Crypto Basics

Cryptography Overview
Public vs. Private Key Cryptography
Classical / ancient ciphers
Modern ciphers: DES
Modes of operation
Stream cipher: RC4
What is a cryptosystem?

- $K = \{0,1\}^l$
- $P = \{0,1\}^m$
- $C' = \{0,1\}^n$, $C \subseteq C'$

- $E: P \times K \rightarrow C$
- $D: C \times K \rightarrow P$

- $\forall p \in P, k \in K: D(E(p,k),k) = p$
  - It is *infeasible* to find inversion $F: P \times C \rightarrow K$

Let's start again!
This time in English ...
What is a cryptosystem?

- A pair of algorithms that take a key and convert plaintexts to ciphertexts and backwards later
  - **Plaintext:** text to be protected
  - **Ciphertext:** should appear like random

- Requires sophisticated math!
  - Do not try to design your own algorithms!
The language of cryptography

- **Symmetric or secret key crypto:** sender and receiver keys are identical and *secret*

- **Asymmetric or Public-key crypto:** encrypt key public, decrypt key secret
Attacks

- Opponent whose goal is to break a cryptosystem is the adversary
  - Assume adversary knows algorithm used, but not key

- Three types of attacks:
  - ciphertext only:
    - adversary has only ciphertext; goal is to find plaintext, possibly key
  - known plaintext:
    - adversary has ciphertext, corresponding plaintext; goal is to find key
  - chosen plaintext:
    - adversary may supply plaintexts and obtain corresponding ciphertext; goal is to find key
Basis for Attacks

- Mathematical attacks
  - Based on analysis of underlying mathematics

- Statistical attacks
  - Make assumptions about the distribution of letters, pairs of letters (digrams), triplets of letters (trigrams), etc.
    - Called *models of the language*
  - Examine ciphertext, correlate properties with the assumptions.
Example: Symmetric key cryptography

Substitution cipher: substituting one thing for another
- Monoalphabetic cipher: substitute one letter for another

plaintext: abcdefghijklmnopqrstuvwxyz
ciphertext: mnbvcxzasdfghjklpoiuytrewq

E.g.: Plaintext: bob. i love you. alice
ciphertext: nkn. s gktc wky. mgsbc
Monoalphabetic Cipher Security

- Total of $26! = 4 \times 1026$ keys

- So many keys, might think is secure

- !!!WRONG!!!

- Problem is language characteristics
Language Redundancy and Cryptanalysis

- Human languages are redundant

- Eg "th lrd s m shphrd shll nt wnt"

- Letters are not equally commonly used

- In English E is by far the most common letter
  - followed by T, R, N, I, O, A, S

- Other letters like Z, J, K, Q, X are fairly rare

- Have tables of single, double & triple letter frequencies for various languages
English Letter Frequencies

The diagram illustrates the relative frequency of each letter in the English language. The letter 'E' appears most frequently, followed by 'T', 'A', 'O', and 'N'. The least frequent letters are 'Z', 'Q', and 'X'. The relative frequency is given in percentage, with 'E' being the highest at approximately 12.702%.
Use in Cryptanalysis

- **Key concept**
  - monoalphabetic substitution ciphers do not change relative letter frequencies

- Discovered by Arabian scientists in 9th century

- Calculate letter frequencies for ciphertext

- Compare counts/plots against known values

- For monoalphabetic must identify each letter
  - tables of common double/triple letters help
Properties of a good cryptosystem

- There should be no way short of enumerating all possible keys to find the key from any reasonable amount of ciphertext and/or plaintext, nor any way to produce plaintext from ciphertext without the key.

- Enumerating all possible keys must be infeasible.

- The ciphertext must be indistinguishable from true random values.
Milestones in modern cryptography

- 1883 Kerckhoffs’ principles

- 1917-1918 Vernam/Mauborgne cipher (one-time pad)

- 1920s-1940s Mathematicization and mechanization of cryptography and cryptanalysis

- 1973 U.S. National Bureau of Standards issues a public call for a standard cipher; this led to the adoption of the Data encryption Standard (DES)
Milestones in modern cryptography: Public key cryptography

- Merkle invents a public key distribution scheme

- 1976: Diffie and Hellman invent public key encryption and digital signatures, but do not devise a suitable algorithms with all desired properties

- 1977: Rivest, Shamir, and Adelman invent their algorithm RSA soon after

  - His discovery, however, was not revealed until 1997 due to its top-secret classification, and Rivest, Shamir, and Adleman devised RSA independently of Cocks' work.
Kerckhoff’s Law

- “The system must not be required to be secret, and it must be able to fall into the hands of the enemy without inconvenience.”

- In other words, the security of the system must rest entirely on the secrecy of the key not the algorithm itself.
Vernam/Mauborgne cipher

- Exclusive-OR a key stream tape with the plaintext

- Online encryption of teletype traffic, combined with transmission

- For a one-time pad – which is provably secure – use true-random keying tapes and never reuse the keying material

- Problem: how to get good long one-time pads
  - Reuse of keying material ) stream cipher
  - Key stream via algorithm ) no one-time pad
Mathematicization and mechanization

- Mechanical encryptors
  (Vernam, Enigma, Hagelin, Scherbius)

- Mathematical cryptanalysis
  (Friedman, Rejewski et al., Bletchley Park)

- Machine-aided cryptanalysis
  (Friedman, Turing et al.)
Hagelin Rotor Machine
Standardized ciphers

- Until the 1970s, most strong ciphers were government secrets

- Spread of computers ⇒ new threats (Reportedly, soviets eavesdropped on U.S. grain negotiators’ conversations)

- NBS (now called NIST) issued public call for cipher; eventually IBM responded

  ⇒ eventual result – via secret process – DES
What we have today

- Encryption is completely computerized and operates on bits

- Basic primitives can be combined to produce powerful results
  - Difficult to verify combined result.

- Encryption is by far the strongest weapon of computer security

- Host and OS software is by far the weakest link

- Bad software breaks crypto – NEVER the cryptanalysis.
Modern Block Ciphers

- Look at modern block ciphers
- One of the most widely used types of cryptographic algorithms
- Provides secrecy / authentication services
- Focus now on DES (Data Encryption Standard)
- Illustrate block cipher design principles
Block vs. Stream Ciphers

- block ciphers process messages in blocks, each of which is then en/decrypted
  - like a substitution on very big characters
    - 64-bits or more

- stream ciphers process messages a bit or byte at a time when en/decrypting

- many current ciphers are block ciphers

- broader range of applications
**Block Cipher Principles**

- most symmetric block ciphers are based on a so called
  - **Feistel Cipher Structure**

- needed since must be able to *decrypt* ciphertext to recover messages efficiently

- block ciphers look like an extremely large substitution

- would need table of $2^{64}$ entries for a 64-bit block

- instead create from smaller building blocks

- using idea of a product cipher
Ideal Block Cipher
Claude Shannon and Substitution-Permutation Ciphers

- Claude Shannon introduced idea of substitution-permutation (S-P) networks in 1949 paper

- form basis of modern block ciphers

- S-P nets are based on the two primitive cryptographic operations seen before:
  - substitution (S-box)
  - permutation (P-box)

- provide confusion & diffusion of message & key
Confusion and Diffusion

- cipher needs to completely obscure statistical properties of original message

- a one-time pad does this

- more practically Shannon suggested combining S & P elements to obtain:
  
  - **diffusion** – dissipates statistical structure of plaintext over bulk of ciphertext
  
  - **confusion** – makes relationship between ciphertext and key as complex as possible
Feistel Cipher Structure

- Horst Feistel devised the **feistel cipher**
  - based on concept of invertible product cipher

- partitions input block into two halves
  - process through multiple rounds which
  - perform a substitution on left data half
  - based on round function of right half & subkey
  - then have permutation swapping halves

- implements Shannon’s S-P net concept
Feistel Cipher Structure
Feistel Cipher Design Elements

- block size
- key size
- number of rounds
- subkey generation algorithm
- round function
- fast software en/decryption
- ease of analysis
Feistel Cipher Decryption
Data Encryption Standard (DES)

- most widely used block cipher in world

- adopted in 1977 by NBS (now NIST)
  - as FIPS PUB 46

- encrypts 64-bit data using 56-bit key

- has widespread use

- has been considerable controversy over its security
DES History

- IBM developed Lucifer cipher
  - by team led by Feistel in late 60’s
  - used 64-bit data blocks with 128-bit key

- then redeveloped as a commercial cipher with input from NSA and others

- in 1973 NBS issued request for proposals for a national cipher standard

- IBM submitted their revised Lucifer which was eventually accepted as the DES
DES Design Controversy

- although DES standard is public

- was considerable controversy over design
  - in choice of 56-bit key (vs Lucifer 128-bit)
  - and because design criteria were classified

- subsequent events and public analysis show in fact design was appropriate

- use of DES has flourished
  - especially in financial applications
  - still standardised for legacy application use
DES Encryption Overview
Initial Permutation IP

- first step of the data computation
- IP reorders the input data bits
- even bits to LH half, odd bits to RH half
- quite regular in structure (easy in h/w)

example:

\[ \text{IP}(675a6967 \ 5e5a6b5a) = (ffb2194d \ 004df6fb) \]
DES Round Structure

- uses two 32-bit L & R halves

- as for any Feistel cipher can describe as:
  
  \[
  L_i = R_{i-1} \\
  R_i = L_{i-1} \oplus F(R_{i-1}, K_i)
  \]

- F takes 32-bit R half and 48-bit subkey:
  - expands R to 48-bits using perm E
  - adds to subkey using XOR
  - passes through 8 S-boxes to get 32-bit result
  - finally permutes using 32-bit perm P
DES Round Structure
Substitution Boxes $S$

- have eight S-boxes which map 6 to 4 bits
- each S-box is actually 4 little 4 bit boxes
  - outer bits 1 & 6 (row bits) select one row of 4
  - inner bits 2-5 (col bits) are substituted
  - result is 8 lots of 4 bits, or 32 bits
- row selection depends on both data & key
  - feature known as autoclaving (autokeying)
- example:
  - $S(18\ 09\ 12\ 3d\ 11\ 17\ 38\ 39) = 5fd25e03$
DES Key Schedule

- forms subkeys used in each round
  - initial permutation of the key (PC1) which selects 56-bits in two 28-bit halves
  - 16 stages consisting of:
    - rotating each half separately either 1 or 2 places depending on the key rotation schedule $K$
    - selecting 24-bits from each half & permuting them by PC2 for use in round function $F$

- note practical use issues in h/w vs. s/w
**DES Decryption**

- decrypt must unwind steps of data computation

- with Feistel design, do encryption steps again using subkeys in reverse order (SK16 ... SK1)
  - IP undoes final FP step of encryption
  - 1st round with SK16 undoes 16th encrypt round
  - ...
  - 16th round with SK1 undoes 1st encrypt round
  - then final FP undoes initial encryption IP
  - thus recovering original data value
Avalanche Effect

- Key desirable property of encryption alg

- Where a change of one input or key bit results in changing approx half output bits

- Making attempts to “home-in” by guessing keys impossible

- DES exhibits strong avalanche
Strength of DES – Key Size

- 56-bit keys have $2^{56} = 7.2 \times 10^{16}$ values

- brute force search looks hard

- recent advances have shown is possible
  - in 1997 on Internet in a few months
  - in 1998 on dedicated h/w (EFF) in a few days
  - in 1999 above combined in 22hrs!

- still must be able to recognize plaintext

- must now consider alternatives to DES – AES
Strength of DES – Analytic Attacks

- now have several analytic attacks on DES
- these utilise some deep structure of the cipher
  - by gathering information about encryptions
  - can eventually recover some/all of the sub-key bits
  - if necessary then exhaustively search for the rest
- generally these are statistical attacks
- include
  - differential cryptanalysis
  - linear cryptanalysis
  - related key attacks
Differential Cryptanalysis

- one of the most significant recent (public) advances in cryptanalysis

- known by NSA in 70's cf DES design

- Murphy, Biham & Shamir published in 90’s

- powerful method to analyse block ciphers

- used to analyse most current block ciphers with varying degrees of success

- DES reasonably resistant to it, cf. Lucifer
Differential Cryptanalysis

- a statistical attack against Feistel ciphers
- uses cipher structure not previously used
- design of S-P networks has output of function $f$ influenced by both input & key
- hence cannot trace values back through cipher without knowing value of the key
- differential cryptanalysis compares two related pairs of encryptions
Differential Cryptanalysis Compares Pairs of Encryptions

- with a known difference in the input
- searching for a known difference in output
- when same subkeys are used

\[
\Delta m_{i+1} = m_{i+1} \oplus m'_{i+1} \\
= [m_{i-1} \oplus f(m_i, K_i)] \oplus [m'_{i-1} \oplus f(m'_i, K_i)] \\
= \Delta m_{i-1} \oplus [f(m_i, K_i) \oplus f(m'_i, K_i)]
\]
Differential Cryptanalysis

- have some input difference giving some output difference with probability \( p \)

- if find instances of some higher probability input/output difference pairs occurring

- can infer subkey that was used in round

- then must iterate process over many rounds (with decreasing probabilities)
Differential Cryptanalysis

\[ \Delta m_{i+1} \parallel \Delta m_i = 40 \ 08 \ 00 \ 00 \ 04 \ 00 \ 00 \ 00 \]

\[ f(\Delta m_i) = 40 \ 08 \ 00 \ 00 \]
\[ \Delta m_i = 04 \ 00 \ 00 \ 00 \]
\[ p = 0.25 \]

\[ f(\Delta m_{i+1}) = 00 \ 00 \ 00 \ 00 \]
\[ \Delta m_{i+1} = 00 \ 00 \ 00 \ 00 \]
\[ p = 1.0 \]

\[ f(\Delta m_{i+2}) = 40 \ 08 \ 00 \ 00 \]
\[ \Delta m_{i+2} = 04 \ 00 \ 00 \ 00 \]
\[ p = 0.25 \]

\[ \Delta m_{i+3} \parallel \Delta m_{i+2} = 40 \ 08 \ 00 \ 00 \ 04 \ 00 \ 00 \ 00 \]
Differential Cryptanalysis

- perform attack by repeatedly encrypting plaintext pairs with known input XOR until obtain desired output XOR

- when found
  - if intermediate rounds match required XOR have a right pair
  - if not then have a wrong pair

- can then deduce keys values for the rounds
  - right pairs suggest same key bits
  - wrong pairs give random values

- for large numbers of rounds, probability is so low that more pairs are required than exist with 64-bit inputs

- Biham and Shamir have shown how a 13-round iterated characteristic can break the full 16-round DES
Linear Cryptanalysis

- another recent development
- also a statistical method
- must be iterated over rounds, with decreasing probabilities
- developed by Matsui et al in early 90's
- based on finding linear approximations
- can attack DES with $2^{43}$ known plaintexts, easier but still in practise infeasible
Linear Cryptanalysis

- find linear approximations with prob \( p \neq \frac{1}{2} \)
  \[
P[i_1, i_2, \ldots, i_a] \oplus C[j_1, j_2, \ldots, j_b] = K[k_1, k_2, \ldots, k_c]
\]
  where \( i_a, j_b, k_c \) are bit locations in \( P, C, K \)

- gives linear equation for key bits

- get one key bit using max likelihood alg

- using a large number of trial encryptions

- effectiveness given by: \(|p - \frac{1}{2}|\)
DES Design Criteria

- as reported by Coppersmith in [COPP94]

- 7 criteria for S-boxes provide for
  - non-linearity
  - resistance to differential cryptanalysis
  - good confusion

- 3 criteria for permutation P provide for
  - increased diffusion
Block Cipher Design

- basic principles still like Feistel’s in 1970’s

- number of rounds
  - more is better, exhaustive search best attack

- function f:
  - provides “confusion”, is nonlinear, avalanche
  - have issues of how S-boxes are selected

- key schedule
  - complex subkey creation, key avalanche
How to use a block cipher

- Direct use of a block cipher is inadvisable
  - Enemy can build up „code book“ of plaintext/ciphertext equivalents
  - Only works for messages that are a multiple of the block size

- Solution: 5 standard modes of operation
  - Electronic Code Book (ECB)
  - Cipher Block Chaining (CBC)
  - Cipher Feedback (CFB)
  - Output Feedback (OFB)
  - Counter (CTR)
Codes vs. Ciphers

- Ciphers operate *syntactically*, on elements of an alphabet (letters) or groups of “letters”: 
  \[ A \rightarrow D, \ B \rightarrow C, \text{ etc.} \]

- Codes operate *semantically*, on words, phrases, or sentences, e.g., per codebooks
Electronic Code Book

- Direct use of block cipher
- Used primarily to transmit encrypted keys
- Very weak for general-purpose encryption
- Problem: block substitution attack
Cipher Block Chaining (CBC)

- IV: Initialization vector, P: plaintext, C: ciphertext
Cipher Block Chaining

Properties of CBC

- Ciphertext of each encrypted block depends on the plaintext of all preceding blocks
- Subsets of blocks appear valid and will decrypt properly
- Message integrity has to be done otherwise

CBC and electronic voting

[Kohno, Stubblefield, Rubin, Wallach]
- Found in the source code for Diebold voting machines:

```c
DesCBCEncrypt((des_c_block*)tmp,
(des_c_block*)record.m_Data, totalSize,
DESKEY, NULL, DES_ENCRYPT)
```
ECB vs. CBC

AES in ECB mode

Similar plaintext blocks produce similar ciphertext blocks (not good!)

AES in CBC mode

[Picture due to Bart Preneel]
Information leakage in ECB mode

[Wikipedia]
n-Bit Cipher Feedback

- Add n-bit shift and move Encrypt operation before X-OR operator
- Retains some of the previous cycle’s ciphertext
- Copes gracefully with deletion of n-bit unit (bit errors)
n-Bit Output Feedback

- No error propagation
- Active attacker can make controlled changes to plaintext
- OFB is a form of stream cipher
Counter mode

- Another form of stream cipher
- Counter often split in message and block number
- Active attack can make controlled changes to plaintext
- Highly parallelizable
- No linkage between stages
- Vital: Counter never to repeat
Which mode for what task

- General file or packet encryption: CBC
  ⇒ Input must be padded to \( n \times \) cipher block size

- Risk of byte or bit deletion: CFB\(_8\) or CFB\(_1\)

- Bit stream: noisy line and error propagation is undesirable: OFB

- Very high-speed data: CTR

- Needed in most situations: integrity checks
  - Actually needed almost always
  - Attack on integrity ⇒ attack on confidentiality
  - Solution: separate integrity check along with encryption
Stream ciphers

- **Operation:**
  - Key stream generator produces a sequence $S$ of pseudo-random bytes
  - Key stream bytes are combined (usually via XOR) with plaintext bytes

- **Properties:**
  - Very good for asynchronous traffic
  - Best-known stream cipher RC4 (used, e.g., in SSL)
  - Key stream must never be reused for different plaintexts
RC4

- Extremely efficient
- After key setup, it just produces a key stream
- Internal state: 256-byte array plus two integers

For as many iterations as are needed, the RC4 modifies the state and outputs a byte of the keystream. In each iteration, it increments $i$, adds the value of $S$ pointed to by $i$ to $j$, exchanges the values of $S[i]$ and $S[j]$, and then outputs the value of $S$ at the location $S[i] + S[j]$ (modulo 256). Each value of $S$ is swapped at least once every 256 iterations.

- No resynchronization except via rekeying + starting over
- Note:
  known weaknesses if used other than as stream cipher
CPU speed vs. key size

- Adding one bit to the key doubles work for brute force attack

- Effect on encryption time is often negligible or even free

- It costs nothing to use a longer RC4 key

- Going from 128-bit AES to 256-bit AES takes (at most) 40% longer for en-/decryption but increases the attacker’s effort by a factor of $2^{128}$

- Using triple DES costs $3 \times$ more to encrypt, but increases the attacker’s effort by a factor of $2^{112}$

- Moore’s Law favors the defender!
Summary

Have considered:
- Block vs. stream ciphers
- Feistel cipher design & structure
- DES
  - details
  - strength
- Differential & Linear Cryptanalysis
- Block cipher design principles
- Use of a block cipher: Modes of operation