# CS915/435 Advanced Computer Security - Elementary Cryptography

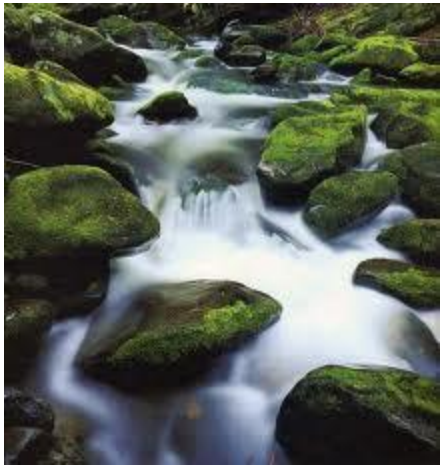
Stream Cipher

# Roadmap

- Symmetric cryptography
  - Classical cryptography
  - Stream cipher
  - Block cipher I, II
  - Hash
  - MAC
- Asymmetric cryptography
  - Key agreement
  - Public key encryption
  - Digital signature

# Stream cipher

- How it works: encrypt individual characters one at a time.



$$
\begin{array}{rl}
 & k0\ k1\ k2\ \ldots \\
\text{XOR} & m0\ m1\ m2\ \ldots \\
\hline
 & c0\ c1\ c2\ \ldots
\end{array}
$$

# Classification

1. The one-time pad
   – The simplest cipher with perfect secrecy
2. Synchronous stream cipher
   – Unable to recover from loss of synchronization
3. Self-synchronizing stream cipher
   – Able to recover from loss of synchronization

# One-time pad

- First described by Gilbert Vernam in 1917

$$c_i = k_i \oplus m_i \text{ for } i = 1,2,3, \ldots$$

# Security of one-time pad

- For many years, OTP was believed to be “unbreakable”, but there was no proof.
- Until 30 years later when Shannon developed the concept of “*perfect secrecy*”
- To understand perfect secrecy, we need to review some basics in probability

# Discrete probability

- A finite sample space $S = \{s_1, s_2, \ldots, s_n\}$

Def: **Probability distribution** P on S is $\{p_1, \ldots, p_n\}$, where $0 \le p_i \le 1$ and

$$\sum_i p_i = 1$$

Example - Uniform distribution

- $p_1 = p_2 = \ldots = p_n$

# Events

- An **event** $E$ is a subset of the sample space $S$.
- The complementary event: $\bar{E}$.
- $\boldsymbol{P(E)}$: the probability that an event occurs.
- $\boldsymbol{P(\bar{E}) = 1 - P(E)}$

H H

H T

T H

T T

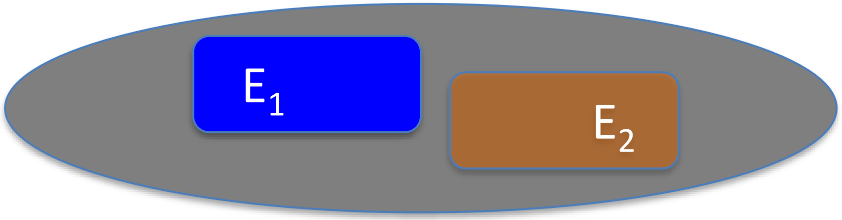
- **Example:**
- S is obtained by tossing a coin 2 times
- $E$ ={one heads, one tails}, $P(E) = \boxed{1/2}$

# Joint probability

- **Joint probability** is the likelihood that two events occur at the same time, denoted as:

$$P[E_1 \cap E_2] \text{ or } P[E_1, E_2]$$

- Two events $E_1$ and $E_2$ are called **mutually exclusive**, if $P[E_1 \cap E_2] = 0$



# Conditional probability

- Let $E_1$ and $E_2$ be two events with $P(E_2) > 0$
- The **conditional probability** of $E_1$ given $E_2$ is defined as:

$$P(E_1|E_2) = \frac{P(E_1 \cap E_2)}{P(E_2)}$$

= P(E1)

- $E_1$ and $E_2$ are called **independent** if:

$$P(E_1 \cap E_2) = P(E_1)P(E_2)$$

- *Question:* what's the probability of failing an exam if the student never attends lectures?

P(fail | never attend) = P(fail ∩ never attend)/P(never attend)

# Bayes' theorem

- One of the most important theorems in probability theory.
- *Theorem*: if $E_1$ and $E_2$ are events with $P(E_2) > 0$, then

$$P(E_1|E_2) = \frac{P(E_1)P(E_2|E_1)}{P(E_2)}$$

# Perfect secrecy

- Def: a cryptosystem has perfect secrecy if $P(m|c) = P(m)$ for all $m \in \mathcal{M}, c \in \mathcal{C}$
- Thm: one-time pad has perfect secrecy
- *Proof*

From Bayes Theorem **P(m|c)** = P(c|m)**P(m)**/P(c).
It suffices to show that P(c|m) = P(c).

(1) $P(c|m) = P(m \oplus k|m) = P(k) = 1/|K|$
(2) $P(c) = \sum P(m_i)P(k_i) = 1/|K| \sum P(m_i) = 1/|K|$

# Perfect secrecy – Example

- Suppose the message space is m = {0, 1}
- We assume the message is not uniformly distributed:
  - $P(m = 0) = 0.3$, and
  - $P(m = 1) = 0.7$
  - **Part (1)**: What is $P(c = 0 \mid m = 0)$?
    - $P(c=0|m=0) = P(k=0) = ½$
  - Similarly, we can compute
    - $P(c =0|m=1) =P(k=1) = 1/2$
    - $P(c =1|m=1) = P(k=0) = 1/2$
    - $P(c =1|m=0) = P(k=1) = 1/2$

# Perfect secrecy – Example

- Suppose the message space is m = {0, 1}
- We assume the message is not uniformly distributed:
  - $P(m = 0) = 0.3$, and
  - $P(m = 1) = 0.7$
  - **Part (2)**: What is $P(c = 0)$?
    - Recall that $P(c) = \sum P(m_i)P(k_i)$
    - So $P(c=0) = P(m=0)P(k=0)+P(m=1)P(k=1)$

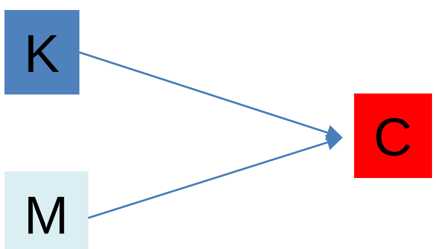
      $= 0.3*0.5 + 0.7*0.5 = 0.5$
  - Similarly, we can compute
    - $P(c =1) = 0.5$

# However, bad news …

- Thm: Perfect secrecy requires $|\mathcal{K}| \geq |\mathcal{M}|$
- Key length must be at least greater than the message length.
- This is why it is called **One Time** Pad.
- As a result, the cipher is “perfect” but “impractical”.

# Stream ciphers: making OTP practical

- Basic idea: use a short secret key to generate a very long key stream
- For example, a short 128-bit key K that can be distributed




- How many possible **different** key streams can we generate from K?
  - $2^{128} - 1$

# Synchronous stream cipher

- Key stream is constructed from the key
- Suppose we start with m-bits $(k_1, k_2, \dots, k_m)$
- We can generate the key stream using a **linear recurrence** of **degree** m:

$$k_{i+m} = \sum_{j=0}^{m-1} c_j k_{i+j} \bmod 2$$

where $c_0, \dots, c_{m-1}$ are constants

# Recurrence

- After a period, the same key stream will recur.
- For example: Vigenère cipher
- What's the recurrence period of Vigenère cipher?
- What is the ideal case for a key K = m bits?
  - It is $2^m - 1$

# Another example

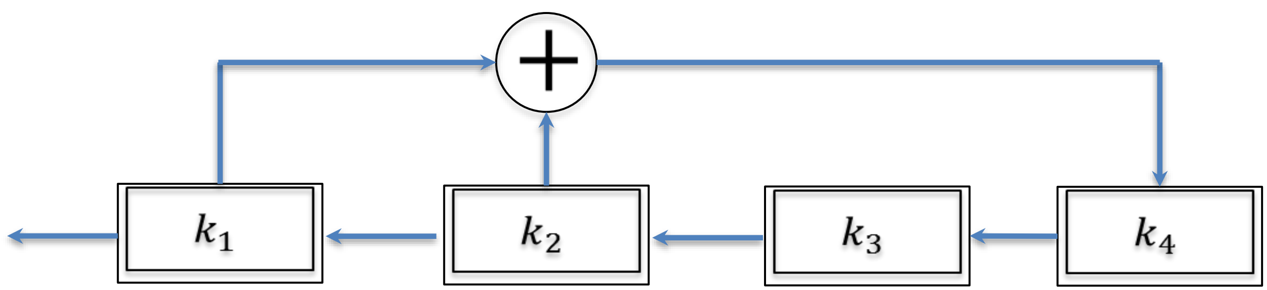
- Suppose $m = 4$ with the following linear recurrence equation for $i \geq 1$:

  $$k_{i+4} = (k_i + k_{i+1}) \bmod 2$$

- For any non-zero k vector, we can obtain a key stream of period 15.
- Starting with (1, 0, 0, 0), we get: 1 0 0 0 1 0 0 1 1 0 1 1 1 …

# Hardware implementation

- This kind of key stream can be efficiently produced in hardware by a ***Linear Feedback Shift Register*** (LFSR)




| | | | | | |
|---|---|---|---|---|---|
| | | 1 | 0 | 0 | 0 |
| | 1 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | ? |

What happens if the shift register contains only 0's?

# Attack on stream cipher: two-time pad

- The mistake of re-using the same key

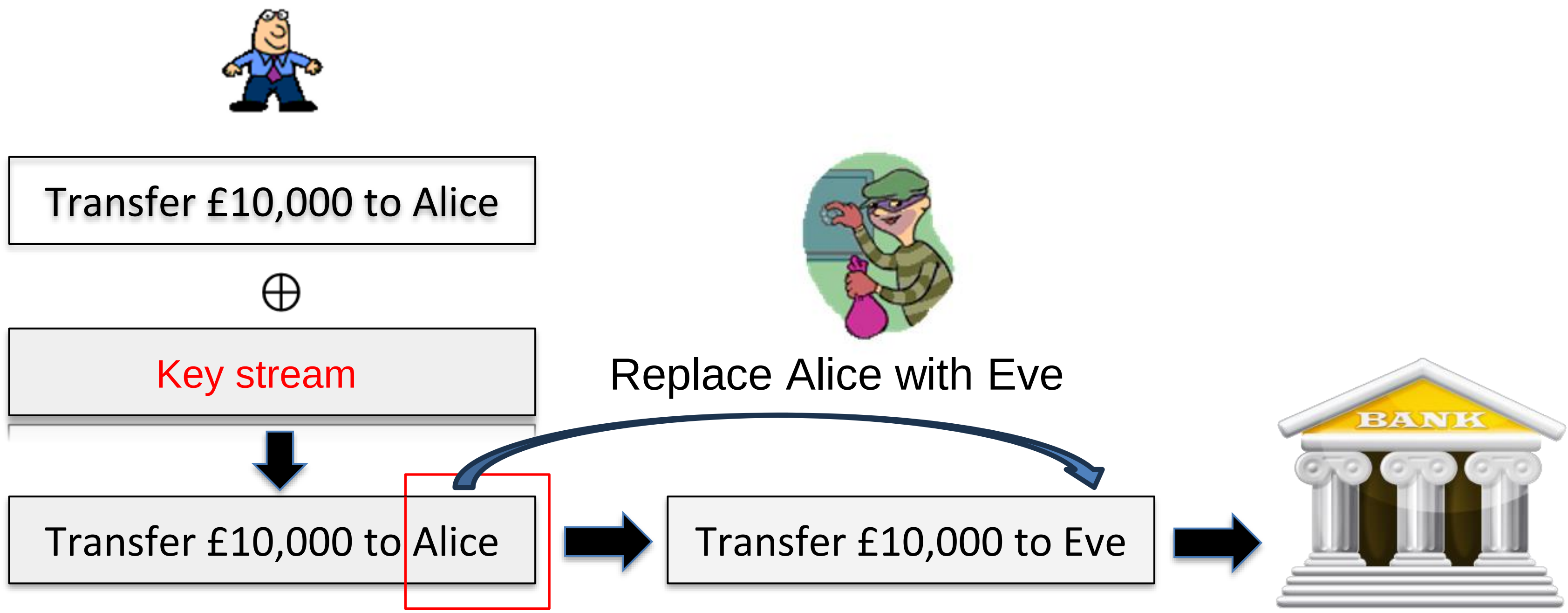
$$c_1 = m_1 \oplus f(k)$$
$$c_2 = m_2 \oplus f(k)$$

Eavesdropper does:

$$c_1 \oplus c_2 = m_1 \oplus m_2 \text{ => } \{m_1, m_2\}$$

# Attack 2: no integrity

- Applies to all stream ciphers




$c' = c \oplus \text{Alice} \oplus \text{Eve}$

But $c = K \oplus \text{Alice}$

So $c' = K \oplus \text{Eve}$

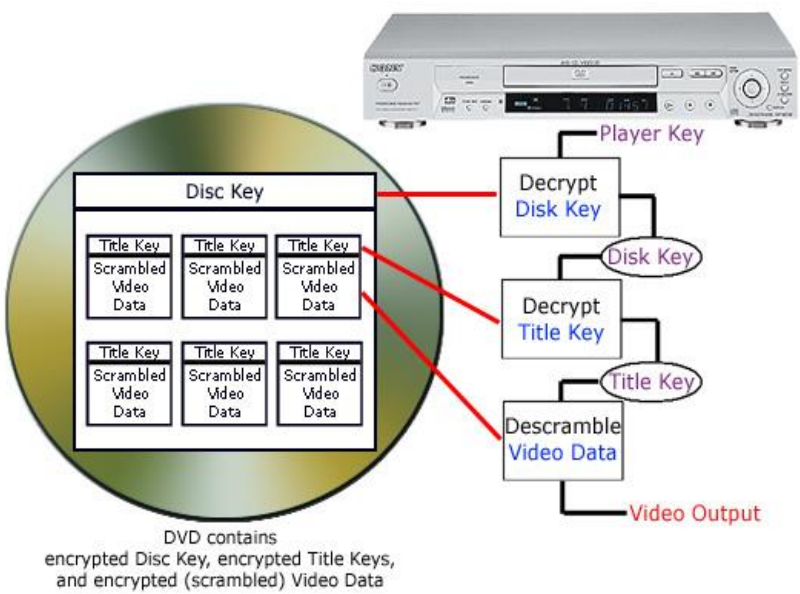
# Attack 3: weakness in the algorithm


- A real-world example: CSS
- What is CSS?
  - Content Scramble System
  - To prevent piracy
  - 40 bit security (US export restriction)
  - Restrict DVD to only licensed players
  - Windows and MAC have CSS license
  - Linux does not

# DeCSS

- Cannot play DVD under Linux
- DeCSS introduced
  - Written by an anonymous German hacker
  - A program to unscramble MPEG-2 video files
  - Jon Johanson, 16-old Norwegian put it on web in September 1999
  - MPAA (The Motion Picture Association of America) took legal action

# DVD encryption

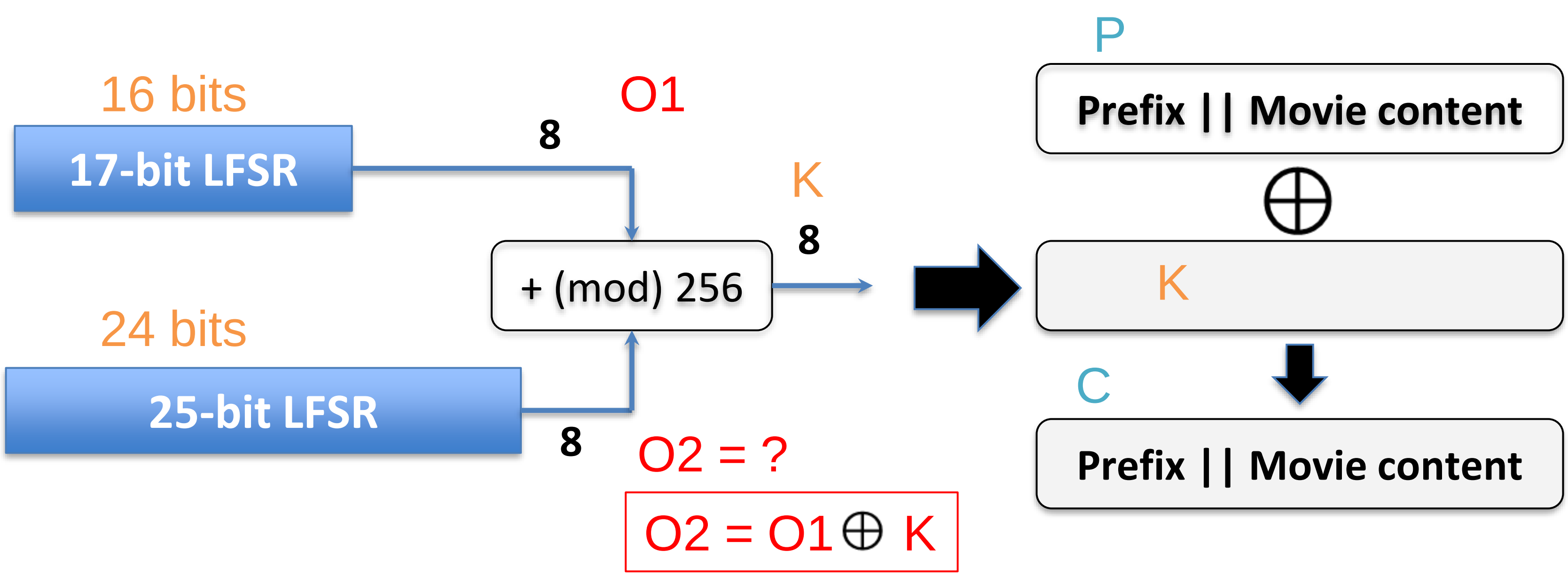



- Weakness in CSS
  - 40 bits key
  - In fact, only about 16 bits security

http://www.math.ucsd.edu/~crypto/Projects/MarkBarry/index.htm

# Breaking CSS

1. Try all possible 17-bit LFSR to get 20 bytes output
2. Subtract from the first 20 bytes of stream output
3. If consistent with 25-bit LFSR, found the key!!



# Lessons from CSS attack

- Never trust a proprietary cipher algorithm
- Choose standard ciphers whenever possible
  - E.g., Salsa20, Trivium etc (from eSTREAM)