: Generate two large random primes, p and q, of approximately equal size such that their product n = pq is of the required bit length (such as 2048 bits, 4096 bits, etc.)
Step 2: Let n = pq
Step 3: Let m = (p-1)(q-1)

Recall that p-1 is the totient of p and q -1 is the totient of q. Further recall that multiplying the totient of p by the totient of q yields the totient of the product of pq.
Step 4: Choose a small number e, co-prime to m (note: Two numbers are co-prime if they have no common factors.)
Step 5: Find d, such that de mod m ≡1. The modulus operator, if you are unfamiliar with it means to divide a number and select the remainder. So, we are looking for a number d, then when multiplied by e and divided by m gives 1 as a remainder. And that is it, you are done, now just publish e and n as the public key. Keep d and n as the secret key.

It can be useful to examine an example. In the following example, I will use very small prime numbers. Real implementations of RSA use integers much larger than these, but these are chosen because they are small enough that calculations can easily be done without the aid of a computer, making the RSA algorithm more understandable.

1. *Select primes: p=17 & q=11*
2. *Compute n = pq =17×11=187*
3. *Compute ø(n)=(p–1)(q-1)=16×10=160*
4. *Select e : gcd(e,160)=1; choose e=7*
5. *Determine d: de=1 mod 160 and d < 160 Value is d=23 since 23×7=161= 10×160+1*
6. *Publish public key (7 and 187)*
7. *Keep secret private key 23*

Now with the keys generated, it is a trivial matter to apply these keys to encryption and decryption. Utilize the number 3 as the plain text. Remember e =7, d=23, and n =187
Cipher text= Plain text mod n

Cipher text = 37 mod 187
Cipher text =2187 mod 187
Cipher text = 130
This produces the cipher text of 130. The private key can then be used to decrypt, as shown below.
Plaintext = Cipher text d mod n
Plaintext = 13023 mod 187
Plaintext = 4.17539054134131116367045797e+48 mod 187
Plaintext = 3
If one attempts to decrypt with the public key (7 and 187) rather than the private key (23) the cipher text will not be decrypted, as shown below.
Plaintext = Cipher text e mod n
Plaintext = 1307 mod 187
Plaintext = 627485170000000 mod 187
Plaintext = 37
This is a relatively simple example, but it illustrates the essence of asymmetric cryptography in general and RSA in particular.

IV. ISSUES WITH RSA

This algorithm has been used effectively for asymmetric cryptography since its publication in 1977. It is widely used today in TLS and other applications. But it is reasonable to ask if the algorithm is still the most appropriate choice. Digital certificates using RSA have been using larger and larger key sizes in order to compensate for smaller key sizes being compromised (Easttom, 2015). Rabah (2006) gave a scathing review of many widely used cryptographic algorithms, including RSA. He argued that these methods have exceeded their useful lifespan and replacements such as Elliptic Curve should be considered. The author provided a critical view of current cryptography modalities. He posited that traditional algorithms are actually now inadequate for modern cryptography needs, and that is it necessary to consider stronger encryption methodologies:

Over the last three decades the traditional cryptosystems like DES, DLP, RSA, DSA, etc. have thus far been the answer to a wide range of issues that impact modern communication including the assurance of privacy, the certainty of a transmitter or receiver's identity, and the integrity of the communication. Today, these traditional crypto-algorithms which where once considered effective have become impractical in light of recent technological development of constrained environment devices (p 204). Rabah (2006) posits that current algorithms are no longer adequate. But Rabah does not specifically address RSA. Recent studies have discovered potential flaws in RSA (Hinek & Lam, 2009; Aciicmez & Schindler, 2008; Swenson, 2008; Zhao & Qi, 2007; Yelik, Rescorla, Shacham, Enright, & Savage, 2009).

Heninger and Shacham (2009) found that RSA implementations that utilized a smaller modulus were susceptible to cryptanalysis attacks. In their study, they considered RSA implementations that utilized a small exponent in the algorithm. A smaller modulus is sometimes used to increase the efficiency of the RSA algorithm. However, the size of the modulus value also could be used to reduce the set of possible factors, and thus decrease the time required to factor the public key. In fact, a great many RSA implementations use e = 216 + 1= 65537. So, a cryptanalysis already has the public key and thus has e and n. And the n is relatively small, making it possible, with extensive computing power and time, to derive the private key. The authors of this study clearly showed that it is possible to derive the private RSA key, which would render that particular RSA encryption implementation useless.

In their methodology Heninger and Shacham (2009) formulated a series of linear equations that would progressively approximate the RSA private key. The approximations were based approximations of factoring the public key. This technique is very similar to the linear cryptanalysis method for cryptanalysis of symmetric key algorithms (Su, Wu, & Zhang, 2011; Alekseychuck, Kovalchuk, & Pal'chenko, 2007; Keliher, 2007; Swenson, 2008)

Zhao and Qi (2007) also utilized implementations that have a smaller modulus operator. The authors of this study also applied modular arithmetic, a subset of number theory, to analyzing weaknesses in RSA. Many implementations of RSA use a shorter modulus operator in order to make the algorithm execute more quickly. Like Heninger and Shacham (2009), Zhao and Qi (2007) showed that, based on the mathematical relationships between the elements of the RSA algorithm, that increases in efficiency resulting from a smaller modulus, will also render a decrease in the efficacy of that RSA implementation. In their study, Zhao and Qi (2007) utilized a lattice matrix attack on the RSA implementation in order to attempt to factor the public key and derive the private key. The specifics of this mathematical methodology are not relevant to this paper. What is significant is that the researchers used a different approach than Heninger and Shacham (2009) and achieved the same results on RSA applications using a small modulus.

Aciicmez and Schindler (2008) examined the RSA Cryptographic algorithm, as implemented in SSL. SSL, or Secure Sockets Layer, is the protocol used to provide encryption on websites (Mao, 2011; Stallings, 2010; Stanoyavich, 2010).

While the actual protocol has since been supplanted by a newer protocol, TLS (Transport Layer Security), many sources still use the term SSL (Stallings, 2010).
Given that SSL/TLS is used for online banking and e-commerce (Ahmed, 2010; Constantinescu, Boldea, & Boboila, 2010; Aïmeur, Brassard, & Onana, 2006), the security of any implementation of the protocol is an important topic. Aciicmez and Schindler (2008) wanted to understand if there were flaws in the implementation that would allow an unintended third party to break the SSL implementation. The authors explained how a particular type of crypto-analysis can be used to break this particular specific implementation of RSA. Since RSA and SSL are both used extensively in e-commerce, exploring weaknesses in either is important. It is important to note that this analysis was dependent upon essential elements of number theory. In their study of SSL using RSA, Aciicmez and Schindler (2008) examined the timing for modular arithmetic operations used in that specific implementation of RSA. This ultimately led to a method for factoring the public key, thus yielding the private key used in that RSA implementation.

This methodology is important because normal approaches to factoring the public key are entirely too time consuming to be of practical use (Swenson, 2008). It is important to derive some additional information about the implementation of RSA in order to attempt a more practical approach to factoring. By utilizing number theory, specifically in respect to the functionality of modular arithmetic, the researchers were able to significantly decrease the time required for factoring the public key.

Given that the authors of this study were able to significantly decrease the time required to factor the public key in order to derive the private key (Aciicmez & Schindler, 2008), clearly these findings are significant. The study clearly shows a problem with some implementations of RSA.

These studies mentioned are simply a sample of known attack vectors against RSA. Many of these attacks depend on a small modulus. And we have seen that many RSA implementations utilize the same small modulus ($e = 216 + 1 = 65537$). It is also true that increases in computing power will make these attacks, as well as brute force attempts to crack RSA, even more practical. So far the cryptography community has reacted by simply using ever larger key sizes for RSA. It seems likely that an entirely new asymmetric algorithm may be needed. But in the meantime, when you implement RSA make sure you not only use a large key size, but be wary of using too small a modulus value.

V. CONCLUSIONS

RSA has been an effective asymmetric algorithm for 30 years. However, there is a growing body of evidence that RSA is no longer the best choice for modern asymmetric applications. There exist other options, such as Elliptic Curve Cryptography. However, there are indications that Elliptic Curve Cryptography may have issues of its own (Mimoso, 2015). In the immediate future, the answer is to use larger RSA keys with more carefully chosen modulus operators. However, the long-term solution is to find a more robust cryptographic algorithm.

REFERENCES

[1]. Aciicmez, O. Schindler, W. (2008). A vulnerability in RSA implementations due to instruction cache analysis and its demonstration on openssl: Proceedings of the 2008 cryptographer's' track at the RSA conference on topics in cryptology. Retrieved from http://portal.acm.org/citation.cfm?id=1791688.1791711&coll=DL&dl=GUIDE&CFID=9605139&CFTOKEN=26457223.
[2]. Aïmeur, E., Brassard, G. , Serge Mani Onana , A. (2006). Blind electronic commerce**.** Journal of Computer Security, 14 (6).
[3]. Alekseychuck, A., Kovalchuk, L., Pal'chenko, S., (2007). Cryptographic parameters of s-boxes that characterize the security of GOST-like block ciphers against linear and differential cryptanalysis. Zakhist Inform, 2, 12-23.
[4]. Chen, C., Wang, T. (2011). A new and extended fault analysis on RSA. ASIACCS '11 Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, 466-470. doi:10.1145/1966913.1966980.
[5]. Constantinescu, N., Boldea, C., Boboila, C. (2010). Elliptic curves cryptosystems for ecommerce applications. Proceedings of the 11th WSEAS international conference on mathematics and computers in business and economics, 216-221. Retrieved from http://portal.acm.org/citation.cfm?id=1863402.
[6]. Easttom, C. (2015). Modern Cryptography: Applied Mathematics for Encryption and Information Security. New York City, NY: McGraw-Hill Publishing
[7]. Heninger, N., Shacham, H. (2009). Reconstructing RSA private keys from random key bit. Advances in Cryptology Lecture Notes in Computer Science, 1 (1). doi:10.1007/978-3-642-03356-8_1.
[8]. Hinek, M., & Lam, C. (2009). Common modulus attacks on small private exponent RSA and some fast variants (in practice). Journal of Mathematical Cryptology, 4 (1).
[9]. Hinek, M. (2009). Cryptanalysis of RSA and its variants. England: Chapman and Hall.
[10]. Kleinjung, K., Aoki, K., Franke, J., Lenstra, A., Thomé, E., Bos, J., …

[11]. Osvik, D. (2010). Advances in Cryptology – CRYPTO 2010. Lecture Notes in Computer Science, 6223(2010), 333-350. doi:10.1007/978-3-642-14623-7_18.
[12]. Ling, Y., Xiang, Y., Wang, X. (2008). RSA-based secure electronic cash payment system. Industrial Engineering and Engineering Management, 1898-1902. doi: 10.1109/IEEM.2007.4419522.
[13]. Mao, W. (2011). Modern cryptography: Theory and practice. Upper Saddle River, New Jersey: Prentice Hall.
[14]. Mimoso, M. (2015). NSA's Divorce from ECC Causing Crypto Hand-Wringing. Retrieved from https://threatpost.com/nsas-divorce-from-ecc-causing-crypto-hand-wringing/115150/
[15]. Rabah, K. (2006). Elliptic curve cryptography over binary finite field gf. Information Technology Journal, 5 (1) 204-229. Retrieved from http://docsdrive.com/pdfs/ansinet/itj/2006/204-229.pdf.
[16]. Rizvi1, S., Wadhwa, N. (2010). Cryptography and mathematics. Proceedings of the 4th National Conference; INDIACom-2010 Computing for Nation Development.
[17]. Skurnick, R., Javadi, M. (2009). An alternative proof of the infinitude of the prime numbers. Mathematics and Computer Education, 43(3), 248.
[18]. Stallings, W. (2010). Cryptography and network security: Principles and practice. Saddle Brook, New Jersey: Prentice-Hall.
[19]. Stanoyavich, A. (2010). Introduction to cryptography with mathematical foundations and computer implementations. London, England : Chapman and Hall.
[20]. Su, B., Wu, W., Zhang, W. (2011). Security of the SMS4 block cipher against differential cryptanalysis. Journal of Computer Science and Technology, 26(1),130 -138. doi: 10.1007/s11390-011-1116-9 Swenson, C. (2008). Modern cryptanalysis: Techniques for advanced code breaking. Hoboken, New Jersey: Wiley.
[21]. Yelik, S., Rescorla, E., Shacham, H., Enright, B., Savage, S. (2009). When private keys are public: Results from the 2008 Debian OpenSSL vulnerability. Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference**.** Retrieved from http://portal.acm.org/citation.cfm?id=164489. doi: 10.1145/1644893.1644896
[22]. Zhao, Y., Qi, W. (2007). Small private-exponent attack on RSA with primes sharing bits. Lecture Notes in Computer Science, 2007, 4779 (2007) 221-229. doi: 10.1007/978-3-540-75496-1_15