CRYPTOGRAPHIC ALGORITHMS AND PROTOCOLS

PART I: CRYPTOGRAPHY

3. Symmetric Key Crypto

3.1. Stream Ciphers

Symmetric Key Crypto

- Stream cipher — based on one-time pad
  - Except that key is relatively short
  - Key is stretched into a long **keystream**
  - Keystream is used just like a one-time pad
- Block cipher — based on codebook concept
  - Block cipher key determines a codebook
  - Each key yields a different codebook
  - Employs both "confusion" and "diffusion"

Stream Ciphers

- Once upon a time, not so very long ago, stream ciphers were the king of crypto
- Today, not as popular as block ciphers
- We'll discuss two stream ciphers…
- A5/1
  - Based on shift registers
  - Used in GSM mobile phone system
- RC4
  - Based on a changing lookup table
  - Used in many places

A5/1: Shift Registers

A5/1 uses 3 shift registers

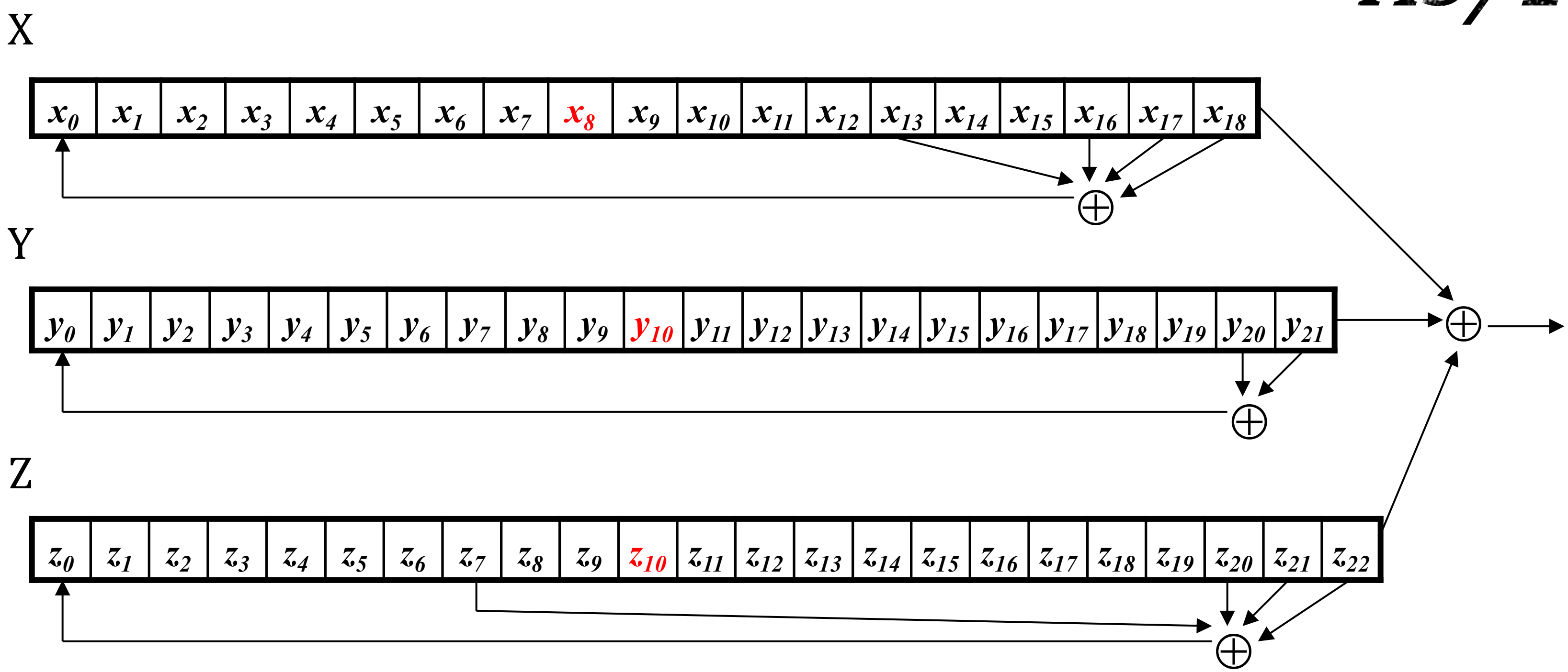
- X: 19 bits $(x_0,x_1,x_2,\ldots,x_{18})$
- Y: 22 bits $(y_0,y_1,y_2,\ldots,y_{21})$
- Z: 23 bits $(z_0,z_1,z_2,\ldots,z_{22})$

- Total: 64 bits → key

A5/1: Keystream

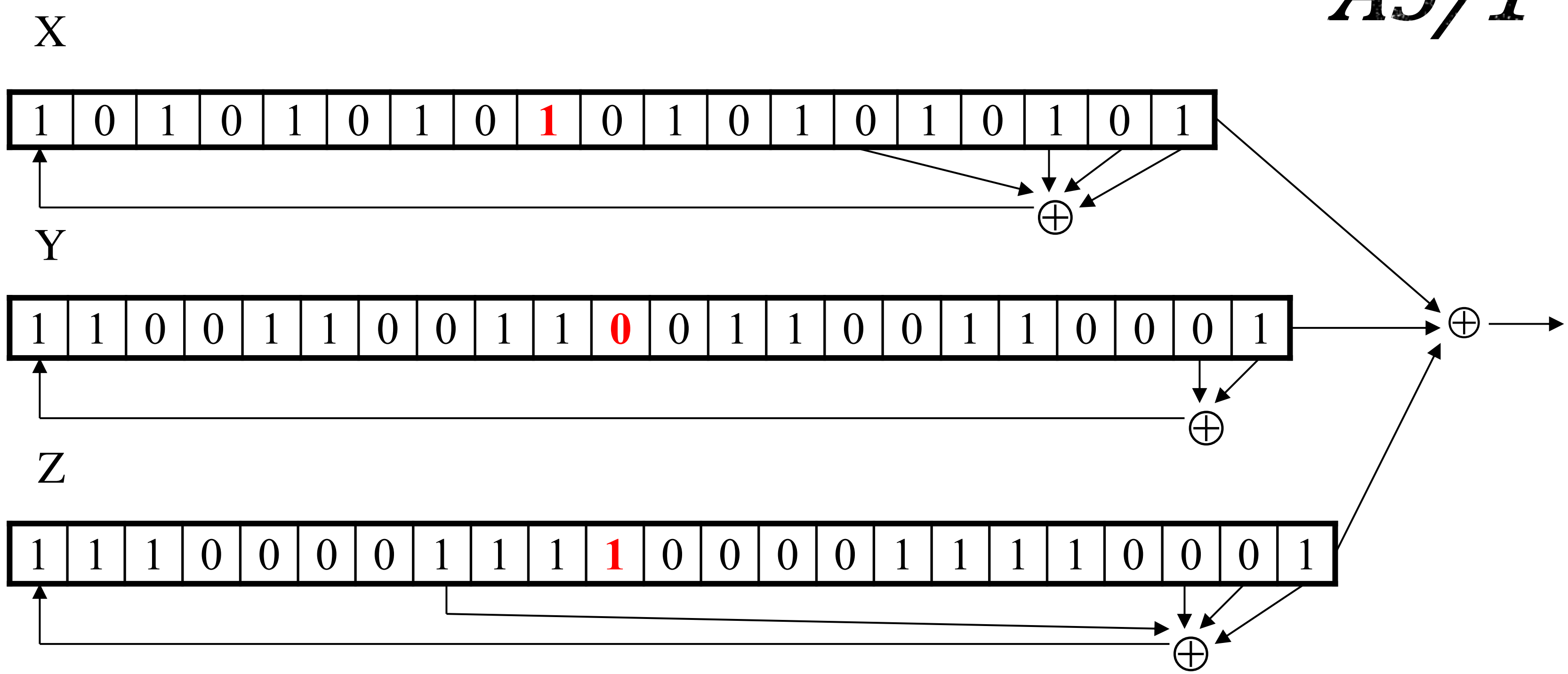
- At each step: $m = \text{maj}(x_8, y_{10}, z_{10})$
  - Examples: maj(0,1,0) = 0 and maj(1,1,0) = 1
- If $x_8 = m$ then X*steps*
  - $x_i = x_{i-1}$ for $i = 18,17,\ldots,1$ and $x_0 = t$
  - $t = x_{13} \oplus x_{16} \oplus x_{17} \oplus x_{18}$
- If $y_{10} = m$ then Y*steps*
  - $y_i = y_{i-1}$ for $i = 21,20,\ldots,1$ and $y_0 = t$
  - $t = y_{20} \oplus y_{21}$
- If $z_{10} = m$ then Z*steps*
  - $z_i = z_{i-1}$ for $i = 22,21,\ldots,1$ and $z_0 = t$
  - $t = z_7 \oplus z_{20} \oplus z_{21} \oplus z_{22}$
- Keystream **bit** is $x_{18} \oplus y_{21} \oplus z_{22}$

A5/1



- Each variable here is a single bit
- Key is used as **initial fill** of registers
- Each register steps (or not) based on $\text{maj}(x_8, y_{10}, z_{10})$
- Keystream bit is XOR of rightmost bits of registers

A5/1



- In this example, $m = \text{maj}(x_8, y_{10}, z_{10}) = \text{maj}(\mathbf{1,0,1}) = \mathbf{1}$
- Register X steps, Y does not step, and Z steps
- Keystream bit is XOR of right bits of registers
- Here, keystream bit will be $0 \oplus 1 \oplus 0 = 1$

Shift Register Crypto

- Shift register crypto efficient in hardware
- Often, slow if implement in software
- In the past, very popular
- Today, more is done in software due to fast processors
- Shift register crypto still used some
  - Resource-constrained devices

RC4

- A self-modifying lookup table
- Table always contains a permutation of the byte values 0,1,…,255
- Initialize the permutation using key
- At each step, RC4 does the following
  - Swaps elements in current lookup table
  - Selects a keystream byte from table
- Each step of RC4 produces a **byte**
  - Efficient in software
- Each step of A5/1 produces only a bit
  - Efficient in hardware

RC4 Initialization

S[ ] is permutation of 0,1,...,255

key[ ] contains N bytes of key

```
for i = 0 to 255
        S[i] = i
        K[i] = key[i (mod N)]
next i
j = 0
for i = 0 to 255
        j = (j + S[i] + K[i]) mod 256
        swap(S[i], S[j])
next i
i = j = 0
```

RC4 Keystream

- For each keystream byte, swap elements in table and select byte

  i = (i + 1) mod 256

  j = (j + S[i]) mod 256

  swap(S[i], S[j])

  t = (S[i] + S[j]) mod 256

  keystreamByte = S[t]

- Use keystream bytes like a one-time pad
- **Note:** first 256 bytes should be discarded, otherwise, related key attack exists

Stream Ciphers

- Stream ciphers were popular in the past
  - Efficient in hardware
  - Speed was needed to keep up with voice, etc.
  - Today, processors are fast, so software-based crypto is usually more than fast enough
- Future of stream ciphers?
  - Shamir declared "the death of stream ciphers"
  - May be greatly exaggerated…
  - Today, stream ciphers are more appropriate than block ciphers for certain applications; like wireless devices, severely resource-constrained devices, and extremely high data-rate systems.