L9: Stream and Block Ciphers

Hui Chen, Ph.D.
Dept. of Engineering & Computer Science
Virginia State University
Petersburg, VA 23806

Acknowledgement

- Many slides are from or are revised from the slides of the author of the textbook
  - Matt Bishop, Introduction to Computer Security, Addison-Wesley Professional, October, 2004, ISBN-13: 978-0-321-24774-5. Introduction to Computer Security @ VSU's Safari Book Online subscription
  - http://nob.cs.ucdavis.edu/book/book-intro/slides/

Outline

- Block ciphers
  - Examples
  - Attacks against direct use of block ciphers
  - Cipher Block Chaining (CBC)
  - Multiple encryption
- Stream ciphers
  - One-time pad: proven secure
  - Synchronous Stream Ciphers
  - Self-Synchronous Stream Cipher

Block Ciphers

- Block ciphers *divide a message into* a sequence of parts, or *blocks*, and encipher each block with the *same* key
- E encipherment function
  - $E_k(b)$ encipherment of message b with key k
  - In what follows, $m = b_1 b_2 \ldots$, each b_i of fixed length
- Block cipher
  - $E_k(m) = E_k(b_1)E_k(b_2) \ldots$

Block Cipher: Example

- DES is a block cipher
  - b_i = 64 bits, k = 56 bits
  - Each b_i enciphered separately using k
- AES is a block cipher
  - b_i = 128 bits, k = 128, or 192, or 256 bits
  - Each b_i enciphered separately using k

Block Ciphers

- Encipher and decipher multiple bits at once
- Each block enciphered independently
- Problem: identical plaintext blocks produce identical ciphertext blocks
  - Example: two database records
    - `MEMBER: HOLLY INCOME $100,000`
    - `MEMBER: HEIDI INCOME $100,000`
  - Encipherment:
    - `ABCQZRME GHQMRSIB CTXUVYSS RMGRPFQN`
    - `ABCQZRME ORMPABRZ CTXUVYSS RMGRPFQN`

Solutions

- Use additional information
- Use *Cipher Block Chaining* (CBC mode)

Additional Information

- Insert *additional varying* information into the plaintext block, then encipher
  - Information about block's position
  - Example:
    - Bits from the preceding ciphertext block (Feistel, 1973)
    - Sequence number on each block (Kent, 1976)
- Disadvantage
  - Effective block size is reduced because a block is in effect {additional bits || bits from plaintext}

Cipher Block Chaining

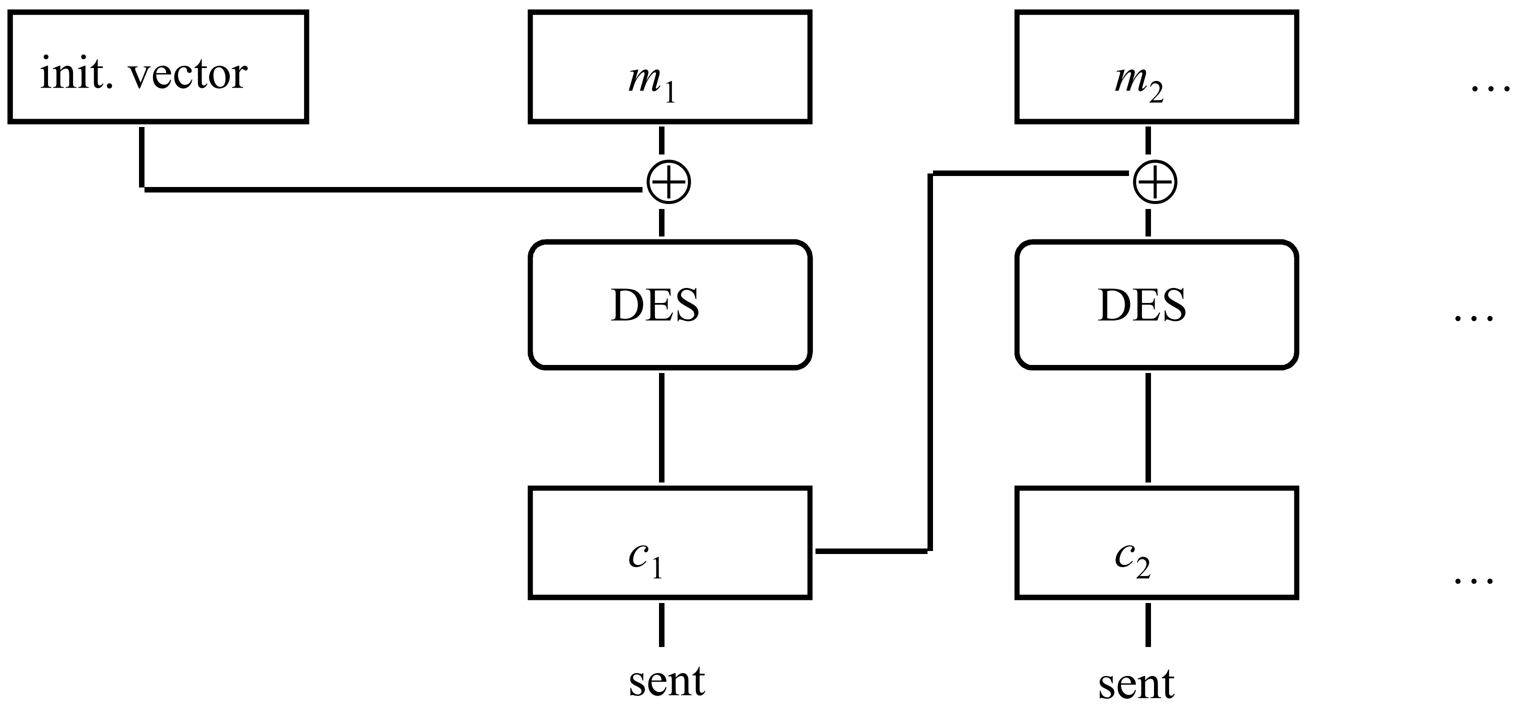
- Cipher block chaining (CBC)
- Exclusive-or current plaintext block with previous ciphertext block:
  - $c_0 = E_k(m_0 \oplus I)$
  - $c_i = E_k(m_i \oplus c_{i-1})$ for i > 0

  where I is the initialization vector

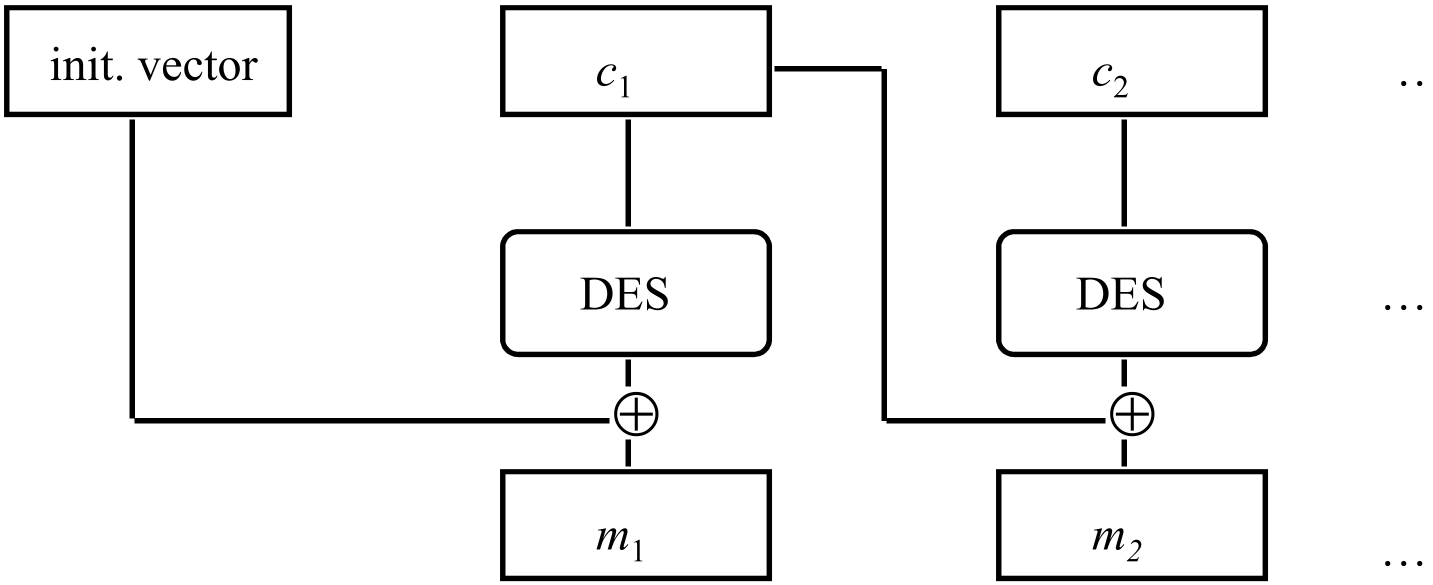
Recap on DES

- Electronic Code Book (ECB): directly use DES, a block cipher
- Cipher Feedback (CFB): generate pseudo one-time pad
- Output Feedback (OFB): generate pseudo one-time pad
- Cipher Block Chaining (CBC): commonly used mode

DES Recap: CBC Mode Encryption



DES Recap: CBC Mode Encryption



Multiple Encryption

- Double encipherment: $c = E_{k'}(E_k(m))$
  - Effective key length is $2n$, if k, k' are length n
  - Problem: breaking it requires 2^{n+1} encryptions, not 2^{2n} encryptions
- Triple encipherment:
  - EDE mode: $c = E_k(D_{k'}(E_k(m)))$
    - Problem: chosen plaintext attack takes $O(2^n)$ time using 2^n ciphertexts
  - Triple encryption mode: $c = E_k(E_{k'}(E_{k''}(m)))$
    - Best attack requires $O(2^{2n})$ time, $O(2^n)$ memory

Stream Ciphers

- Stream ciphers use a nonrepeating stream of key elements to encipher characters of a message
- E encipherment function
  - $E_k(b)$ encipherment of message b with key k
  - In what follows, $m = b_1 b_2 \ldots$, each b_i of fixed length
- Stream cipher
  - $k = k_1 k_2 \ldots$
  - $E_k(m) = E_{k1}(b_1)E_{k2}(b_2) \ldots$
  - If $k_1 k_2 \ldots$ repeats itself, cipher is *periodic* and the kength of its period is one cycle of $k_1 k_2 \ldots$

Examples

- Vigenère cipher
  - b_i = 1 character, $k = k_1k_2 \ldots$ where k_i = 1 character
  - Each b_i enciphered using $k_{i \bmod \text{length}(k)}$
  - Stream cipher
- One-time pad
  - A stream cipher
  - Not periodic because the key stream never repeats
  - Proven secure

Bit-Oriented Stream Ciphers

- Bit-oriented stream ciphers: each "character" is a bit
- Often (try to) implement one-time pad by xor'ing each bit of key with one bit of message
  - Example:

$$m = 00101$$
$$k = 10010$$
$$c = 10111$$

- But how to generate a good key?

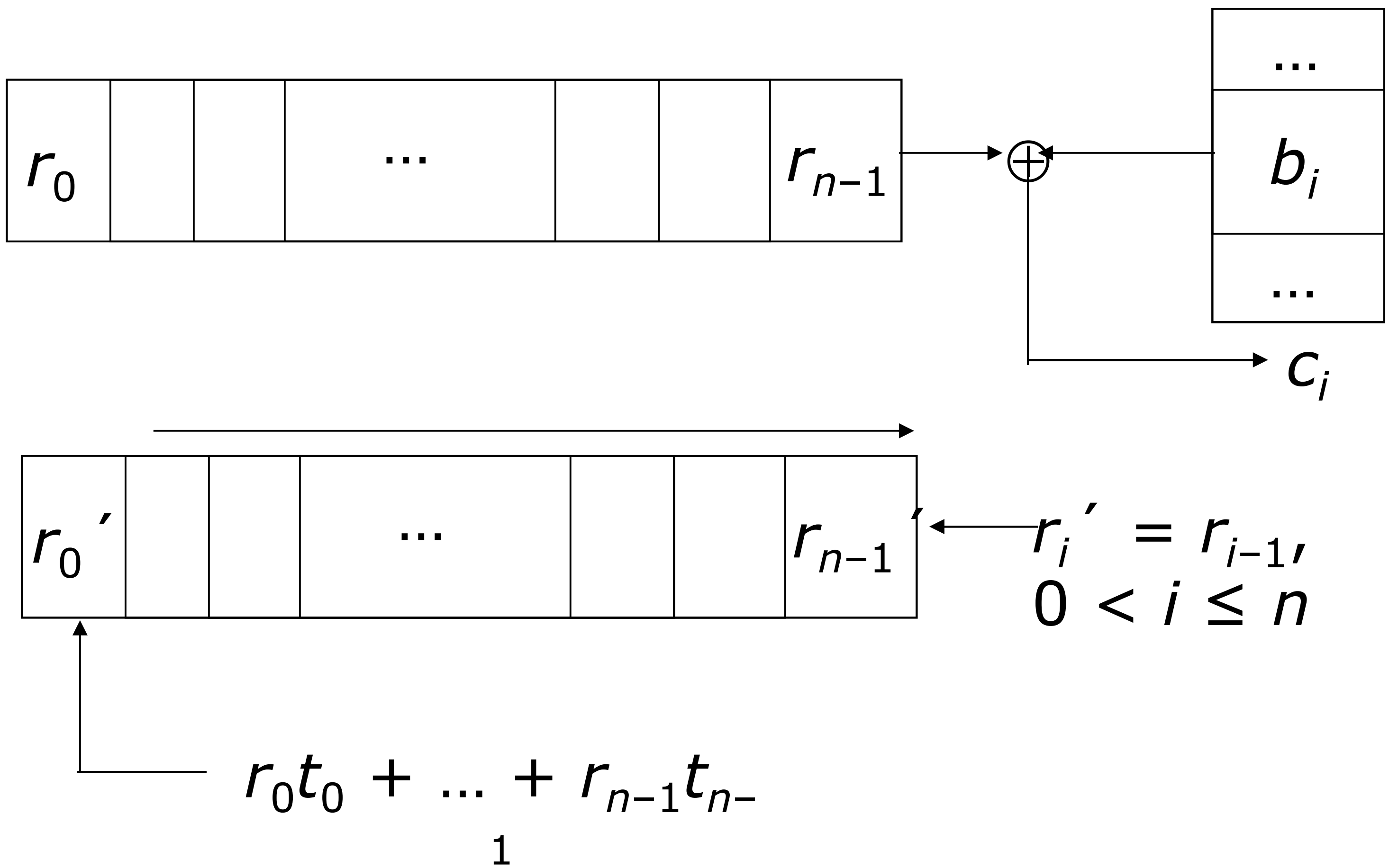
Synchronous Stream Ciphers

- To simulate a random, infinitely long key, synchronous stream ciphers generate key bits from a course other than the message itself
- Simplest approach: extracts bits from a register to use as the key
- Example: n-stage Linear Feedback Shift Register (LFSR)

n-stage Linear Feedback Shift Register

- n bit register $r = r_0 \ldots r_{n-1}$
- n bit tap sequence $t = t_0 \ldots t_{n-1}$
- Operation
  - Use r_{n-1} as key bit
  - Compute $x = r_0 t_0 \oplus \ldots \oplus r_{n-1} t_{n-1}$
  - Shift r one bit to right, dropping r_{n-1}, x becomes r_0

Operation



Example

- The least significant bit is the right-most bit
- 4-stage LFSR; *t* = 1001

r	k_i	*new bit computation*	*new r*
0010	0	$01 \oplus 00 \oplus 10 \oplus 01 = 0$	0001
0001	1	$01 \oplus 00 \oplus 00 \oplus 11 = 1$	1000
1000	0	$11 \oplus 00 \oplus 00 \oplus 01 = 1$	1100
1100	0	$11 \oplus 10 \oplus 00 \oplus 01 = 1$	1110
1110	0	$11 \oplus 10 \oplus 10 \oplus 01 = 1$	1111
1111	1	$11 \oplus 10 \oplus 10 \oplus 11 = 0$	0111
0111	0	$11 \oplus 10 \oplus 10 \oplus 11 = 1$	1011

  - Key sequence has period of 15 (010001111010110)

Notes on n-stage LFSR

- A known plaintext attack can reveal parts of the key sequence
- If the known plaintext is of length 2n, the tap sequence of an n-stage LFSR can be determined completely

n-stage Non-Linear Feedback Shift Register

- Do not use tap sequences. New key bit is any function of the current register bits

- n bit register $r = r_0 \ldots r_{n-1}$
- Use:
  - Use r_{n-1} as key bit
  - Compute $x = f(r_0, \ldots, r_{n-1})$; f is any function
  - Shift r one bit to right, dropping r_{n-1}, x becomes r_0

  Note same operation as LFSR but more general bit replacement function

Example

- The least significant bit is the right-most bit
- 4-stage NLFSR; $f(r_0, r_1, r_2, r_3) = (r_0 \,\&\, r_2) \mid r_3$

r	k_i	*new bit computation*	*new r*
`1100`	`0`	`(1 & 0)	0 = 0`	`0110`
`0110`	`0`	`(0 & 1)	0 = 0`	`0011`
`0011`	`1`	`(0 & 1)	1 = 1`	`1001`
`1001`	`1`	`(1 & 0)	1 = 1`	`1100`
`1100`	`0`	`(1 & 0)	0 = 0`	`0110`
`0110`	`0`	`(0 & 1)	0 = 0`	`0011`
`0011`	`1`	`(0 & 1)	1 = 1`	`1001`

  - Key sequence has period of 4 (0011)

Eliminating Linearity

- NLFSRs not common
  - No body of theory about how to design them to have long period
- Alternate approach: *output feedback mode*
  - For *E* encipherment function, *k* key, *r* register:
    - Compute $r' = E_k(r)$; key bit is rightmost bit of r'
    - Set *r* to r' and iterate, repeatedly enciphering register and extracting key bits, until message enciphered
  - Variant: use a counter that is incremented for each encipherment rather than a register
    - Take rightmost bit of $E_k(i)$, where *i* is number of encipherment

Self-Synchronous Stream Cipher

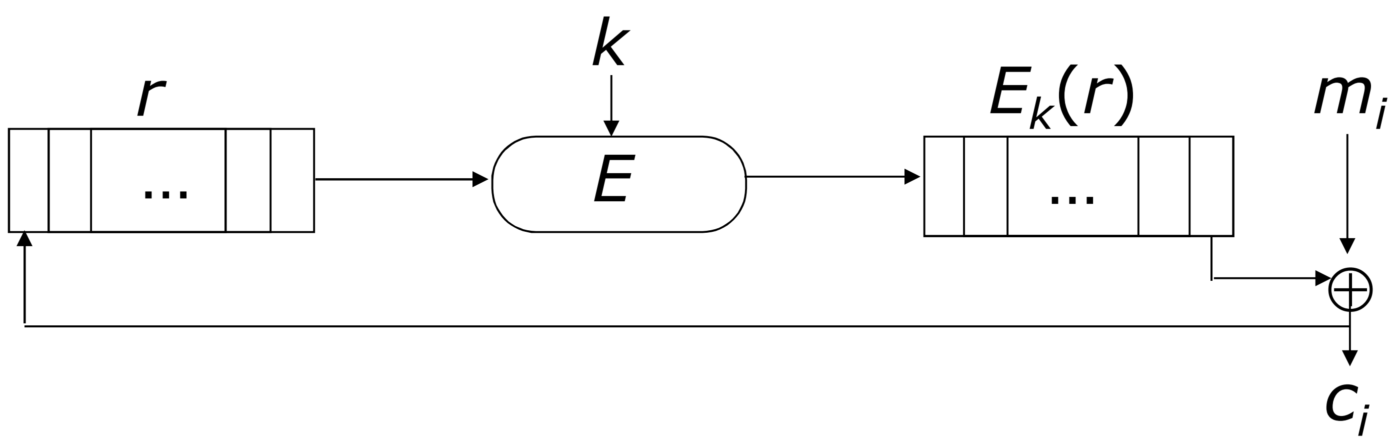
- Take key from message itself (*autokey*)
- Example: Vigenère, key drawn from plaintext
  - *key* XTHEBOYHASTHEBA
  - *plaintext* THEBOYHASTHEBAG
  - *ciphertext* QALFPNFHSLALFCT
- Problem:
  - Statistical regularities in plaintext show in key
  - Once you get any part of the message, you can decipher more

Another Example

- Take key from ciphertext (*autokey*)
- Example: Vigenère, key drawn from ciphertext
  - *key* XQXBCQOVVNGNRTT
  - *plaintext* THEBOYHASTHEBAG
  - *ciphertext* QXBCQOVVNGNRTTM
- Problem:
  - Attacker gets key along with ciphertext, so deciphering is trivial

Variant

- Cipher feedback mode: 1 bit of ciphertext fed into *n* bit register
  - Self-healing property: if ciphertext bit received incorrectly, it and next *n* bits decipher incorrectly; but after that, the ciphertext bits decipher correctly
  - Need to know *k*, *E* to decipher ciphertext



Summary

- Block ciphers
  - Examples: DES, AES
  - Attacks against direct use of block ciphers
  - Cipher Block Chaining (CBC)
  - Multiple encryption
- Stream ciphers
  - Examples: Vigenère cipher, One-time pad
  - One-time pad: proven secure
  - Synchronous Stream Ciphers (LFSR, NLFSR)
  - Self-Synchronous Stream Cipher