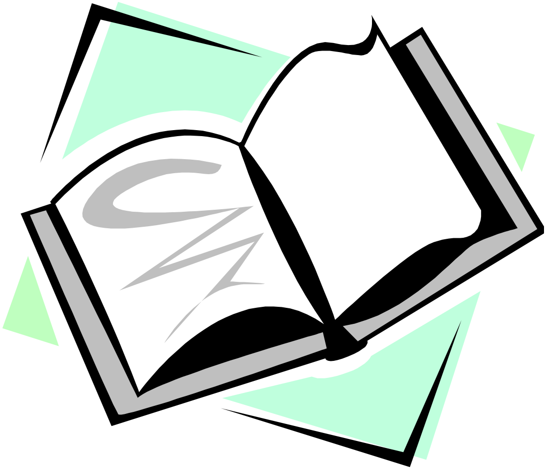
Cryptography
CS 555

Topic 5: Pseudorandomness and Stream Ciphers

Outline and Readings

- Outline
  - Stream ciphers
  - LFSR
  - RC4
  - Pseudorandomness



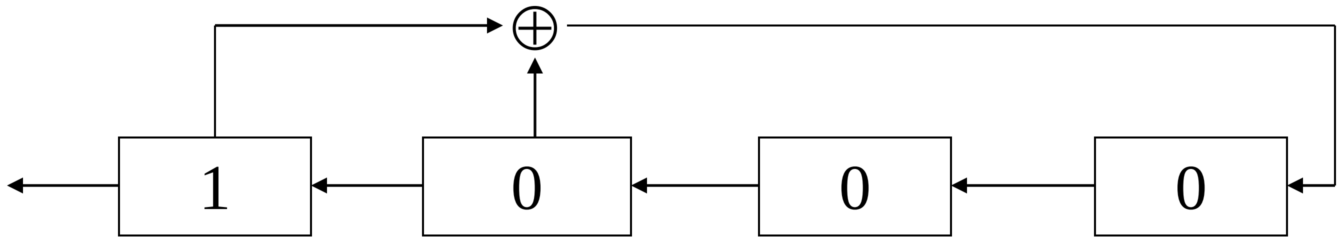
- Readings:
  - Katz and Lindell: 3.3, 3.4.1

Stream Ciphers

- In One-Time Pad, a key is a random string of length at least the same as the message
- Stream ciphers:
  - Idea: replace "rand" by "pseudo rand"
  - Use a Pseudo Random (Number) Generator
  - G: $\{0,1\}^s \rightarrow \{0,1\}^n$
    - expand a short (e.g., 128-bit) random seed into a long (e.g., 10^6 bit) string that "looks random"
  - Secret key is the seed
  - Naïve encryption: $E_{key}[M] = M \oplus G(key)$
  - To encrypt more than one messages, need to be more sophisticated.

Linear Feedback Shift Register (LFSR)

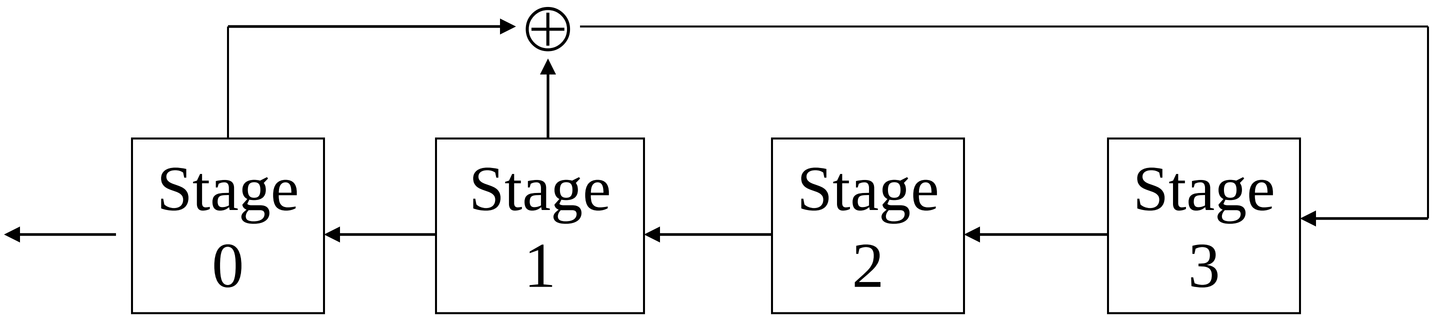
- Example:



- Starting with 1000, the output stream is
  - 1000 1001 1010 1111 000
- Repeat every $2^4 - 1$ bit
- The seed is the key

Linear Feedback Shift Register (LFSR)

- Example:



- $z_i \quad = z_{i-4} + z_{i-3} \bmod 2$
  $= 0 \cdot z_{i-1} + 0 \cdot z_{i-2} + 1 \cdot z_{i-3} + 1 \cdot z_{i-4} \bmod 2$

- We say that stages 0 & 1 are selected.

Properties of LFSR

- Fact: given an L-stage LFSR, every output sequence is periodic if and only if stage 0 is selected
- Definition: An L-stage LFSR is maximum-length if some initial state will results a sequence that repeats every $2^L - 1$ bit
- Whether an LFSR is maximum-length or not depends on which stages are selected.

Cryptanalysis of LFSR

- Vulnerable to know-plaintext attack
  - A LFSR can be described as
    $z_{m+i} = \sum_{j=0}^{m-1} c_j \, z_{i+j} \bmod 2$
  - Knowing 2*m* output bits, one can
    - construct *m* linear equations with *m* unknown variables $c_0, \ldots, c_{m-1}$
    - recover $c_0, \ldots, c_{m-1}$

Cryptanalysis of LFSR

- Given a 4-stage LFSR, we know
  - $z_4=z_3c_3+z_2c_2+z_1c_1+z_0c_0$ mod 2
  - $z_5=z_4c_3+z_3c_2+z_2c_1+z_1c_0$ mod 2
  - $z_6=z_5c_3+z_4c_2+z_3c_1+z_2c_0$ mod 2
  - $z_7=z_6c_3+z_5c_2+z_4c_1+z_3c_0$ mod 2
- Knowing $z_0,z_1,\ldots,z_7$, one can compute c_0,c_1,c_2,c_4.
- In general, knowing 2n output bits, one can solve an n-stage LFSR

$$z_j = c_1 z_{j-1} + c_2 z_{j-2} + \cdots + c$$

The RC4 Stream Cipher

- A proprietary cipher owned by RSA, designed by Ron Rivest in 1987.
- Became public in 1994.
- Simple and effective design.
- Variable key size (typical 40 to 256 bits),
- Output unbounded number of bytes.
- Widely used (web SSL/TLS, wireless WEP).
- Extensively studied, not a completely secure PRNG, when used correctly, no known attacks exist

The RC4 Cipher: Encryption

- The cipher internal state consists of
  - a 256-byte array S, which contains a permutation of 0 to 255
    - total number of possible states is $256! \approx 2^{1700}$
  - two indexes: i, j

```
i = j = 0
Loop
   i = (i + 1) (mod 256)
   j = (j + S[i]) (mod 256)
   swap(S[i], S[j])
   output (S[i] + S[j]) (mod 256)
End Loop
```

RC4 Initialization

- Generate the initial permutation from a key k; maximum key length is 2048 bits
- First divide k into L bytes
- Then

```
for i = 0 to 255 do
    S[i] = i
j = 0
for i = 0 to 255 do
    j = (j + S[i] + k[i mod L])(mod 256)
    swap (S[i], S[j])
```

Randomness and Pseudorandomness

- For a stream cipher (PRNG) is good, it needs to be "pseudo-random".
- Random is not a property of one string
  - Is "000000" "less random" than "011001"?
  - Random is the property of a distribution, or a random variable drawn from the distribution
- Similarly, pseudo-random is property of a distribution
- We say that a distribution $\mathcal{D}$ over strings of length-ℓ is pseudorandom if it is indistinguishable from a random distribution.
- We use "random string" and "pseudorandom string" as shorthands

Distinguisher

- A distinguisher D for two distributions works as follows:
  - D is given one string sampled from one of the two distributions
  - D tries to guess which distribution it is from
  - D succeeds if guesses correctly

- How to distinguish a random binary string of 256 bits from one generated using RC4 with 128 bites seed?

Pseudorandom Generator Definition (Asymptotic version)

- Definition 3.14. We say an algorithm G, which on input of length n outputs a string of length $\ell(n)$, is a pseudorandom generator if
  1. For every n, $\ell(n) > n$
  2. For each PPT distinguisher D, there exists a negligible function negl such that
     $$|\Pr[D(r)=1 - \Pr[D(G(s))=1| \leq \text{negl}(n)$$

  Where r is chosen at uniformly random from $\{0,1\}^{\ell(n)}$ and s is chosen at uniform random from $\{0,1\}^s$

Security of using Stream Cipher for Encrpytion

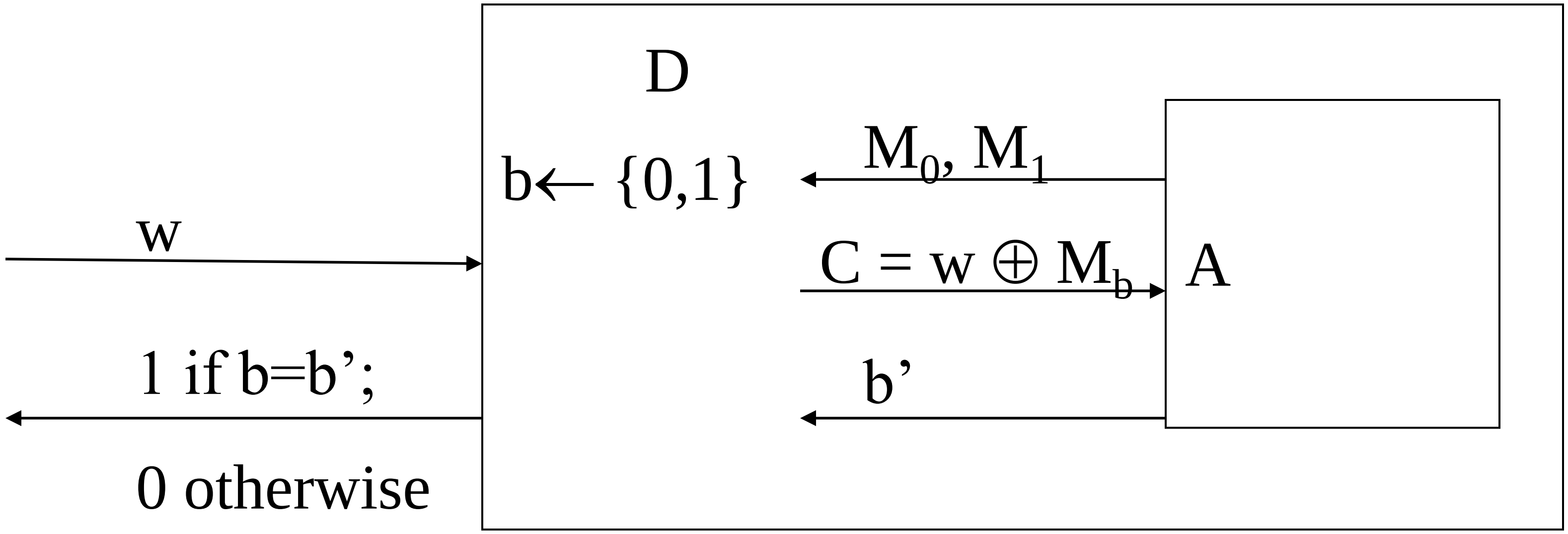
- Consider the construction Π of using $G(k) \oplus m$ as the encryption of m

- Theorem 3.16. If G is a pseudorandom generator, then Π has indistinguishable encryptions in the presence of an eavesdropper.

- Proof idea?

Proof of Theorem 3.16

- If Π does not have indistinguishable encryptions in the presence of an eavesdropper; then there exists adversary A that can break Π with non-negligible prob; we construct a distinguisher D as follows



A Bit More Details on the Proof

- Let $\varepsilon(n)$ be $|\Pr[\mathbf{PrivK}^{\mathbf{eav}}_{\mathbf{A},\Pi}=1] - ½ |$

- Then $|\Pr[D(r)=1 – \Pr[D(G(s))=1|$
  $= | ½ - \Pr[\mathbf{PrivK}^{\mathbf{eav}}_{\mathbf{A},\Pi}=1] | = \varepsilon(n)$

Recap of Pseudo Random Generator

- Useful for cryptography and for simulation
  - Stream ciphers, generating session keys
- The same seed always gives the same output stream
- Simulation requires uniform distributed sequences
  - E.g., having a number of statistical properties
- **Definition 3.14 is equivalent to** requiring unpredictable sequences
  - satisfies the "next-bit test“: given consecutive sequence of bits output (but not seed), next bit must be hard to predict
- Some PRNG’s are weak: knowing output sequence of sufficient length, can recover key.
  - Do not use these for cryptographic purposes

Coming Attractions …

- Number Theory Basics
- Reading: Katz & Lindell: 7.1