RSA Encryption: Math Club Notes

Markus Hoehn

Overview

- RSA (Rivest–Shamir–Adleman) is one of the oldest and most widely used public-key cryptosystems.
- Public-key cryptography involves individuals generating a private key (kept to yourself) and a public key, which they share with others.
- Messages can be encrypted using someone's public key, and only the recipient possessing the corresponding private key can decrypt them.

Modular Arithmetic

- Modular arithmetic is a system for working with integers under a specified modulus.
- The expression $x \equiv a \pmod{N}$ means that x is congruent to a modulo N. In other words, x and a yield the same remainder when divided by N.
- For example, $15 \equiv -9 \pmod{3}$ since both yield a remainder of 0 when divided by 3.
- Clock arithmetic is a specific case of modular arithmetic with a modulus of 12. For instance, if it is $9:00$ now, then in 6 hours, the time will be $15:00$, which is congruent to $3:00$ modulo 12.

Euler's Totient Function

- Euler's totient function, denoted $\phi(n)$, counts the number of positive integers m such that $1 \leq m < n$ and $\gcd(m, n) = 1$.
- Two numbers are said to be coprime if their greatest common divisor (gcd) is 1, indicating they share no common factors other than 1. Notably, primes are coprime to all preceding positive integers, so $\phi(p) = p - 1$ for prime p.
- For example, $\phi(15) = 8$ since $\{1, 2, 4, 7, 8, 11, 13, 14\}$ are coprime to 15, leaving out $\{3, 5, 6, 9, 10, 12\}$.
- It is a known fact that $\phi(nm) = \phi(n)\phi(m)$ when n and m are coprime.

Euler's Theorem

Let n be a positive integer, and let a be an integer coprime to n. Then, according to Euler's theorem:

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

For example, for any number a coprime to 15, since $\phi(15) = 8$, we have $a^8 \equiv 1 \pmod{10}$.

Modular Inverses

- The modular inverse of a modulo m is the integer x such that $ax \equiv 1 \pmod{m}$. If a and m are coprime, then there exists an integer x that serves as the modular inverse of a.
- We can use the Extended Euclidean algorithm to compute modular inverses, which is an algorithm relying on the computationally easy multiplication.

Factoring Problem

- The security of RSA encryption hinges on the challenge of factoring the product of two large prime numbers.
- Today, known algorithms would require an infeasible amount of time, extending beyond the age of the universe, to factorize sufficiently large primes, ensuring our security.
- The practicality of RSA encryption is rooted in the computational simplicity of multiplying numbers, a task easily handled by computers.
- For instance, computing $31 \cdot 37 = 1147$ is straightforward, but finding the factors of 1147 without prior knowledge of its construction is significantly more time-consuming.

RSA Algorithm

The Setup

1. The receiver first selects two large prime numbers, p and q. Their product, $n = pq$, forms part of the public key.
2. Next, the receiver computes Euler's totient function: $\phi(n) = \phi(pq) = \phi(p)\phi(q) = (p-1)(q-1)$. Then, they choose a number e coprime to $\phi(n)$. This e constitutes the rest of the public key.
3. Finally, the receiver calculates the modular inverse d of e modulo $\phi(n)$. This d serves as the private key. Computing d is computationally challenging without knowing $n = pq$, as it requires the knowledge of $\phi(n) = (p-1)(q-1)$.

RSA Algorithm

Transmitting Messages

The receiver broadcasts their public key (n, e) and keeps their private key d confidential.

1. The sender converts their message into a number m, typically using a system like ASCII encoding. It's important that $m < n$ to avoid losing the message in the encryption process when taking modulo n.
2. Next, the sender computes the encrypted ciphertext: $c \equiv m^e \pmod{n}$. This ciphertext, along with the public key, is the only information accessible to a potential attacker.
3. Upon receiving the ciphertext, the receiver decrypts it to retrieve the original message: $m \equiv c^d \pmod{n}$.

Step 3 relies on Euler's theorem, which states that $m^{\phi(n)} \equiv 1 \pmod{n}$, and the choice of d such that $de \equiv 1 \pmod{\phi(n)}$. This ensures the existence of an integer k such that $de = k\phi(n) + 1$. Consequently,

$$m^{de} \equiv m^{k\phi(n)+1} \equiv m^{k\phi(n)} \cdot m \equiv (m^{\phi(n)})^k \cdot m \equiv 1^k \cdot m \equiv m \pmod{n}.$$

Example

The Setup

Let's have the receiver select primes $p = 13$ and $q = 17$. In practice, primes would be much larger to avoid falling victim to brute force attacks.

1. $n = pq = 13 \times 17 = 221$, which is half of the public key.
2. $\phi(n) = (13 - 1)(17 - 1) = 192$. The receiver chooses $e = 5$.
3. Then, the receiver calculates $d = 77$, since $de \equiv 1 \pmod{\phi(n)}$.
4. Finally, the receiver distributes their public key $(221, 5)$.

Example

Message Transmission

Note that the public key is $(221, 5)$.
Let's say the sender wants to transmit the message "DAVID". We can use an ASCII table to convert each character to its ASCII number. So we have $m = 6869766868$. However, note that $m > n$, which would cause our message to be lost if we try to send it all at once. Therefore, we send our message piece by piece.

1. Starting with the letter "D", we have $m = 68$.
2. To encrypt the message m, the sender calculates $c \equiv m^e \equiv 68^5 \equiv 204 \pmod{221}$. Thus $c = 204$ and it is sent to the receiver. Any third-party attackers will also be able to see this encrypted message.
3. The receiver, however, will be able to decrypt it with their private key $d = 77$. $c^d = 204^{77} \equiv 68 \pmod{221}$, thus getting the message $m = 68$.
4. The receiver translates this to the letter "D".

Applications

- Online Banking
- Account logins
- Digital Signatures
- VPN Encryption
- Secure Shell (remote file access and transfer)
- Email encryption
- Web browser authentification
- Token-based logins

Vulnerabilities

- The strength of RSA is measured by the number of bits in n. To prevent brute force attacks, most newly generated keys are 4096 bits long. A 512-bit RSA key can be cracked in a few hours, and in 2010, a brute force attack on a 768-bit key was successful.
- Computers can easily compute the greatest common divisor of two numbers using the Euclidean algorithm. Thus, an attacker can use this algorithm on two public keys. If their greatest common divisor is not 1, then the attacker has found a prime number dividing both keys, compromising both keys and allowing the attacker to promptly find the private key.
- If p and q are close together such that $m^e < n$, the attacker can determine the private key efficiently.
- Quantum computers can factor numbers exponentially faster than current algorithms (Shor's algorithm). However, as of today, quantum computers are too small to factor numbers greater than 16-bits.