Chapter 2 – Stream Ciphers

(Based on Lecture Slides: Understanding Cryptography – Paar & Pelzl)

□ 1. Introduction to Stream Ciphers (in Cryptology Context)

- **Cryptology** divides into two main branches:
- **Cryptography:** The science of creating secure communication systems.
- **Cryptanalysis:** The science of breaking or analyzing these systems.
- Within **symmetric cryptography**, there are two main cipher types:
- **Block Ciphers**
- **Stream Ciphers**
- **Stream Ciphers** were first conceptualized in **1917 by Gilbert Vernam**.
  His system, known as the **Vernam Cipher**, introduced the idea of **XOR (exclusive OR)** addition.
  This concept later became the mathematical basis for the **One-Time Pad (OTP)** — the only known perfectly secure cipher.

□ 2. Stream Cipher vs. Block Cipher

Aspect	Stream Cipher	Block Cipher
Encryption Unit	Processes single bits (or bytes) sequentially.	Processes a full block (e.g., 64 or 128 bits) at once.
Speed & Size	Usually lightweight and fast — ideal for embedded or mobile systems.	Heavier computation — common for Internet and file encryption.
Examples	A5/1 (GSM), RC4, Trivium	AES, DES, 3DES
Error Behavior	One bit error affects only that bit.	One bit error corrupts the entire block.

Lecture note:
"Stream ciphers encrypt bits individually; usually small and fast — common in embedded devices (e.g., A5/1 for GSM phones)."

□ 3. Encryption and Decryption with Stream Ciphers

- Both encryption and decryption rely on **modulo-2 addition (XOR, ⊕)**.
- The **same operation** is used for both encryption and decryption.

Equations:

Encryption: $y_i = x_i + s_i \pmod 2$

Decryption: $x_i = y_i + s_i \pmod 2$

where:
$x_i, y_i, s_i \in \{0, 1\}$

Explanation:
Each plaintext bit (x_i) is XORed with one keystream bit (s_i).
If the keystream is truly random and never reused, the cipher achieves **theoretical perfect secrecy** (same as the One-Time Pad).

□ 4. Synchronous vs. Asynchronous Stream Ciphers

Type	Definition	Key Dependence	Error Tolerance
Synchronous	Keystream depends only on the secret key (and optionally an Initialization Vector, IV).	Independent of ciphertext.	If a bit is lost → desynchronization; must re-sync manually.
Asynchronous	Keystream also depends on previous ciphertext bits (feedback-based).	Feedback from transmitted data.	Can automatically re-synchronize after errors.

Key Requirement:
The security of a stream cipher depends entirely on the randomness of its keystream (s_i):
$Pr(s_i = 0) = Pr(s_i = 1) = 0.5$
and it must be **reproducible** by both sender and receiver.

□ 5. Why Modulo-2 Addition is a Good Encryption Function

- XOR has **excellent statistical properties**.
  If the keystream (s_i) is truly random, each ciphertext bit (y_i) has an equal 50% probability of being 0 or 1.
- XOR is **self-inverse**, meaning the same operation decrypts the message.

Truth Table:

x_i	s_i	y_i
0	0	0
0	1	1
1	0	1
1	1	0

Conclusion:
XOR provides both **simplicity** and **strong confusion** properties.

□ 6. Throughput Comparison (Slide 10)

Cipher	Key Length (bits)	Throughput (Mbit/s)	Type

DES	56	36.95	Block
3DES	112	13.32	Block
AES	128	51.19	Block
RC4	Variable	211.34	Stream

Observation:
Stream ciphers like **RC4** are much faster and lighter than block ciphers such as **DES**, **3DES**, or **AES**, making them ideal for hardware or real-time encryption.

7. Random Number Generators (RNGs)

Classification (Slide 12):

1. True RNG (TRNG)
2. Pseudorandom Number Generator (PRNG)
3. Cryptographically Secure PRNG (CSPRNG)

True RNG (TRNG)

- Based on **physical random processes**, such as:
  • Semiconductor noise
  • Clock jitter in digital circuits
  • Radioactive decay
  • Mouse movement or keyboard timing
- Outputs are **non-deterministic**, **unpredictable**, and **non-reproducible**.
- Desired property: $\Pr(s_i = 0) = \Pr(s_i = 1) = 0.5$
- Used for **key generation**, **nonces**, and **initialization vectors (IVs)**.

Pseudorandom Number Generator (PRNG)

- Deterministic algorithm that produces a pseudo-random sequence from an **initial seed (S_0)**.
- **Example:** rand() in C — *Linear Congruential Generator (LCG)*

$$S_{i+1} = (A \times S_i + B) \bmod m$$

- **Weakness:**
  Linear structure → predictable → unsuitable for cryptographic use.

Cryptographically Secure PRNG (CSPRNG)

- A PRNG designed with **unpredictability**:
  Knowing *n* output bits must not allow prediction of the next bit (s_{n+1}) in polynomial time.
- **Required** for secure cryptographic systems, especially **stream ciphers**.

□ 8. One-Time Pad (OTP)

- Developed by **Mauborgne**, based on **Vernam's** stream cipher.
- **Encryption:** $y_i = x_i \oplus k_i$
- **Decryption:** $x_i = y_i \oplus k_i$

Unconditional Security:
Even with infinite computing power, OTP cannot be broken if:

1. The key is **truly random**.
2. The key is **used only once**.
3. The key length **equals the message length**.

Disadvantage:
Impractical — key must be as long as the message, making key distribution difficult.

□ 9. Linear Feedback Shift Registers (LFSRs)

- **Structure:** Concatenated flip-flops (registers) with feedback via XOR of selected bits.
- **Degree (m):** Number of flip-flops in the register.
- **Maximum Period:** $2^m - 1$
- **Recursive Equation:**
  $$s_{i+m} = (p_1 \cdot s_{i+m-1} \oplus p_2 \cdot s_{i+m-2} \oplus \dots \oplus p_m \cdot s_i)$$
- LFSRs are used to generate **pseudo-random keystreams**.

Example:
For m = 3, feedback pattern (1, 0, 1) produces a repeating sequence of length 7.

□ 10. Security of LFSRs

- Each LFSR is defined by a **feedback polynomial:**
  $$P(x) = 1 + p_1x + p_2x^2 + \dots + p_mx^m$$
- **Weakness:**
  Single LFSRs are linear → predictable.
  If 2m output bits are known, the feedback coefficients (p_i) can be determined by solving linear equations.
- **Solution:**
  Combine **multiple LFSRs** and introduce **non-linear components** to resist linear attacks and increase unpredictability.

□ 11. Trivium – Modern Stream Cipher (Slides 25–26)

- Combines **three nonlinear LFSRs (NLFSRs)** of lengths **93, 84, and 111 bits**.
- **Registers:** A, B, and C → total **288 bits** of internal state.

Initialization Steps:

1. Load **80-bit IV** into Register A.
2. Load **80-bit key** into Register B.
3. Set last **three bits of Register C to 1**, others to 0.
4. Perform **1152 warm-up clock cycles** (no output yet).

Keystream Generation:
The keystream bit (s_i) = XOR sum of outputs from all three NLFSRs.

Design Features:

- 3 AND gates for non-linearity.
- 7 XOR gates (four with triple inputs).
- Highly efficient in hardware.
- Can be parallelized to produce up to **64 bits per clock cycle**.

□ 12. Lessons Learned (Slide 27)

- Stream ciphers are **less common** than block ciphers in Internet security but are **useful for low-resource devices** (e.g., GSM, IoT).
- The **security** of a stream cipher depends entirely on the **quality of the keystream generator**.
- The **One-Time Pad (OTP)** is theoretically perfect but **impractical**.
- **Single LFSRs** are weak, but **combinations with non-linear functions** (like Trivium) can yield **strong stream ciphers**.

□ End of Chapter Summary

Key Takeaways:

- Stream ciphers operate **bit-by-bit** using XOR.
- Keystream **randomness** and **nonlinearity** are essential for security.
- **True RNGs** provide physical randomness; **CSPRNGs** make it usable for cryptography.
- **OTP** is perfectly secure but impractical.
- **LFSRs** are efficient but predictable — combining them with non-linearity (as in **Trivium**) ensures modern security.