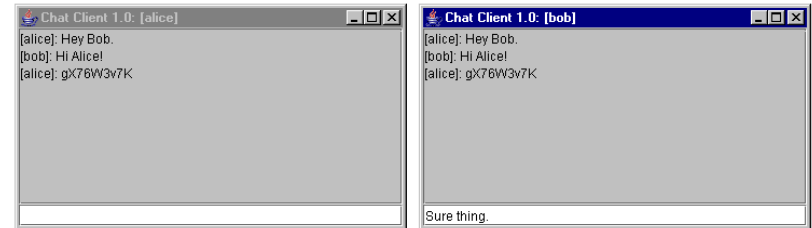


1.0. Overview

Secure Chat

Alice wants to send a secret message to Bob?

- Can you read the secret message `gX76W3v7K` ?
- But Bob can. How?



A Cryptographic Example

How to make a simple machine that produces "random" bits.

- Linear feedback shift register.

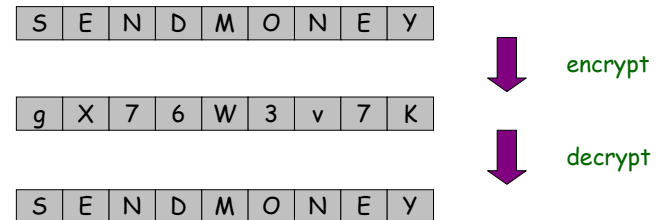
What we can do with it.

- ➔ • Encrypt and decrypt secret messages.
- Encrypt DVDs with CSS.
- Decrypt DVDs with DeCSS !
- Use as subroutine in military cryptosystems.

Science behind it.

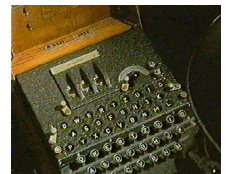
Encryption Machine

Goal: design a machine to encrypt and decrypt data.



Enigma encryption machine.

- "Unbreakable" German code during WWII.
- Broken by Turing bombe.
- One of first uses of computers.
- Helped win Battle of Atlantic by locating U-boats.



Digital Data

Digital data.

- Computers store all data as a sequence of bits.
- Text, images (JPEG), audio (MP3), video (DivX).

Bit = 0 or 1

Base64 encoding.

- Use 6 bits to represent each alphanumeric symbol.

Binary Char	Binary Char	Binary Char	Binary Char	Binary Char	Binary Char
000000	A	001011	L	010110	W
000001	B	001100	M	010111	X
000010	C	001101	N	011000	Y
000011	D	001110	O	011001	Z
000100	E	001111	P	011010	a
000101	F	010000	Q	011011	b
000110	G	010001	R	011100	c
000111	H	010010	S	011101	d
001000	I	010011	T	011110	e
001001	J	010100	U	011111	f
001010	K	010101	V	100000	g
				101011	h
				101100	s
				101101	t
				101110	u
				101111	v
				110000	w
				110001	x
				110010	y
				110011	z
				111000	0
				111001	1
				111010	2
				111011	3
				111100	4
				111101	5
				111110	6
				111111	7
					8
					9
					+
					/

6

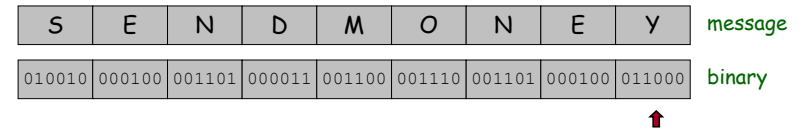
One-Time Pad Encryption

1. Convert text message to N bits.

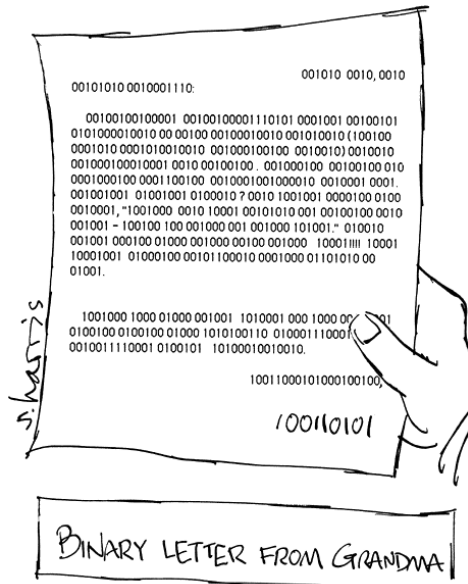
Bit = 0 or 1

Base64 Encoding

char	dec	binary
A	0	000000
B	1	000001
...
Y	24	011000
...



7

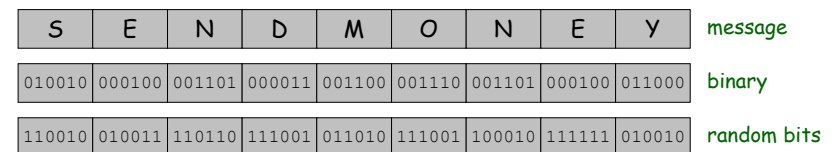


Copyright 2004, Sidney Harris

8

One-Time Pad Encryption

- Convert text message to N bits.
- Generate N random bits (one-time pad).



9

A Russian One-Time Pad



Random Numbers

Are these 2000 numbers random?

```
11001001001110110111001011010111001100010111110100100001001101001011100110010011111
10111000001010110001000011101010011010000111001001100111011111110101000001000010001010
010101000110000010111000100100101010111000010100110110011101011110010001001110101
011101000001010010001000110101011100000010110000010011000101110101001010100110000
1111110011000001111100011000010111001110100111010011100100110111010101010101000
0000000100000001010000010001000010101010010000001010000011100100010111010111010100
0101000010100010001001010110101000011000010011100101110011100101111011100100101011011
00001010111001000010111010010010100110100011101101100101010111000000100110000101111
1001001000111011010101011000110001101110101001011000011001110011111011100001010
011001000111110101010000100011001010101110000101011001110001111010100010101011010
0110101001111000011001100110111111101000000100100000101101000100110010101111100001
00001100101001111000110001101101011011010101010101010100000101110001110101010100011
0110010111011100101010011100000110110001101011011100010101010100000010010000111110
100110001001111010111000100010110101010011000000111100001100011001111011111001010000
11100010011010101110110001001011101011001010001110001011001101001111100111000011110
1100110010111111100100000011101000011010010011100110111011110101010001000000101010000
10000010010100010110001010011101000110100101101001100110011111111000000001100000001
11110000011001100011111110110000001011100001001011001011001111001111001111001110011
0110011110111100010100011010001011100101001011100011001010111100110100011110010110
00111001101011101010100100010011010111110001000001101010001110000101101001001010
111101110100101001001100011011110111010001010100101000001100010001111010101100100001
111010001100100101111011001000101110101001001000001101101001101100111010111101000100
01001010101010000000111000000110110000110111001101010111100000100011000101010111010
```

If not, what is the pattern?

One-Time Pad Encryption

- Convert text message to N bits.
- Generate N random bits (one-time pad).
- Take bitwise XOR of two strings.
 - Sum pair of bits (1 if sum is odd, 0 if even).

XOR Truth Table

x	y	x ^ y
0	0	0
0	1	1
1	0	1
1	1	0

S	E	N	D	M	O	N	E	Y	message
010010	000100	001101	000011	001100	001110	001101	000100	011000	binary
110010	010011	110110	111001	011000	111001	100010	111111	010010	random bits
100000	010111	111011	111010	010110	110111	101111	111011	001010	XOR

One-Time Pad Encryption

- Convert text message to N bits.
- Generate N random bits (one-time pad).
- Take bitwise XOR of two strings.
- Convert binary back into text.

Base64 Encoding

char	dec	binary
A	0	000000
B	1	000001
...
X	23	010111
...

S	E	N	D	M	O	N	E	Y	message
010010	000100	001101	000011	001100	001110	001101	000100	011000	binary
110010	010011	110110	111001	011010	111001	100010	111111	010010	random bits
100000	010111	111011	111010	010110	110111	101111	111011	001010	XOR
g	X	7	6	W	3	v	7	K	encrypted

One-Time Pad Decryption

1. Convert encrypted message to binary.

g	X	7	6	W	3	v	7	K
---	---	---	---	---	---	---	---	---

encrypted

14

One-Time Pad Decryption

1. Convert encrypted message to binary.

Base64 Encoding

char	dec	binary
A	0	000000
B	1	000001
...
X	23	010111
...

g	X	7	6	W	3	v	7	K
---	---	---	---	---	---	---	---	---

encrypted

100000	010111	111011	111010	010110	110111	101111	111011	001010
--------	--------	--------	--------	--------	--------	--------	--------	--------

binary

15

One-Time Pad Decryption

1. Convert encrypted message to binary.
2. Use same N random bits (one-time pad).

g	X	7	6	W	3	v	7	K
---	---	---	---	---	---	---	---	---

encrypted

100000	010111	111011	111010	010110	110111	101111	111011	001010
--------	--------	--------	--------	--------	--------	--------	--------	--------

binary

110010	010011	110110	111001	011010	111001	100010	111111	010010
--------	--------	--------	--------	--------	--------	--------	--------	--------

random bits

16

One-Time Pad Decryption

1. Convert encrypted message to binary.
2. Use same N random bits (one-time pad).
3. Take bitwise XOR of two strings.

XOR Truth Table

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

g	X	7	6	W	3	v	7	K
---	---	---	---	---	---	---	---	---

encrypted

100000	010111	111011	111010	010110	110111	101111	111011	001010
--------	--------	--------	--------	--------	--------	--------	--------	--------

binary

110010	010011	110110	111001	011010	111001	100010	111111	010010
--------	--------	--------	--------	--------	--------	--------	--------	--------

random bits

010010	000100	001101	000011	001110	001110	001101	000100	011000
--------	--------	--------	--------	--------	--------	--------	--------	--------

XOR

17

One-Time Pad Decryption

1. Convert encrypted message to binary.
2. Use same N random bits (one-time pad).
3. Take bitwise XOR of two strings.
4. Convert back into text.

Base64 Encoding

char	dec	binary
A	0	000000
B	1	000001
...
Y	24	011000
...

g	X	7	6	W	3	v	7	K	encrypted
100000	010111	111011	111010	010110	110111	101111	111011	001010	binary
110010	010011	110110	111001	011010	111001	100010	111111	010010	random bits
010010	000100	001101	000011	001100	001110	001101	000100	011000	XOR
S	E	N	D	M	O	N	E	Y	message

18

Why Does It Work?

Notation	Meaning
a	original message
b	one-time pad
^	XOR operator
a ^ b	encrypted message
(a ^ b) ^ b	decrypted message

Crucial property: $(a \oplus b) \oplus b = a$.

- Decrypted message = original message.

Why is crucial property true?

- Use properties of XOR.
- $(a \oplus b) \oplus b = a \oplus (b \oplus b) = a \oplus 0 = a$

↑
always 0

19

An Cryptographic Example

How to make a simple machine that produces "random" bits.

- Linear feedback shift register.

➔ What we can do with it.

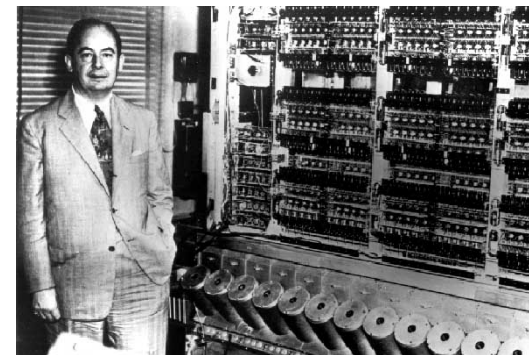
- Encrypt and decrypt secret messages.
- Encrypt DVDs with CSS.
- Decrypt DVDs with DeCSS!
- Use as subroutine in military cryptosystems.

Science behind it.

20

Random Numbers

"Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin."



Jon von Neumann (left), ENIAC (right)

21

Linear Feedback Shift Register

How might the "random number machine" be built?

- Linear feedback shift register.
- Linear congruential generator.

Some terminology

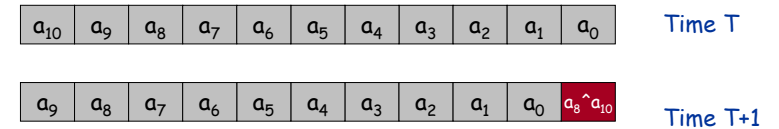
- Bit: 0 or 1.
- Cell: storage element that holds one bit.
- Register: sequence of cells.
- Shift register: when clock ticks, bits propagate one position to left.

22

Linear Feedback Shift Register

Linear feedback shift register.

- Machine consists of 11 bits.
- Bit values change at discrete time points.
- Bit values at time T+1 determined by bit values at time T.
 - new bits 1 - 10 are old bits 0 - 9
 - new bit 0 is XOR of previous bits 8 and 10
 - output bit 0



LFSR Demo 

23

Random Numbers

Are these 2000 numbers random?

```
11001001001111011011100101101011100110001011111101001000010011010010111100110010011111
101110000010101100010000111010100110100001110010011001110111110101000001000010001010
01010100011000001011110001001001101011110001101001101100111101011110010001001110101
0111010000010100100010001101010111000000010110000100111001011101010010101100110000
1111110011000001111100011000011011100111010011110100111001001110111010101010101000
00000001000000010100000100010000101010100000001101000001100100011011010111010100
01010000101000100100010101010100001100001001110010111001110010111011100100101011011
0000101011100100001011101001001010011011000111101110110010101011100000010011000010111
100100100011101010101010001100011011110101010010110000110011100111110111100001010
0110010001111101010100001000111001010101110000110101001110011110110110001010111010
011010100111000011100110011011111110100000010010000010110100010011001010111100001
000011001010011111000111000110110110110110110101010110000001101110001101011010100011
0110010110111001010100111000011101100011010110111000101010101000000110010000111110
1001100010011111010111000100010110101001100000011110000110001100111101111001010000
111000100101010111011000100101110101100101000111000101100110100111110011100001110
1100110010111110010000001101000010100100111001101101110101010001000000101010000
100000100101000101100010100111010001101001011010011001100111111111000000000110000000
1111000001001100011111101100000010111000010010110010110011110011110011110011110011
011001111011110001010001011000101100101001011100011001010111100110100011110010110
00111001110101111010101001000110011010111111000100000110101000110101100010110110010010
1111011110100101001001100011011110110100010101001010000011000100011110101010010000010
1110100011001001011111011001000101111010100100100000110101001110100111010111101000100
01001010101011000000001110000011011000011011100110101011111000001000110001010111010
```

 No. This is output of LFSR!

24

The Science Behind It

Are the bits really random?

 No! Real machines are deterministic.  fill in answer


How did the computer scientist die in the shower?

 The instructions on the shampoo read "lather, rinse, repeat."

Will bit pattern repeat itself?

 Yes, after $2^{11} - 1 = 2047$ steps.


What if I need more bits?

 Scalable: 20 cells for 1 million bits, 30 for 1 billion.

Will the machine work equally well if we XOR bits 4 and 10?

 No, need to understand theory of abstract rings.

How many cells do I need to guarantee a certain level of security?

 Subject of active research.

25

LFSR and "General Purpose Computer"

Important properties.

- Built from simple components.
- Scales to handle huge problems.
- Requires a deep understanding to use effectively.

Basic Component	LFSR	Computer
Control	Start, stop, load	same
Clock	Regular pulse	2.8 GHz pulse
Memory	11 bits	1 GB
Input	Seed	Sequence of bits
Computation	Shift, XOR	Logic, arithmetic, ...
Output	Pseudo-random bits	Sequence of bits

Critical difference. General purpose machine can be programmed to simulate ANY abstract machine.

26

Simulating The Abstract Machine in Java

Java program prints exactly same bits as LFBSR.

- You'll understand this program by next week.

```
public class LFBSR {
    public static void main(String[] args) {
        int N = Integer.parseInt(args[0]);
        boolean b10 = false, b9 = true, b8 = true, b7 = false;
        boolean b6 = true, b5 = false, b4 = false, b3 = false;
        boolean b2 = false, b1 = true, b0 = false;

        for (int i = 0; i < N; i++) {
            boolean bit = b8 ^ b10;
            b10 = b9; b9 = b8; b8 = b7; b7 = b6; b6 = b5;
            b5 = b4; b4 = b3; b3 = b2; b2 = b1; b1 = b0;
            b0 = bit;

            if (bit) System.out.print(1);
            else System.out.print(0);

            System.out.println();
        }
    }
}
```

← update

← print

```
% java LFBSR 2000
010011001000000110001000101
110101011110010011110011101
00100001101111111100101 ...
```

27