Substitution Cipher

• A substitution cipher replaces one symbol for another in the alphabet
  ‣ Caesar cipher and rot13 are a specific kind (rotation)
  ‣ The most common is a *random permutation* cipher
Why are substitution ciphers breakable?

- Substitution ciphers are breakable because they don’t hide the underlying frequency of characters. You can use this information if you know the target language frequency count.

- For example, in English ...
  - e, t, a, o, i, n, s, r, h, d, l, u, c, m, f, y, w, g, p, b, v, k, x, q, j, z

- Q: how do you exploit this?
Using frequency ..

- Vg gbbx n ybg bs oybbq, fjrng naq grnef gb trg gb jurer jr ner gbqnl, ohg jr unir whfg ortha. Gbqnl jr ortva va rnearfg gur jbex bs znxvat fher gung gur jbeyq jr yrnir bhe puvyqera vf whfg n yvggyr ovg orggre guna gur bar jr vaunovg gbqnl.
Using frequency ..

- Vg gbbx n ybg bs oybbq, fjrng naq grnef gb trg gb jurer jr ner gbqnl, ohg jr unir whfg ortha. Gbqnl jr ortva va rnearfg gur jbex bs znxvat fher gung gur jbevyq jr yrni r bhe puvyqera vf whfg n yvggyr ovg orggreguna gur bar jr vaunovg gbqnl.

- It took a lot of blood, sweat and tears to get to where we are today, but we have just begun. Today we begin in earnest the work of making sure that the world we leave our children is just a little bit better than the one we inhabit today.

‘r’ appears very frequently so very likely is one of the top frequency letters.
Using frequency ..

• Vg gbbx n ybg bs oybbq, fjrng naq grnef gb trg gb jurer jr ner gbqnl, ohg jr unir whfg ortha. Gbqnl jr ortva va rnearfg gur jbex bs znxvat fher gung gur jbeyq jr yrnir bhe puvyqera vf whfg n yvggyr ovgrggre guna gur bar jr vaunovg gbqnl.

• It took a lot of blood, sweat and tears to get to where we are today, but we have just begun. Today we begin in earnest the work of making sure that the world we leave our children is just a little bit better than the one we inhabit today.

Repeat this process, picking out more letters, then common words, e.g., ‘the’ ... which gives (e to r), (g to t), and (u to h)
Attacking a Cipher

• The attack mounted will depend on what information is available to the adversary

  ‣ **Ciphertext-only attack**: adversary only has the ciphertext available and wants to determine the plaintext encrypted

  ‣ **Known-plaintext attack**: adversary learns one or more pairs of ciphertext/plaintext encrypted under the same key, tries to determine plaintext based on a different ciphertext

  ‣ **Chosen-plaintext attack**: adversary can obtain the encryption of any plaintext, tries to determine the plaintext for a different ciphertext

  ‣ **Chosen-ciphertext attack**: adversary can obtain the plaintext of any ciphertext except the one the adversary wants to decrypt
Generic Block Encryption

- Converts one input *plaintext block of fixed size n bits* to output ciphertext block also of n bits
- Benefits of large n? of short n?

![Diagram of block encryption process]
Two Principles for Cipher Design

• **Confusion**: Make the relationship between the <plaintext, key> input and the <ciphertext> output as complex (non-linear) as possible
  ‣ Mainly accomplished by *substitution*

• **Diffusion**: Spread the influence of each input bit across many output bits
  ‣ Mainly accomplished by *permutation*

• **Idea**: use *multiple, alternating* permutations and substitutions
  ‣ S→P→S→P→S→... or P→S→P→S→P→...
  ‣ Does it have to alternate?, e.g., S→S→S→P→P→P→S→S→...
Basic Form of Modern Block Ciphers

Preprocessing

Rounds of Encryption
$i=1,2,\ldots,n$

Postprocessing

Ciphertext block

Sub-Key Generation

Sub-Key #1
Sub-Key #2
Sub-Key #3
...
Sub-Key #n
Feistel Cipher

- Very influential “template” for designing block ciphers
- Major benefit: do encryption and decryption w/ same hardware
- One “round” of Feistel Encryption
  1. Break input block i into left and right halves L_i and R_i
  2. Copy R_i to create output half block L_{i+1}
  3. Half block R_i and key K_i are “scrambled” by function f
  4. XOR result with input half-block L_i to create output half-block R_{i+1}
One “Round” of Feistel Decryption

• Just reverse the arrows!
Complete Feistel Cipher: Encryption

Round 1

Round i

Round n

note this final swap!

Plaintext (2w bits)

Ciphertext (2w bits)
Complete Feistel Cipher: Decryption

Ciphertext (2w bits)

Plaintext (2w bits)

Round 1

Round i

Round n

note this final swap!

L₀ → f → R₀ = Kₙ

L₁ → f → R₁ = Kₙ₋₁

L₂ → f → R₂ = K₁

Lₙ → f → Rₙ = K₁

Lᵢ₊₁ → f → Rᵢ₊₁

note this final swap!
Data Encryption Standard (DES)

- Introduced by the US NBS (now NIST) in 1972
- Signaled the beginning of the modern area of cryptography

- Basics
  - Feistel Cipher
  - 8-byte (64 bit) input
  - 8-byte (64 bit) output
  - 8-byte key (56-bits+8 parity bits)
  - 16 rounds
DES Round: $f$ (Mangler) Function

Input block $i$

$\begin{align*}
L_i & \quad R_i \\
\downarrow & \quad \uparrow \quad f \\
\downarrow & \quad \downarrow \\
L_{i+1} & \quad R_{i+1}
\end{align*}$

Output block $i+1$

function $f = “Mangler”$

32-bit half block

Expansion

48 bits

S-Box (substitution)

Permutation

32-bit half block
Advanced Encryption Standard (AES)

- International NIST bakeoff between cryptographers
  - Rijndael (pronounced “Rhine-dall”)
- Replacement for DES/accepted symmetric key cipher
  - Substitution-permutation network, not a Feistel network
  - Variable key lengths (128, 192, or 256 bits)
  - Block size: 128 bits
  - Fast implementation in hardware and software
  - Small code and memory footprint
• Most ciphers work on blocks of fixed (small) size

• How to encrypt long messages?

• Modes of operation
  ‣ ECB (Electronic Code Book)
  ‣ CBC (Cipher Block Chaining)
  ‣ OFB (Output Feedback)
  ‣ CFB (Cipher Feedback)
  ‣ CTR (Counter)
Issues for Block Chaining Modes

• **Information leakage**: Does it reveal info about the plaintext blocks?

• **Ciphertext manipulation**: Can an attacker modify ciphertext block(s) in a way that will produce a predictable/desired change in the decrypted plaintext block(s)?
  
  ‣ Note: assume the structure of the plaintext is known, e.g., first block is employee #1 salary, second block is employee #2 salary, etc.

• **Parallel/Sequential**: Can blocks of plaintext (ciphertext) be encrypted (decrypted) in parallel?

• **Error Propagation**: If there is an error in a plaintext (ciphertext) block, will there be an encryption (decryption) error in more than one ciphertext (plaintext) block?
Electronic Code Book (ECB)

- Easiest mode of operation; each block *independently* encrypted
ECB Issues

- **Information leaks**: two ciphertext blocks that are the same
- **Manipulation**: switch ciphertext with predictable results on plaintext
- **Parallel**: yes
- **Propagate**: no

**ECB Decryption**
Cipher Block Chaining (CBC)

• Chaining dependency: each ciphertext block depends on all preceding plaintext blocks
Initialization Vectors

- **Initialization Vector (IV)**
  - Used along with the key; not secret
  - For a given plaintext, changing either the key, or the IV, will produce a different ciphertext
  - Why is that useful?

- **IV generation and sharing**
  - Random; may transmit with the ciphertext
  - Incremental; predictable by receivers
CBC Properties

- **Info leak:** identical plaintext blocks produce different ciphertext blocks
- **Manipulation:** flipping bit \( i \) of ciphertext block \( b \) results in flipping bit \( i \) of decrypted block \( b+1 \)
- **Parallel:**
  - enc: no; dec: yes
- **Error prop:**
  - enc: yes; dec: a little

How many ciphertext blocks does each plaintext block depend on?
Output Feedback Mode (OFB)

Initialization Vector

Key

$E$

$C_1\rightarrow C_2\rightarrow C_3\rightarrow C_4$

$M_1\rightarrow M_2\rightarrow M_3\rightarrow M_4$

one-time pad

64

46 + padding

64

64

64

64

64

64
OFB Properties

- **Info leak**: identical plaintext blocks produce different ciphertext blocks

- **Manipulation**: flipping bit \(i\) of ciphertext block \(b\) results in flipping bit \(i\) of decrypted block \(b\)

- **Parallel**:
  - no (gen pad), yes (XORing w/ blocks)

- **Error prop**:
  - ???

---

**OFB Decryption**

If you know one plaintext/ciphertext pair, can easily derive one-time pad that was used!

**IV must be different every time!**

No block decryption required!
Cipher Feedback Mode (CFB)

- Ciphertext block $C_j$ depends on all preceding plaintext blocks
CFB Properties

- **Info leak**: identical plaintext blocks produce different ciphertext blocks
- **Manipulation**: modify single block, but next plaintext block will be garbled
- **Parallel**:
  - enc: no; dec: yes
- **Error prop**:
  - ???

CFB Decryption

No block decryption required!
Counter Mode (CTR)
CTR Mode Properties

- **Info leak**: identical plaintext blocks produce different ciphertext blocks

- **Manipulation**: modify any single block in a predictable way

- **Parallel**:
  - Yes (both gen pad and XORing)

- **Error prop**:
  - ???

- Allow decryption of the ciphertext at any location
  - Ideal for random access to ciphertext
Birthday Attack

• A birthday attack is a name used to refer to a class of brute-force attacks.
  – birthday paradox: the probability that two or more people in a group of 23 share the same birthday is >than 50%

• General formulation
  – function f() whose output is uniformly distributed
  – On repeated random inputs $n = \{ n_1, n_2, \ldots, n_k \}$
    • $Pr(f(n_i) = f(n_j)) = 1.2\sqrt{k}$, for $1 \leq i, j \leq k$ and $i \neq j$
    • E.g., $1.2\sqrt{365} \approx 23$

• Q: Why is resilience to birthday attacks important?
Cryptanalysis and Protocol Analysis

• Cryptographic Algorithms
  ‣ Complex mathematical concepts
  ‣ May be flawed
  ‣ What approaches are used to prove correct/find flaws?

• Cryptographic Protocols
  ‣ Complex composition of algorithms and messages
  ‣ May be flawed
  ‣ What approaches are used to prove correct/find flaws?
Timing Attacks

• Use the timing behavior of system to extract secret

• Suppose a smartcard stores your private key
  ‣ By precisely measuring the time it takes to perform private key ops, we can recover the key
  ‣ Due to Kocher
  ‣ At most 2^n operations required, where n is the number of bits in the key

• Attack summary
  ‣ Adversary asks smartcard to generate signatures on several messages
  ‣ Recover one bit at a time starting with least significant
  ‣ Compare times to those measured offline

• Solution: blinding
Power Analysis Attacks

• Also, Discovered by Kocher
  ‣ Power usage is higher than normal in these computations
  ‣ Measure the timing of high power consumption

• Simple Power Analysis
  ‣ Direct interpretation of power measurements
  ‣ Reveals instructions executions
  ‣ Some crypto ops may be sensitive to data, e.g., DES S-boxes

• Differential Power Analysis
  ‣ Statistical analysis of power data correlations

• Solution: Gotta change the code
Power and Timing

• What is the threat model in power/timing attacks?
• How does this conflict with the trust model?
• What is the vulnerability?
Textbook Problems

• 2.1, 2.3, 2.6
• 20.2, 20.8

• Review questions from chapters 2, 20, 21

• 2nd edition of book