# CSC 580
# Cryptography and Computer Security

*Block Cipher Operation*
*Multiple Encryption and Modes*

*(Sections 7.1-7.6)*

February 16, 2017

## Overview

Today:

- HW4 Quiz
- Block cipher operations - multiple encryption and modes

To do before Tuesday:

- Do HW5 problems
- Finish reading Chapter 7 through section 7.7

## Chapter Theme: Block Cipher Use

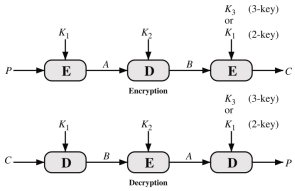
Two questions for this chapter:

Can you use a block cipher multiple times to increase security?

How to use a block cipher to encrypt more than a single block?

# Triple-DES

## Using a block cipher multiple times to increase security




Sometimes called "EDE Encryption"

Two-key version: 112-bit effective key length
Three-key version: 168-bit effective key length

Constructing in HW: $K_1$=$K_2$ gives 1-key DES (backward compatibility)

*Similar "double-DES" construction is insecure (meet-in-middle attack)*

# Block Cipher Modes

Question: How to use a block cipher to encrypt multiple blocks?

Four modes introduced with DES standard

- Electronic Codebook (ECB)
- Cipher Block Chaining (CBC)
- Cipher Feedback (CFB)
- Output Feedback (OFB)

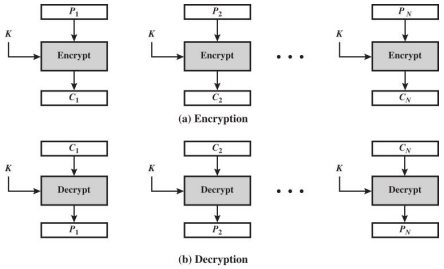
An additional mode introduced later (standardized with AES)

- Counter (CTR)

Each mode has tradeoffs in terms of flexibility, security, parallelizability, ...

# Electronic Codebook (ECB) Mode

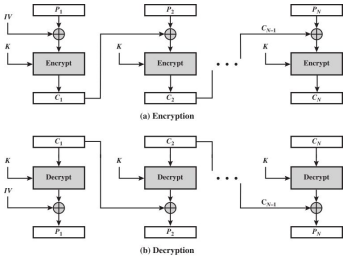
## Encrypting plaintext longer than one block




"Common Sense" solution

Does not hide repeated block patterns - ***insecure, so don't use***!

## Cipher Block Chaining (CBC) Mode



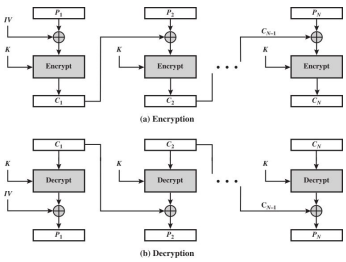

IV must be random
- Not sensitive
- Transmit with ciphertext

Randomizes next block
- Breaks up pattern
- Changing block affects all following blocks

But: Can't parallelize

Breaking assumptions
- BEAST attack (2011)

## Cipher Block Chaining (CBC) Mode




Questions

1. If transmission error in ciphertext block, how many errors in recovered plaintext?
2. If 500 MB encrypted, how can you decrypt the second half?
3. What if input is not a multiple of block size?

## Padding

ECB and CBC modes ***must*** encrypt full blocks of plaintext!

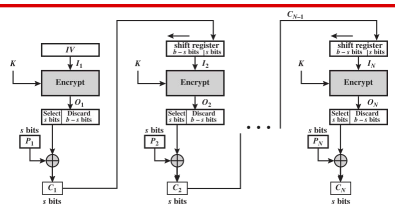
What if you have 192 bits of plaintext with AES/CBC?

Technique 1 (bit padding):
- No matter how long the plaintext, always append a 1 bit, followed by as many 0's as needed to fill out block.
- Example: 8-bit blocks, 10111010 110 becomes 10111010 110<u>10000</u>
- Advantage: plaintext can be any number of bits
- Question: Why "always append 1"? What if plaintext is already a multiple of block size?

Technique 2 (byte count padding - or PKCS#7 / PKCS#5):
- Count how many bytes of padding needed (at least 1), say c
- Add c bytes each with value c
- Ex (32-bit blocks, hex): 42 1a 49 c3 21 becomes 42 1a 49 c3 21 <u>03 03 03</u>
- Only works for padding full bytes! (Note: Used by JCA)

# Cipher Feedback (CFB) Mode (s-bit)

## Only encryption shown




Benefit: Can encrypt in units less than a full block (stream cipher)

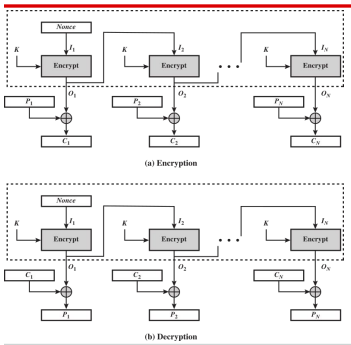
- For example, can encrypt character by character (terminals)

Can't parallelize and multiple block encryptions per plaintext "block"

- Question: What about decryption?

Not really used these days...

# Output Feedback (OFB) Mode

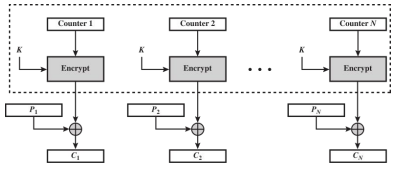



Can't parallelize
But *can* precompute!

DES definition also had
*s*-bit mode similar to CFB

No real advantage over
CTR mode, so….
Not really used these days

# Counter (CTR) Mode




Fully parallelizable! (Compare to OFB mode)

How to view this: Block cipher makes a "pseudo random one-time pad"

Just like one-time pad

- Must never repeat counter values (then not one-time!)
- Question 1: What about malleability?
- Question 2: How do ciphertext errors propagate in recovered plaintext?