# EECS 498-7/8
# Computer Security

Peter Honeyman
Center for Information Technology Integration

## Course Overview

- **www.citi.umich.edu/u/honey/security**
- Monday & Friday, 9:00 – 10:30
  Wednesday, 9:00 - 10:00
  1005 Dow
  4 credits
- EECS Technical Elective?
  - Presumably -- working on it.

## Course Requirements

- Cryptography and Network Security: Principles and Practice (Third Edition)
  William Stallings
  Prentice-Hall
  ISBN 0130914290
- Weekly (or more) homework + programming assignments: 50%
- Exams: 25% ea.

## Outline of Lectures

- Models of security
- Classical encryption
  - Substitution and transposition ciphers
- Symmetric key cryptography
  - DES, AES, others
  - Confidentiality
  - Key distribution
  - Random number generation

## Outline of Lectures

- Number theory
- Asymmetric key cryptography
- Message authentication
- Digital signatures
- Applications
  - Kerberos
  - SSL, X.509, and PKI
  - IPSec

## Computer Security

- Host security & network security
  - Equally important
  - Often no clear boundary between them
- Examples
  - Morris' Internet worm
  - Mitnick's attack on Shimomura
  - Credit card theft from e-commerce sites
  - Distributed denial of service attacks

## X.800 Security Services

- Authentication
  - Peer entity identification
    - Guards against masquerade and unauthorized replay
  - Data origin
    - Useful in connectionless communication
- Access control
  - Prevent unauthorized use of resources
  - Presupposes some sort of authentication

## X.800 Security Services

- Data confidentiality
  - Protection from unauthorized disclosure
  - Granularity
    - Session
    - Message
    - Fields in a message
  - Traffic analysis

## X.800 Security Services

- Data integrity
  - Protection from unauthorized modification, insertion, deletion, replay
  - Granularity
    - Session, message, or field(s)
    - Connection-oriented or connectionless
  - Detection and/or recovery

## X.800 Security Services

- Nonrepudiation
  - Origin (sender)
  - Destination (receiver)
- Availability
  - Security? Reliability?

## X.800 Security Mechanisms

- Encryption
- Digital signature
- Access control
- Data integrity
- Authentication exchange
- Traffic padding
- Routing control
- Notarization

## Security Attacks

- Passive attacks
  - Interception
  - Traffic analysis
- Active attacks
  - Masquerade
  - Replay
  - Content modification
  - Denial of service

## Model for Network Security

- Principals
  - Sender
  - Receiver
  - Adversary
  - Trusted third party

## Model for Network Security

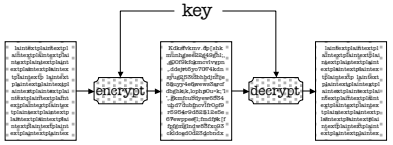
- Sender injects message, receiver extracts it
- Sender and receiver communicate over information channel
- Sender and receiver provide security-related information
  - Possibly shared with or generated by T3P
- Security-related transformation is applied to message
- Adversary may control information channel

## Designing a Security Service

- Select an algorithm for the security-related transformation (cipher)
- Generate the security-related information to be used by the algorithm (keys)
- Select a method for distribution of security-related information (key distribution)
- Select a protocol for the communicating principals that uses the security algorithm (cryptographic protocol)

## Classical Encryption

- *Symmetric*, or single-key encryption
- Model: Fig 2.1, p. 25




## Symmetric Key Cryptography

- Sender combines *plaintext* and key to produce *ciphertext*
  - Called *enciphering* or *encryption*
  - $Y = E(K, X)$ or $Y = E_K(X)$
- Receiver combines ciphertext and key to recover ciphertext
  - Called *deciphering* or *decryption*
  - $X = D(K, Y)$ or $X = D_K(Y)$
- *Cryptography* is the study of ciphers

## Dimensions of Cryptography

- Type of operations used in cipher
  - Substitution
  - Transposition
- Number of keys
  - Symmetric vs. asymmetric
- Plaintext processing
  - Block cipher
  - Stream cipher

## Cryptosystem Model

- Fig. 2.2, p. 26 augments earlier model in two ways
  - Key distribution via secure channel
  - Adversary cryptanalyzes ciphertext
- Adversary has complete information about the encryption and decryption methods
  - Only the key is secret
  - Kerckhoff's principle, 1883
  - Necessary for any practical cipher
  - Alternatively, refer to all the secret information as the key
    - Example: gzip | dd conv=swab | tr -c

## Goals of Cryptanalysis

- Recover plaintext
- Recover key

## Cryptanalytic Attacks

- In all cases, cryptanalyst has complete knowledge of the cipher and some ciphertext to be decoded
- Ciphertext only
  - Most common attack
- Known plaintext
  - Cryptanalyst has plaintext-ciphertext pair(s)
  - Surprisingly easy to obtain or infer plaintext
- Chosen plaintext
  - Cryptanalyst has plaintext-ciphertext pair(s)
  - Cryptanalyst (somehow) was able to select the plaintext and force its encryption

## Cryptanalysis

- Chosen ciphertext
  - Cryptanalyst has plaintext-ciphertext pair(s)
  - Cryptanalyst (somehow) was able to select the ciphertext and force its decryption
- Chosen text
  - Cryptanalyst is able to produce chosen plaintext and chosen ciphertext pairs

## Unconditionally Secure Cipher

- A cipher is unconditionally secure if no amount of ciphertext suffices to determine uniquely the plaintext
  - Shannon showed that there is only one cipher that is unconditionally secure
  - It is not practical in most instances

## Computationally Secure Cipher

- A cipher is computationally secure if
  - The cost of breaking the cipher exceeds the value of the encrypted information, or
  - The time required to break the cipher exceeds the useful lifetime of the information
- Key size plays an important role
- So does computational power
- Table 2.2, p. 26

## Exhaustive Search and Key Size

| Key | @ 1 per μsec | @ 1 per picosec |
|---|---|---|
| 32 bits | 35.8 min | 2.15 ms |
| 56 bits | 1,142 years | 10.01 hours |
| 128 bits | $5.4 \times 10^{24}$ years | $5.4 \times 10^{18}$ years |
| Substitution | $6.4 \times 10^{12}$ years | $6.4 \times 10^{6}$ years |

## Computationally Secure Cipher

- Note that DES can no longer be considered computationally secure
- *Cracking DES: Secrets of Encryption Research, Wiretap Politics & Chip Design*, Electronic Frontier Foundation, John Gilmore (Editor), O'Reilly & Associates, ISBN: 1565925203

## Substitution Ciphers

- Plaintext characters are replaced by other plaintext characters according to some rule
- Caesar cipher: E(*C*) = P + 3 (mod 26), D(P) = *C* - 3 (mod 26)
- ROT13: E(*C*) = P + 13 (mod 26), D = E
- General Caesar cipher: E(*C*) = P + k (mod 26)
  - k is the key
- Cryptanalysis: try k = 0, …, 25
  - Works for known (or probable) plaintext

## Caesar Ciphers

- Cryptanalysis is easy because
  - Algorithm is known
  - Only 26 keys to try
  - Known or probable plaintext
- Defeating cryptanalysis
  - Pre-scramble plaintext, e.g., compress it
  - Increase the key space
    - E(*C*) = P + k (mod 26), k = 0, …, 1,000,000? :-)

## Monoalphabetic Substitution Cipher

- Let $\Sigma$ = {A, B, …, Z}
- Let $\pi: \Sigma \rightarrow \Sigma$ be a permutation
- Key space is now 26! $\approx 2^{88}$
  - Much too large to search
- But this is still easy to cryptanalyze through letter frequency analysis
  - ETAOINSHRDLU or something like that

## Polygram Substitution Cipher

- Playfair
  - $E(P_iP_{i+1}) = C_iC_{i+1}$ through key-based 5 × 5 transformation table
  - Cryptanalysis: digram frequency
- Hill cipher
  - $\mathbf{C} = \mathbf{KP}$, where $\mathbf{C}$ and $\mathbf{P}$ are $d$-dimensional column vectors and $\mathbf{K}$ is a nonsingular $d \times d$ matrix
  - $\mathbf{P} = \mathbf{K}^{-1}\mathbf{C}$
  - Hides $d$-1 letter sequence analysis
  - Easily broken with known plaintext

## Polyalphabetic Substitution Cipher

- E: □□ 2□, pick one
  - Typically a set of monoalphabetic substitution rules is used
  - Key determines which rule to use

## Periodic Substitution Ciphers

- Special class of polyalphabetic substitution ciphers
- Example: Vigenère cipher
  - Each key letter determines one of 26 Caesar ciphers
  - $C_i = E(P_i) = P_i + k_{i \bmod(\text{key length})}$
  - Given a sufficient amount of ciphertext, common sequences are repeated, exposing the period
  - Frequently occurring letters in the key will be used to encrypt frequently occur plaintext letters

## Periodic Substitution Ciphers

- Vigenère autokey system: after key is exhausted, use plaintext for running key
- Can still detect regularities, e.g., E encrypted with E

## Vernam Cipher

- Key length equal to plaintext length
- A.k.a. "one-time pad"
- Plaintext and ciphertext are statistically independent
- Unconditionally secure (Shannon, 1948)
- Key generation and distribution are difficult

## Transposition

- Rail-fence technique
  - `Ri-ec ehiu`
    `alfnetcnqe`
- Generalization: columnar technique
  - `Cuathq`
    `omrenu`
    `ln cie`
  - Augment with permuted rows
- Generalization: multiple transpositions
- Does not change letter frequencies

## Rotor Machines

- Enigma, ca. WWII
- Each rotor corresponds to a substitution cipher
- A one-rotor machine produces a polyalphabetic cipher with period 26
- Output of each rotor is input to next rotor

## Rotor Machines

- After each symbol, the "fast" rotor is rotated
- After a full rotation, the adjacent rotor is rotated
  - An *n* rotor machine produces a polyalphabetic cipher with period $26^n$

## Chapter 2 Homework

- Pick any five problems
- Extra credit for more than five

## Chapter 3

- Simplified DES
- Block cipher principles
- DES
- Block cipher modes of operation

## Simplified DES

- Block cipher: 8-bit blocks
  - Input and output
- 10-bit key
- Complex, multi-stage algorithm

## Simplified DES

- Five stages
  - Initial permutation (IP)
  - Key-dependent scrambler (f)
    - Mixes permutation and substitution
    - 8-bit key
  - Swap of L and R
  - f again (different key)
  - Inverse permutation $IP^{-1}$
- DES: $IP^{-1} \circ f_{K2} \circ swap \circ f_{K1} \circ IP$
- DES $^{-1}$: $IP^{-1} \circ f_{K1} \circ swap \circ f_{K2} \circ IP$

## S-DES Key Schedule

- 10-bit key is generator for two 8-bit keys, $K_1$ and $K_2$
- $K_1$ = select and $permute_8 \circ$ paired circular left $shift_1 \circ permute_{10}(K)$
- $K_2$ = select and $permute_8 \circ$ paired circular left $shift_2 \circ$ paired circular left $shift_1 \circ permute_{10}(K)$

## S-DES Key Schedule

- $\text{Permute}_{10}$
  3 5 2 7 4 10 1 9 8 6
- Paired circular left $\text{shift}_1$
  2 3 4 5 1 7 8 9 10 6
- Select and $\text{permute}_8$
  6 3 7 4 8 5 10 9
- Paired circular left $\text{shift}_2$
  4 5 1 2 3 9 10 6 7 8

## S-DES Initial Permutation

- IP = 2 6 3 1 4 8 5 7
- $IP^{-1}$ = 4 1 3 5 7 2 8 6

## $f_K$

- Combination of substitution and permutation
- Let $\text{input}_8 = L_4 R_4$
- Let F: $\{0,1\}^4 \times \{0,1\}^4$, not necessarily 1-1
- $f_K(L, R) = L \oplus F(R, K_i), R)$
- Note the structure: Feistel network
  - We will revisit this soon

## F

- F takes a 4-bit input, expands it to 8 bits
  - 4 1 2 3 2 3 4 1
- Then adds the key

$$\begin{matrix} n_4 + K_{11} & n_1 + K_{12} & n_2 + K_{13} & n_3 + K_{14} \\ n_2 + K_{15} & n_3 + K_{16} & n_4 + K_{17} & n_1 + K_{18} \end{matrix}$$

## F

- View this as a matrix

$$\begin{matrix} p_{0,0} & p_{0,1} & p_{0,2} & p_{0,3} \\ p_{1,0} & p_{1,1} & p_{1,2} & p_{1,3} \end{matrix}$$

- First row is fed into S-box $S_0$
  Second row is fed into S-box $S_1$
- Each produces two bits
- Results are concatenated for 4-bit output

## S-boxes

- $S_0$: 4×4 matrix of 2-bit entries
  1 0 3 2
  3 2 1 0
  0 2 1 3
  3 1 3 2
- $S_1$
  0 1 2 3
  2 0 1 3
  3 0 1 0
  2 1 0 3

## S-boxes

- Select row $p_{x,0}$, $p_{x,3}$ of $S_x$
- Select column $p_{x,1}$ and $p_{x,2}$ of $S_x$
- Concatenate and permute the resulting 2-bit S-box entries
  - 2 4 3 1
- Complete the computation of f
  - xor with L, append R
- N.B.: F is applied only to R, but after swapping, F is applied to the former L.

## S-DES Example

- Big, hairy example

## Analysis of S-DES

- Exhaustive search (brute force) on key space
- Known plaintext attack
- For each ciphertext bit, can write an equation
  - $c_i = g(p_1, p_2, \ldots, p_8, k_1, k_2, \ldots, k_{10})$
- 8 equations in 10 unknowns, many (like $2^9$) terms

## Relationship to DES

- DES has 64-bit blocks, 56-bit key, 48-bit subkeys, 16 rounds
- F acts on 32-bits
- 8 S boxes, 4 by 16, containing 4-bit values
- $IP^{-1} \circ f_{K16} \circ SW \circ f_{K15} \circ SW \circ \ldots \circ f_{K1} \circ IP$

## Block Cipher Principles

- Stream cipher: one bit or byte at a time
  - E.g., Vigenere, Vernam
- Block cipher: large block, typically 8 bytes, at a time
  - Large block thwarts statistical attacks

## Block Cipher Principles

- F: $2^n \times 2^n$
- F must be reversible, i.e., 1-1 correspondence
- $2^n!$ bijections $\Rightarrow O(n \times 2^n)$ bit keys
  - 64 bit block $\Rightarrow \approx 2^{70} \approx 10^{21}$ bit key, which is far too long

## Product Ciphers

- Apply *confusion* and *diffusion* operations to thwart statistical analysis
- Diffusion
  - Thwart letter frequency analysis by dissipating statistical structure into long-range statistics relating plaintext and ciphertext
  - Accomplished by making each plaintext character affect many ciphertext characters
  - Equivalently, each ciphertext is affected by many plaintexts

## Product Ciphers

- Confusion
  - Makes statistical relationship between ciphertext and key complex
  - Generally a complex, key-dependent substitution algorithm

## Feistel Cipher Structure

- Substitution on L input
  - Substitution is based on operations applied to R
    - Result is modified L and original R, allowing decryption
  - Followed by a permutation, swapping L and R
- Multiple rounds

## Feistel Cipher Features

- Block size
  - Bigger is better, but slower
- Key size
  - Bigger is better, but slower
- Number of rounds
  - S-DES: 8-bit block, 10-bit key, 2 rounds
  - DES: 64-bit block, 56-bit key, 16 rounds

## Feistel Cipher Features

- Subkey generation algorithm
  - Greater complexity leads to more difficult cryptanalysis
- Round function
  - Greater complexity leads to more difficult cryptanalysis
- Hardware vs. software speed
  - Fast software implementation is desirable
- Ease of analysis

## Decryption

- Reverse the encryption steps
- Recall the Feistel round step: (L, R) □ (R, L ⊕ F(K, R))
- Reverse with (L, R) = (R, R ⊕ F(K, L))
- F need not be reversible

## DES

- 64-bit block, 56-bit key, 16 rounds
- 48-bit subkeys
- Initial and final (inverse) permutations

## DES Round Function

- Operates on 32-bit units
- 32-bit  48-bit expansion/permutation (E table)
- XOR with 48 bit subkey
- S-box computation returns 32 bits
- Round permutation
- Followed by Feistel XOR and swap

## S-box Details

- Eight S-boxes, each maps 6-bits to 4-bits
- One S-box contains 64 entries, each 4-bits
- Can be viewed as four permutations of {0, …, 15}
- The particular permutation is selected with the additional bits added by the E table

## DES Key Generation

- 56-bit key is split into 28-bit L and R
- Subkeys are generated by various circular left shifts of L and R
- Bits are permuted and selected

## DES Decryption

- Just as in S-DES, apply the subkeys in reverse order
- The Feistel structure does the rest

## DES Avalanche Effect

- In any good cipher, any change in the key or plaintext, no matter how large or small, should change approximately half the ciphertext bits
- Examples in Table 3.5 on p. 81
  - Change one bit in the plaintext or key
  - After 3 or 4 rounds, approximately half of the ciphertext bits are changed
  - After 16 rounds, a lot of scrambling has taken place
- Thwarts key guessing attacks

## Strength of DES

- EFF DES cracker
  - Brute force attack on 56-bit key based on Weiner's design
  - $250K custom h/w exhausts key space in about a week
  - Final nail in DES' coffin
- Timing and power attacks
  - Paul Kocher smart card attacks
  - Use knowledge of an actual implementation under test to infer key bits
  - Relies on internal calculations varying in time or power depending on input value

## DES Design Criteria

- S-box weaknesses?
  - No publicly known weaknesses
  - Design criteria remain classified
    - Why?
- What we know is based mostly on D. Coppersmith, "The Data Encryption Standard (DES) and Its Strength Against Attacks," *IBM J. of R. and D.* (May 1994)

## DES S-Box Design Criteria

- The S-box is the only source of nonlinearity in DES
- No S-box output bit should be too close to a linear function of the input bits (or any subset of them)
- In practical terms, if we select any output bit and any subset of the input bits, then the fraction of inputs for which the output bit is the xor of the input bits should be close to 1/2

## DES S-Box Design Criteria

- Each row of an S-box should be a permutation
- If two inputs to an S-box differ in exactly one bit, then the outputs must differ in at least two bits
- If two inputs to an S-box differ in exactly the middle two bits, then the outputs must differ in at least two bits

## DES S-Box Design Criteria

- If two inputs to an S-box differ in their first two bits and are identical in their last two bits, then the two outputs should not be the same
- Etc., see text. The remaining criteria have mostly to do with avalanche and resistance to differential cryptanalysis
- See the text for general information about design criteria for the round function, S-box design, and key schedule algorithm design

## Statistical Attacks on DES

- Differential cryptanalysis
  - Attack on Feistel structure
  - Murphy (FEAL), then Biiham and Shamir (DES)
  - Look for inputs with some fixed difference that produce outputs with a predictable difference
  - This allows inference of key bits
  - Round by round analysis, so more rounds  more difficult analysis
  - Leads to a $2^{47}$ chosen plaintext attack

## Statistical Attacks on DES

- Linear cryptanalysis
  - Matsui
  - Also iterated over rounds with decreasing effectiveness
  - $2^{47}$ known plaintexts

## Statistical Attacks on DES

- Smonewhat surprisingly, DES resists differential cryptanalysis
- Apparently known to NSA in 70s
  - Eight round LUCIFER requires $2^8$ chosen plaintexts
  - Eight round DES requires $2^{14}$ chosen plaintexts

## Block Cipher Modes of Operation

- How to encrypt more than one block?
- Block modes
  - Electronic codebook (ECB)
  - Cipher block chaining (CBC)
- Stream modes
  - Cipher feedback (CFB)
  - Output feedback (OFB)
  - Counter mode (CTR)

## Electronic Codebook (ECB)

- Each plaintext block is independently encrypted with the same key
- Last block is padded appropriately
- Useful for transmission of a single block or a small number of blocks
- Called a codebook because, for a given key, each block of plaintext produces a unique ciphertext
- ECB has certain weaknesses …

## Cipher Block Chaining

- Prior to encrypting a plaintext block, xor it with the previous ciphertext block
- $C_i = DES(K, C_{i-1} \oplus P_i)$
- $P_i = DES^{-1}(K, C_i) \oplus P_{i-1}$
- For first block, need "initialization vector"

## Initialization Vector Considerations

- IV must be known to sender and receiver
- Stallings (and others) recommend security for the IV
  - This prevents the adversary from modifying selected bits in the plaintext by complementing corresponding bits in the IV
- In practice, it is effective to use a fixed value, say, 0

## Cipher Feedback Mode

- Allows use of DES as a stream cipher
- Start with IV
- Encrypt
- XOR *j* bits of output with *j* bit plaintext
  - Result is ciphertext
- Shift IV by *j* bits, insert ciphertext
- Most efficient to use *j* = 64

## Cipher Feedback Decryption

- Reverse steps
- Start with IV
- Encrypt
- XOR *j* bits of output with *j* bit ciphertext
  - Result is plaintext
- Shift IV by *j* bits, insert ciphertext

## Output Feedback Mode

- Encrypt IV
- Shift IV by *j* bits, insert *j* bits of DES output
- XOR same *j* bits of output with *j* bit plaintext
  - Result is ciphertext
  - Similar to a Vernam cipher: XOR with PRNG bits
- Only use *j* = 64
- Decryption reverses these steps
- Must not reuse IV+key pair

## CFB and OFB comparison

- Errors do not propagate in OFB
- This makes OFB vulnerable to modification

## Counter Mode

- Similar to OFB but encrypts a counter (or Gray code or …) instead
- $C_i = P_i \oplus DES_K(i)$
- Supports random access
- Provably as secure as other modes
- Must not reuse counter+key pair

## Other Block Ciphers

- Triple DES
- Blowfish
- Rijndael

## Triple DES

- First consider double DES
- $C = E_{K2}(E_{K1}(P))$
- 112 bit key is safe from brute force attack
- But … $\exists K_3$ s.t. $E_{K2}(E_{K1}(P)) = E_{K3}(P)$
- No — DES is not closed
  - Not surprising, as DES occupies only a tiny portion of the space of 64-bit substitution ciphers

## Meet In The Middle Attack

- Let $X = E_{K1}(P)$. Clearly $X = D_{K2}(C)$
- Given a known $\langle P, C \rangle$, construct a table of size $2^{56}$ with all values of $K_1$ and $E_{K1}(P)$
- Sort on $E_{K1}(P)$
- Now decrypt $C$ with all values of $K_2$. Check all results against table
- Any match is a candidate $\langle K_1, K_2 \rangle$ pair
- Total effort is $O(2^{56})$, not $2^{112}$
- Works for any $F = f_2 \circ f_1$
- Hence, need three encryptions

## Triple DES

- $C = E_{K1}(D_{K2}(E_{K3}(P)))$ (EDE)
- No cryptographic significance to middle decrypt operation
  - D and E have equivalent security
  - Allows single DES by setting $K_1 = K_2 = K_3$
- Two-key 3DES: $K_1 = K_3$ (EDE)
  - Shorter key is easier to work with
  - Textbook describes several impractical attacks
- Three-key 3DES

## Other Block Ciphers

- IDEA
- Blowfish
- RC5 (not covered)
- CAST-128
- RC2
- Rijndael (AES)

## IDEA

- Xuejia Lai and James Massey, ETH
- Patented
- Used in PGP
- 128-bit key, 64-bit block
- Variant Feistel network
- Eight rounds + final transformation

## IDEA Round Function

- Round function mixes four 16-bit blocks
- Each round takes six 16-bit subkeys
  - Final transformation uses four 16-bit subkeys
  - Total of 52 subkeys
  - Subkeys are derived through circular shifts
- Uses three operations
  - XOR
  - Addition modulo $2^{16}$
  - Multiplication modulo $2^{16} + 1$

## Blowfish

- Bruce Schneier, ca. '93
- Freely available
- Used in SSH, OpenBSD, IPSec
- 64-bit block, 32- to 448-bit keys
- Fast encryption, small memory footprint
- Slow key schedule computation
  - Four S-boxes, each containing 256 32-bit values
  - 18 32-bit subkeys ("P array")

## Blowfish Initialization

- P and S are initialized with fractional part of π
- P is XORed with the key (reusing key bits if necessary)
- P and S are churned through Blowfish
- 521 executions in total
  - Excellent defense against dictionary attack

## Blowfish Encryption

- Slight variant on classic Feistel network
  - L and R are both processed in each round
  - 16 rounds
  - Two extra XORs at the end
- Uses addition modulo $2^{32}$ and XOR
- Round function processes four bytes
  - $F(a, b, c, d) = ((S_{1,a} + S_{2,b}) \oplus S_{3,c}) + S_{4,d}$
  - Followed by Feistel swap
- This is fast

## CAST-128

- Carlisle Adams and Stafford Tavares
- Used in IPSec
- 64-bit block, 40- to 128-bit keys
- Classic Feistel network
- Sixteen rounds

## CAST-128 Round Function

- Two subkeys per round, one 32-bit, one 5-bit
- Three different round functions
- Four operations: addition and subtraction modulo $2^{32}$, XOR, and (variable) circular shift
  - 5-bit subkey determines shift amount

## CAST-128 Round Function

- Round function uses four S-boxes
  - Fixed values
  - Indexed by four 8-bit blocks
  - Each contains 256 32-bit values
  - 32-bit values are combined in round-dependent ways
  - Produces 32-bit output

## Characteristics of Advanced Block Ciphers

- Variable key length
  - Blowfish, RC5, CAST-128, RC2
- Mixed operators
  - Especially ones that are not associative or distributive
- Data-dependent rotation
  - RC5
- Key-dependent rotation
  - CAST-128

## Characteristics of Advanced Block Ciphers

- Key-dependent S-boxes
- Expensive key schedule computation
  - Blowfish
- Variable round function
  - CAST-128
- Variable block length
- Variable number of rounds
- Operation on L and R in the round function

## Rijndael

- Selected as AES
- 128-bit block, 128-, 192-, or 256-bit key, 9, 11, or 13 rounds
- Simple and fast

## Rijndael Round Function

- Substitution
- Permutation
- Multiplication over GF($2^8$)
- Addition over GF($2^8$)
- Not a Feistel network

## GF($2^8$)

- All representations of a finite field over a prime power (in this case 2) are isomorphic
- Treat a byte $b_7b_6b_5b_4b_3b_2b_1b_0$ as a polynomial
  - $b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x^1 + b_0x^0$
- Addition: bitwise modulo 2, i.e., XOR
  - 0x57 + 0x83 = 0xD4
  - 
```
  0101 0111
+ 1000 0011
  1101 0100 = 0xD4
```

## GF($2^8$) Multiplication

- Multiplication
  - Multiplication of polynomials modulo an irreducible binary polynomial of degree 8
  - For Rijndael, it is 0x11B
  - $x^8 + x^4 + x^3 + x + 1$

## GF($2^8$) Multiplication Example

- 0x57 • 0x83 = 0xC1
- ```
          0101 0111
   •      1000 0011
          0101 0111
        0 1010 1110
   01 0101 1100 0000
  010 1011 0111 1001 = 0x2B79
  ```
- 0x2B79 modulo 0x11B = 0xC1

## GF($2^8$)

- Addition is an Abelian group: closed, associative, commutative, every element has an inverse (itself), and there is an additive identity (0)
- Multiplication: closed, associative, commutative, there is a multiplicative identity (0x01), every element (except 0) has an inverse
  - Also an Abelian group (ignoring 0)

## GF($2^8$)

- Distributive law holds over addition and multiplication
  - $a \cdot (b + c) = ab + ac$
- GF($2^8$) is a field

## Polynomials with coefficients in GF($2^8$)

- Treat a four-byte vector as four coefficients of a polynomial of degree less than four
- Addition is still XOR
- For multiplication, use polynomial $x^4 + 1$
  - Not irreducible, but relatively prime to the polynomials we multiply with, so the step is invertible.

## Rijndael Round Function

- Substitution
  - S-box replaces each byte with its multiplicative inverse followed by addition of 0x63
- Permutation
  - Represent 16 bytes as 4 × 4 block
  - $i^{th}$ row is shifted by $i$ elements ($0 \le i \le 3$)
  - Same for 24 bytes (4 × 6)
  - For 32 bytes (4 × 8), extra shift for rows 2 and 3

## Rijndael Round Function

- Multiplication of polynomials over GF($2^8$)
  - Can be represented as matrix multiplication
- Addition over GF($2^8$)
  - Add round key
- Initialize with round key addition
- Last round omits multiply
  ```

## Rijndael Key Schedule

- Expand cipher key into (block length) □ (number of rounds + 1) bits
- Round keys are taken from the expanded key in the natural way
- Key expansion is defined recursively, using rotation, S-boxes, and XOR with a round constant

## Rijndael Decryption

- Each step is invertible

## Rijndael Performance

- 8-bit □P
  - ~ 1 KB code size, < 100 bytes RAM, 10-50K clock cycles
- 32-bit □P
  - 300-3,000 cycles for key schedule, 1,500-4,000 cycles for inverse key schedule
  - 20-50 Mbits/sec encryption speed on 200 Mhz Pentium