NUMBER THEORY IN CRYPTOGRAPHY

JASON JACOBS

Abstract. In this paper, we will discuss some important cryptosystems. This will involve proving why they work as well as discussing potential attacks on them. Number theory is crucial to their existence, and this paper will begin by providing the necessary background in this field to be able to understand the material.

Contents

Date: August 28, 2021.

1. Introduction

Sending messages in secret has been necessary for thousands of years. If two parties want to communicate without a third party knowing what they are saying, they must correspond in a fashion that the third party couldn't understand even if they saw the message. For example, if ally military leaders want to discuss key battle tactics, they cannot risk their foes intercepting and understanding their messages. This overall idea gave rise to the concept of cryptography. Individuals were enlisted to create ciphers in order to encrypt messages. One famous historical technique is the Caesar Cipher, a primitive method of encryption named after Julius Caesar. This is an example of a shift cipher, as its idea is to replace each letter with a different letter by shifting the alphabet a specific number of places (e.g. "at" becomes "bu" if the alphabet is shifted by 1). If this was used for the English alphabet, obviously any number but a multiple of 26 would work (as this would shift a letter back to itself). However, since there are only 25 possible ways to shift the alphabet, this was easily broken by codebreakers. Even though more complex ciphers of the same sort are possible, they are often easily broken by frequency analysis, a technique that uses the frequency of letters in words and attempts to match the most common symbols of the encrypted text to the most common letters in the alphabet (e.g. a circle is the most common symbol in the intercepted message and e is the most common letter in the English alphabet, therefore there is a solid chance that the circle represents e).

Following this process, there has been a race between codemakers and codebreakers for many years. One wants to construct an indecipherable code, and the other will keep attempting to crack the cipher. As math advances, so do the different techniques used to construct ciphers. Overall, this paper will demonstrate that number theory is a crucial component of cryptography by allowing a coherent way of encrypting a message that is also challenging to decrypt. The discussion in this paper follows the set of notes [1] [2] [3] by Evan Dummit.

2. Number Theory Background

2.1. Basic Principles.

We must begin by explaining the math that is useful in cryptography to allow for easier comprehension of specific cryptosystems.

2.1.1. *Divisibility and Prime Numbers.*

Prime numbers are an elementary part of number theory that all readers must understand. First, consider all positive integers besides 1, e.g. 2, 3, 4, etc. We can divide these numbers into two types: prime numbers and composite numbers. However, prior to going into the definition, we first need to explain the statement "a divides b."

Definition 2.1. For any two integers, we say that "a divides b" or "$a|b$" if b is divisible by a. In other words, a divides b if $b = ac$ for some integer c.

Example 2.2. $4 \mid 12$, since $12 = 4(3)$.

Example 2.3. $8 \mid 56$, since $56 = 8(7)$.

Now, we can explore the idea of prime numbers and composite numbers.

Definition 2.4. An integer $n \geq 2$ is prime if the only positive integers that divide n are 1 and n.

Definition 2.5. An integer n is composite if more than two positive integers divide n.

To clarify, every positive integer besides 1 is either prime or composite, as it will always be divisible by at least 1 and itself.

2.1.2. *Modular Arithmetic.* We will next discuss a part of number theory that has played a role in a vast array of ciphers: modular arithmetic. To understand modular arithmetic, picture a clock. The maximum number is 12, and no number is larger than that. If one were to reference 5 hours after 12, they would not be referencing 17, as there is no 17 on the clock. They would be talking about 5. This is the idea of modular arithmetic, and this is what we will call "modulo 12."
We define modular arithmetic formally as follows:

Definition 2.6. We say that $a \equiv b \pmod{m}$ if m divides $a - b$.

In arithmetic modulo (or "mod") 12, all numbers are equivalent to some number in the ranges 0-11 or 1-12. If we were speak about 20 hours after 6, we would not be referring to 26, but instead be talking about 2. To reduce a large number to a smaller number modulo 12, we repeatedly subtract 12 from that number until we arrive at a number between 0 and 11. [1]

Additionally, the following are some (but not all) arithmetic rules which still apply:

If $a \equiv c \pmod{n}$ and $b \equiv d \pmod{n}$, then $a + b = c + d \pmod{n}$ and $ab = cd \pmod{n}$.

Example 2.7. (1) 10 is congruent to 2 modulo 4, because $10 - 2 = 8$, which is a multiple of 4.

(2) $127 \equiv 13 \pmod{19}$, because $127 - 13 = 114 = 6 \cdot 19$.

The Caesar Cipher from the introduction can be described more succinctly using arithmetic modulo 26. If one wants to shift all letters by 3, then the easy way to accomplish this is the following:

(1) Convert all letters into numbers, with a being 0, b being 1, etc., with z eventually representing 25.
(2) Add 3 to each number, ensuring that one uses modular arithmetic here. For instance, to encrypt c, one uses (2+3) mod 26 = 5 mod 26 = 5.
(3) Convert each number back into a letter. Now, c is represented by f, y is represented by b, etc., and z is represented by c. We have our new alphabet.

2.2. **Definitions and Theorems to Know.**

2.2.1. *Definitions and Theorems.* We should also express the following definitions and theorems before we begin to discuss cryptography.

Definition 2.8. Two positive integers a and b are relatively prime if there does not exist a positive integer c greater than 1 such that $c|a$ and $c|b$.

Theorem 2.9. *Chinese Remainder Theorem: Let m_1, m_2, ..., m_k be relatively prime positive integers such that the greatest common divisor of m_i and m_j is 1 when $i \neq j$. Also let a_1, a_2, ..., a_k be arbitrary integers. Then there exists an integer a such that the set of values x satisfying the equations*

$$\begin{aligned} x &= a_1 \pmod{m_1} \\ x &= a_2 \pmod{m_2} \\ &\vdots \\ x &= a_k \pmod{m_k} \end{aligned}$$

consists of those integers x congruent to a modulo $m_1 m_2 ... m_k$. Essentially, this system of equations has a unique solution modulo $m_1 m_2 ... m_k$.

Proof. See [2]. □

Definition 2.10. We define $\varphi(n)$ as the number of integers between 1 and n, inclusive, that are relatively prime to n. This function is known as Euler's totient function.

Example 2.11. $\varphi(7) = 6$.
The numbers between 1 and 7, inclusive, that are relatively prime to 7 are 1, 2, 3, 4, 5, and 6. It is important to note here that 7 is prime and $\varphi(7) = 6$, which is $7 - 1$. More generally, $\varphi(p) = p$ - 1 for every prime number p, as every number less than p shares no factors with p besides 1 and is thus relatively prime to p.

Lemma 2.12. *If $N = pq$ where p and p are prime numbers, then $\varphi(N) = \varphi(p) \cdot \varphi(q)$.*

Proof. By the definition, we know that $\varphi(N)$ will tell us the number of integers between 1 and N (inclusive) that are relatively prime to N. We also know that two integers are relatively prime if no positive integers greater than 1 divide both of them. We can picture N as the prime number p which is then multiplied by the other prime q. As a result, N only has one more positive divisor than p (which is q), as q is only divided by 1 and itself. Therefore, only 4 numbers divide N: 1, p, q, and N.

We can conceptually think about $\varphi(N)$ as follows: $\varphi(N)$ will not include p, q, and all the multiples of p and q up to and including N, as those will share a common factor with N (either p or q). There are precisely q multiples of p up to N, and there are precisely p multiples of q up to N. Since we only multiplied p and q together once, there is no overlap except for N, which we double counted. Thus, $\varphi(N) = N - p - q + 1 = pq - p - q + 1 = (p-1)(q-1) = \varphi(p) \cdot \varphi(q)$. □

Definition 2.13. The inverse of x modulo m is some number y that satisfies $xy = 1 \pmod{m}$. If x has an inverse modulo m, we say that x is a unit modulo m.

Example 2.14. Suppose $x = 5$ and $m = 19$. Take $y = 4$. Then, $xy = (5)(4) = 20 \equiv 1 \pmod{19}$. Therefore, 5 is a unit modulo 19.

Note that an inverse does not always exist. In fact, the inverse of a modulo m only exists if a is relatively prime to m.

Definition 2.15. Suppose b is a unit modulo m. The order of b is the smallest integer $e > 0$ such that $b^e \equiv 1 \pmod{m}$.

Example 2.16. Consider $b = 2$ and $m = 7$.
$2^1 = 2$, which is congruent to 2 mod 7.
$2^2 = 4$, which is congruent to 4 mod 7.
$2^3 = 8$, which is congruent to 1 mod 7.
Thus, the order of 2 is 3.

Definition 2.17. We say that a is a primitive root modulo m if a is a unit modulo m and the order of a is $\varphi(m)$.

Example 2.18. Since 5 is prime, we know that $\varphi(5) = 5 - 1 = 4$. Additionally, 3 is a unit modulo 5 since 7 satisfies $3(7) = 21 \equiv 1 \pmod 5$. The order of 3 mod 5 is 4, since $3^1 = 3 \equiv 3 \pmod 5$, $3^2 = 9 \equiv 4 \bmod 5$, $3^3 = 27 \equiv 2 \pmod 5$, $3^4 = 81 \equiv 1 \pmod 5$. Thus, since 3 is a unit modulo 5 and the order of 3 is 4, which is $\varphi(5)$, 3 is a primitive root modulo 5.

Theorem 2.19. *Fermat's Little Theorem: Suppose a is an integer. If p is prime, then $a^{p-1} \equiv 1$ (mod p) if p is prime.*

Proof. See [2]. □

3. RSA Encryption

3.1. Background.

3.1.1. *Terms to Know.*

We are about to discuss one of the most popular and well-known cryptosystems: RSA encryption (this acronym originates from the last names of its creators). This is a method of encryption that originated in 1977 and is still used today. For example, RSA is used for digital signatures on documents [4]. Before explaining RSA encryption, we first need to establish a few terms that are necessary to understand.

(1) Cryptosystem: A cryptosystem is a general term that describes the entire process for encrypting and decrypting a message.
(2) Key: This is a "string of bits used by a cryptographic algorithm to transform plain text into cipher text or vice versa" [5]. For instance, if we were to use a Caesar Cipher and shift all letters over 3 places in the alphabet, our key would be the number 3.
(3) Alice, Bob, and Eve: The names of three hypothetical individuals used to describe a situation involving a cryptosystem. Alice and Bob want to communicate in private, and Eve desires to intercept their message. Sometimes the name Carol is also used if we depict a situation where three parties want to communicate secretly.
(4) Asymmetric versus Symmetric Cryptosystems: Asymmetric, or Public-Key, cryptography is when the key is not kept secret. Symmetric cryptography is when the same key is used for encryption and decryption and therefore should only be known by Alice and Bob. For instance, if Alice and Bob were to use a Caesar Cipher, that is an example of Symmetric cryptography, as if Eve knows the key, she will be able to decrypt any message that Alice and Bob send to one another.

3.1.2. *Introduction to RSA Encryption.*

We are finally ready to proceed with learning about RSA Encryption. If Alice wants to send a secure message to Bob under RSA Encryption, they proceed as in [2]:

(1) Bob makes 3 choices:
   a) He picks some prime number p
   b) He picks some other prime number q, and computes $N = pq$.
   c) He picks some positive integer e that is relatively prime to $\varphi(N)$.
(2) Bob releases the values of N and e.
(3) Alice writes her message and then converts it to some number m modulo N by a process which she and Bob have previously discussed (not necessarily in secret).
(4) Using the formula $c \equiv m^e \bmod N$, Alice determines the value of c.
(5) Alice sends the number c to Bob.
(6) Bob receives the number c and needs to find m.
(7) Bob finds an inverse d of e modulo $\varphi(N)$.
(8) Bob finds m by computing $c^d \pmod N$.

We would like to clarify some parts of the process for the readers. First, e in practice is often 3 and p and q are virtually always very large numbers. We are going to demonstrate the necessity of this shortly, but the reason why is security; if Eve is able to factor N, then she will have all the information that Bob possesses and will easily be able to decrypt Alice's message. For this reason, RSA numbers are generally between 1024-2048 bits, which means often more than $2^{1024} - 1$ [6]. Next, recall that $\varphi(x) = x - 1$ if x is prime. Additionally, we know from Lemma 2.12 that if $N = pq$ with p and q being prime, then $\varphi(N) = \varphi(p) \cdot \varphi(q)$. Therefore, $\varphi(N) = (p-1)(q-1)$. Also, remember that Bob releases the values of N and e; this is why RSA is a public-key cryptosystem. We see that Alice, Eve, and everyone else can find out both N and e.

Having described the process, we need to show why it works.

3.1.3. *Proof that RSA works.* We need to prove that $m \equiv c^d \pmod N$. Remember that m is the secret message, $c \equiv m^e \pmod N$, d is the inverse of e modulo $\varphi(N)$, and $N = pq$ (where p and q are two prime numbers). Additionally, remember that d exists because e is relatively prime to $\varphi(N)$.

We want to show that $m \equiv c^d \pmod p$ and $m \equiv c^d \pmod q$, since this will let us apply the Chinese Remainder Theorem to guarantee that $m \equiv c^d \pmod N$. Keep in mind that the other condition for the Chinese Remainder Theorem to apply is met: p and q are relatively prime because they are prime numbers. Therefore, p only has 1 and p as its divisors, and q only has 1 and q as its divisors. Thus, the only number that divides both p and q is 1, and therefore they are relatively prime.

We are only going to show the p case for the sake of brevity, as the q case is virtually identical (one just has to replace the "p"s with "q"s).

It's time to simplify some terms so we can see if the RSA decryption process works. First, we will examine $c^d \bmod p$.

We know that $c \equiv m^e \pmod p$. Therefore,

$$c^d \equiv (m^e)^d \equiv m^{ed} \pmod p.$$

Next, we can simplify d. We know that d is the inverse of $e \bmod \varphi(N)$. Thus, by the definition of an inverse, d is the number that satisfies $ed \equiv 1 \pmod{\varphi(N)}$. This means that $ed - 1 = k\varphi(N)$ for some k, hence $ed - 1 = k(p-1)(q-1)$. In particular, $ed = 1 + k'(p-1)$ for $k' = k(q-1)$. Then, $m^{ed} = m^{1+k'(p-1)} = m \cdot (m^{p-1})^{k'}$. Now, we will proceed to the crux of the proof. It centers around Fermat's Little

Theorem, which if we recall, states that if x is prime, then

$$a^{x-1} = 1(\mathrm{mod}\, x).$$

In the event that $m \equiv 0 \pmod{p}$, then $c^d \equiv (m)(m^{p-1})^{k'} \equiv 0 \equiv m \pmod{p}$, and we are done. In the more likely scenario that $m \neq 0 \pmod{p}$, then we need Fermat's Little Theorem. Since p (and also q, which is relevant for that case) is prime, $m^{p-1} \equiv 1 \pmod{p}$. Therefore, $c^d \equiv (m)(m^{p-1})^{k'} \equiv m \pmod{p}$, and we are done.

To stress once again, the same steps can be repeated to prove the q case.

Since we have proven that $m = c^d \pmod{p}$ and $m = c^d \pmod{q}$,
we can use the Chinese Remainder Theorem, which tells us that $m = c^d \pmod{\mathrm{N}}$. Thus, the RSA decryption process is mathematically legitimate. [2]

3.1.4. *Sample RSA Process.* We are going to determine c and then determine m from c.

Let $m = 25$, $N = 187$, and $e = 3$. Bob chose 11 and 17 for his prime numbers, but Alice does not know that. Also, m and N would traditionally be significantly larger, but this is for ease of understanding.

To encode this, recall that Alice needs to determine c:

$$c \equiv m^e(\mathrm{mod}\, N) \equiv 25^3(\mathrm{mod}\, 187) = 15625(\mathrm{mod}\, 187) = 104.$$

We are done encoding the message. Now, we are going to decode it. We need to determine $e \bmod \varphi(N) = 3 \bmod ((11\text{-}1)(17\text{-}1)) = 3 \bmod ((10)(16)) = 3 \bmod 160 = 3$. Next, we need to find d. Remember that d is the inverse of 3, which is the number that satisfies $d(3) \equiv 1 \pmod{160}$. Since $160 - 3(53) = 1$, we can determine that $3 \cdot (-53) = 1 + 160 \pmod{160}$, so $3 \cdot (-53) \equiv 1 \pmod{160}$. Thus, $d =$ -53 mod $160 = 107$.

Lastly, we can compute m. Recall that $m = c^d \bmod N$, so $m = 104^{107} \bmod 187 = 25$.

As a whole, we can see why RSA is thought to be secure. It is extremely challenging to "undo" modular exponentiation, especially when the numbers become large.

3.2. **Attacks.**

3.2.1. *General.* Suppose Eve has intercepted the message and knows Bob's public key. Therefore, she has the values N, e, and c, and now she has to solve for m. She has to figure out m from $c = m^e \bmod N$.

3.2.2. *Factoring.* One seemingly obvious yet necessary to mention method for trying to break RSA is factoring. If Eve is able to factor N and obtain p and q, then she can determine m in the same fashion that Bob does. Now, the question is of course how Eve can factor N.

The reader might assume that a brute-force attack could work, as there are obviously much fewer prime numbers than composite numbers. However, remember that N is an incredibly large number. It is effectively impossible for a human to stage a brute-force attack, and so it is irrelevant to discuss any further. Something more relevant and interesting for the future is a technological brute-force attack. Computers could try and brute-force the solution significantly quicker than a human. Surprisingly enough, this is currently believed to be infeasible. Mathematicians speculate that an incredibly powerful computer can factor a 1048 bit

RSA key. Some RSA keys are of that length, but many are multiple times more than 1048 bits and thus impossible to factor at the moment. [2]

3.2.3. *Hastad's Attack.* Hastad's Attack is an excellent example of why $e = 3$ should not be used in some circumstances.

Assume that $e = 3$ and Alice sends a message to three different people with three different public keys. These three keys are $(N_1, 3)$, $(N_2, 3)$, and $(N_3, 3)$.

Eve can then use the Chinese Remainder Theorem to break RSA encryption by solving the following equations:

$$C = c_1 (\text{mod } N_1)$$
$$C = c_2 (\text{mod } N_2)$$
$$C = c_3 (\text{mod } N_3)$$

to find a residue C modulo $N_1N_2N_3$. Note that c_1, c_2, and c_3 are the three different encryptions.

Remember that Eve knows all of those above numbers (besides C, obviously). Then, Eve knows that $C \equiv m^3 \pmod{N_1N_2N_3}$. We know that $0 \leq m < N_1$, N_2, and N_3. Thus, $0 \leq m^3 < N_1N_2N_3$, and it also is the case that $0 \leq C < N_1N_2N_3$. Since we know that C is congruent to m^3, it means that $C = m^3$. Finally, since Eve has solved for C and $C = m^3$, Eve can determine m by taking the cube root of C.

The above applies only if N_1, N_2, and N_3 are relatively prime, but if they aren't, then the cryptosystem is quite easy to decrypt. Eve can just determine the greatest common divisor of two of the keys and thus receive a factorization of each. [2]

What Hastad's Attack overall demonstrates is that we should not send the same message to many different people if they are all using the same small value of e.

4. Diffie-Helman Key Exchange

4.1. **Background.**

4.1.1. *Introduction to Diffie-Helman Key Exchange.* RSA encryption is arguably the gold standard for modern-day cryptosystems [7]. Asymmetric encryption is more logical than symmetric encryption, as Alice and Bob don't need to worry about Eve intercepting the key if they use the former method. For symmetric encryption, Eve finding out the key is disastrous and ruins the cryptosystem. However, though asymmetric encryption is obviously more secure, there is the issue of time. It takes quite a bit of time to both encrypt and decrypt long messages under an RSA system. As a result, symmetric encryption, while inferior, is the more practical approach. [3]

However, the story luckily does not end here. There is a way to approach asymmetric levels of security while still maintaining efficiency. One can use an asymmetric system to agree on a key, and then use a symmetric cryptosystem to send a message using this key. This makes a lot of sense, as it has many of the benefits of asymmetric encryption but is still efficient, as the key is obviously a small message.

This is effectively the idea of Diffie-Helman Key Exchange. It is a method for two parties to agree on a key in an asymmetric fashion, which they can then use to exchange messages with a symmetric cryptosystem. [3]

We will now proceed with showing how Diffie-Helman Key Exchange works. Here is the process as articulated by [3], which is fairly straightforward yet extremely effective:

(1) Alice and Bob publicly agree on two values: a prime number, p, and g, a primitive root modulo p.
(2) Alice privately picks an integer a.
(3) Bob privately picks an integer b.
(4) Alice computes $g^a \bmod p$.
(5) Bob computes $g^b \bmod p$.
(6) The two publicly send their results to each other.
(7) Alice and Bob determine the key by taking $g^{ab} \bmod p$, which just entails raising the number they received from the other to the power of their individual number (a for Alice, b for Bob).

Another useful property of Diffie-Helman Key Exchange is that it can include more than two parties. For instance, assume that Alice and Bob want to include Carol. The process is quite similar to the two-person process:

(1) Alice, Bob, and Carol publicly agree on two values: p and g. Again, p is a prime number, and g is a primitive root modulo p.
(2) Alice privately picks an integer a.
(3) Bob privately picks an integer b.
(4) Carol privately picks an integer c.
(5) Alice computes $g^a \bmod p$.
(6) Bob computes $g^b \bmod p$.
(7) Carol computes $g^c \bmod p$.
(8) All of these results are made public.
(9) Alice computes $g^{ab} \bmod p$.
(10) Alice computes $g^{ac} \bmod p$.
(11) Bob computes $g^{bc} \bmod p$.
(12) All of these numbers are made public.
(13) Alice determines the key by taking $g^{bc} \bmod p$ and raising it to the power of a.
(14) Bob determines the key by taking $g^{ac} \bmod p$ and raising it to the power of b.
(15) Carol determines the key by taking $g^{ab} \bmod p$ and raising it to the power of c.

4.1.2. *Sample Diffie-Helman Process.* We are going to illustrate an example of using Diffie-Helman Key Exchange. Alice and Bob decide that $p = 5$ and $g = 3$. Again, p would be larger in a real-world scenario. Remember from Example 2.18 that 3 is a primitive root modulo 5.

Next, Alice picks $a = 2$ and determines that $g^a \pmod p = 3^2 \pmod 5 \equiv 9 \equiv 4 \pmod 5$.

At the same time, Bob picks $b = 4$ and determines that $g^b \pmod p = 3^4 \pmod 5 \equiv 81 \equiv 1 \pmod 5$. Alice and Bob send these numbers to each other. Alice computes $g^{ab} \pmod p = 4^4 \pmod 5 \equiv 256 \equiv 1 \pmod 5$. Bob computes $g^{ab} \bmod p = 1^2 \pmod 5 \equiv 1 \equiv 1 \pmod 5$.

Now, Alice and Bob both know that the key is 1. Notice that they both received the same answer.

4.1.3. *Proof that Diffie-Helman Works.* This proof is quite easy. Suppose that there are m people and person n (where $n \leq m$) needs to find the key. They have received $g^{a_1 a_2 \ldots a_{n-1} a_{n+1} \ldots a_m}$, and raising that number to the power of a_n will yield $g^{a_1 a_2 \ldots a_{n-1} a_n a_{n+1} \ldots a_m}$ (which they reduce mod p) and therefore the secret key. This is considered to be a secure form of encryption, as a and b are kept private and figuring them out is extremely challenging.

4.2. **Attacks.**

4.2.1. *General.* Again, recall the information that Eve possesses. She has all of the public values, which includes g, g^a, g^b, g^{ab}, etc. She needs to determine the value of $g^{abc\cdots}$ to obtain the key. [3]

4.2.2. *Discrete logarithm computation.* First, we need to begin with a definition.

Definition 4.1. If b is a unit modulo m and a is another unit with $a \equiv b^d \pmod{m}$, we say that d is the discrete logarithm of a modulo m to the base b and write $d = \log_b(a)$.

This is a fairly straightforward attack, but it is one of the most mathematical and therefore beneficial to include. Remember from above that Eve has g and every value when it is raised to some exponent. Therefore, Eve could attempt to calculate $\log_g(g^a)$ to figure out a. From that, Eve can simply take her known value of g^b and raise it to the power of a. [3]

Again, this seems fairly simple, but it is virtually impossible absent incredibly powerful technology. This is also another attack that demonstrates the value of using large numbers; if p is extremely large, then this attack becomes nigh impossible to execute. [3]

5. Conclusion

5.1. **Final Notes.** As a whole, number theory and cryptography are closely related. As number theory has advanced, so has the security of cryptosystems. In this paper, we examined two techniques that are well-known and important in the field of cryptography. Both RSA encryption and Diffie-Helman Key Exchange are still used and the former is extraordinarily popular, but they are relatively old. In this field, the rapid development of technology means that constant attacks are being staged on cryptosystems. As a result, cryptologists are looking for new ideas to deliver secret messages. One more modern example is elliptic curve cryptography, which has a similar premise to RSA and Diffie-Helman. The idea is that these are all probably solvable with powerful enough technology and infinite time, but they are virtually impossible to solve in any reasonable amount of time with our current technology. Elliptic curve cryptography centers around the idea that calculating discrete logarithms on elliptic curves is very difficult with modern technology, and thus it is extremely interesting for the future of cryptography. Long-term, cryptologists are considering post-quantum methods of encrypting messages. Since the advent of quantum computers may spell doom for modern cryptosystems, cryptologists need to find new techniques. It is unclear exactly how this will work, but it is certain that math will be crucial. Nonetheless, cryptography is a fascinating field and the main way in which number theory has proven to be extremely useful

outside of inherent academic purposes. Technology will continue to advance and attack complex mathematical problems, but mathematicians will continue to explore the outer reaches of their field and invent new ways of encoding messages.

6. Acknowledgments

I would like to thank my brother, parents, and grandparents for always being there for me and being the main reasons why I was fortunate enough to receive the opportunity to attend the University of Chicago and its Mathematics REU. Additionally, I would like to thank Sam Quinn for being an excellent mentor. This paper would not be nearly what it is without his guidance, and I sincerely appreciate all that he has done to enrich my REU experience. Lastly, I would like to thank Professor May for hosting the REU and ensuring that it is a great program.

References

[1] Evan Dummit. Cryptography (part 1): Classical Cryptosystems and Modular Arithmetic. Northeastern University, 2016. Located online at: https://web.northeastern.edu/dummit/docs/cryptography_1_classical_cryptosystems.pdf.

[2] Evan Dummit. Cryptography (part 2): Public-Key Cryptography. Northeastern University, 2016. Located online at: https://web.northeastern.edu/dummit/docs/cryptography_2_public_key_cryptography.pdf.

[3] Evan Dummit. Cryptography (part 3): Discrete Logarithms in Cryptography Northeastern University, 2016. Located online at: https://web.northeastern.edu/dummit/docs/cryptography_3_discrete_logarithms_in_cryptography.pdf.

[4] Kenneth Levasseur. Digital Signatures using RSA. University of Massachusetts Lowell, 2013. Located online at: https://faculty.uml.edu//klevasseur/math/RSA_Signatures/RSA_Signatures.pdf.

[5] Cryptographic Key. techopedia. Located online at: https://www.techopedia.com/definition/24749/cryptographic-key.

[6] How to generate Large Prime numbers for RSA Algorithm. GeeksforGeeks. Located online at: https://www.geeksforgeeks.org/how-to-generate-large-prime-numbers-for-rsa-algorithm/.

[7] Israel Koren. Fault-Tolerant Systems, 2nd Edition. University of Massachusetts Amherst, 2020. Located online at: http://www.ecs.umass.edu/ece/koren/FaultTolerantSystems/simulator/RSA/new_page_5.htm.