COMPSCI 4CR3 - Applied Cryptography

Jake Doliskani

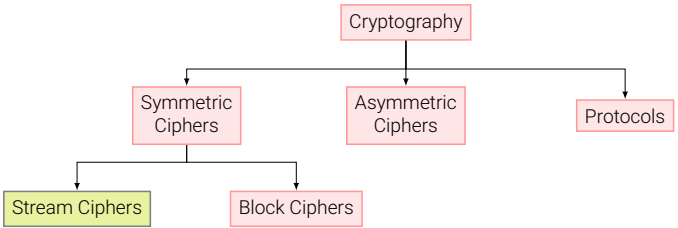
McMaster University

Stream Ciphers

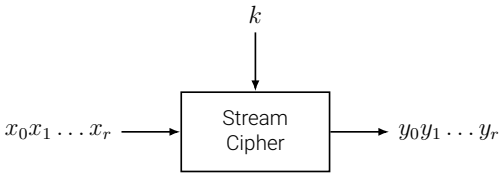
This lecture

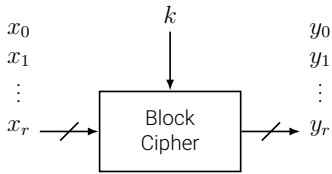
- Introduction to stream ciphers
- Random and pseudorandom number generators
- The One-Time Pad (OTP)
- Linear feedback shift registers

Stream Ciphers



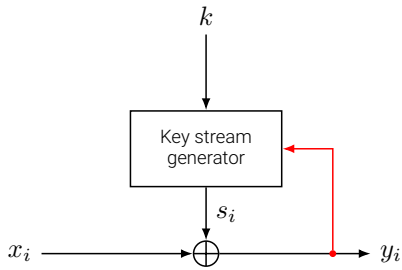
Stream vs Block Cipher





- Stream cipher: 1 bit at a time
  - Xor 1 bit of the key stream with 1 bit of the input
- Block cipher: 1 block at a time
  - Encode an entire block using the same key

Stream cipher: synchronous vs asynchronous



- **Synchronous:** key stream depends on the key
- **Asynchronous:** key stream depends on the key and the ciphertext
- Most practical stream ciphers are synchronous

Block ciphers

- Encrypt an entire block of plaintext bits at a time
- Example block lengths:
  - 128 bits in AES
  - 64 bits in DES

- In practice, block ciphers are used more often than stream ciphers.
- Stream ciphers are small and fast, they are used smaller devices with less resources.
- Traditionally, it was assumed that stream ciphers are more efficient. Modern block ciphers are now very fast.

Stream Ciphers: Encryption, Decryption

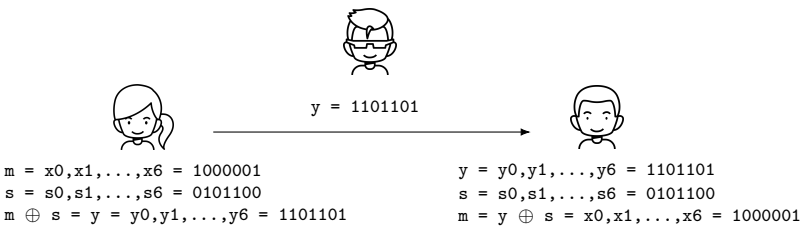
$x_i, y_i, z_i \in \{0, 1\}$
Encryption: $y_i = e_{s_i}(x_i) = x_i + s_i \bmod 2$
Decryption: $x_i = d_{s_i}(y_i) = y_i + s_i \bmod 2$

Encryption and Decryption are the same functions

Why addition mod 2?

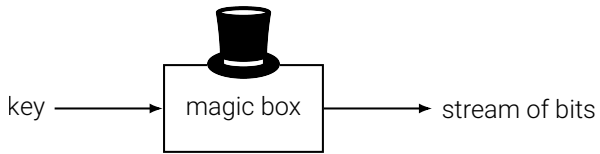
- The same as XOR
- If the key is random, it is impossible to extract the plaintext from the ciphertext.

Example:



The key stream

- The security of the stream cipher depends completely on the key stream.
- It is an algorithm that generate an unbounded number of secret bits.
- The actual key is used as an input/seed
- The more “random looking” the bits the better.



Random Number Generators

- True Random Number Generators
  - The output cannot be reproduced
- How can we build them?
- Maybe using a quantum computer?

- Pseudorandom Number Generators (PRNG)
  - Generated using a function
  - Uses a seed
- Example:

  $s_0 =$ seed
  $s_{i+1} \equiv as_i + b \bmod m$

  The `rand()` function in ANSI C:
  $s_0 = 12345$
  $s_{i+1} \equiv 1103515245 s_i + 12345 \bmod 2^{31}$

Cryptographically secure PRNG

- A PRNG which is unpredictable.
  - Given a bunch of bits of the stream, it is computationally infeasible to compute the subsequent bits.
  - More precisely: for some parameter n, given a polynomials (in n) number of consecutive bits of the stream, there is no polynomial time (in n) algorithm that can predict the next bit.

Do cryptographically secure PRNGs exist?

The One-Time Pad

Unconditional Security

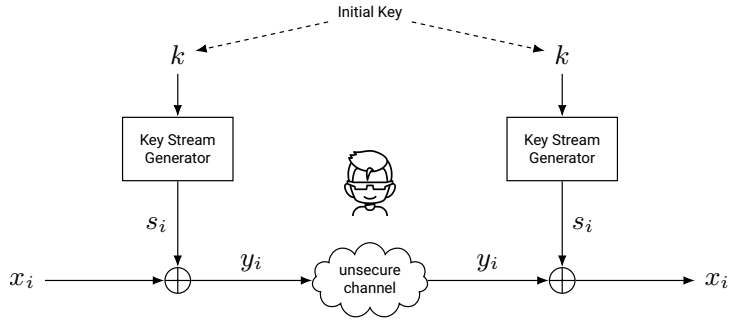
A cryptosystem is unconditionally (or information-theoretically) secure if it cannot be broken even with infinite computational resources.

One-Time Pad (OTP)

1. the key stream $s_0, s_1, \ldots$ is generated by a true random number generator
2. every key stream bit s_i is used only once

OTP is Unconditionally Secure.

Practical stream ciphers



Computational Security

A cryptosystem is computationally secure if there is no known polynomial time algorithm that can break it.

Key streams from PRNGs

- Many PRNGs have good statistical properties, e.g., the output bit stream look random
- Example: for the key $k = (a, b)$,

  $s_0 = \text{seed}$
  $s_{i+1} \equiv as_i + b \bmod m$

- But, is looking random good enough?

- Attack: Trudy obtains

  $s_2 \equiv as_1 + b \bmod m$
  $s_3 \equiv as_2 + b \bmod m$

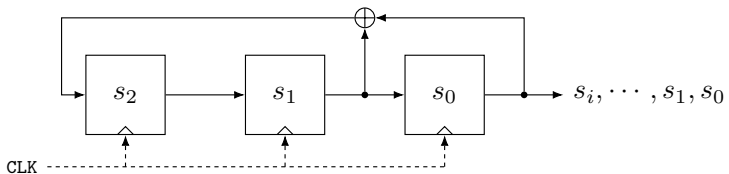
- Then he can easily solve for $a, b \bmod m$.

We need cryptographically secure PRNGs.

Shift Register-Based Stream Ciphers

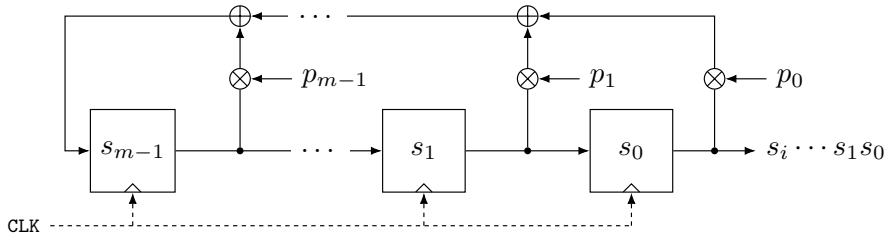
- Linear Feedback Shift Registers (LFSR)
  - An LFSR consists of clocked storage elements (flip-flops) and a feedback path
  - The number of storage elements gives us the degree of the LFSR

Example:



$$s_3 = s_1 + s_0 \bmod 2$$

General LFSRs



- If $p_i = 1$, the feedback is active. If $p_i = 0$, the feedback is inactive.
- Mathematically:

$$s_m = p_{m-1}s_{m-1} + \cdots + p_1 s_1 + p_0 s_0 \bmod 2$$

LFSRs are periodic

- Mathematically, LFSRs are described by linear recurrences (as we just saw)
- After a finite number of output bits the sequence will repeat
- The length of the period is called the length of the LFSR

Example: the LFSR $s_3 = s_1 + s_0 \bmod 2$ has length 7. For the initial value $(s_0 = 0, s_1 = 0, s_2 = 1)$, the output is `0010111 0010111 0010111 ...`

Theorem

The maximum length of an LFSR of degree m is $2^m - 1$.

LFSRs for stream ciphers?

- If we use an LFSR as a stream cipher, the secret key is the set of coefficients $(p_{m-1}, \ldots, p_1, p_0)$.
- If Trudy knows some plaintexts, he can lunch an attack called the **Known-Plaintext Attact.**
- Suppose Trudy knows the plaintext bits

$$x_0, x_1, \ldots, x_{2m-1}$$

and the corresponding ciphertext bits

$$y_0, y_1, \ldots, y_{2m-1}$$

Then he can compute the first $2m$ key stream bits:

$$s_i = x_i + y_i \bmod 2, \qquad i = 0, 1, \ldots, 2m-1.$$

Known-Plaintext Attack

The recurrence relation for the LFSR is

$$s_{i+m} = p_{m-1}s_{i+m-1} + \cdots + p_1 s_{i+1} + p_0 s_i \bmod 2, \quad s_i, p_j \in \{0,1\}, \quad i = 0, 1, \ldots$$

This gives us

$$\begin{array}{lll} i=0 & s_m & = p_{m-1}s_{m-1} + \cdots + p_1 s_1 + p_0 s_0 \mod 2 \\ i=1 & s_{m+1} & = p_{m-1}s_m + \cdots + p_1 s_2 + p_0 s_1 \mod 2 \\ \vdots & \vdots & \vdots \\ i=m-1 & s_{2m-1} & = p_{m-1}s_{2m-2} + \cdots + p_1 s_m + p_0 s_{m-1} \mod 2 \end{array}$$

- A system of linear equations $\bmod 2$!
- So, what should we do?
  $\Rightarrow$ Use colorred multiple LFSRs with nonlinear components

Multiple LFSRs

- Use nonlinear components to avoid attacks like the one we just saw

  Example: A5/1