Occidental College

Number Theory

Secure Communications using RSA Encryption

Dylan Morison

November 14, 2020

Abstract

One of the most prominent aspects of modern technology is the Internet. The ability for computers to communicate on such a wide scale has made data, information, and knowledge widely accessible to people world wide. Yet with all the positive attributes of the Internet comes great risk. The amount of sensitive, personal, and private information that people store on the internet is massive and therefore must be kept protected from unwanted possession. RSA Encryption is one of many components of the internet that allows people to send sensitive data securely. Invented in the late 1970s by mathematicians and engineers, RSA encryption is an algorithm that conceals or *encrypts* data such that data can be transmitted securely and decrypted at the destination. The RSA algorithm would not be possible without two important theorems derived from Number Theory: the Euler phi-function, and Euler's Theorem.[1]

1 Introduction

Before the birth of RSA Encryption in the 1970s, the world of cryptography revolved around using the same key to encrypt and decrypt information, defined as *symmetric key cryptography.* [2] It became very apparent that symmetric key encryption and decryption was widely impractical because people had to send a unique key to every person they are communicating with. (Note: we will use common cryptography convention and use Bob and Alice to denote two parties trying to securely communicate). RSA Encryption solves the issue of symmetric key encryption by getting rid of the need to have unique keys. More specifically, RSA Encryption allows any individual to only have to keep track of one key. As such, if Bob were to send a secure message to 10 of his friends he could use the same encryption key for each person.

RSA encryption is used world wide because it solves one very important problem very efficiently. That is, it allows two parties to *easily* send secure data and information to each other whilst making it near *impossible* for any eavesdropper to obtain any of the information. RSA encryption makes this possible by breaking the key into two parts: an *encryption key* and a *decryption key.* These two keys are created using *Euler's Theorem*, which finds a relationship between the two keys using a common function in Number Theory known as the Euler phi-function. In section two we will examine the Euler phi-function and Euler's Theorem in detail.

[1]Khan Academy. Accessed November 24, 2019. https://www.khanacademy.org.
[2]Khan Academy. Accessed November 24, 2019. https://www.khanacademy.org.

The last important aspect of RSA encryption to introduce is that of prime factorization. The reason our eavesdropper, Eve, cannot interpret intercepted data sent between Alice and Bob is that the necessary key Eve would need is near impossible for her to find. Without the decryption key, all Eve can see is the encrypted version of the information sent between Alice and Bob. The reason this decryption key is so hard for Eve to obtain is that it depends on the prime factorization of a very large number. However, how can we be sure that Eve cannot find the prime factorization of this very large number? Simply put, it a trivial computation for a computer to multiply two 150 digit numbers together, however it is quite the opposite to ask a computer to find the prime factorization of a 300 digit number. We will explore this idea more in section 3.

2 Required Definitions and Theorems

The only two required definitions is that of modular arithmetic and the Euler Phi Function.

Definition 2.1. *We say a is equivalent to b and $a \equiv b$ mod m if $m|a-b$, where a, b are integers and m is a positive integer.*

Note: all calculations in the RSA algorithm are done using modular arithmetic.

Definition 2.2. *The Euler Phi Function is denoted $\phi(n)$ and is defined by the number of positive integers less than or equal to n that are relatively prime to n*

Remark. *We know by definition if n is prime, then only two numbers can divide n: 1 and n. Thus the only integer x such that $gcd(n, x) \neq 1$ is $x = n$* ***iff*** *n is prime. From this insight, we can observe that there are $n-1$ numbers less than or equal to n that are relatively prime to n, thus $\phi(n) = n - 1$.*

Now for our last theorem: Euler's Theorem. Euler's Theorem is responsible for allowing the core component of RSA encryption's modular exponentiation to work as it does. Moreso, it is used to obtain a relationship between the encryption key and decrpytion key.

Theorem 2.1. *Let a and n be natural numbers and $(a, n) = 1$, then $a^{\phi(n)} \equiv 1$ mod n.*

Proof. Let us denote the system that is made up of integers that are relatively prime and less than n as $\{r_1, r_2, ..., r_{\phi(n)}\}$, also known as the reduced residue system mod n. Similarly, we have $(a, n) = 1$, so we can write $\{ar_1, ar_2, ..., ar_{\phi(n)}\}$, which is also a reduced residue system mod n.

Notice that the set $\{ar_1 \bmod n, ar_2 \bmod n, ..., ar_{\phi(n)} \bmod n\}$ is just $\{r_1, r_2, ..., r_{\phi(n)}\}$ written in a different order because each $r_1, r_2, r_3, ..., r_{\phi(n)}$ is only represented once mod n.

Lastly, we can take the product of all the terms we just listed to obtain: $(r_1)(r_2)(r_3)...(r_{\phi(n)}) \equiv (\mathrm{ar}_1)(ar_2)(ar_3), ...(ar_{\phi(n)}) \equiv \mathrm{a}^{\phi(n)}(r_1 r_2 r_3 ... r_{\phi(n)}) \bmod n$. From here we observe that n is relatively prime to any r_i thus
$\gcd(n, r_1 r_2 r_3 ... r_{\phi(n)}) = 1$ and thus by modular cancellation
$a^{\phi(n)}(1) \equiv 1 \bmod n$. □

We now have a foundation to be able to fully understand RSA Encryption. Next we will discuss the algorithm itself using the number theory terms we've defined and proven.

3 RSA Algorithm Overview

3.1 The General Algorithm

First and foremost, the RSA Algorithm is classified as a 'Trapdoor'function [3], meaning it is very easy to compute in the forward direction, yet very hard to compute in the backwards direction unless necessary 'trapdoor' information is known. Specifically, it is very easy for Alice to compute the encryption and decryption keys given she has access to specific trapdoor information, however it is very hard for Eve to do the same because she does *not* have access to any of the trapdoor information.

3.2 The Trapdoor Computations

We now know that the RSA algorithm is easy to compute in one direction, and that computations in the inverse direction are hard without special 'trapdoor' information. But what are these computations and what gives them this trapdoor property? There are several computations that are done in the forward direction. In the following subsections of *3.2 The Trapdoor Computations* we will go over each specific trapdoor computation. We will start with the first computation: the trivial multiplication of two large prime numbers.

[3]Cryptography I. Accessed November 24 ,2019. https://www.coursera.org/lecture/crypto/the-rsa-trapdoor-permutation-JrGlH

3.2.1 Finding N

We denote the two primes p_1, and p_2 and we assume they are both very large randomly generated prime numbers, say 150 digits long each. We then compute
$N = p_1 p_2$ which we assume to be around 300 digits long.

3.2.2 Finding $\phi(N)$

Calculating $\phi(N)$ is only possible because:

1. $\phi(N)$ is multiplicative
2. We know the prime factorization of N

As such we can write $\phi(N) = \phi(p_1 p_2) = \phi(p_1)\phi(p_2)$ and since we know p_1, and p_2 are prime, we have $\phi(p_1)\phi(p_2) = (p_1 - 1)(p_2 - 1)$. The last trapdoor computation needed is to calculate our encryption and decryption keys, however first we must make some modifications to Euler's Theorem.

3.3 Euler's Theorem Modification

Recall Euler's Theorem: $a^{\phi(n)} \equiv 1 \bmod n$, when $(a, n) = 1$.

1. First let m be a numerical form of some message, and let N be the same N introduced in 3.2.1. We now have $m^{\phi(N)} \equiv 1 \bmod N$ given that $(m, N) = 1$.
2. Next, let k be an integer. We write $m^{\phi(N)\cdot k} \equiv 1 \bmod N$. We know this equation is true for any value k.
3. Finally, we would like to get m on the right hand side of our equation, and we know that $1 \cdot m = m$, thus we have $m \cdot m^{\phi(N)\cdot k} \equiv 1 \cdot m \bmod N$ which can be simplified as: $m^{\phi(N)\cdot k+1} \equiv m \bmod N$

We now have everything we need to finish our trapdoor computations. We will use our modification of Euler's Theorem to get our encryption and decryption keys.

3.4 The Final Trapdoor Computation: Finding d

Let e be our encryption key, and d be our decryption key. Before determining an equation for encryption and decryption, recall what we are looking for in our algorithm. Say Alice wants Bob to be able to send her secure messages so that she can easily decrypt them, yet if our eavesdropper Eve got a hold of the same message, Eve would be unable to decrypt them. Alice would need to find some trapdoor computation such that she can easily compute d while Eve cannot. This sounds exactly like the computation of N! Therefore, all we need to do is to find d in terms of N. *Thus*, let by setting $e \cdot d = \phi(N) \cdot k + 1$ we have done just that! Dividing both sides by e gives us $d = \frac{\phi(N) \cdot k+1}{e}$.

Interestingly, Alice actually does not care if Eve knows the value of e, only that Even does not have a value for d; thus e is Alice's public exponent, while d is her private (trapdoor) exponent. The importance of the equation $d = \frac{\phi(N) \cdot k+1}{e}$ cannot be overstated, and its power comes from the fact that Eve will never be able to find $\phi(N)$ without knowing the factorization of N, and no resource exists that would allow Eve to determine the factorization of N.

3.5 Encryption and Decryption

So far in section 3 we have gone over all trapdoor computations needed for Alice and Bob to communicate. To summarize, we have our public information: N and e, and we have our private information: d and the prime factorization of N. All that is left is to put everything together and examine how RSA Encryption actually encrypts data.

Say Bob wants to securely receive messages from anyone in the world using RSA Encryption. Bob needs to send out his public encryption key to anyone that wants to send him data. This means Bob has to send e and N. Say Alice is one such person needing to send a message, m, to Bob. Alice will take her message and raise it to the e power mod N in order to obtain an encrypted version of the message, denoted c. We thus have $c \equiv m^e \bmod N$. Alice sends c to Bob, and finally Bob can use his private key, d, to undo e as such: $m \equiv c^d \bmod N$. Lastly, we can expect Eve to have been eavesdropping and to have intercepted all information sent between Alice and Bob. However given Eve only has access to the encrypted message c, as well as the public key e and N, we know she will never be able to find message m. Finally, we can be sure Eve will never be able to obtain the decryption key, d, needed decrypt c because she doesn't have the prime factorization of N.

4 RSA Encryption example

Recall that Bob and Alice are two parties trying to send messages to each other securely. Let us assume that Bob wants to securely send the message "cat" to Alice.

1. First Bob encodes "cats" to numerical form using some Unicode format. Note that the specificity of the Unicode format is not important for the implementation of RSA. Let us denote Bob's message as m, and let 151 be its numerical value.

2. Next, Alice generates two random prime numbers, denoted p_1 and p_2. Let $p_1 = 167$, $p_2 = 199$. Under normal circumstances p_1 and p_2 would be 150 digits or larger, but they are shortened here for the sake of simplicity.

3. We then say N $= (p_1)(p_2) = (167)(199) = 33233$, we also say N is our modulus.

4. We now find $\phi(N) = \phi(33233)$. We know $\phi(N)$ is a multiplicative function and since $(167, 199) = 1$ we can compute: $\phi(33233) = \phi(167)\phi(199)$ $= (166)(198) = 32868$

5. Now Alice determines her public exponent e, keeping in mind e must be odd and relatively prime to $\phi(N)$. Let Alice pick $e = 5$.

6. The last variable we need to calculate is the private exponent, d. We set $d = (k * \phi(N) + 1)/e$ where k is any integer that satisfies the equation for d. In this case we have $d = (k * 32868 + 1)/5$ and we see that setting $k = 3$ results in $d = 19721$, which is a whole number as desired.

7. Now it's time for the "trapdoor" aspect of this algorithm, where Alice takes everything other than her public exponent e, and modulus N, and hides them in a figurative "trapdoor", meaning she keeps everything private. As such, we classify e and N as Alice's "Public Key".

8. Alice sends her Public Key to Bob so Bob can encrypt his message with it. Recall that Bob's message is $m = 151$. Bob uses $m^e \bmod N \equiv c$ to encrypt his message, he computes $151^5 \bmod 33233 \equiv 32248 = c$ Bob then sends his encrypted message, c, back to Alice.

9. Finally, Alice decrypts Bob's message using her private key, d. She computes $c^d \bmod n$. We have $32248^{19721} \bmod 33233 \equiv 151$. And thus she is able to securely obtain Bob's message "cats".