

# Lecture 22: Network Security

COMP 332, Spring 2024

Victoria Manfredi



**Acknowledgements:** materials adapted from Computer Networking: A Top Down Approach 7<sup>th</sup> edition: ©1996-2016, J.F Kurose and K.W. Ross, All Rights Reserved as well as from slides by Abraham Matta at Boston University, and some material from Computer Networks by Tannenbaum and Wetherall.

# Today

## 1. Announcements

- Homework 8 due next Wed. (no coding)

## 2. Network Security

- overview
- principles of cryptography

## 3. Symmetric encryption

- overview
- block ciphers
- Data Encryption Standard (DES)
- Advanced Encryption Standard (AES)

# Network Security

## **OVERVIEW**

# What is network security?

Goal: enable secure communication over insecure channel

## Confidentiality

- only sender, intended receiver understand message contents
  - sender **encrypts** message
  - receiver **decrypts** message

## Authentication

- sender, receiver want to **confirm identity** of each other

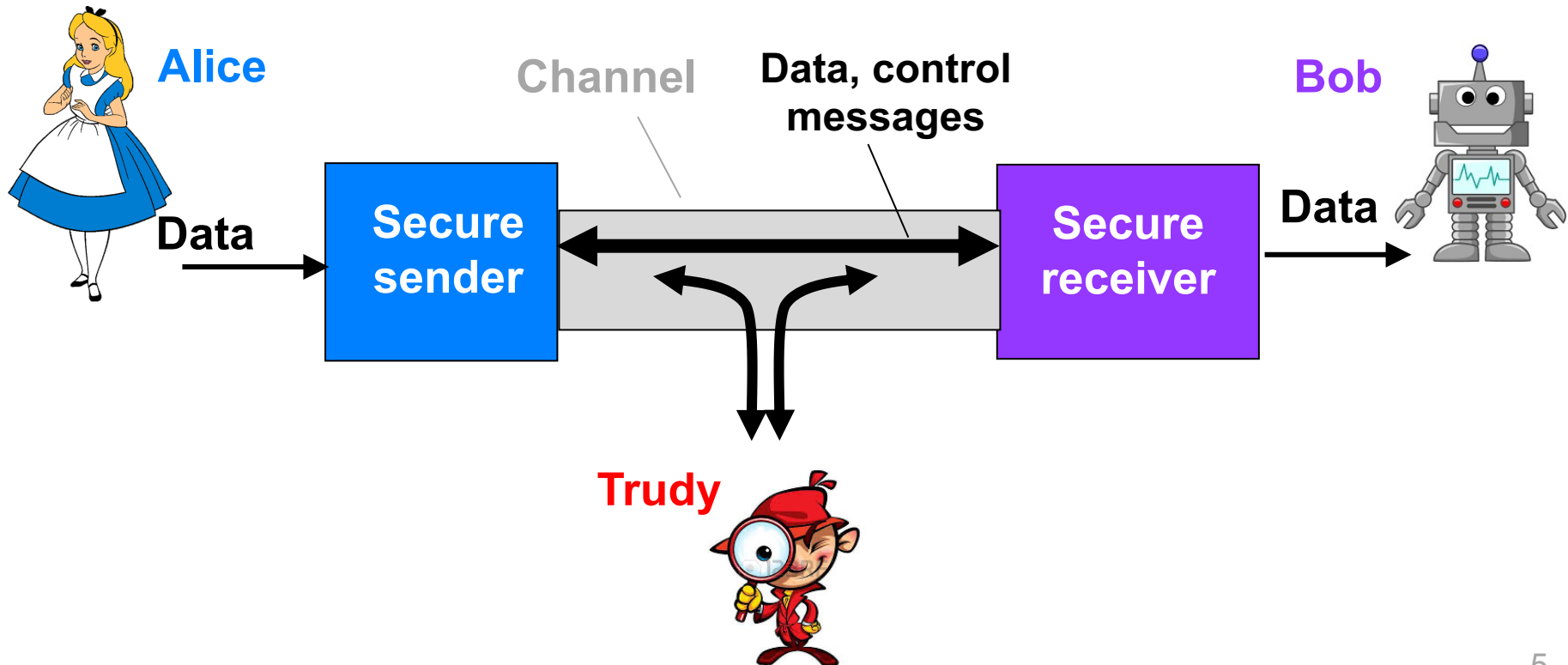
## Message integrity

- sender, receiver want to ensure **message not altered** (in transit, or afterwards) without detection

# Friends and enemies: Alice, Bob, Trudy

## Well-known in network security world

- Alice and Bob want to communicate securely
- Trudy (intruder) may intercept, delete, add messages



# Real-life Alices and Bobs?

Web browser and server for on-line purchases

On-line banking client and server

Email client and server

DNS servers

Routers exchanging routing table updates

Other examples?

# What can enemies do?

## Passive attack

- sniff and record messages
- analyze traffic patterns of messages

## Active attack

- replay and/or modify messages
- impersonate
  - spoof source addr (or any other field) in new packet
- hijack
  - take over ongoing connection
  - by removing sender or receiver, and inserting oneself in their place
- denial of service
  - prevent service from being used by others
  - e.g., by overloading resources

# Network Security

## **PRINCIPLES OF CRYPTOGRAPHY**

# Confidentiality

How can Alice hide msg she wants to send to Bob?

- so only Bob and no-one else can read msg?

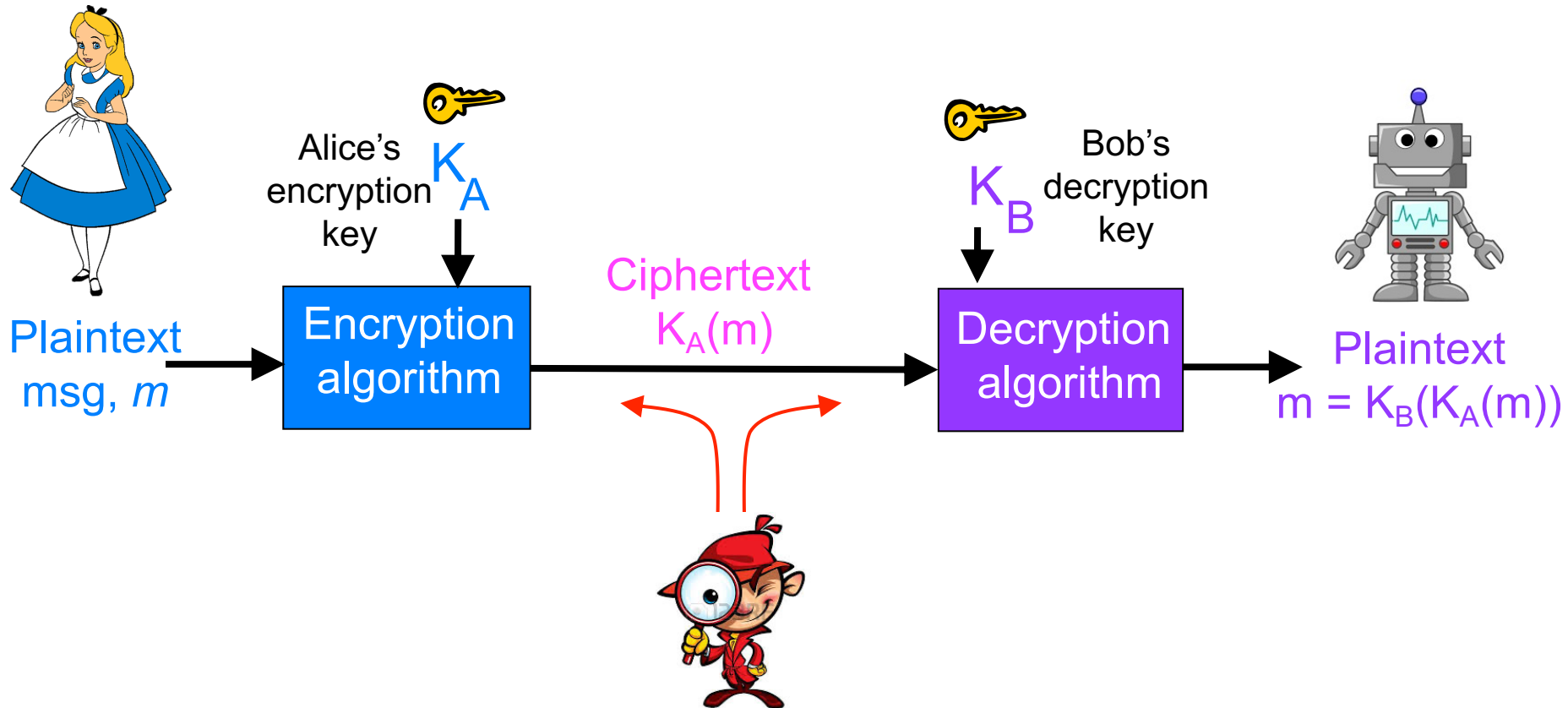
## Encryption

- used to disguise a msg and hide its contents
- **plaintext**: unencrypted msg
- **ciphertext**: encrypted msg

## Encryption algorithm

- **substitute/rearrange** pieces of plaintext with pieces of ciphertext
- **known and publicly available**
  - keys (secret info) used to prevent intruder from decrypting data

# The language of cryptography



# How to break an encryption scheme?

## Cipher-text only attack

- Trudy has **ciphertext** she can analyze
  - brute force: search through all keys
  - statistical analysis

## Known-plaintext attack

- Trudy has **plaintext corresponding to ciphertext**
  - e.g., monoalphabetic cipher, Trudy determines pairings for a,l,i,c,e,b,o,

## Chosen-plaintext attack

- Trudy can get **ciphertext for chosen plaintext**

## Q: When is an encryption scheme computationally secure?

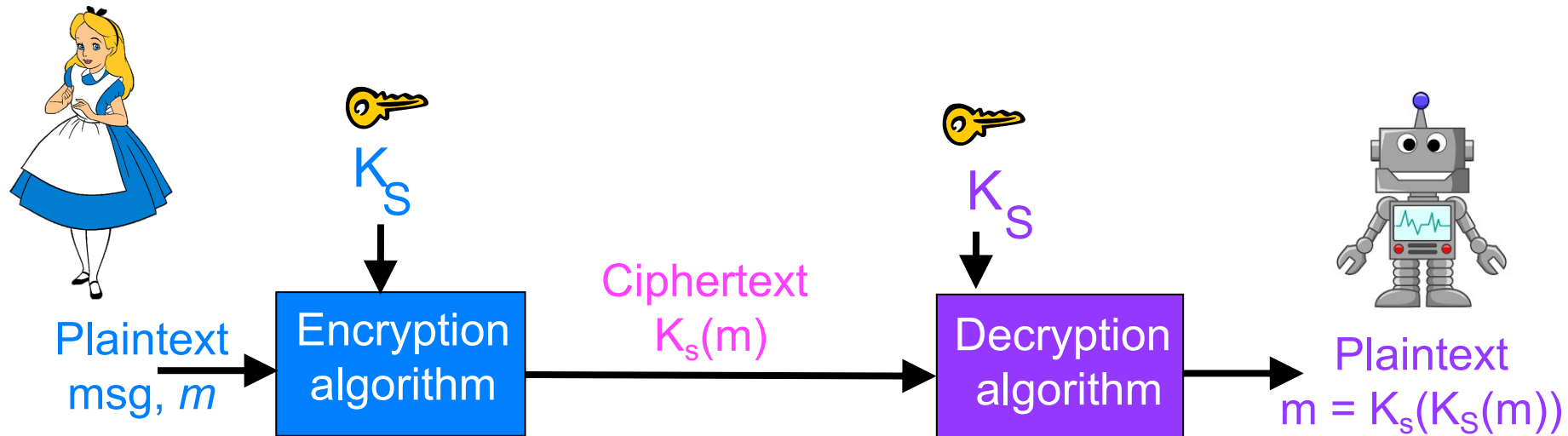
- if cost to break cipher > value of info
- If time to break cipher > time info is useful

# Symmetric Key Cryptography

## **OVERVIEW**

# Symmetric key cryptography

Both Alice and Bob use same encryption/decryption key:  $K_s$



Q: how do Bob and Alice agree on key value?

# Caesar cipher

## Take each letter of plaintext

- substitute letter that is **k letters** later in alphabet

**plaintext:**    abcdefghijklmnopqrstuvwxyz

↓                                 ↓

**ciphertext:**   defghijklmnopqrstuvwxyzabc

e.g.,   plaintext: hello world  
          ciphertext: koor zruog

🔑 *Encryption key:  $k=3$*   
*# of possible keys? 26*

# A more sophisticated encryption scheme

Substitution cipher: substitute one thing for another

- monoalphabetic cipher: substitute one letter for another

plaintext:	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	
		↓																								↓	
ciphertext:		m	n	b	v	c	x	z	a	s	d	f	g	h	j	k	l	p	i	o	u	y	t	r	e	w	q

e.g., plaintext: bob. meet me in wonderland  
ciphertext: nkn. hccu hc sj rkjvcogmjv

 *Encryption key:* mapping from set of 26 letters  
to set of 26 letters

*# of possible keys?* 26!


# An even more sophisticated encryption scheme

## Polyalphabetic cipher

- use  $n$  substitution ciphers,  $M_1, M_2, \dots, M_n$  + cyclic pattern
  - e.g.,  $n=4$ :  $M_1, M_3, M_4, M_3, M_2$ ;  $M_1, M_3, M_4, M_3, M_2$ ; ..
- for each new plaintext symbol
  - use subsequent substitution pattern in cyclic pattern

e.g., plaintext: dog

ciphertext: d from  $M_1$   
                  o from  $M_3$   
                  g from  $M_4$

 *Encryption key:*  $n$  substitution ciphers, and cyclic pattern  
*# of possible keys?*  $(26!)^n$

# Back to the modern world

## 2 classes of symmetric key encryption techniques

### 1. Block ciphers

- process **one block of elements** at a time
- produce output block for each input block
- used in many **secure Internet protocols**
  - PGP: secure email
  - TLS/SSL: secure TCP connections
  - IPSec: secure network layer communication



**Our focus**

### 2. Stream ciphers

- process **input elements continuously**
- produce output one element at a time as it goes along
- used for **wireless LANs**

# Symmetric Key Cryptography

## **BLOCK CIPHER**

# Block cipher

## Process msg to encrypt in $k$ -bit blocks

- $k=64$ : msg broken into 64-bit blocks
  - each block encrypted independently
  - each  $k$ -bit block of plaintext mapped to  $k$ -bit block of ciphertext

## 2 approaches

### 1. Electronic Codebook (ECB) mode

- 1 block of plaintext encrypts to same block of ciphertext

### 2. Cipher Block Chaining (CBC) mode

- 1 block of plaintext can encrypt to different blocks of ciphertext

Q: which do you think is more secure?

# Electronic Codebook (ECB) mode

1 block of plaintext always encrypts to same block of ciphertext

*Electronic Codebook,  $k=3$*

<b><i>Input</i></b>	<b><i>Output</i></b>
---------------------	----------------------

000	110
-----	-----

001	111
-----	-----

010	101
-----	-----

011	100
-----	-----

100	011
-----	-----

101	010
-----	-----

110	000
-----	-----

111	001
-----	-----

$2^3!$  choices for map.

Huge table for even just  $k=64$ , so use functions that simulate randomly permuted table

**plaintext:** 010 110 001 111

**ciphertext:** 101 000 111 001

# Problems with ECB

## Trudy can start to build codebook without knowing key

- given plaintext and ciphertext for a few msgs
- bits of msgs **repeat** in real world
  - 2 or more blocks of plaintext may be identical
- msg to be encrypted may have **regular structure**, similar **start/end**
  - e.g., email, webpage

## Trudy can modify ciphertext without knowing key

- transfers \$100 between 2 banks several times, watches exchange
- correlates msgs that authorize transaction, **replays msg**

# Cipher-block chaining (CBC) mode

1 block of plaintext maps to different blocks of ciphertext

## Assume 64 bit blocks

- $m(i)$ :  $i^{\text{th}}$  plaintext block
- $c(i)$ :  $i^{\text{th}}$  ciphertext block
- $c(0)$ : initialization vector (iv), random 64 bit string
- $k_s$ : symmetric key

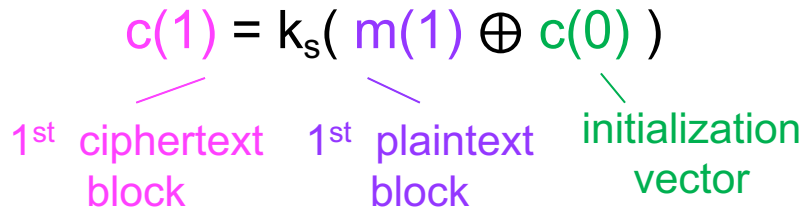
$a \oplus b$ : exclusive or (XOR) of 2 bit strings,  $a$  and  $b$

- 1 if and only if 1 of bits is 1
- 0 otherwise

# CBC steps

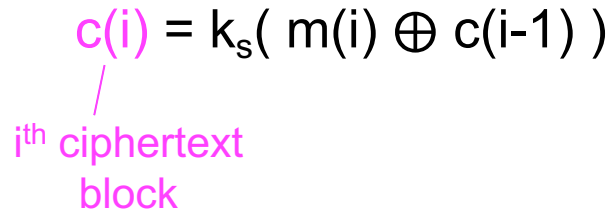
1. Sender generates IV and sends to receiver in plaintext
2. Sender computes 1<sup>st</sup> block,  $c(1)$ , and sends to receiver

$$c(1) = k_s( m(1) \oplus c(0) )$$



3. Sender computes i<sup>th</sup> block,  $c(i)$ , and sends to receiver

$$c(i) = k_s( m(i) \oplus c(i-1) )$$



4. Receiver decrypts  $c(i)$  to get  $m(i) \oplus c(i-1)$ 
  - knows initialization vector,  $c(0)$ , does  $\oplus$  to recover original msg

Identical plaintext blocks will almost certainly map to different ciphertexts

# Symmetric Key Cryptography

## **DATA ENCRYPTION STANDARD**

# Data Encryption Standard (DES)

## Features

- 56-bit symmetric key,  $2^{56}$  possible keys
- 64-bit plaintext block input
- block cipher with CBC

## Q: How secure is DES?

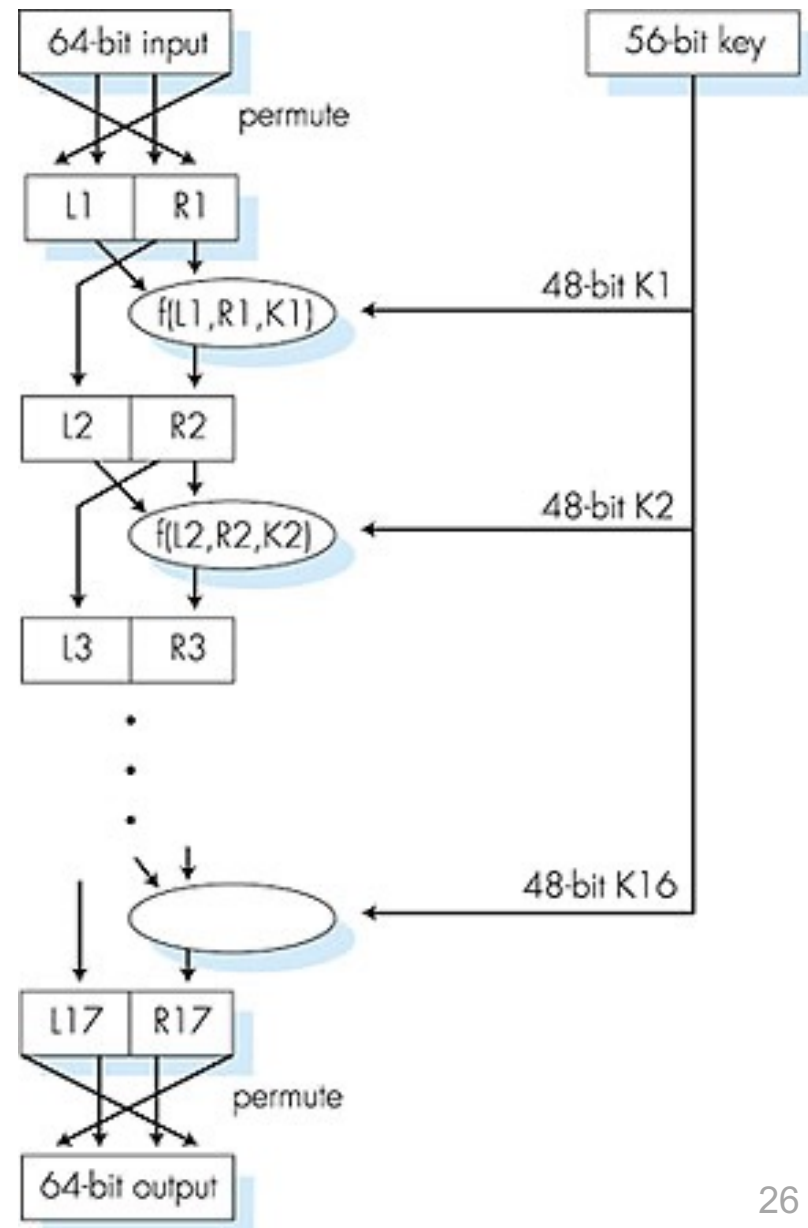
- brute force attack: 1 sec for DES
- no known good analytic attack

## Making DES more secure

- 3DES: encrypt 3 times with 3 different keys

# DES operation uses Feistel cipher structure

1. Divide plaintext block in half
  - $L_1$  and  $R_1$
2.  $L_1$  and  $R_1$  pass through  $n=16$  rounds of processing
  - each round  $i$  uses
    - inputs  $L_{i-1}$  and  $R_{i-1}$  from previous round
    - different 48 bits of 56-bit key
3. Combine halves at end to produce the ciphertext block



# Symmetric Key Cryptography

## **ADVANCED ENCRYPTION STANDARD**

# Advanced Encryption Standard (AES)

## Replaced DES (Nov 2001)

- DES is insecure, (3)DES is slow in software, small block size

## Features

- 128, 192, or 256 bit symmetric keys, up to  $2^{256}$  possible keys
- 128-bit plaintext block input
- block cipher with CBC
- does not use Feistel structure

## Q: How secure is AES?

- brute force attack
  - 1 sec for DES
  - 149 trillion years for AES