Guarding Secrets: Understanding Stream Ciphers, Random Numbers, and the One-Time Pad

WOMANIUM GLOBAL ONLINE QUANTUM MEDIA PROJECT #Quantum30 Challenge Day 3

Listen

Introduction:

Cryptography, the art of secure communication, plays a crucial role in safeguarding sensitive information in the digital age. It can be broadly classified into three types: symmetric cryptography, asymmetric cryptography, and cryptographic protocols. In our previous articles, we have delved into the world of symmetric, asymmetric, and cryptographic protocols. Now, let's focus on symmetric cryptography and its two primary categories: **stream ciphers** and **block ciphers**. We will also explore the significance of **random number generation** and delve into the intriguing concept of the **one-time pad**.

Symmetric Cryptography:

In symmetric cryptography, the same key is used for both encryption and decryption, making it more efficient but requiring a secure channel for key exchange. Stream ciphers and block ciphers fall under this category.

Stream Ciphers:

Stream ciphers are a type of symmetric encryption that encrypts data one bit or one byte at a time. Unlike block ciphers, which divide data into fixed-size blocks, stream ciphers encrypt the data stream continuously. The encryption process involves combining the plaintext with a stream of random or pseudorandom bits, known as the keystream, using a bitwise XOR (exclusive OR) operation.

Photo by Christian Lendl on Unsplash

Encryption Process:

1. **Key Generation:** The most crucial aspect of stream ciphers is the generation of a secure keystream. The key is used as a seed for the pseudorandom number generator (PRNG) to produce the keystream. The key length significantly impacts the strength of the cipher.

2. **Keystream Generation:** The PRNG produces a stream of random bits based on the seed (key) provided. This keystream is combined with the plaintext using the XOR operation to produce the ciphertext.

3. **Ciphertext Generation:** The XOR operation between the keystream and plaintext results in the ciphertext, which appears random and indecipherable without the proper key.

Decryption Process:

1. **Key Generation:** The same key used during encryption is required for decryption.

2. **Keystream Generation:** Using the key, the PRNG generates the same keystream as during encryption.

3. **Plaintext Recovery:** The decryption process involves XORing the ciphertext with the keystream to retrieve the original plaintext.

Importance of Mod 2 Operator:

The use of the mod 2 (XOR) operator is essential in stream ciphers as it ensures that the encryption and decryption operations are reversible. XORing the ciphertext with the keystream during decryption effectively cancels out the XOR operation applied during encryption, revealing the original plaintext.

Example of Stream Cipher Encryption and Decryption:

Let's illustrate the encryption and decryption process of the *ASCII* character '*B*' (01000010) using a stream cipher with a random keystream (10101010).

Encryption: Plaintext: 01000010 (*ASCII 'B'*) Keystream: 10101010 (Random) Ciphertext: 11101000 (Encrypted *ASCII 'B'*)

Decryption: Ciphertext: 11101000 (Encrypted *ASCII 'B'*) Keystream: 10101010 (Same random keystream used during encryption) Plaintext: 01000010 (*ASCII 'B'*).

. . .

Random Number Generators (RNG):

Random number generators play a critical role in cryptography, providing the randomness necessary for generating strong encryption keys, initialization vectors, and more. RNGs are categorized into four types:

1. ***True Random Number Generators (TRNG):*** TRNGs harness physical processes, such as electronic noise or thermal noise, to generate truly random and unpredictable numbers.

2. ***Pseudorandom Number Generators (PRNG):*** PRNGs use deterministic algorithms and a seed value (initial value) to generate seemingly random numbers. While they appear random, PRNGs are not truly unpredictable and repeat their sequence after a certain period.

3. ***Cryptographically Secure Pseudorandom Number Generators (CPRNG):*** CPRNGs are designed explicitly for cryptographic applications. They possess properties that prevent an attacker from predicting future outputs based on previous ones.

4. ***Linear Congruential Generators (LCG):*** LCGs are a specific type of PRNG that generates sequences of numbers using a linear congruential formula. However, they have certain weaknesses and are not suitable for cryptographic purposes.

Photo by rc.xyz NFT gallery on Unsplash

How do PRNGs ensure unpredictability?

Pseudorandom Number Generators (PRNGs) ensure unpredictability by simulating randomness through deterministic algorithms and utilizing a seed value as the starting point for number generation. While PRNGs are not truly random like True Random Number Generators (TRNGs), they exhibit properties that make their output appear random and unpredictable, provided certain conditions are met.

The unpredictability of PRNGs depends on the following factors:

- ***Seed Value:*** PRNGs use a seed value, which is typically a fixed-length binary value. The seed acts as the initial state of the PRNG algorithm. If the seed is truly random or adequately unpredictable, it helps create a more unpredictable sequence of numbers. However, if the seed is known or can be easily guessed, it may lead to the predictability of the entire sequence.

- *Algorithm Design:* The underlying algorithm in a PRNG is responsible for generating subsequent numbers based on the seed. The algorithm should be carefully designed to avoid patterns or correlations that could compromise the unpredictability of the output. Common algorithms used in PRNGs include linear congruential generators (LCGs), Mersenne Twister, and XOR-shift algorithms.
- *Period Length:* PRNGs have a finite period length, which refers to the number of unique values the generator can produce before it starts repeating the sequence. A longer period length is desirable to delay the recurrence of the generated sequence, making it more challenging to predict the next number.
- *Reproducibility:* PRNGs should produce the same sequence of numbers when given the same seed value. However, they should not reveal any patterns or correlations that could be exploited to predict future numbers based on past outputs.
- *Seed Unpredictability:* The seed value should be unpredictable to ensure that an attacker cannot guess or deduce it easily. If the seed is predictable, an adversary could reconstruct the entire sequence and compromise the unpredictability of the PRNG.

The primary goals of CPRNGs are:

- *Unpredictability:* CPRNGs should produce output that is statistically indistinguishable from true randomness. This means that the generated sequence should not exhibit any discernible patterns, correlations, or biases, making it impossible to predict future outputs based on observed ones.
- *Resistance to Attacks:* CPRNGs are designed to withstand cryptographic attacks, including state recovery attacks, where an adversary attempts to reverse-engineer the internal state of the generator to predict future outputs. A CPRNG should have a large internal state space to resist such attacks effectively.
- *Reproducibility:* Similar to regular PRNGs, CPRNGs should be able to reproduce the same sequence of numbers when given the same initial seed value. This feature is essential for generating cryptographic keys and synchronization in secure communication protocols.

- *Forward Secrecy:* Even if an adversary compromises the current state of a CPRNG, it should not be able to deduce the previous states or any previously generated numbers. This property ensures that past outputs remain secure even if the current state is compromised.

. . .

One-Time Pad:

The one-time pad is a unique encryption technique with a fascinating concept of perfect secrecy. It involves using a key of the same length as the plaintext, ensuring that each key bit is used only once and then discarded. The key is truly random and remains secret between the sender and the receiver.

Essence of One-Time Pad:

One-time pad encryption provides absolute security, as long as the key is truly random, and the key is used only once. It achieves perfect secrecy because every possible plaintext maps to an equally likely ciphertext, making it impossible for an attacker to determine the original message, regardless of computational power or resources.

Drawbacks:

Despite its perfect security, the one-time pad has several practical limitations:

- *Key Management:* Generating and securely distributing truly random keys of equal length to the plaintext can be challenging, especially for long messages.
- *Key Reuse:* Reusing the same key or using a key more than once compromises the security of the one-time pad.
- *Key Storage:* Safely storing and managing the key material is crucial, and any mishandling can lead to security breaches.

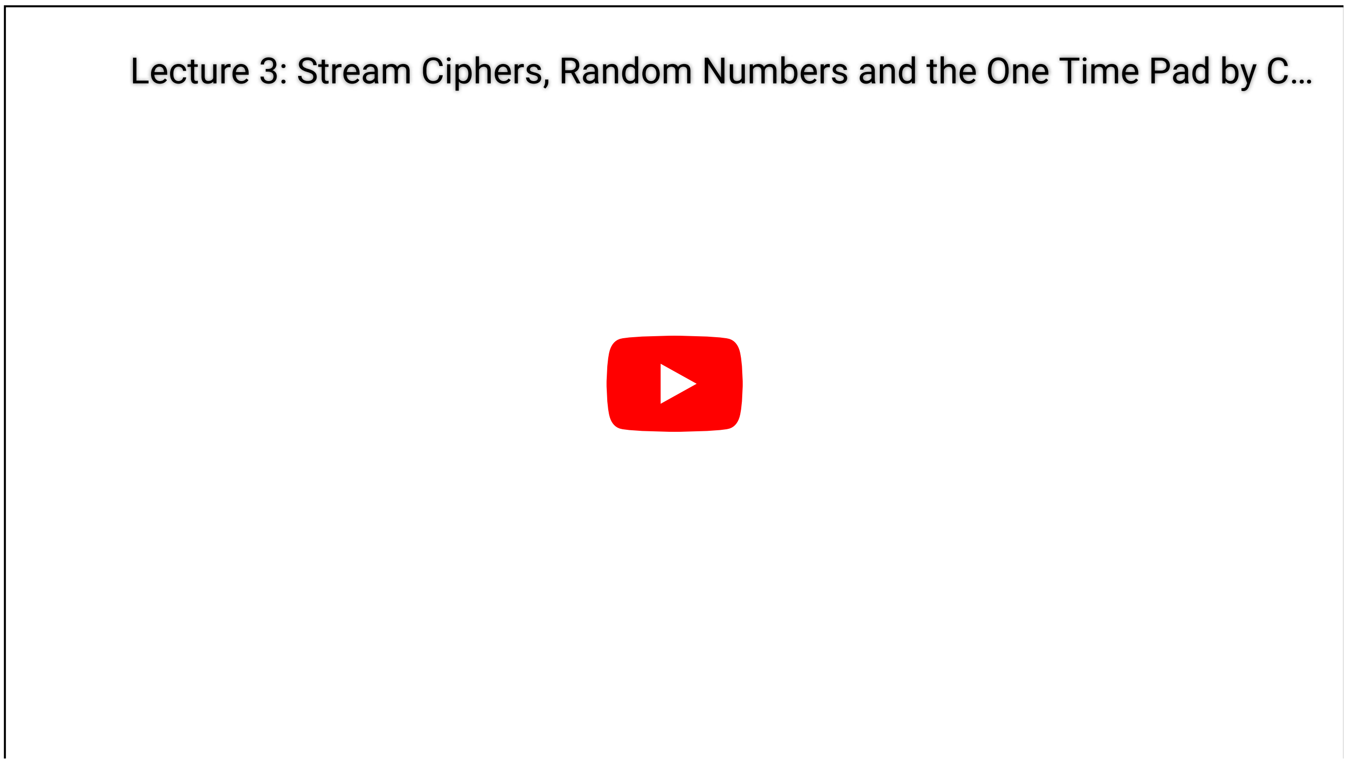
. . .

Conclusion:

Stream ciphers offer a lightweight and efficient encryption solution for securing data in symmetric cryptography. The generation of random numbers is fundamental for cryptographic applications, while the one-time pad exemplifies perfect secrecy, albeit with challenges in key management. As technology advances, the quest for secure communication continues, pushing the boundaries of cryptographic research and innovation.

. . .

This article is written based on the video lecture



Lecture 3: Stream Ciphers, Random Numbers and the One Time Pad by C...

provided by QuantumComputingIndia as a part of the #Quantum30 learning challenge and some topics are collected from the internet.

I have been exploring Cryptography since the start of this month and throughout this month I will gain in-depth knowledge of this field. Your suggestion will be really helpful for my future endeavor.

This is a part of **WOMANIUM GLOBAL ONLINE QUANTUM MEDIA PROJECT**. This project will help me to dive into the cryptographic world(From Classical to Quantum Approach). From onwards I shall share my learning log with others who are curious about this particular and promising field.

I want to take a moment to express my gratitude to **Marlou Slot** and **Dr. Manjula Gandhi** for this initiative and encouragement and sincere thanks to **Moses Sam Paul Johnraj** for providing the 30-day schedule.

#Quantum30 #QuantumComputing #QuantumJourney #QuantumEnthusiast #Womanium #Cryptography #QuantumCryptography

Written by Murshed SK

Physics Undergrad | Quantum Information Science and Computation Enthusiast | Passionate about Quantum Machine Learning | <Womanium | Quantum> Scholar