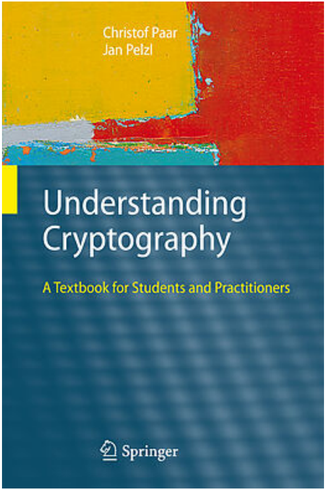
Introduction to Cryptography and Security

Stream ciphers

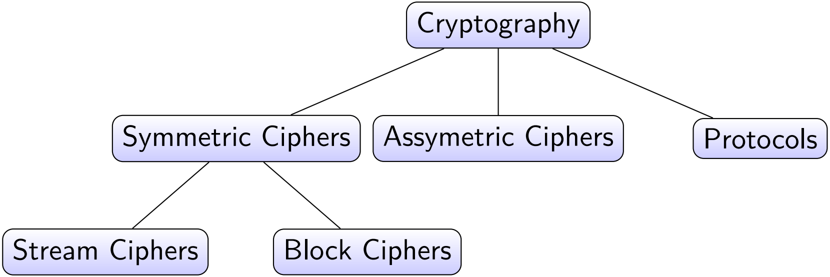
Textbook

https://www.crypto-textbook.com

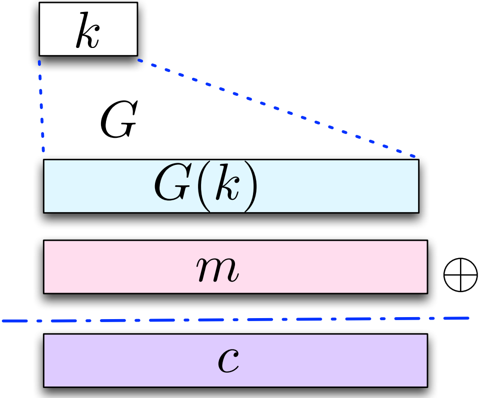


Outline

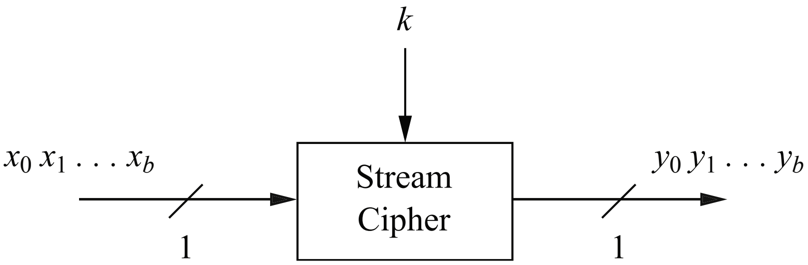
Cryptography



One-time pad is a Stream Cipher

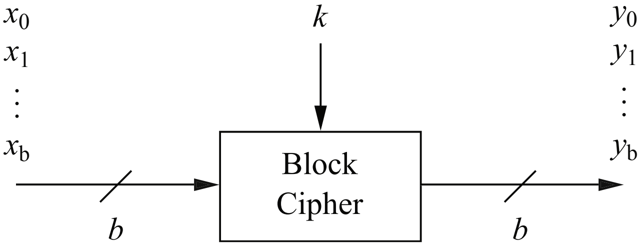


Stream ciphers



Stream ciphers encrypt bits individually.

Block ciphers



Block ciphers encrypt an entire block of plaintext bits at a time with the same key.

Example

- Block length of AES is 128 bit;
- Block length of DES and 3DES is 64 bit.

Outline

Definition (Stream Cipher Encryption and Decryption)

The plaintext, the ciphertext and the key stream consist of individual bits, i.e

$$x_i,\ \ y_i,\ \ s_i \in \{0, 1\}.$$

- Encryption: $y_i = \mathrm{Enc}(s_i, x_i) = x_i \oplus s_i$.
- Decryption: $x_i = \mathrm{Dec}(s_i, y_i) = y_i \oplus s_i$.

Review: One Time Pad

A stream cipher for which

- the key stream $s_0, s_1, s_2, \ldots$ is generated by a true random number generator, and
- the key stream is only known to the legitimate communicating parties, and
- every key stream bit s_i is only used once

is called a one-time pad.

Example

Alice wants to encrypt the letter `A`, where the letter is given in ASCII code. The ASCII value for 'A' is $65_{10} = 1000001_2$. Let's furthermore assume that the first key stream bits are $(s_0, \ldots, s_6) = 0101100$.

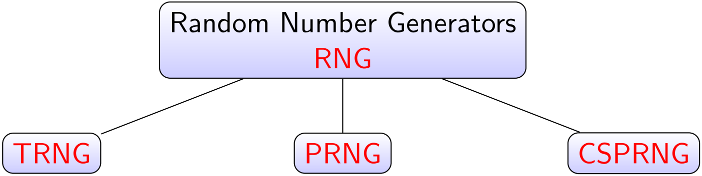
Alice	**Oscar**	**Bob**
$x_0, \ldots, x_6 = 1000001 = \mathtt{A}$		
$\oplus$		
$s_0, \ldots, s_6 = 0101100$		
$y_0, \ldots, y_6 = 1101101 = \mathtt{m}$		
	$\xrightarrow{m=1101101}$	
		$y_0, \ldots, y_6 = 1101101 = \mathtt{m}$
		$\oplus$
		$s_0, \ldots, s_6 = 0101100$
		$x_0, \ldots, x_6 = 1000001 = \mathtt{A}$

What Exactly Is the Nature of the Key Stream?

- Question: how do we get the key stream s_i?
- Response: This is a major topic and is discussed later in this lecture.
- Guess: The key stream bits should be that they appear like a random sequence to an attacker.

Outline

Random Number Generators



Three types of random number generators:

- Random Number Generator
- True Random Number Generators
- (General) Pseudorandom Number Generators
- Cryptographically Secure Pseudorandom Number Generators

True Random Number Generators (TRNG)

- True random number generators (TRNGs) are characterized by the fact that their output cannot be reproduced.
- Example: coin flipping, rolling of dice, semiconductor noise, clock jitter in digital circuits and radioactive decay.
- In GNU/Linux, random bytes can be taken from /dev/random.

```
> hexdump -C -n 8 /dev/random
0000000010596941 66    |.Yi....f|
00000008
```

(General) Pseudorandom Number Generators (PRNG)

- Pseudorandom number generators (PRNGs) generate sequences which are computed from an initial `seed` value.
- Often they are computed recursively in the following way:

$$
\begin{aligned}
s_0 &= \texttt{seed} \\
s_{i+1} &= f(s_i), \quad i = 0, 1, \ldots
\end{aligned}
$$

- A generalization of this are generators of the form

$$s_{i+1} = f(s_i, s_{i-1}, \ldots, s_{i-t})$$

where t is a fixed integer.

Example

The `rand()` function used in ANSI C.

$$
\begin{aligned}
s_0 &= 12345 \\
s_{i+1} &= 1103515245 \cdot s_i + 12345 \mod 2^{31}, \quad i = 0, 1, \ldots
\end{aligned}
$$

Example with `rand()`

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(int argc, char **argv)
{
    srand(time(0));
    printf("%d\n", rand());0;
}
```

- The `time(0)` function returns the time as the number of seconds since the Epoch, 1970-01-01 00:00:00 +0000 (UTC).
- This value is used as a seed.

Cryptographically Secure Pseudorandom Number Generators (CSPRNG)

CSPRNGs are a special type of PRNG which possess the following additional property:

A CSPRNG is PRNG which is unpredictable.

Definition (Unpredictable)

Given n consecutive bits of the key stream, there is no polynomial time algorithm that can predict the next bit s_{n+1} with better than 50% chance of success.

Question

Suppose $s = s_1, \ldots, s_n$ such that

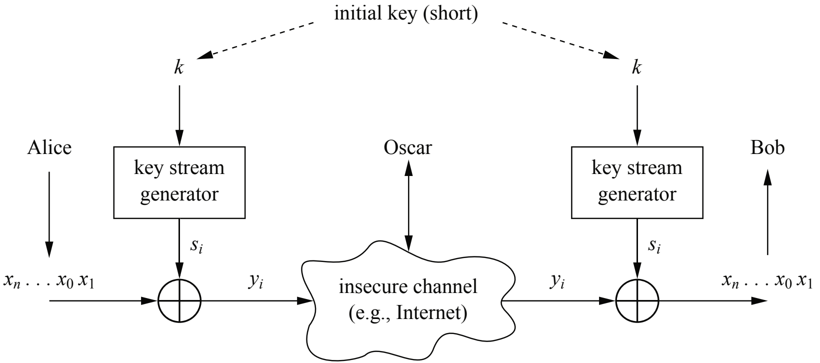
$$s_1 \oplus \cdots \oplus s_n = 1.$$

Is s is predictable?

- Yes, given the first bit I can predict the second
- No, s is unpredictable
- Yes, given the first $(n-1)$ bit I can predict the n'th bit
- It depends

Outline

Practical stream ciphers



Building Key Streams from PRNGs

Example (Linear Congruential Generator)

$$
\begin{aligned}
S_0 &= \texttt{seed} \\
S_{i+1} &= A \cdot S_i + B \mod m, \quad i = 0, 1, \ldots
\end{aligned}
$$

where we choose m to be 100 bits long and

$$S_i, A, B \in \{0, 1, \ldots, m-1\}.$$

The secret key comprises the values (A, B) and possibly the seed S_0.

Attack LCG

Assume Oscar knows the first 300 bits of plaintext, he can now compute the first 300 bits of key stream $s_1, \ldots, s_{300}$. How?

1. Oscar computes

$$S_1 = (s_1, \ldots, s_{100}),\ S_2 = (s_{101}, \ldots, s_{200}),\ S_3 = (s_{201}, \ldots, s_{300}).$$

2. Oscar can now generate two equations:

$$\begin{aligned} S_2 &= A \cdot S_1 + B \mod m \\ S_3 &= A \cdot S_2 + B \mod m \end{aligned}$$

This is a system of linear equations over $\mathbb{Z}_m$ with two unknowns A and B!

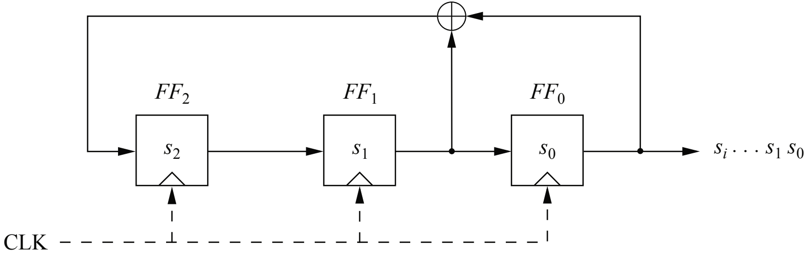
Attack LCG

$$A = (S_2 - S_3)(S_1 - S_2)^{-1} \mod m$$
$$B = S_2 - S_1(S_2 - S_3)(S_1 - S_2)^{-1} \mod m$$

Do not use LCG to generate key streams!

Outline

Linear Feedback Shift Registers (LFSR)



Hình: Linear feedback shift register of degree $m = 3$ with initial values FF_2, FF_1 và FF_0

In general, the output bit is computed as:

$$s_{i+3} = s_{i+1} + s_i \mod 2.$$

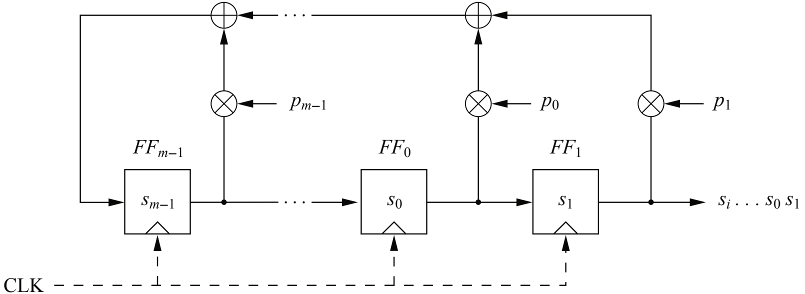
Example

$$s_{i+3} = s_{i+1} \oplus s_i$$

clk	FF_2	FF_1	$FF_0 = s_i$
0	1	0	0
1	0	1	0
2	1	0	1
3	1	1	0
4	1	1	1
5	0	1	1
6	0	0	1
7	1	0	0
8	0	1	0

Output: $0010111\ 0010111\ 0010111\ \ldots$

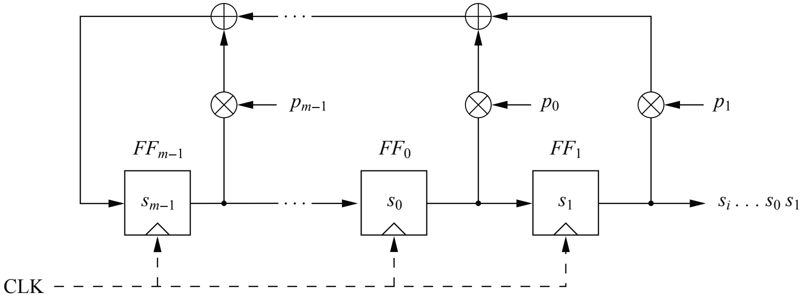
General LFSR of degree m



Hình: General LFSR with feedback coefficients p_i and and initial values $s_{m-1}, \ldots, s_0$

Linear recurrences

$$s_{i+m} = \sum_{j=0}^{m-1} p_j \cdot s_{i+j} \mod 2$$



Exercise 1

- Given an LFSR of degre $m = 4$, the feedback path

$$(p_3 = 0, p_2 = 0, p_1 = 1, p_0 = 1).$$

and the initial state

$$s_3 = 0,\ s_2 = 1,\ s_1 = 0,\ s_0 = 0.$$

- Compute the next 15 output bits.

Exercise 2

- Given an LFSR of degree $m = 4$, the feedback path

$$(p_3 = 1, p_2 = 1, p_1 = 1, p_0 = 1).$$

and the initial state

$$s_3 = 0,\ s_2 = 1,\ s_1 = 0,\ s_0 = 1.$$

- Compute the next 15 output bits.

Maximum length of an LFSR

- LFSR is defined by linear recurrence:

$$s_{i+m} = \sum_{j=0}^{m-1} p_j s_{i+j} \mod 2; \qquad s_i, p_j \in \{0, 1\}; \quad i = 0, 1, 2, \ldots$$

- LFSR can produce output sequences of different lengths, depending on the feedback coefficients.

Theorem

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

Example

Given an LFSR of degree $m = 4$ and the feedback path

$$(p_3 = 0, p_2 = 0, p_1 = 1, p_0 = 1).$$

the output sequence of the LFSR has a period $2^4 - 1 = 15$, i.e., it is a maximum-length LFSR.

Example

Given an LFSR of degree $m = 4$ and

$$(p_3 = 1, p_2 = 1, p_1 = 1, p_0 = 1).$$

Then the output sequence has period of 5; therefore, it is not a maximum-length LFSR.

LSFR and polynomial

LFSR of degree m with a feedback coefficient vector $(p_{m-1}, \ldots, p_1, p_0)$ is represented by the polynomial

$$P(x) = x^m + p_{m-1}x^{m-1} + \cdots + p_1 x + p_0$$

Example

LFSR of degree $m = 4$ with a feedback coefficient vector

$$(p_3 = 0, p_2 = 0, p_1 = 1, p_0 = 1)$$

is represented by the polynomial

$$P(x) = x^4 + x + 1.$$

Example

LFSR of degree $m = 4$ with a feedback coefficient vector

$$(p_3 = 1, p_2 = 1, p_1 = 1, p_0 = 1)$$

is represented by the polynomial

$$P(x) = x^4 + x^3 + x^2 + x + 1.$$

Primitive polynomials and LSFR

- Maximum-length LFSRs have what is called primitive polynomials!
- Primitive polynomials are a special type of irreducible polynomial.
- Irreducible polynomials are roughly comparable with prime numbers, i.e., their only factors are 1 and the polynomial itself.
- Example:

$$(0, 2, 5) \rightarrow 1 + x^2 + x^5$$

is a primitive polynomial.

Primitive polynomials

(0,1,2)	(0,1,3,4,24)	(0,1,46)	(0,1,5,7,68)	(0,2,3,5,90)	(0,3,4,5,112)
(0,1,3)	(0,3,25)	(0,5,47)	(0,2,5,6,69)	(0,1,5,8,91)	(0,2,3,5,113)
(0,1,4)	(0,1,3,4,26)	(0,2,3,5,48)	(0,1,3,5,70)	(0,2,5,6,92)	(0,2,3,5,114)
(0,2,5)	(0,1,2,5,27)	(0,4,5,6,49)	(0,1,3,5,71)	(0,2,93)	(0,5,7,8,115)
(0,1,6)	(0,1,28)	(0,2,3,4,50)	(0,3,9,10,72)	(0,1,5,6,94)	(0,1,2,4,116)
(0,1,7)	(0,2,29)	(0,1,3,6,51)	(0,2,3,4,73)	(0,11,95)	(0,1,2,5,117)
(0,1,3,4,8)	(0,1,30)	(0,3,52)	(0,1,2,6,74)	(0,6,9,10,96)	(0,2,5,6,118)
(0,1,9)	(0,3,31)	(0,1,2,6,53)	(0,1,3,6,75)	(0,6,97)	(0,8,119)
(0,3,10)	(0,2,3,7,32)	(0,3,6,8,54)	(0,2,4,5,76)	(0,3,4,7,98)	(0,1,3,4,120)
(0,2,11)	(0,1,3,6,33)	(0,1,2,6,55)	(0,2,5,6,77)	(0,1,3,6,99)	(0,1,5,8,121)
(0,3,12)	(0,1,3,4,34)	(0,2,4,7,56)	(0,1,2,7,78)	(0,2,5,6,100)	(0,1,2,6,122)
(0,1,3,4,13)	(0,2,35)	(0,4,57)	(0,2,3,4,79)	(0,1,6,7,101)	(0,2,123)
(0,5,14)	(0,2,4,5,36)	(0,1,5,6,58)	(0,2,4,9,80)	(0,3,5,6,102)	(0,37,124)
(0,1,15)	(0,1,4,6,37)	(0,2,4,7,59)	(0,4,81)	(0,9,103)	(0,5,6,7,125)
(0,1,3,5,16)	(0,1,5,6,38)	(0,1,60)	(0,4,6,9,82)	(0,1,3,4,104)	(0,2,4,7,126)
(0,3,17)	(0,4,39)	(0,1,2,5,61)	(0,2,4,7,83)	(0,4,105)	(0,1,127)
(0,3,18)	(0,3,4,5,40)	(0,3,5,6,62)	(0,5,84)	(0,1,5,6,106)	(0,1,2,7,128)
(0,1,2,5,19)	(0,3,41)	(0,1,63)	(0,1,2,8,85)	(0,4,7,9,107)	
(0,3,20)	(0,1,2,5,42)	(0,1,3,4,64)	(0,2,5,6,86)	(0,1,4,6,108)	
(0,2,21)	(0,3,4,6,43)	(0,1,3,4,65)	(0,1,5,7,87)	(0,2,4,5,109)	
(0,1,22)	(0,5,44)	(0,3,66)	(0,8,9,11,88)	(0,1,4,6,110)	
(0,5,23)	(0,1,3,4,45)	(0,1,2,5,67)			

Outline

Assumption

Oscar knows:

- Let the known plaintext be given $(x_0, x_1, \ldots, x_{2m-1})$
- and the corresponding ciphertext by $(y_0, y_1, \ldots, y_{2m-1})$
- the degree m m.

Step 1

Oscar computes

$$y_i = x_i + s_i \mod 2$$
$$s_i = y_i + x_i \mod 2$$

for $i = 0, 1, \ldots, 2m - 1$

Step 2

- Goal: Get the key stream

$$s_{2m},\ s_{2m+1},\ \dots$$

- Question: How to compute the feedback coefficients?

$$p_0,\ p_1,\ \dots,\ p_{m-1}?$$

Recall:

$$s_{i+m} = \sum_{j=0}^{m-1} p_j \cdot s_{i+j} \mod 2$$

Step 2: Compute p_i

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

- This is a system of linear equations in m unknowns $p_0, p_1, \ldots, p_{m-1}$.
- This system can easily be solved by Oscar using Gaussian elimination, matrix inversion or any other algorithm for solving systems of linear equations!

Consequences

As soon as Oscar knows $2m$ output bits of an LFSR of degree m, the p_i coefficients can exactly be constructed by merely solving a system of linear equations

$$p_0, \; p_1, \; \ldots, \; p_{m-1}.$$

Oscar can now clock the LFSR and produce the entire output sequence.

Step 3

- Using the p_i coefficients

$$(p_{m-1}, \ldots, \; p_1, p_0)$$

  to construct LFSR.
- Compute the key stream

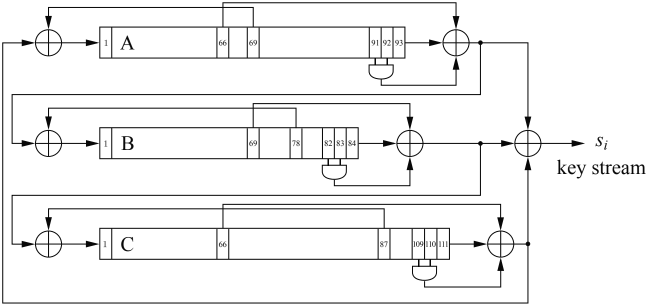
$$s_0, s_1, \ldots, s_{2m}, \ldots$$

- Decode

$$x_i = y_i + s_i \mod 2.$$

Outline

What is Trivium?



- Trivium is a relatively new stream cipher which uses an 80-bit key.
- It is based on a combination of three shift registers.
- There are nonlinear components used to derive the output of each register

Description of Trivium

	register length	feedback bit	feedforward bit	AND inputs
A	93	69	66	91,92
B	84	78	69	82,83
C	111	87	66	109,110

- The AND operation is equal to multiplication in modulo 2 arithmetic.
- The resulting equations are no longer linear as they contain products of two unknowns.
- Feedforward paths involving the AND operation are crucial for the security of Trivium as they prevent attacks that exploit linearity of the cipher.

Encryption with Trivium

Initialization

- An 80 bit IV is loaded into the 80 leftmost locations of register A.
- 80 bit key is loaded in the 80 leftmost locations of register B.
- All other register bits are set to zero with the exception of the three rightmost bits of register C:

$$c_{109} = c_{110} = c_{111} = 1.$$

Encryption with Trivium

Warm-up Phase

- In the first phase, the cipher is clocked

$$4 \times (93 + 84 + 111) = 1152$$

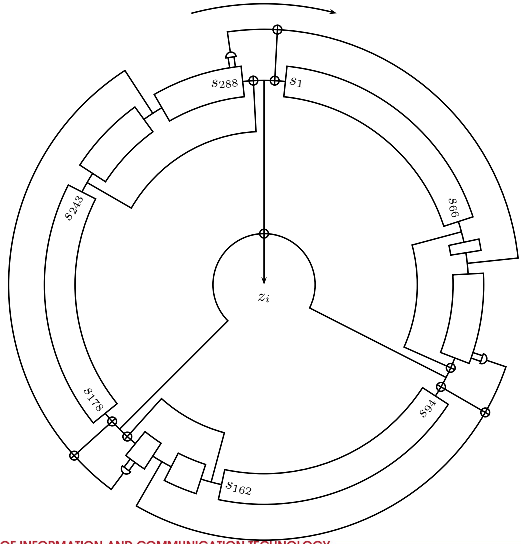
  times. No cipher output is generated.
- This phase is required to generate enough random elements for the cipher.
- It ensures the key stream depends on both k and IV.

Encryption with Trivium

- The bits produced hereafter, i.e., starting with the output bit of cycle 1153, form the key stream s_i.
- An attractive feature of Trivium is its compactness, especially if implemented in hardware.
- The encryption speed is very high: About 1Gbit/s on Intel's 1.5 GHz processor.
- Even though there are no known attacks at the time of writing, one should keep in mind that Trivium is a relatively new cipher and attacks in the future are certainly a possibility.

Specification of Trivium

https://www.ecrypt.eu.org/stream/p3ciphers/trivium/trivium_p3.pdf



Programming Assigment 2

1. Implement the Trivium stream cipher.
2. Encrypt files with Trivium. You can generate a random IV and place it at the beginning of the ciphertext.

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Thank you!

soict.hust.edu.vn/ fb.com/groups/soict