Algebraic Structures and Prime Number Theory in Cryptographic Systems

Alex Shaw 40362741
Peter Corvan 40333991
Matteo Melis 40324932

December 12, 2024

1 Introduction

Cryptography is the study of techniques for transmitting messages in an altered form so as to enable only intended recipients to undo this alteration and read them. This is called *encryption*. In today's financially and data-driven world, it is more vital and lucrative than ever to ensure privacy of communication; naturally encryption is the foundation of this. Algebraic structures and their relevance to the difficulty of factoring large numbers underpin these methods and as such will be examined here, alongside the relatively lower complexity of obtaining large prime numbers.

2 Units in $\mathbb{Z}/n\mathbb{Z}$

We briefly examine the group of units of $\mathbb{Z}/n\mathbb{Z}$, which underpins secure modular arithmetic and is central to the cryptographic algorithms discussed later. To begin, we recall the definition of a unit and a fundamental result in number theory:

Definition 2.1. Let R be a ring. An element $x \in R$ is said to be a *unit* if there is an element $y \in R$ such that $xy = yx = 1_R$. The set of all units is denoted $R^\times$.

Proposition 2.2. Given $x, n \in \mathbb{Z}, n > 0$. Then x is invertible modulo $n \iff gcd(x, n) = 1$.

Proof. Suppose $[x]$ is a unit. Then there exists $y \in \mathbb{Z}$ such that $[x] \cdot [y] = [1] \implies xy \equiv 1 \pmod{n} \implies 1 = xy + qn$ for some $q \in \mathbb{Z} \implies \gcd(x, n) = 1$.
To see the converse, we deploy Bézout's identity and the result follows. □

Thus, in the case of $\mathbb{Z}/n\mathbb{Z}$, the group of units $(\mathbb{Z}/n\mathbb{Z})^\times$ consists of all integers modulo n that are coprime to n, or equivalently:

$$(\mathbb{Z}/n\mathbb{Z})^\times = \{0 \leq x < n \mid \gcd(x, n) = 1\}.$$

Example 2.3. Is $[17]$ a unit in $\mathbb{Z}/3120\mathbb{Z}$? If so, find its modular inverse.
We first compute $\gcd(17, 3120)$ using Euclidean division.

$$\begin{aligned} 3120 &= 183 \cdot 17 + 9 \\ 17 &= 1 \cdot 9 + 8 \\ 9 &= 1 \cdot 8 + 1 \\ 8 &= 8 \cdot 1 + 0 \end{aligned}$$

The final remainder not equal to zero is what we seek, hence $\gcd(17, 3120) = 1$, which implies that $[17]$ is invertible and thus a unit. We find an inverse by using the extended Euclidean algorithm:

$$\begin{aligned} 1 &= 9 - 8 \\ &= 9 - (17 - 9) \\ &= 2 \cdot 9 - 17 \\ &= 2 \cdot (3120 - 183 \cdot 17) - 17 \\ &= 2 \cdot 3120 - 367 \cdot 17. \end{aligned}$$

That is, $-367 \cdot 17 = 1 + 2 \cdot 3120 \equiv 1 \pmod{3120}$. Hence the modular inverse of 17 is:

$$-367 + 3120 \equiv 2753 \pmod{3120}.$$

3 Euler's Totient Theorem

Definition 3.1. Let n be a positive integer. *Euler's totient function*, denoted by $\varphi(n)$, is defined to be the number of non-negative integers x less than n which are coprime to n:

$$\varphi(n) := \#\{0 \leq x < n \mid \gcd(x, n) = 1\} = |(\mathbb{Z}/n\mathbb{Z})^\times|$$

Example 3.2. One can immediately see that $\varphi(1) = 1$, and $\varphi(p) = p - 1$ for any prime p. Less obviously, we show the following for all p prime and $\alpha \in \mathbb{Z}$:

$$\varphi(p^\alpha) = p^\alpha - p^{\alpha-1} = p^{\alpha-1}(p-1).$$

Indeed, the numbers from 0 to $p^\alpha - 1$ which are not prime to p^α are exactly those that are divisible by p. Those numbers are $p^{\alpha-1}, p^{\alpha-2}, \ldots, p^2, p$. So we have $p^{\alpha-1}$ fewer numbers than p^α.

Theorem 3.3. (Euler's Totient Theorem). *Let $n \geq 1$ and $x \in (\mathbb{Z}/n\mathbb{Z})^\times$. Then*

$$x^{\varphi(n)} = 1.$$

Proof. From Lagrange's theorem in group theory we know that for G a group and $x \in G$, $x^{|G|} = 1$. Let $G = (\mathbb{Z}/n\mathbb{Z})^\times$, the group of units of the ring. Then by definition, we have the result. $\square$

Corollary 3.4. (Fermat's little theorem). *Let p be a prime and $x \in \mathbb{Z}$ with $p \nmid x$. Then*

$$x^{p-1} \equiv 1 \pmod{p}.$$

Proof. Observe that x is a unit mod $p \iff p \nmid x$. Then we see that $p - 1 = \varphi(p) = |(\mathbb{Z}/n\mathbb{Z})^\times|$ and the result holds by Euler's totient theorem. $\square$

4 Finite Fields of Order p

Finite fields, or Galois fields, are algebraic structures consisting of a finite number of elements where all standard arithmetic operations are defined. Equivalently, finite fields are commutative division rings with a finite number of elements. A finite field with p elements is denoted by $\mathbb{F}_p$, where p is a prime number.

Corollary 4.1. *Let n be a positive integer. Then $\mathbb{Z}/n\mathbb{Z}$ is a field if and only if n is prime.*

Proof. If $n = p$ is prime, then the number of units is $\varphi(p) = p - 1$, so $\mathbb{Z}/n\mathbb{Z}$ is a field. If n is not prime, then there exists some divisor d such that $1 < d < n$ and $gcd(d, n) = d$. Altogether, $[d]$ is a non-zero element of $\mathbb{Z}/n\mathbb{Z}$ which is not invertible. $\square$

Definition 4.2. Let p be a prime. The *field of order p* is $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$.

5 Public Key Cryptosystems

In the 1970s, mathematicians started building on the concept of symmetric-key cryptography which was limited in its security, since encryption and decryption keys were often the same or could be derived from each other. This brought about the genesis of *Public Key Cryptography* and in particular the *RSA Cryptosystem.* The RSA algorithm is an encryption scheme designed in 1977 by Ronald Rivest, Adi Shamir and Leonard Adleman. It allows encrypting a message with a key (the encryption key) and decrypting it with a different key (the decryption key). The encryption key is public and can be given to everybody. The decryption key is private and is known only by the recipient of the encrypted message. We require the following proposition to understand the role of Euler's totient function in RSA.

Proposition 5.1. *If m, n are coprime, then $\varphi(mn) = \varphi(m)\varphi(n)$.*

Proof. By the Chinese remainder theorem,

$$\mathbb{Z}/mn\mathbb{Z} \cong \mathbb{Z}/m\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}.$$

Recognising that when an element (a, b) of $R \times S$ (where R and S are rings) is a unit, we know that a and b are units of R and S respectively, we now have that

$$(\mathbb{Z}/mn\mathbb{Z})^\times \cong (\mathbb{Z}/m\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z})^\times \cong (\mathbb{Z}/m\mathbb{Z})^\times \times (\mathbb{Z}/n\mathbb{Z})^\times.$$

Applying the definition of φ relating to the number of units of $\mathbb{Z}/n\mathbb{Z}$, we have

$$\varphi(mn) = |(\mathbb{Z}/mn\mathbb{Z})^\times| = |(\mathbb{Z}/m\mathbb{Z})^\times| \cdot |(\mathbb{Z}/n\mathbb{Z})^\times| = \varphi(m)\varphi(n). \qquad \square$$

The RSA algorithm is based on the following facts. Given two prime numbers p and q, and a positive number m relatively prime to p and q, we use the previous proposition along with Euler's theorem to tell us that:

$$m^{\varphi(pq)} = m^{\varphi(p)\varphi(q)} = m^{(p-1)(q-1)} \equiv 1 \pmod{pq},$$

noting the use of the obvious condition that p and q are coprime since they are prime, when applying the proposition.
Assume now that we have two integers e and d such that $e \cdot d \equiv 1 \pmod{\varphi(pq)}$. Then we have that:

$$(m^e)^d = m^{e \cdot d} \equiv m \pmod{pq}.$$

So, given m^e we can recover m modulo pd by raising m^e to the dth power.
The RSA algorithm consists of the following:

1. Generate two large primes p and q. Find their product $n = pq$.

2. Find two numbers e and d (in the range from 2 to $\varphi(n)$) such that $e \cdot d \equiv 1 \pmod{\varphi(n)}$. This requires some trial and error. First, e is chosen at random, and the Euclidean algorithm is used to find $gcd(e, m)$, solving at the same time the equation $ex + my = gcd(e, m)$. If $gcd(e, m) = 1$ then the value obtained for x is d. Otherwise, e is not relatively prime to $\varphi(n)$ and we must try a different value for e.

3. The *public encryption key* will be the pair (n, e). The *private decryption key* will be the pair (n, d). The encryption key is given to everybody, while the decryption key is kept secret by the future recipient of the message.

4. The message to be encrypted is divided into small pieces, and each piece is encoded numerically as a positive integer $m < n$.

5. The number m^e is reduced modulo n to give $m^e \equiv m' \pmod{n}$

6. The recipient recovers m by computing $(m')^d \equiv m \pmod{n}$

Consider the following simplified example:

Example 5.2. Let $p = 3$, $q = 11$. Calculating $n = pq = 33$. Compute $\varphi(n) = \varphi(pq) = (p-1)(q-1) = 2 \cdot 10 = 20$. Now, we choose $e \in \mathbb{N}$ such that $1 < e < \varphi(n)$ and $gcd(e, \varphi(n)) = 1$. In this simple case, it is easy to see that we can choose $e = 7$. We now need $d \in \mathbb{N}$ such that $1 < d < \varphi(n)$ and $e \cdot d \equiv 1 \pmod{n}$. If we take $d = 3$, we have $e \cdot d = 7 \cdot 3 = 21 \equiv 1 \pmod{\varphi(n)}$. We obtain the following public and private key:

$$\text{Public key} = (e, n) = (7, 33), \text{ Private key} = (d, n) = (3, 33)$$

Now if we let our message $m = 2$, our encrypted message becomes $m^e = 2^7 = 128 \equiv 29 \pmod{n} = m'$. The recipient receives m' and computes $(m')^d = 29^3 = 24389 \equiv 2 \pmod{n} = m$.

Remark 5.3. The above example outlines the utilisation of RSA by the intended users, but the reason for using encryption at all is to conceal information from everyone else. Thus, we must discuss this method's security.

The security of the RSA algorithm is built upon the asymmetry between two mathematical tasks: it is much easier to find two extremely large primes p and q than it is for someone else (usually a bad actor) who knows $n = pq$ but neither p nor q to determine the two factors in n. In general, finding large prime numbers is computationally feasible using probabilistic algorithms such as the Miller-Rabin primality test [1], yet factoring n into its prime components p and q is significantly harder when n is sufficiently large.

The prime generation process relies on testing candidate numbers for primality, which, on average, can be achieved efficiently in polynomial time with modern algorithms. By contrast, factoring n when its prime factors are unknown is a problem for which no efficient algorithm exists within classical computation. Specifically, the most advanced algorithms, such as the Quadratic Sieve Method [2] or General Number Field Sieve (GNFS) [3], have sub-exponential complexity but still require vast computational resources when n exceeds several hundred digits.

To illustrate this disparity concretely:

- Generating two 512-bit prime numbers p and q can be done in a matter of seconds on modern hardware, with many Python, MATLAB, etc., tools available for free.
- Factoring the resulting 1024-bit composite number $n = pq$ could take years, even using state-of-the-art factoring techniques and large-scale computational power [4].

This computational imbalance guarantees that the private key d, derived from $\varphi(n)$, remains secure as long as p and q remain secret. Any adversary attempting to decrypt an RSA-encrypted message would need to factor n to recover $\varphi(n)$ and ultimately d, an infeasible task for sufficiently large n.

Thus, the RSA cryptosystem's security is not merely about encryption but relies fundamentally on the practical impossibility of solving the integer factorization problem for large numbers within a reasonable time frame.

6 Elliptic Curve Cryptography

Until this point, the underlying groups have been those based on modular arithmetic. There are many other classes of groups that have a purpose and niche within cryptography - with one of those being the groups consisting of points on *elliptic curves*. These curves, when limited to finite fields specifically, yield an inexhaustible supply of abelian groups which are manageable within computation owing to the robust properties that this structure offers. For our purposes, we will furthermore only consider fields of prime order $p > 3$.

Definition 6.1. Let $p > 3$ be prime. The *elliptic curve* over $\mathbb{Z}/p\mathbb{Z}$ is the set of all pairs $(x, y) \in \mathbb{Z}/p\mathbb{Z}$ which satisfy

$$y^2 \equiv x^3 + a \cdot x + b \bmod p,$$

together with a single element $\mathcal{O}$, called the "point at infinity", where $a, b \in \mathbb{Z}/p\mathbb{Z}$ and $4a^3+27b^2 \neq 0 \pmod{p}$. Let $E(\mathbb{Z}/p\mathbb{Z})$ denote the set of pairs $(x, y) \in \mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/p\mathbb{Z}$ fulfilling the above equation (known as the Weierstrass equation) along with $\mathcal{O}$ and we have

$$E(\mathbb{Z}/p\mathbb{Z}) := \{(x, y) : x, y \in \mathbb{Z}/p\mathbb{Z} \text{ and } y^2 \equiv x^3 + a \cdot x + b \bmod p\} \cup \{\mathcal{O}\}.$$

As opposed to RSA cryptography, what distinguishes Elliptic Curve Cryptography (ECC) as an alternative encryption method? Consider the following theorem:

Theorem 6.2. (Every Elliptic Curve (E) is an abelian group). The set E is an abelian group with neutral element $\mathcal{O}$, provided addition $P + Q$ and inverses $-P$ are defined as follows:

1. -$\mathcal{O} = \mathcal{O}$ and $P + \mathcal{O} = P = \mathcal{O} + P$
2. If $\mathcal{O} \neq P = (x, y)$ then $-P = (x, -y)$
3. For $Q = -P$ set $P + Q = \mathcal{O}$ and $Q + P = \mathcal{O}$
4. For $P = (x, y)$ and $Q = (x', y')$ with $x \neq x'$, set $P + Q = (x'', y'')$ where:
$$x'' = \left(\frac{y' - y}{x' - x}\right)^2 \quad \text{and} \quad y'' = -y + \frac{y' - y}{x' - x}(x - x'').$$
5. If $P = Q \neq \mathcal{O}$, define $P + Q = (x'', y'')$, where:
$$x'' = \left(\frac{3x^2 + a}{2y}\right)^2 - 2x \quad \text{and} \quad y'' = -y + \left(\frac{3x^2 + a}{2y}\right)(x - x'').$$

Where we cannot have $y = 0$ in case 5.

This means that any elliptic curve is a group, namely an abelian one, and if defined over a finite field, it will of course form a finite group. The above set of rules allows us to make a very large set of groups. Determining the private key under such a construction will be extremely difficult. This is because with the elliptic curve defined as a group, we can consider the discrete logarithm problem:

Definition 6.3. 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 $d = log_b(a)$. All usual rules of logarithms still apply here.

Example 6.4. Modulo 14, we have $log_3(11) = 4$, since $3^4 \equiv 11 \pmod{14}$. It is easier to write $log_3(11) = 4 \pmod{6}$ as the order of 3 modulo 14 is 6.

Remark 6.5. In general, it is believed to be computationally infeasible to evaluate discrete logarithms or extract roots modulo m for sufficiently large m. This elliptic curve discrete logarithm problem (ECDLP), which forms the foundation of ECC, is significantly harder to solve compared to the discrete logarithm problem in other algebraic structures, such as finite fields.

Until 1990, no algorithms existed at all for solving discrete logarithms on elliptic curves that forced the group's structure to achieve subexponential time complexity. Even after advances by Menezes, Okamoto, and Vanstone [2], it was shown that the only curves vulnerable to attacks are the so-called *supersingular* curves. These curves can be identified and avoided during implementation.

The strength of ECC hence stems from two key points:

- The best known algorithms for solving the ECDLP on a general elliptic curve still require exponential time complexity, as opposed to subexponential methods such as the GNFS for RSA.

- For a properly chosen curve (non-supersingular and with order divisible by a large prime factor), ECC offers equivalent security to RSA or Diffie-Hellman but with significantly smaller key sizes.

For example, a 256-bit ECC key provides comparable security to a 3072-bit RSA key [5]. This efficiency makes ECC particularly attractive for modern applications, including resource-constrained environments such as embedded systems, mobile devices, and secure messaging protocols.

In conclusion, the computational complexity of the ECDLP, combined with careful curve selection, ensures that ECC remains a highly secure and efficient alternative to RSA and other public-key systems.

References

[1] Stanford University. Miller-rabin primality test. `https://crypto.stanford.edu/pbc/notes/numbertheory/millerrabin.html`. Accessed: Dec 12, 2024.

[2] Neal Koblitz. *A Course in Number Theory and Cryptography.* Graduate Texts in Mathematics. Springer, 7th edition, 1994.

[3] D. J. Bernstein. The number field sieve method. `https://www.cs.umd.edu/~gasarch/TOPICS/factoring/NFSmadeeasy.pdf`, 2001. Accessed: Dec 12, 2024.

[4] Arjen K. Lenstra, Hendrik W. Lenstra Jr., and Bruce Schneier. Factoring large numbers with the number field sieve. In *Advances in Cryptology*, pages 287–299. Springer, 2003.

[5] Nina Heninger and Tanja Lange. Elliptic curve cryptography key size comparisons with RSA. In *Lecture Notes in Computer Science.* Springer, 2011.

[6] David M. Burton. *Elementary Number Theory.* McGraw-Hill Education, New York, seventh (international) edition, 2011.