# University of Washington
# CSEP 590TU – Practical Aspects of Modern Cryptography

*Instructors:* Josh Benaloh, Brian LaMacchia, John Manferdelli

*Tuesdays: 6:30-9:30, Allen Center 305*

*Webpage: http://www.cs.washington.edu/education/courses/csep590/06wi/*

*Recommended texts:*

*Stinson,* Cryptography, Theory and Practice. 2nd Edition, CRC Press, 2002.

*Menezes, vanOrtshot, Vanstone.* Handbook of Applied Cryptography.

*Ferguson and Schneier*, Practical Cryptography.

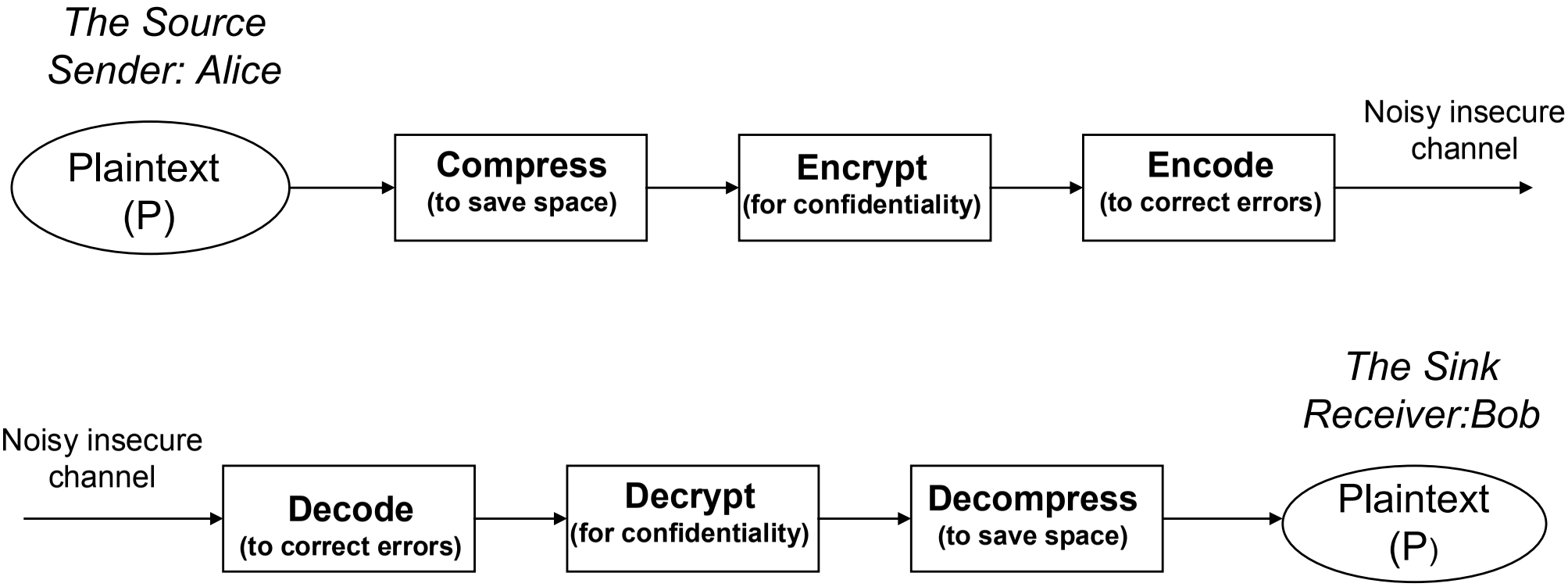
# New Lecture Schedule

| | Date | Topic | Lecturer |
|---|---|---|---|
| 1 | 1/3 | Practical Aspects of Cryptography | Josh |
| 2 | 1/10 | Symmetric Key Ciphers and Hashes | John |
| 3 | 1/17 | Public Key Ciphers | Josh |
| 4 | 1/24 | Cryptographic Protocols I | Brian |
| 5 | 1/31 | Cryptographic Protocols II | Brian |
| 6 | 2/7 | Security of Block Ciphers | John |
| 7 | 2/14 | AES and Cryptographic Hashes | John |
| 8 | 2/21 | Trust, PKI, Key Management [Last HW Assignment) | Brian |
| 9 | 3/1 | Random Numbers/Elliptic Curve Crypto | Josh |
| 10 | 3/8 | Three topics: Elections, ITAR/Politics, Side Channels/Timing Attacks, DRM, BigNum Implementation | All |

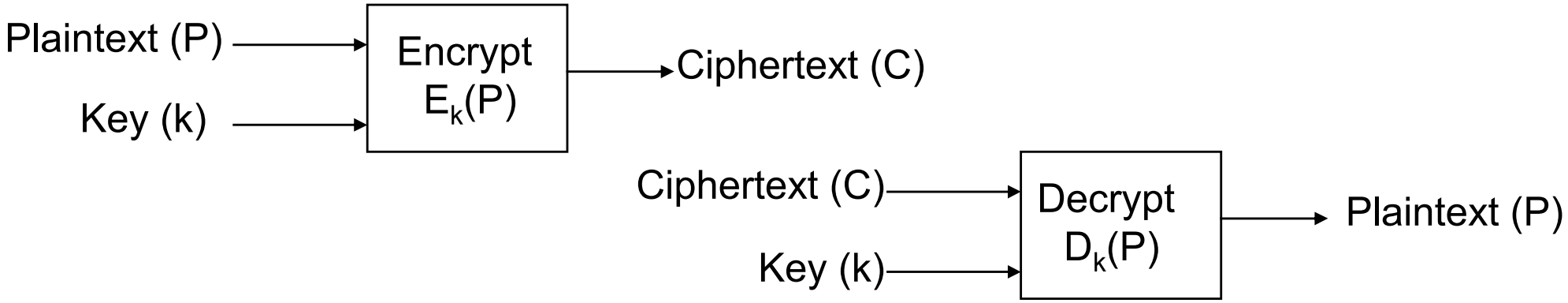
# Symmetric Key Cryptography and Cryptographic Hashes - I

John Manferdelli
jlm@cs.washington.edu
jmanfer@microsoft.com

# Communications Engineers Coat of Arms



# Symmetric Encryption




- Symmetric Key cryptographic algorithms use a secret known to the authorized parties called a “key”. Encryption and Decryption use the same key.
  - The transformations are simple and fast enough for practical use and implementation.
  - “Keyspace” large enough to protect against exhaustive search.
  - The encryption algorithm must be efficiently invertible.
  - Two major types: Stream ciphers and Block ciphers

# Why go Ugly?

| **Algorithm** | **Speed** |
|---|---|
| RSA-1024 Encrypt | .32 ms/op (128B), 384 KB/sec |
| RSA-1024 Decrypt | 10.32 ms/op (128B), 13 KB/sec |
| AES-128 | .53 $\mu$s/op (16B), 30MB/sec |
| RC4 | .016 $\mu$s/op (1B), 63 MB/sec |
| DES | .622 $\mu$s/op (8B), 12.87 MB/sec |

RSA implementation uses CRT, Karasuba and Montgomery.
Timings do not include setup. All results are for an 850MHz x86.

# Ugly II

| Algorithm | Speed |
|---|---|
| SHA-1 | 48.46 MB/sec |
| SHA-256 | 24.75 MB/sec |
| SHA-512 | 8.25 MB/sec |

Timings do not include setup. All results are for an 850MHz x86.

# Ugly III

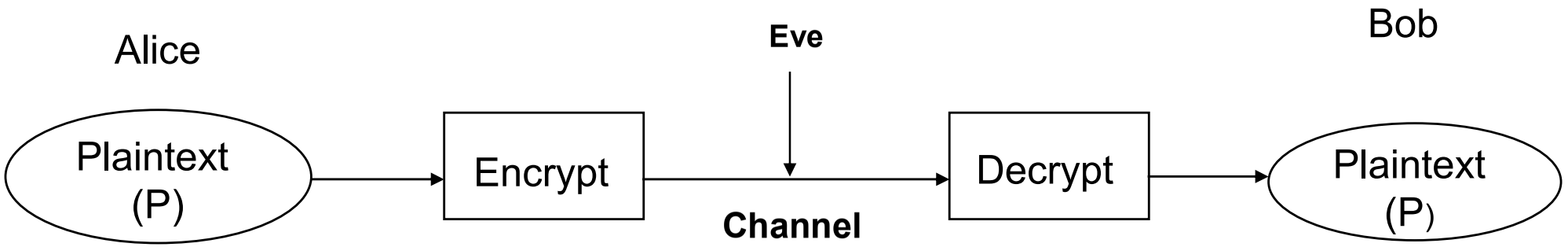
| Symmetric Key Size | RSA/DH Key Size | Elliptic Curve Key Size |
|---|---|---|
| 80 | 1024 | 160 |
| 112 | 2048 | 224 |
| 128 | 3072 | 256 |
| 192 | 8192 | 384 |
| 256 | 15360 | 521 |

# Suite B - Computation Cost

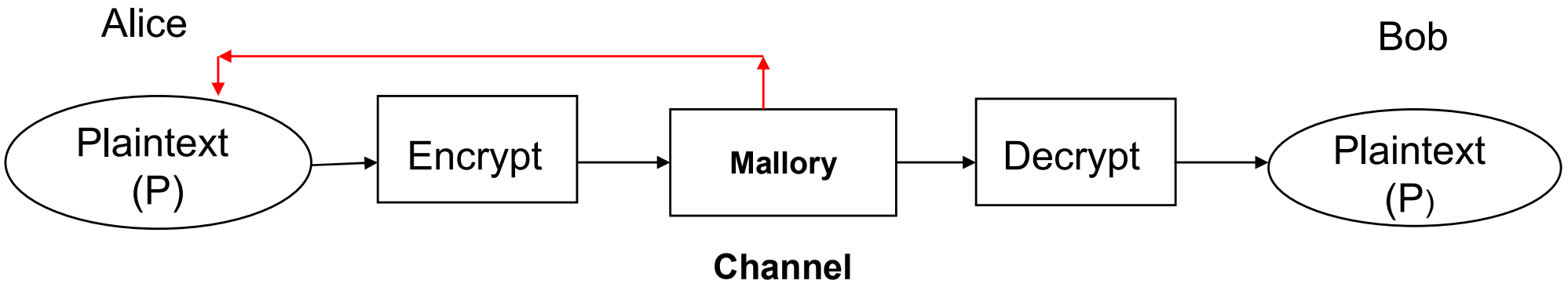
| Symmetric Key Size | Ratio RSA/DH:EC |
|---|---|
| 80 | 3:1 |
| 112 | 6:1 |
| 128 | 10:1 |
| 192 | 32:1 |
| 256 | 64:1 |

# Adversaries and their Discontents

**Wiretap Adversary**




**Man in the Middle Adversary**



# Adversaries

- Cryptography is computing/communicating in the presence of an Adversary
- An Adversary's strength is characterized by:
  - Computational resources available to the adversary:
    - Exponential time/memory
    - Polynomial time/memory
  - Nature of access to cryptographically protected data:
    - Probable plaintext attacks
    - Known plaintext/ciphertext attacks
    - Chosen plaintext attacks
    - Adaptive interactive chosen plaintext attacks (oracle model)
  - Physical Access
    - Outsider threat
    - Insider Threat (Timing, side channels)
    - Trusted Insider Threat (Some key access)

# Cipher Requirements

- WW II
  - Universally available (simple, light instrumentation) - interoperability
  - Compact , rugged: Easy for people to use
  - Security in key only: We assume that the attacker knows the complete details of the cryptographic algorithm and implementation
  - Adversary has access to some corresponding plain and ciphertext
- Now
  - Adversary has access to unlimited ciphertext and lots of chosen text
  - Implementation in digital devices (power/speed) paramount
  - Easy for computers to use
  - Resistant to ridiculous amount of computing power

# Practical Attacks

- Exhaustive Search of Key space
- Exhaustive Search of Keyspace Restricted by poor practice.
  - For example non random key generation
- Exploiting bad key Management/Storage
- Bribing Keyholder
- Side Channel/Timing
- Exploiting encryption Errors
- Spoofing (ATM PIN)
- Leaking due to size, position, language choice, repetition, frequency, intersymbol transitions

# Some Formal  Attack Requirements

- Indistinguishability:
  - Given two messages and the encryption of one of the messages (the target ciphertext), it is hard to determine which message is encrypted
  - Related to Semantic Security
- Non-Malleability：
  - Given a target ciphertext *y*, it is hard to find another ciphertext *y'* such that the corresponding plaintexts are "meaningfully related"

# Security In Practice

- Balances risk of loss, usability and operational efficiency against costs
- Risk assessed against “known” threat and absolute computing model. (e.g.- Linear cryptanalysis in $2^{43}$ steps rather that O(f(n))).
- This requires an intelligent assessment of each problem case and computing/communication environment.
- The greatest threat to “good security” is the insistence on perfect security.

# Block Cipher

- In a block cipher, plaintext is encrypted in blocks of m elements of the alphabet. In the binary block case, m is the block length in bits and $E_k$: P →C where
  - P=C= {0,1,…, $2^m$-1}.
- Since $E_k$ is invertible, $E_k \in S_N$, the permutations on N = $2^m$ elements. Thus $E_k$: k → $S_N$.
  - Think of k as selecting an element from $S_N$ , which is enormous.
- | $S_N$| = N! $\approx \sqrt{(2\ \pi\ N)}\ (N/e)^N$. For example, in DES, the keyspace has N=$2^{56}$ elements, $S_N$ has more than $(2^{64})^N$ elements which is way, way more than the number of elementary particles in the universe

# Block Ciphers: Modes

- ECB : $c_i = E_k(p_i)$
  - Same plaintext gets same ciphertext anywhere in stream
  - Not semantically secure
  - All other modes safer for long communications.
- CBC: $c_0 = IV$, $c_i = E_k(p_i + c_{i-1})$
  - Errors propagate two blocks
  - Can't use parrallel HW or pre-processing, slower
- OFB: $z_0 = IV$, $z_i = E_k(z_{i-1})$, $c_i = p_i + z_i$
  - Limited error propagation
- CFB: $c_0 = IV$, $z_i = E_k(c_{i-1})$, $c_i = p_i + z_i$
  - Error propagates for a few blocks but self-synchronizing
- CTR: $z_i = E_k(\text{nonce} \,||\, i)$, $c_i = p_i + z_i$
  - Never reuse nonce key combination
  - Current favorite: fast, random access, allows preprocessing, similar security to CBC

# Stream Cipher – Definition and Example

- A Stream Cipher takes an arbitrary sequence of elements from the (plaintext) input alphabet to the (ciphertext) output alphabet.
- Example for the binary alphabet:
  - Let $m_t$ represents the bits of the plaintext, t =1,2,…
  - Let $k_t$ be bits selected from a random source, t =1,2,…
  - Let $c_t$ represent the ciphertext bits. $c_t = m_t \oplus k_t$ defines a self-reciprocal (self inverting) stream cipher called a one-time pad.
  - You can't beat one time pads for security (can you spot why they are seldom used?)

# The “Manual” Ciphers

- Simple Transposition: Grilles, columnar transpositions.
- Monoalphabetic Substitution: Caesar, simple substitution
- Polyalphabetic substitution: Vigeniere
- Polygraphic substitution
  - Hill, Playfair

- Additional Reference : Army cryptanalysis manual, http://www.umich.edu/~umich/fm-34-40-2/

# Group Theory in Cryptography - 1

- Groups are sets of elements that have a binary operation with the following properties:
  1. If x,y,z $\epsilon$ G, xy $\epsilon$ G and (xy)z=x(yz). It is not always true xy=yx.
  2. There is an identity element 1 $\epsilon$ G and 1x=x1=x for all x in G
  3. For all, x in G there is an element $x^{-1}$ $\epsilon$ G and x $x^{-1}$ =1= $x^{-1}$ x
- One very important group is the group of all bijective maps from a set of n elements to itself denoted $S_n$ or $\Sigma_n$. These are called the "permutations of n elements." The "binary operation" is the composition of mappings. The identity element leaves every element alone. The inverse of a mapping, x, "undoes" what x does.
- If $\sigma \epsilon S_n$ and the image of x is y we can write this two ways: y= $\sigma$ (x); this is the usual functional notation your used to where mappings are applied "from the left". When mappings are applied from the left and $\sigma$ and $\delta$ are elements of $S_n$ $\sigma\delta$ denotes the mapping obtained by applying $\delta$ first and then $\sigma$ - i.e. y= $\sigma(\delta(x))$.
- Some people apply mappings from the right. They would write y=(x) $\sigma$. For them, $\sigma\delta$ denotes the mapping obtained by applying $\sigma$ first and then $\delta$ - i.e. y= ((x)$\sigma$)$\delta$.

# Group Theory in Cryptography - 2

- The smallest k such that $\sigma^k=1$ is called the order of $\sigma$. G is finite if it has a finite number of elements. In a finite group, all elements have finite order and the order of each element divides the number of elements of G (Lagrange's Theorem).
- Example. Let G= $S_4$.
  - $\sigma$= 1→2, 2→3, 3→4, 4→1, $\delta$= 1→3, 2→4, 3→1, 4→2.
    $\sigma^{-1}$= 1→4, 2→1, 3→2, 4→3
  - Applying mappings "from the left", $\sigma\delta$= 1→4, 2→1,3→2,4→3.
  - Sometimes $\sigma$ is written like this:

    $\sigma$= 1 2 3 4
    2 3 4 1
  - Sometimes permutations are written as products of cycles: $\sigma$=(1234) and $\delta$=(13)(24).

# Transpositions

- Grilles

```
B U L L
W I N K
L E I S   →  BWLAEUINEDLNIOLKSP
A D O P
E
```

$c_i = p_{S(i)}$ where

S=(1)(2,5,17,16,12,11,7,6)(3,9,14,4,13,15,8,10)

# Breaking Completely filled Columnar Transposition

**Message (from Sinkov)**

**EOEYE GTRNP SECEH HETYH SNGND DDDET OCRAE RAEMH**
**TECSE USIAR WKDRI RNYAR ABUEY ICNTT CEIET US**

Procedure

1. Determine rectangle dimensions (l,w) by noting that message length=m = l x w. Here m=77, so l=7, w=11 or l=11, w=7
2. Anagram to obtain relative column positions

Note a transposition is easy to spot since letter frequency is the same as regular English.

# Anagramming

- Look for words, digraphs, etc.
- Note: Everything is very easy in corresponding plain/ciphertext attack

| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|
| E | E | G | A | E | R | C |
| O | C | N | E | U | N | N |
| E | E | D | R | S | Y | T |
| Y | H | D | A | I | A | T |
| E | H | D | E | A | R | C |
| G | E | D | M | R | A | E |
| T | T | E | H | W | N | I |
| R | Y | T | T | K | U | E |
| N | H | O | E | D | E | T |
| P | S | C | C | R | Y | U |
| S | N | R | S | I | I | S |

→

| 3 | 6 | 1 | 5 | 7 | 2 | 4 |
|---|---|---|---|---|---|---|
| G | R | E | E | C | E | A |
| N | N | O | U | N | C | E |
| D | Y | E | S | T | E | R |
| D | A | Y | I | T | H | A |
| D | R | E | A | C | H | E |
| D | A | G | R | E | E | M |
| E | N | T | W | I | T | H |
| T | U | R | K | E | Y | T |
| O | E | N | D | T | H | E |
| C | Y | P | R | U | S | C |
| R | I | S | I | S | N | S |

# Alphabetic Substitution

- A *monoalphabetic* cipher maps each occurrence of a plaintext character to one ciphertext character
- A *polyalphabetic* cipher maps each occurrence of a plaintext character to more than one ciphertext character
- A *polygraphic* cipher maps more than one plaintext character at a time
  - Groups of plaintext characters are replaced by assigned groups of ciphertext characters

# Et Tu Brute?: Substitutions

Caeser Cipher

**B U L L W I N K L E I S A D O P E**

**D W N N Y K P M N G K U C F Q S G**

**c= $pC^k$, C= (ABCDEFGHIJKLMNOPQRSTUVWXYZ), k= 2 here**

**k=3 for classical Caeser**

# Attacks on Monoalphabetic Substitution

- Letter Frequency

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| A | .0651738 | B | .0124248 | C | .0217339 | D | .0349835 |
| E | .1041442 | F | .0197881 | G | .0158610 | H | .0492888 |
| I | .0558094 | J | .0009033 | K | .0050529 | L | .0331490 |
| M | .0202124 | N | .0564513 | O | .0596302 | P | .0137645 |
| Q | .0008606 | R | .0497563 | S | .0515760 | T | .0729357 |
| U | .0225134 | V | .0082903 | W | .0171272 | X | .0013692 |
| Y | .0145984 | Z | .0007836 | sp | .1918182 | | |

- Probable word.
- Corresponding plain/cipher text makes this trivial.

# Polygraphic Frequencies

- Bigraphs

| | | | | |
|---|---|---|---|---|
| EN | RE | ER | NT | TH |
| ON | IN | TE | AN | OR |
| ST | ED | NE | VE | ES |
| ND | TO | SE | AT | TI |

- Trigraphs

| | | | | |
|---|---|---|---|---|
| ENT | ION | AND | ING | IVE |
| TIO | FOR | OUR | THI | ONE |

- Words

| | | | | |
|---|---|---|---|---|
| THE | OF | AND | TO | A |
| IN | THAT | IS | I | IT |
| FOR | AS | WITH | WAS | HIS |
| HE | BE | NOT | BY | BUT |
| HAVE | YOU | WHICH | ARE | ON |

# Letter Frequency Bar Graph

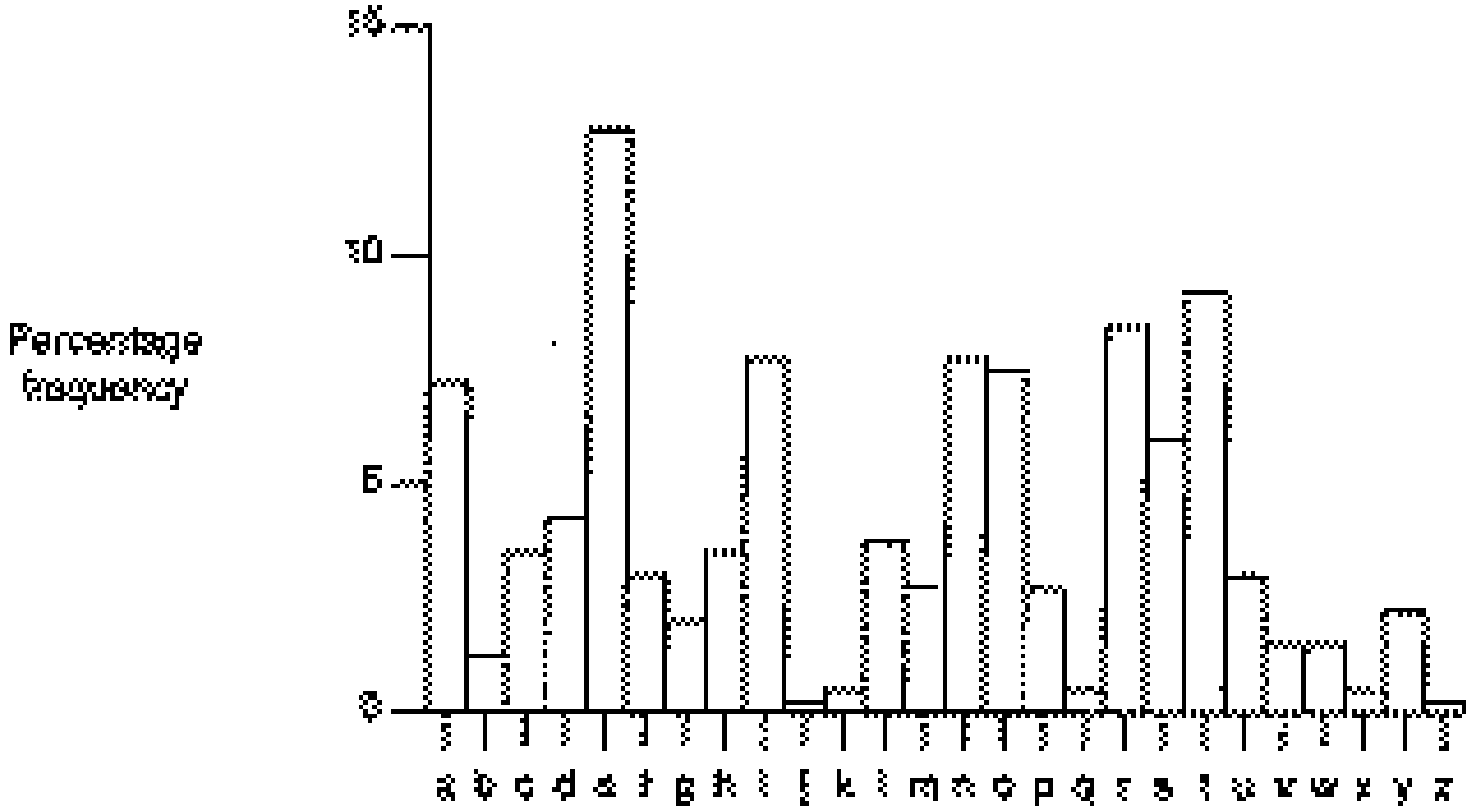
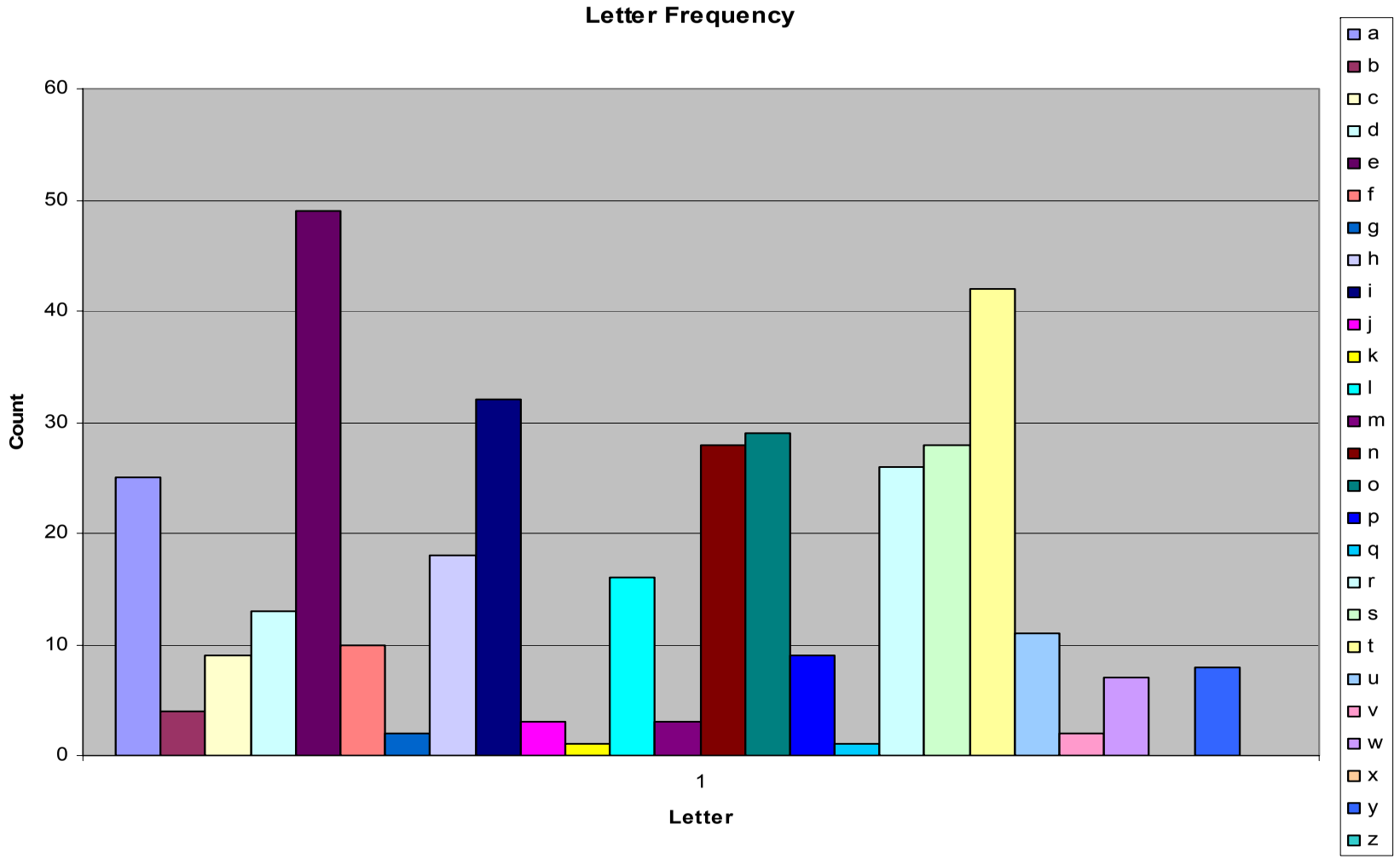



Figure 3.1 English character frequencies

# Letter Frequency Bar Graph



# Shifted Letter Frequency Bar Graph

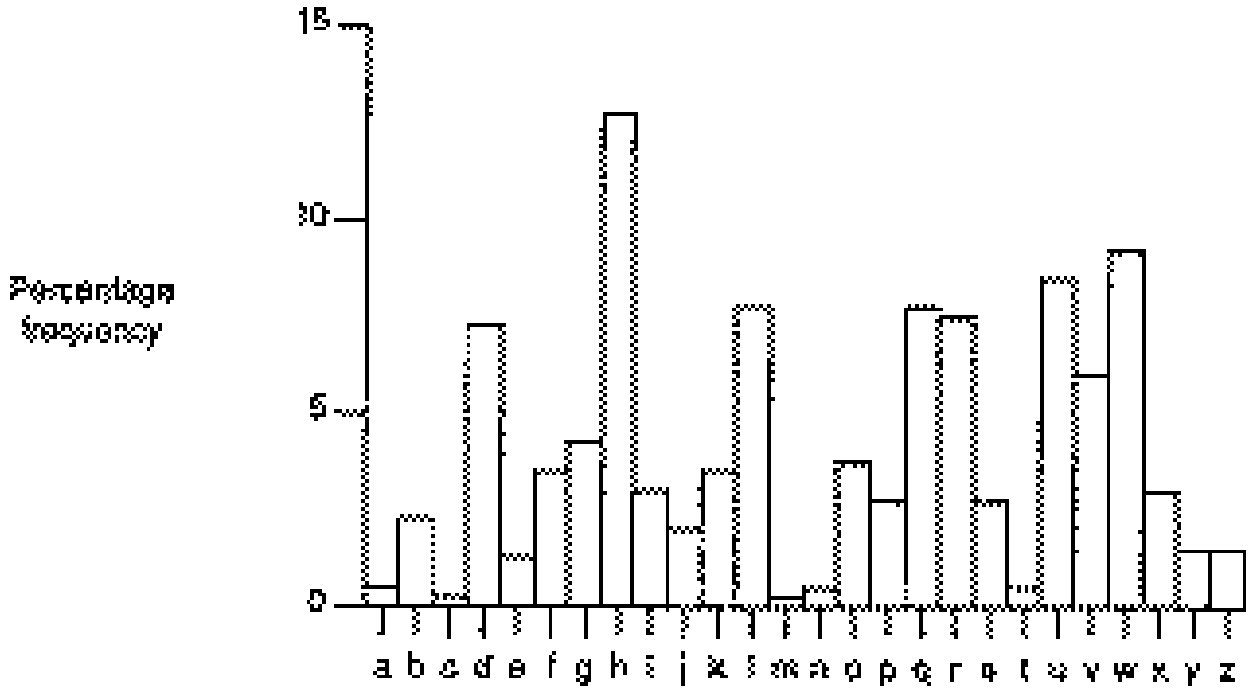



Figure 3.2 Encryption character frequencies with i→i+3

# Vigenere Multialphabetic Cipher

6 Alphabet Direct Standard Example (Keyword: SYMBOL)

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
--------------------------
STUVWXYZABCDEFGHIJKLMNOPQR
YZABCDEFGHIJKLMNOPQRSTUVWX
MNOPQRSTUVWXYZABCDEFGHIJKL
BCDEFGHIJKLMNOPQRSTUVWXYZA
OPQRSTUVWXYZABCDEFGHIJKLMN
LMNOPQRSTUVWXYZABCDEFGHIJK
```

```
PLAIN:   GET OUT NOW
  KEY:   SYM BOL SYM
CIPHER:  YCF PIE FMI
```

# Constructing Vig Alphabets

Direct Standard:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Reverse Standard:

ZYXWVUTSRQPONMLKJIHGFEDCBA

Keyword Direct (Keyword: NEW YORK CITY):

NEWYORKCITABDFGHJLMPQRSUVZ

Keyword Transposed (Keyword: CHICAGO):

CHIAGO

BDEFJK

LMNPQR

STUVWX

YZ

CBLSYHDMTZIENUAFPVGJQWOKRX

# Solving Vigenere

1. Determine Number of Alphabets
   - Repeated runs yield interval differences. Number of alphabets is the gcd of these. (Kasiski)
   - Statistics: Index of coincidence
2. Determine Plaintext Alphabet
3. Determine Ciphertext Alphabets

# Statistical Tests for Alphabet Identification

- Index of coincidence for letter frequency
  - Can choose same letters $f_i$ choose 2 ways

    $IC = \sum_i f_i(f_i-1)/(n(n-1))$, so $IC \approx \sum_i p_i^2$
  - For English Text $IC \approx .07$
  - For Random Text IC= 1/26=.038
  - IC is useful for determining number of alphabets (key length) and aligning alphabets. For n letters enciphered with m alphabets:
  - IC(n,m)= 1/m (n-m)/(n-1) (.07) + (m-1)/m n/(n-1) (.038)

- Other Statistics
  - Vowel Consonant pairing
  - Digraph, trigraph frequency

# Review of Attacks on Polyalphabet

- Letter Frequency, multigram frequencies, transition probabilities
- Index of Coincidence
- Alphabet Chaining
- Sliding Probable Text
- Limited Keyspace search
- Long repeated sequences in ciphertext
- Markoff like contact processes
- Decimation of sequences
- Log weights
- Generatrix
- Direct and indirect symmetries

# Polygraphic Substitution

- PlayFair Digraphic Substitution

```
OHNMA
FERDL
IBCGK          TH→ QM
PQSTU
VWXYZ
```

# Hill Cipher

- The Hill cipher is a block cipher with block size is 2 over the “normal” alphabet.
- Assign each letter a number between 0 and 25 (inclusive)
  - For example, a = 0, b = 1, . . ., z = 25 (z is used as space)
- Let $p_1p_2$ be two successive plaintext letters. $c_1c_2$ are the ciphertext output where

$$\mathbf{c_1 = k_{11}p_1 + k_{12}p_2 \pmod{26}}$$

$$\mathbf{c_2 = k_{21}p_1 + k_{22}p_2 \pmod{26}}$$

- Apply the inverse of the “key matrix” $[k_{11}\ k_{12} \mid k_{21}\ k_{22}]$ to transform ciphertext into plaintext
- Works better if we add space ($27=3^3$ letters) or throw out a letter ($25=5^2$)

# Breaking Hill

- The Hill cipher is resistant to a ciphertext only attack with limited ciphertext.
  - Increasing the block size increases the resistance.
- It is trivial to break using a known plaintext attack.
  - The process is much like the method used to break an affine cipher. Corresponding plaintext/ciphertext are used to set up a system of equations whose solutions are the key bits.

# Information Theory Motivation

- How much information is in a binary string?
- Game: I have a value between 0 and $2^n$-1 (inclusive), find it by asking the minimum number of yes/no questions.
  - Write the number as $[b_{n-1}b_{n-2}\ldots b_0]_2$ .
  - Questions: Is $b_{n-1}$ 1?, Is $b_{n-2}$ 1? , … , Is $b_0$ 1?
- So, what is the amount of information in a number between 0 and $2^n$-1?
  - Answer: n bits
  - The same question: Let X be a probability distribution taking on values between 0 and $2^n$-1 with equal probability. What is the information content of a observation?
  - There is a mathematical function that measures the information in an observation from a probability distribution. It's demoted H(X).
- $H(X)= \sum_i -p_i lg(p_i)$

# Information Theory

- The “definition” of H(X), which is called entropy, has two desireable properties
  - Doubling the storage (the bits your familiar with) doubles the information content
  - H(1/2, 1/3, 1/6)= H(1/2, 1/2) + ½ H(2/3,1/3)
- It was originally developed to study how efficiently one can reliably transmit information over “noisy” channel.
- Applied by Shannon to Cryptography (Communication Theory of Secrecy Systems. BTSJ, 1949)
- Entropy which measures “amount” of uncertainty that is represented in a sample drawn from a stochastic distribution
- Thus information learned about Y by observing X is I(Y,X)= H(Y)-H(Y|X).
- Can be used to estimate requirements for cryptanalysis of a cipher.

# Information Theory in Cryptography

- Studying key search
  - Distribution A: 2 bit key each key equally likely
  - Distribution B: 4 bit key each key equally likely
  - Distribution C: n bit key each key equally likely
  - Distribution A': 2 bit key selected from distribution (1/2, 1/6, 1/6, 1/6)
  - Distribution B': 4 bit key selected from distribution (1/2, 1/30, 1/30, …, 1/30)
  - Distribution C': n bit key selected from distribution (1/2, ½ $1/(2^n-1)$,…, ½ $1/(2^n-1)$)

# Information Theory in Cryptography

- How much information is there in a key drawn from a random variable X
  - Distribution A: $H(X)= ¼ \lg(4) + ¼ \lg(4) + ¼ \lg(4) +1/4 \lg(4) = 2$ bits
  - Distribution B: $H(X)= 16 \times (1/16 \lg(16))= 4$ bits
  - Distribution C: $H(X)= 2^n \times (1/2^n) \lg(2^n) = n$ bits
  - Distribution A': $H(X) = ½ \lg(2) + 3 \times(1/6 \lg(6))= 1.79$ bits
  - Distribution B': $H(X) = ½ \lg(2) + 15 \times(1/30 \lg(30))= 2.95$ bits
  - Distribution C': $H(X) = ½ \lg(2) + 1/2\ 2^n\text{-}1 \times(1/(2^n\text{-}1) \lg(2^n\text{-}1)) \approx n/2+1$ bits

# Information Theory in Cryptography

- What is the expected number of keys that must be tried to determine a key drawn from a random variable X
  - Distribution A: E(X)= ¼ (1+2+3) = 1.5 trials
  - Distribution B: E(X)= 1/16 (1+2+…+15)=7.5 trials
  - Distribution C: E(X)= $1/2^n$ (1+2+ … +$2^n$) = ($2^n$-1)/2 trials
  - Distribution A': E(X) = ½ x 1 + 1/6(2+3)= 4/3 trials
  - Distribution B': E(X) = ½ X1 + 1/30 (2+3+…+15)= 125/12 trials
  - Distribution C': E(X) = ½ X1 + ½ 1/($2^n$-1)(2+3+…+$2^n$-1)=

    ½ + ¼ 1/($2^n$-1) ($2^n$-1)($2^n$+1)trials

# Equivocation and Bayes Theorem

- $H_E = \text{Lim}_{n\to\infty} \Sigma_{(x[1],\ldots,x[n])} (1/n) Pr(X=(x[1],\ldots,x[n])) \lg(Pr(X=(x[1],\ldots,x[n])))$
- For random stream of letters
  - $H_R = \Sigma_i (1/26) \lg(26) = 4.7004$
- For English
  - $H_{English}$ = 1.2-1.5 (so English is about 75% redundant)
  - There are approximately $T(n) = 2^{nH}$ n symbol messages that can be drawn from the meaningful English sample space.
- H(X,Y)= H(Y)+H(X|Y)
- Bayes: P(x|y) P(y)= P(x)P(y|x)
- X and Y are independent if P(X,Y)= P(X)P(Y)

# Some Information Theory Theorems

- $H(K|C) = H(M|C) + H(K|M,C)$

  Proof:

  $H(K|C) = H(K,C) - H(C)$, $H(K|M,C) = H(M, K, C) - H(M,C)$.

  $H(M|C) = H(M,C) - H(C)$. Thus, $H(K|C) = H(K,C) - H(M,C) + H(M|C)$.

  $H(M|K,C) = H(M,K,C) - H(K,C)$, but $H(M|K,C) = 0$ since there is no uncertainty in the message given the ciphertext and the key. $H(K|M,C) = H(M,K,C) - H(M,C)$. So $H(K|M,C) = H(K,C) - H(M,C)$. Thus $H(K|C) = H(K|M,C) + H(M|C)$.

- Perfect Security: $P(C|M) = P(C)$

# Unicity and Random Ciphers

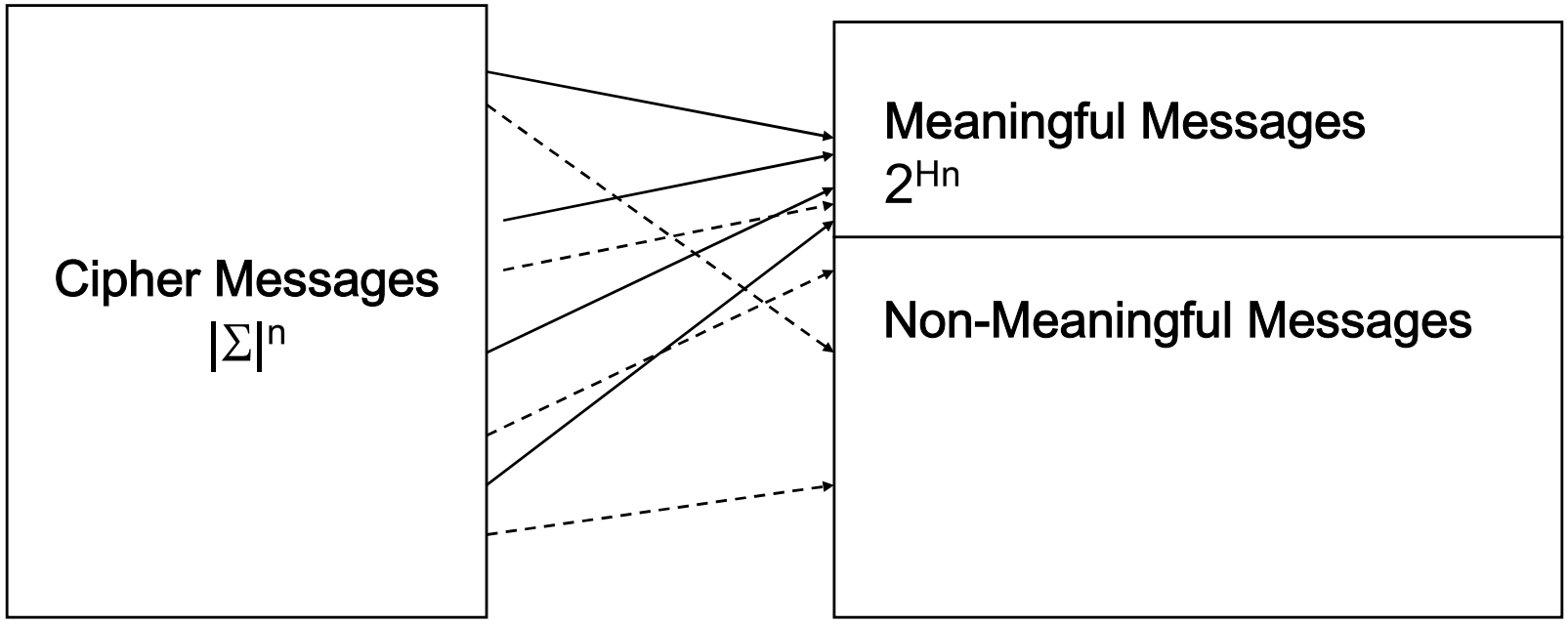
Question: How many messages do I need to trial decode so that the expected number of false keys for which all m messages land in the meaningless subset is less than 1?

Answer: The unicity point.

Nice application of Information Theory.

Theorem: Let H be the entropy of the source (say English) and let $\Sigma$ be the alphabet. Let K be the set of (equiprobable) keys. Then $u = \lg(|K|)/(\lg(|\Sigma|)-H)$.

# Unicity for Random Ciphers




Decoding with correct key

Decoding with incorrect key

# Unicity Distance for Monoalphabet

$H_{CaeserKey} = H_{random} = \lg(26) = 4.7004$

$H_{English} \approx 1.2$.

For Caeser, $u \approx \lg(26)/(4.7-1.2) \approx 4$ symbols, for ciphertext only attack. For known plaintext/ciphertext, only 1 corresponding plain/cipher symbol is required for unique decode.

For arbitrary substitution, $u \approx \lg(26!)/(4.7-1.2) \approx 25$ symbols for ciphertext only attack. For corresponding plain/ciphertext attack, about 8-10 symbols are required.

Both estimates are remarkably close to actual experience.

# Application: One Time Pads are "unbreakable"

M, C, K are n bits long

$P(M=x|C=y)=P(M=x \text{ and } C=y)/P(C=y)$.

$P(M=x \text{ and } C=y) = P(M=x \text{ and } K=x \oplus y)=P(M=x)P(K=x \oplus y)=P(M=x)2^{-n}$

$P(C=y)= \Sigma_x P(M=x \text{ and } C=y)= \Sigma_x P(M=x)2^{-n}=2^{-n}$

So $P(M=x|C=y)= P(M=x)\ 2^{-n}/2^{-n}=P(M=x)$

# Information Theoretic Estimates to break monoalphabet

| Cipher | Type of Attack | Information Resources | Computational Resources |
|---|---|---|---|
| **Caeser** | Ciphertext only | U= 4.7/1.2=4 letters | 26 computations |
| **Caeser** | Known plaintext | 1 corresponding plain/cipher pair | 1 |
| **Substitution** | Ciphertext only | ~30 letters | O(1) |
| **Substitution** | Known plaintext | ~10 letters | O(1) |

# Mixing cryptographic elements to produce strong cipher

- Diffusion – transposition
  - Using group theory, the action of a transposition $\tau$ on $a_1\ a_2 \ldots a_k$ could be written as $a_{\tau(1)}\ a_{\tau(2)} \ldots a_{\tau(k)}$ .
- Confusion – substitution
  - Using group theory, the action of a substitution $\sigma$ on $a_1\ a_2 \ldots a_k$ could be written as $\sigma(a_1)\ \sigma(a_2) \ldots \sigma(a_k)$ .
- Transpositions and substitutions may depend on keys. Keyed permutations may be written as $\sigma_k(x)$. Incidentially, a block cipher on b bits is nothing more than a keyed permutation on $2^b$ symbols.

- Iterative Ciphers – key dependant staged iteration of combination of basic elements is very effective way to construct cipher. (DES, AES)

# The “Machine” Ciphers

- Simple Manual Wheels
  - Wheatstone
  - Jefferson
- Rotor
  - Enigma
  - Heburn
  - SIGABA
  - TYPEX
- Stepping switches
  - Purple
- Mechanical Lug and cage
  - M209

# Jefferson Cipher

# Enigma

# Group Theory for Rotors

- Theorem: If $\sigma = (a_{11}\ a_{12}\ \ldots\ a_{1i})\ (a_{11}\ \ldots\ a_{1j})\ \ldots\ (a_{11}\ \ldots\ a_{1k})$ then $\delta\sigma\delta^{-1} = (\delta a_{11}\ \delta a_{12}\ \ldots\ \delta a_{1i})\ (\delta a_{11}\ \ldots\ \delta a_{1j})\ \ldots\ (\delta a_{11}\ \ldots\ \delta a_{1k})$.
- When permutations are written as products of cycles, it is very easy to calculate their order. (It is the LCM of the length of the cycles).
- Writing cryptographic processes as group operation can be very useful. For example, if R denotes the mapping of a "rotor" and C=(1,2,…,26), the mapping of the rotor "turned" one position is $CRC^{-1}$.

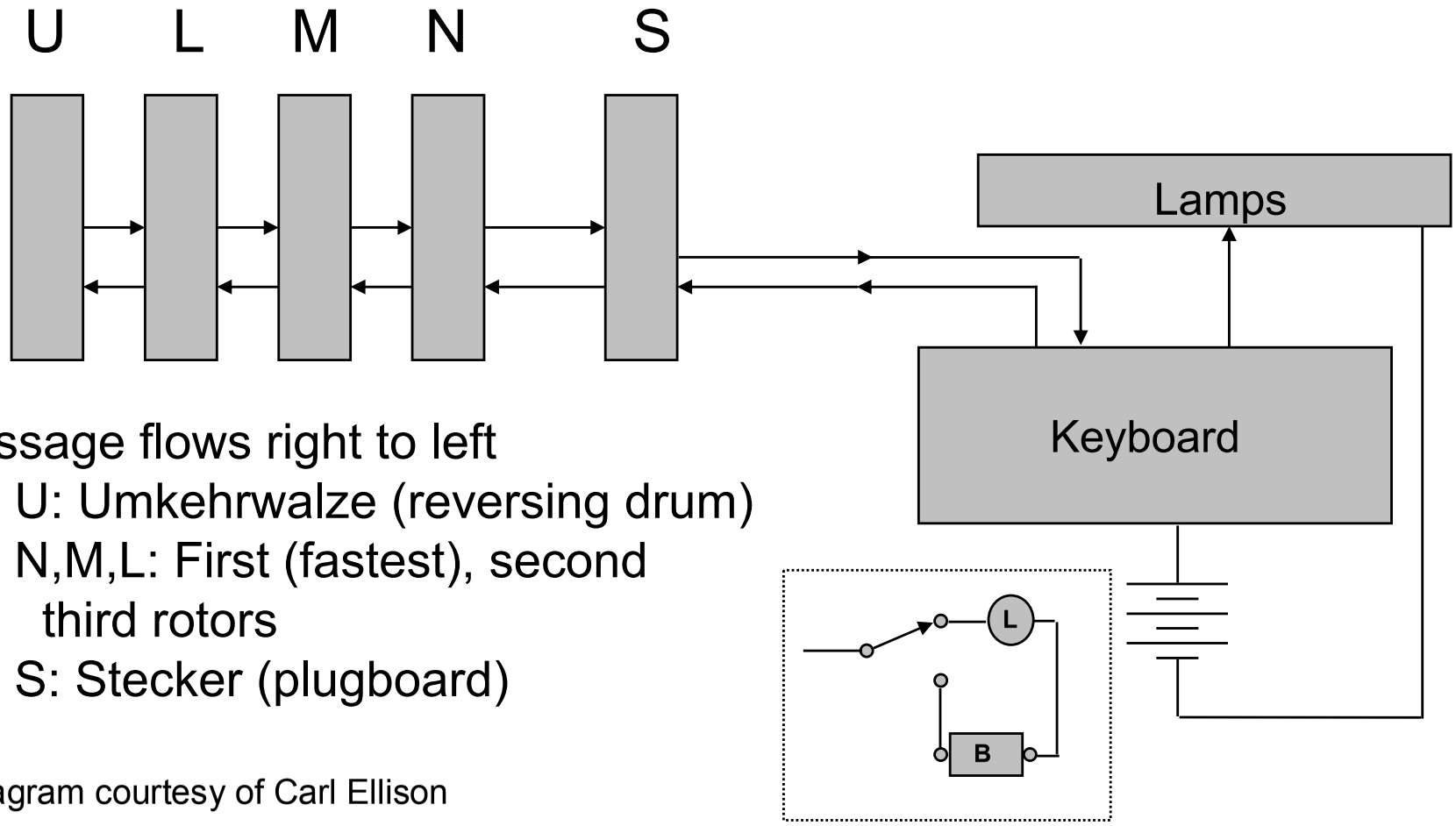
Reference:


Rotman, Group Theory.

Lang, Algebra.

Hall, Group Theory. Chelsea.

# Diagrammatic Enigma Structure




Message flows right to left

- U: Umkehrwalze (reversing drum)
- N,M,L: First (fastest), second third rotors
- S: Stecker (plugboard)

Diagram courtesy of Carl Ellison

# Military Enigma

Encryption Equation

$$c = (p)\ P^{i}NP^{-i}\ P^{j}MP^{-j}\ P^{k}LP^{-k}\ U\ P^{k}L^{-1}P^{-k}\ P^{j}M^{-1}P^{-j}\ P^{i}N^{-1}P^{-i}$$

- K: Keyboard
- P=(ABCDEFGHIJKLMNOPQRSTUVWXYZ)
- N: First Rotor
- M: Second Rotor
- L: Third Rotor
- U: Reflector. Note: $U=U^{-1}$.
- i,j,k: Number of rotations of first, second and third rotors respectively.

- Later military models added plug-board (S) and additional rotor (not included). The equation with Plugboard is:

$$c = (p)S\ P^{i}NP^{-i}\ P^{j}MP^{-j}\ P^{k}LP^{-k}\ U\ P^{k}L^{-1}P^{-k}\ P^{j}M^{-1}P^{-j}\ P^{i}N^{-1}P^{-i}\ S^{-1}$$

# Enigma Data

**Rotors**

| | |
|---|---|
| Input | ABCDEFGHIJKLMNOPQRSTUVWXYZ |
| Rotor I | EKMFLGDQVZNTOWYHXUSPAIBRCJ |
| Rotor II | AJDKSIRUXBLHWTMCQGZNPYFVOE |
| Rotor III | BDFHJLCPRTXVZNYEIWGAKMUSQO |
| Rotor IV | ESOVPZJAYQUIRHXLNFTGKDCMWB |
| Rotor V | VZBRGITYUPSDNHLXAWMJQOFECK |
| Rotor VI | JPGVOUMFYQBENHZRDKASXLICTW |
| Rotor VII | NZJHGRCXMYSWBOUFAIVLPEKQDT |

**Ring Turnover**

| | |
|---|---|
| Rotor I | R |
| Rotor II | F |
| Rotor III | W |
| Rotor IV | K |
| Rotor V | A |
| Rotors VI | A/N |

| | |
|---|---|
| **Reflector B** | (AY) (BR) (CU) (DH) (EQ) (FS) (GL) (IP) (JX) (KN) (MO) (TZ) (VW) |
| **Reflector C** | (AF) (BV) (CP) (DJ) (EI) (GO) (HY) (KR) (LZ) (MX) (NW) (TQ) (SU) |

# Military Enigma Key Length

- Key Length (rotor order, rotor positions, plugboard)
  - 60 rotor orders. lg(60)= 5.9 bits.
  - 26*26*26 = 17576 initial rotor positions. lg(17576)= 14.1 bits of key
  - 10 exchanging steckers were specified yielding C(26,2) C(24,2)…C(8,2)/10! = 150,738,274,937,250. lg(150,738,274,937,250)= 47.1 bits as used
  - Bits of key: 5.9 + 14.1 + 47.1 = 67.1 bits
  - Note: plugboard triples entropy of key!
- Rotor Wiring State
  - lg(26!) = 88.4 bits/rotor.
- Total Key including rotor wiring:
  - 67.1 bits + 3 x 88.4 bits = 312.3 bits

# Method of Batons

- Applies to Enigma
  - Without plugboard
  - With fast rotor ordering known and only the fast rotor moving
  - With a “crib”
- Let N be the fast rotor and Z the combined effect of the other apparatus, then $N^{-1}ZN(p)=c$.
- Since $ZN(p)=N(c)$, we know the wiring of N and a crib, we can play the crib against each of the 26 possible positions of N for the plaintext and the ciphertext. In the correct position, there will be no “scritches” or contradictions in repeated letters.
- This method was used to “analyze” the early Enigma variants used in the Spanish Civil War and is the reason the Germans added the plugboard.
- Countermeasure: Move fast rotor next to reflector.

# German Key Management before 5/40

- Daily keys were distributed on paper monthly and distributed by courier. Each daily key consisted of a line specifying:
  - (date, rotor order, ring settings, plug settings -10)
- For each message
  1. Operator chooses a 3-letter sequences, the “indicator” and the “text”
  2. Operator set rotor positions to indicator and encrypted text *twice*.
  3. Machine rotor positions were reset to “text” positions and the message encrypted.
  4. Operator prepended indicator and twice encrypted rotor order text to transmitted message.
- This was done to avoid starting rotors in the same position for each message without enormous key lists.
  - Good motivation. Bad idea.

# Changes German use of Enigma

1. Plugboard added– 6/30
2. Key setting method – 1/38
3. Rotors IV and V – 12/38
4. More plugs - 1/39
5. End of message key pair encipherment – 5/40

# Polish (Rejewski) Attack

- Rejewski exploited weakness in German keying procedure to determine rotor wiring
  - Rejewski had ciphertext for several months but no German Enigma.
  - Rejewski had Stecker settings for 2 months (from a German spy via the French in 12/32), leaving 265.2 bits of key (the wirings) to be found. He did.
- Poles determine the daily keys
  - Rejewski catalogued the characteristics of rotor settings to detect daily settings. He did this with two connected Enigmas offset by 3 positions (the "cyclotometer").
  - In 9/38, when the "message key" was no longer selected from standard setting (the Enigma operator to choose a different encipherment start called the indicator), Rejewski's characteristics stopped working.
  - Zygalski developed a new characteristic and computation device ("Zygalski sheets") to catalog characteristics which appeared when $1^{st}/4^{th}$, $2^{nd}/5^{th}$,$3^{rd}/6^{th}$ ciphertext letters in encrypted message keys ("Females") were the same.

# How Rejewski did it

Recall $_1$ $c=(p)S\ P^iNP^{-i}\ P^jMP^{-j}\ P^kLP^{-k}\ U\ P^kL^{-1}P^{-k}\ P^jM^{-1}P^{-j}\ P^iN^{-1}P^{-I}\ S^-$

Let $Q=\ MLUL^{-1}M^{-1}=Q^{-1}$

Then the first 6 permutations (used to encrypt settings twice) are:

- $A=A^{-1}=\ SP^1NP^{-1}QP^1N^{-1}P^{-1}S^{-1}$, $B=B^{-1}=\ SP^2NP^{-2}QP^2N^{-1}P^{-2}S^{-1}$
- $C=C^{-1}=\ SP^3NP^{-3}QP^3N^{-1}P^{-3}S^{-1}$, $D=D^{-1}=\ SP^4NP^{-4}QP^4N^{-1}P^{-4}S^{-1}$
- $E=E^{-1}=\ SP^5NP^{-5}QP^5N^{-1}P^{-5}S^{-1}$, $F=F^{-1}=\ SP^6NP^{-6}QP^6N^{-1}P^{-6}S^{-1}$

Their products and what they reveal about encrypted settings ($c_1c_2c_3c_4c_5c_6$):

- $AD=\ SP^1NP^{-1}QP^1N^{-1}P^3NP^{-4}QP^4N^{-1}P^{-4}S^{-1}$. $(c_1)AD=\ c_4$.
- $BE=\ SP^2NP^{-2}QP^2N^{-1}P^3NP^{-5}QP^5N^{-1}P^{-5}S^{-1}$. $(c_2)BE=\ c_5$.
- $CF=\ SP^3NP^{-3}QP^3N^{-1}P^3NP^{-6}QP^6N^{-1}P^{-6}S^{-1}$. $(c_3)CF=\ c_6$.

So we can find $AD$, $BE$ and $CF$ after about 80 messages.

# How Rejewski did it (continued)

We don't know S, N or Q. Suppose

- `AD= (dvpfkxgzyo)(eijmunqlht)(bc)(rw)(a)(s)`
- `BE= (blfqveoum)(hjpswizrn)(axt)(cgy)(d)(k)`
- `CF= (abviktjgfcqny)(duzrehlxwpsmo)`

Now the following two simple Group Theory Theorems help:

1. The product of two permutations of the same degree consisting of products of disjoint transpositions has an even number of cycles of the same length and the each element of a transposition ends up in different cycles of the same length.
2. Two permutations in $S_n$ are conjugate ($\sigma=\rho^{-1}\delta\rho$) iff they have the same cycle structure.

# How Rejewski did it (conclusion)

- Using the first theorem, the expression for CF gives 13 possibilities for C and F, 27(=3x9) possibilities for B and E and 20(=2x10) possibilities for A and D.
- Histograms of the enciphered message keys showed spikes. Rejewski deduced, correctly, that these were often for plaintext keys AAA, BBB, CCC, etc. --- that allowed him to line up cycles between the three Enigma pairs.
- This determines A,B,C,D,E and F.
- Rejewski was given 2 months of key sheets, carrying the Stecker settings. From them he removed the Stecker and solved the equations for the fast rotor. For example:
  - $S^{-1}AS = P^{1}NP^{-1}QP^{1}N^{-1}P^{-1}$ with $S^{-1}AS$ and P, known.
  - This leaves only two permutations Q and N (the fast rotor) unknown and Q is in precisely the right form to apply the second group theory result.
  - Thus we can obtain both Q and N (see postscript).
- Since all the rotors are in the fast position at some time, we (as Rejewski) can obtain all the wirings.

# How Rejewski did it (postscript)

$A = SP^{1}NP^{-1}QP^{1}N^{-1}P^{-1}S^{-1} \rightarrow A' = S^{-1}P^{-1}AP^{1}S = NP^{-1}QP^{1}N^{-1}$

$B = SP^{2}NP^{-2}QP^{2}N^{-1}P^{-2}S^{-1} \rightarrow B' = S^{-1}P^{-2}BP^{2}S = NP^{-2}QP^{2}N^{-1}$

So,

$A'B' = NP^{-1}QP^{1}QP^{2}N^{-1}$

Similarly,

$C'D' = NP^{-2}QP^{1}QP^{3}N^{-1}$

So

$C'D' = NP^{-1}N^{-1}\ A'B'\ NPN^{-1}$

# Digital Block Ciphers

Complicated keyed invertible functions constructed from iterated elementary rounds.

- Confusion: non-linear functions (ROM lookup)
- Diffusion: permute round output bits
- Key mixing: xor “key schedule” at beginning of round

***Characteristics:***

- *Fast*
- *Data encrypted in fixed “block sizes” (64,128,256 bit blocks are common).*
- *Key and message bits non-linearly mixed in ciphertext*

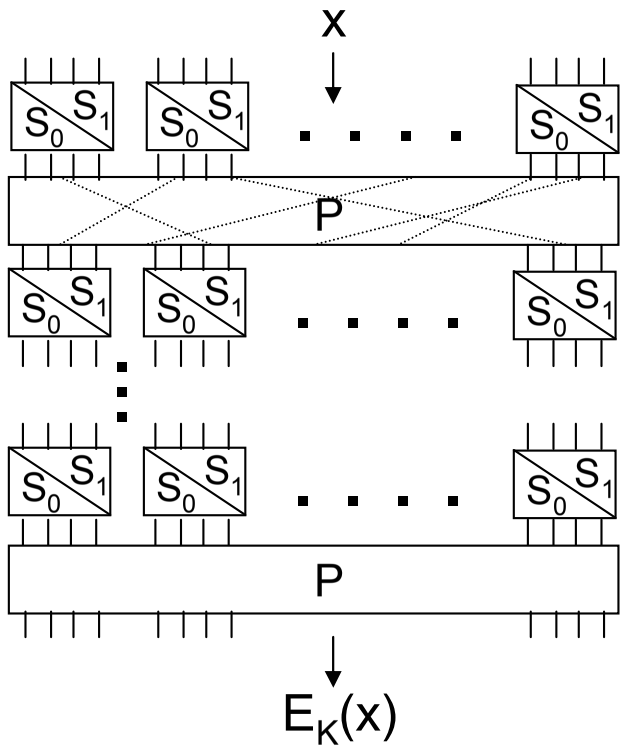
DES (1974) design was watershed in public symmetric key crypto.

# Data Encryption Standard

- Federal History
  - 1972 study
  - RFP: 5/73, 8/74
  - NSA: S-Box influence, key size reduction
  - Published in Federal Register: 3/75
  - FIPS 46:  January, 1976.
- *IBM*
  - Descendant of Feistel's Lucifer
  - Designers: Horst Feistel, Walter Tuchman, Don Coppersmith, Alan Konheim, Carl Meyer, Mike Matyas, Roy Adler, Edna Grossman, Bill Notz, Lynn Smith, and Bryant Tuckerman
- Brute Force Cracking
  - EFS DES Cracker: $250K, 1998. 1,536 custom chips. Can brute force a DES key in days
  - Deep Crack and distributed.net break a DES key in 22.25 hours.

# Horst Feistel: Lucifer

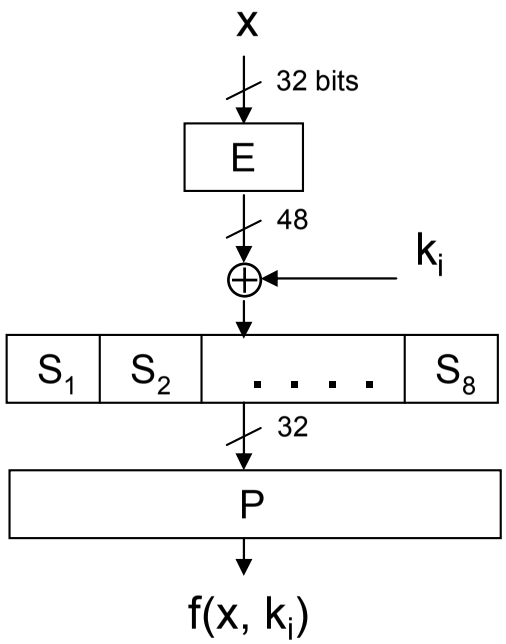
- Early 1970s: First serious needs for civilian encryption (in electronic banking)
- IBM's response: Lucifer, an iterated SP cipher
- Lucifer (v0):
  - Two fixed, 4x4 s-boxes, $S_0$ & $S_1$
  - A fixed permutation P
  - Key bits determine which s-box is to be used at each position
  - 8 x 64/4 = 128 key bits (for 64-bit block, 8 rounds)



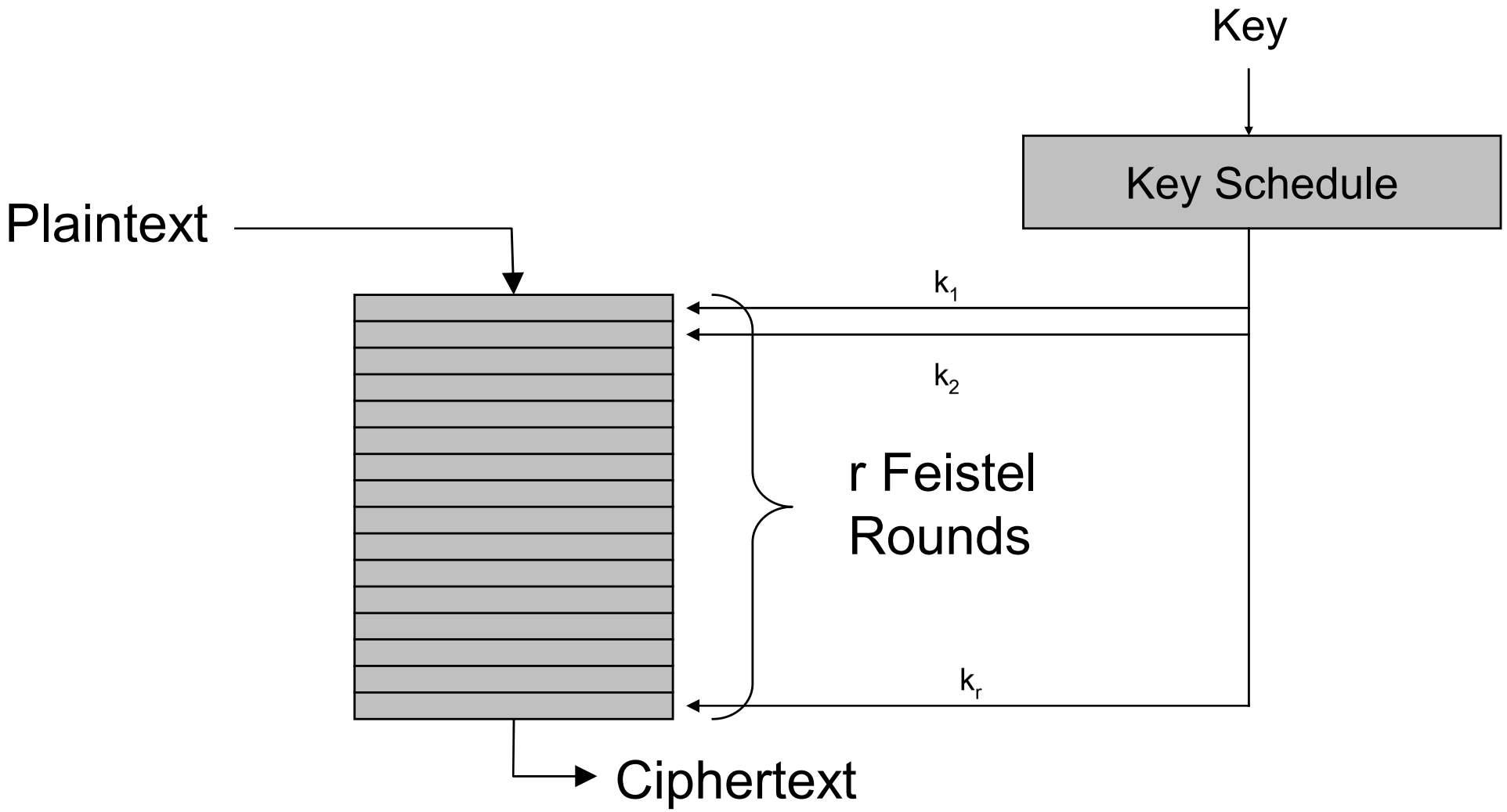

Graphic by cschen@cc.nctu.edu.tw

# From Lucifer to DES

- 8 fixed, 6x4 s-boxes (non-invertible)
- expansion E  (simple duplication of 16 bits)
- round keys are used only for xor with the input
- 56-bit key size
- 16 x 48 round key bits are selected from the  56-bit master key by the “key schedule”.

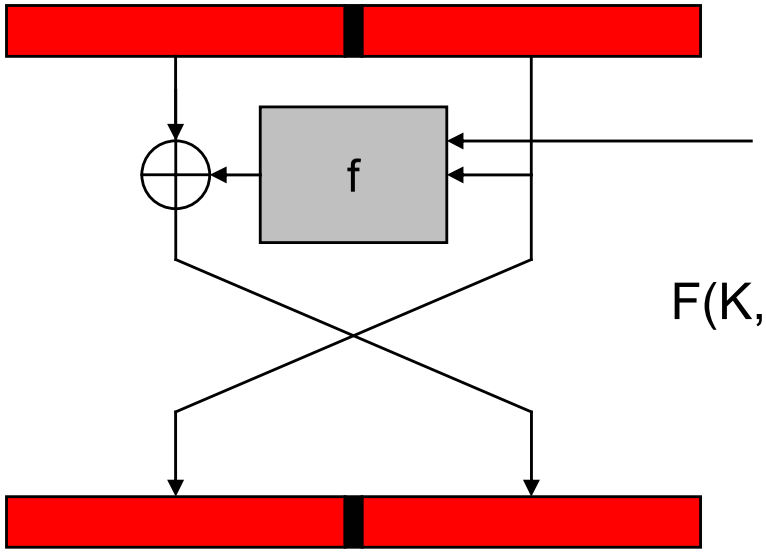


# Iterated Feistel Cipher



# Feistel Round

Graphic courtesy of Josh Benaloh

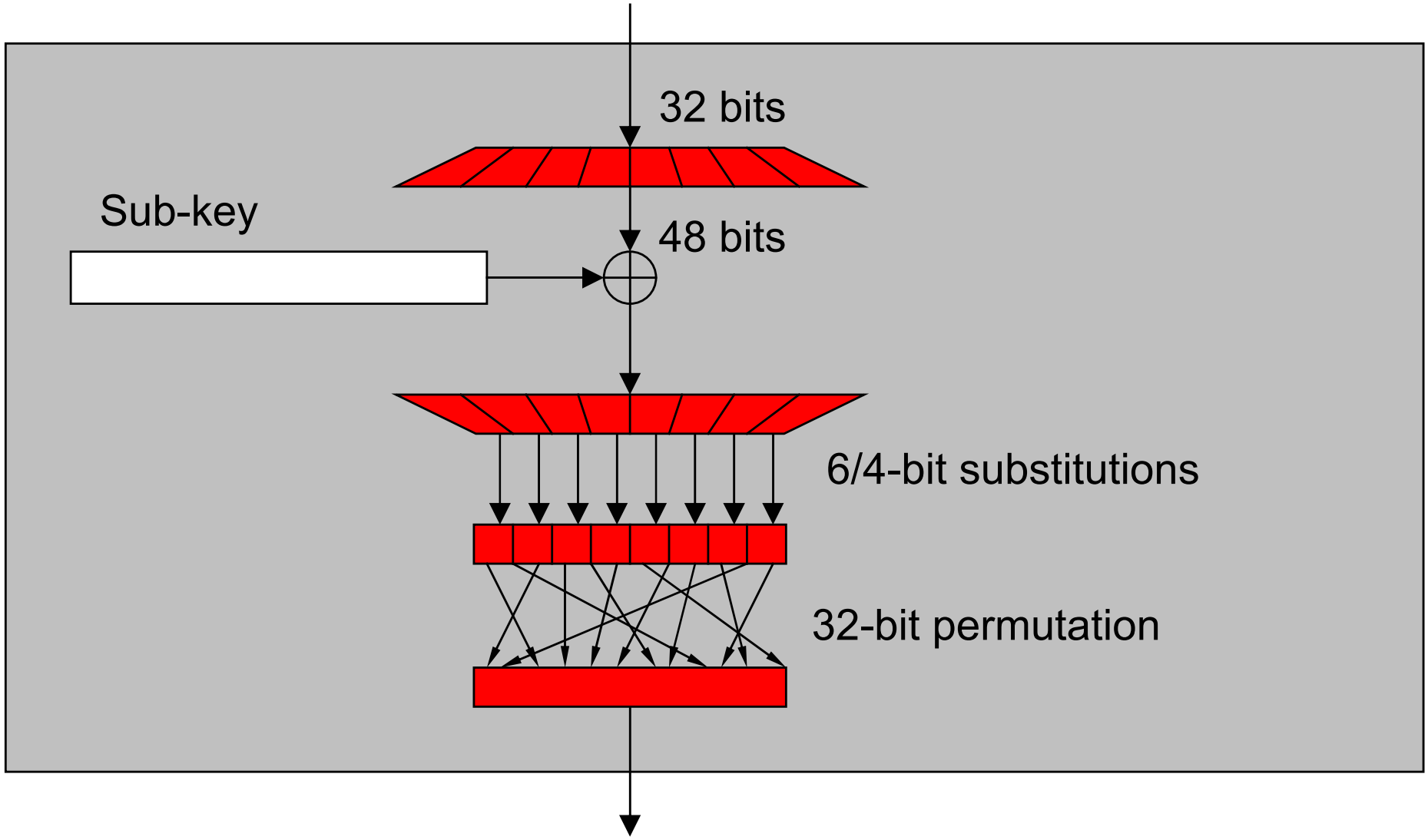



F(K,X)= non-linear function

Note: If $\sigma_i$(L,R)= (L$\oplus$f(E(R) $\oplus k_i$) ,R) and $\tau$(L,R)= (R,L), this round is $\tau\sigma_i$(L,R).

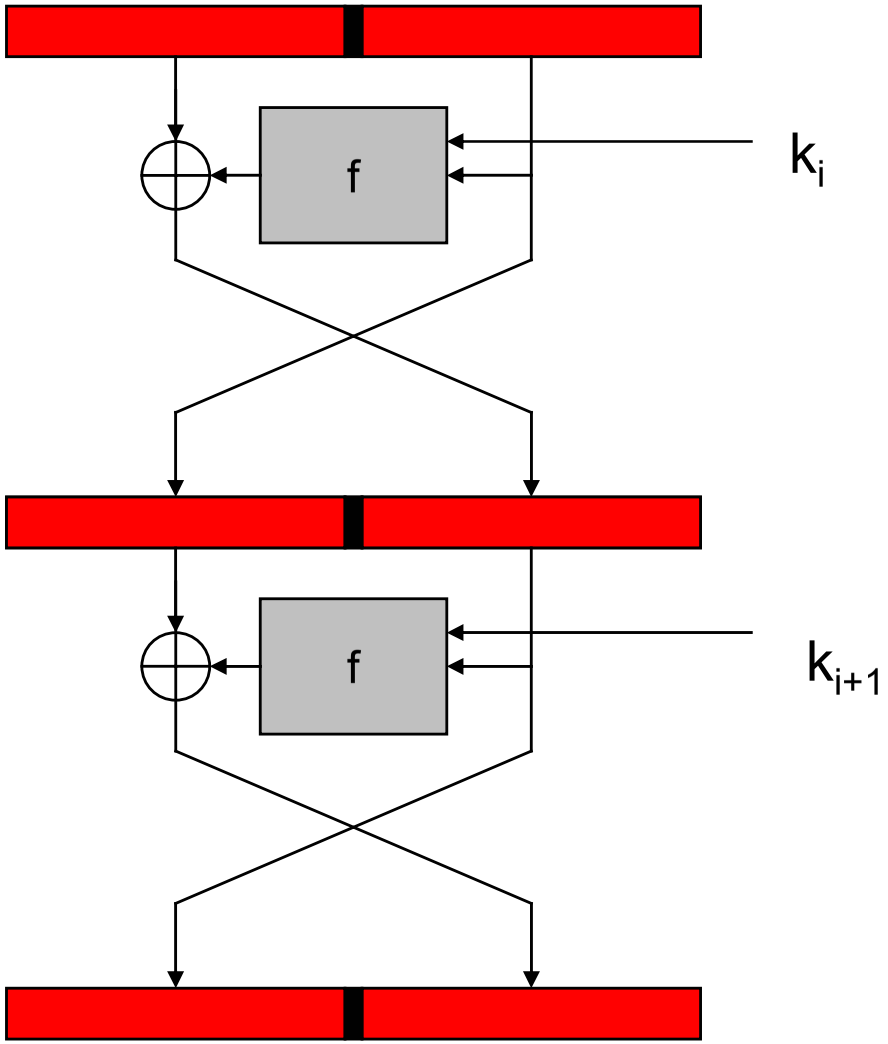
To invert: swap halves and apply same transform with same key: $\sigma_i\tau\tau\sigma_i$(L,R)=(L,R).

# DES Round Function




Slide courtesy of Josh Benaloh

# Chaining Feistel Rounds



# Feistel Ciphers defeat simple attacks

- After 2 to 4 rounds to get flat statistics.
- Parallel system attack
  - Solve for key bits or constrain key bits

    $k_{i(1)} = a_{11}(K)p_1\, c_1 + a_{12}(K)p_2\, c_1 + \ldots + a_{1N}(K)p_n c_n$

    … … … …

    $k_{i(m)} = a_{m1}(K)p_1\, c_1 + a_{m2}(K)p_2\, c_1 + \ldots + a_{mN}(K)p_n c_n$
- Solving Linear equations for coefficients determining cipher

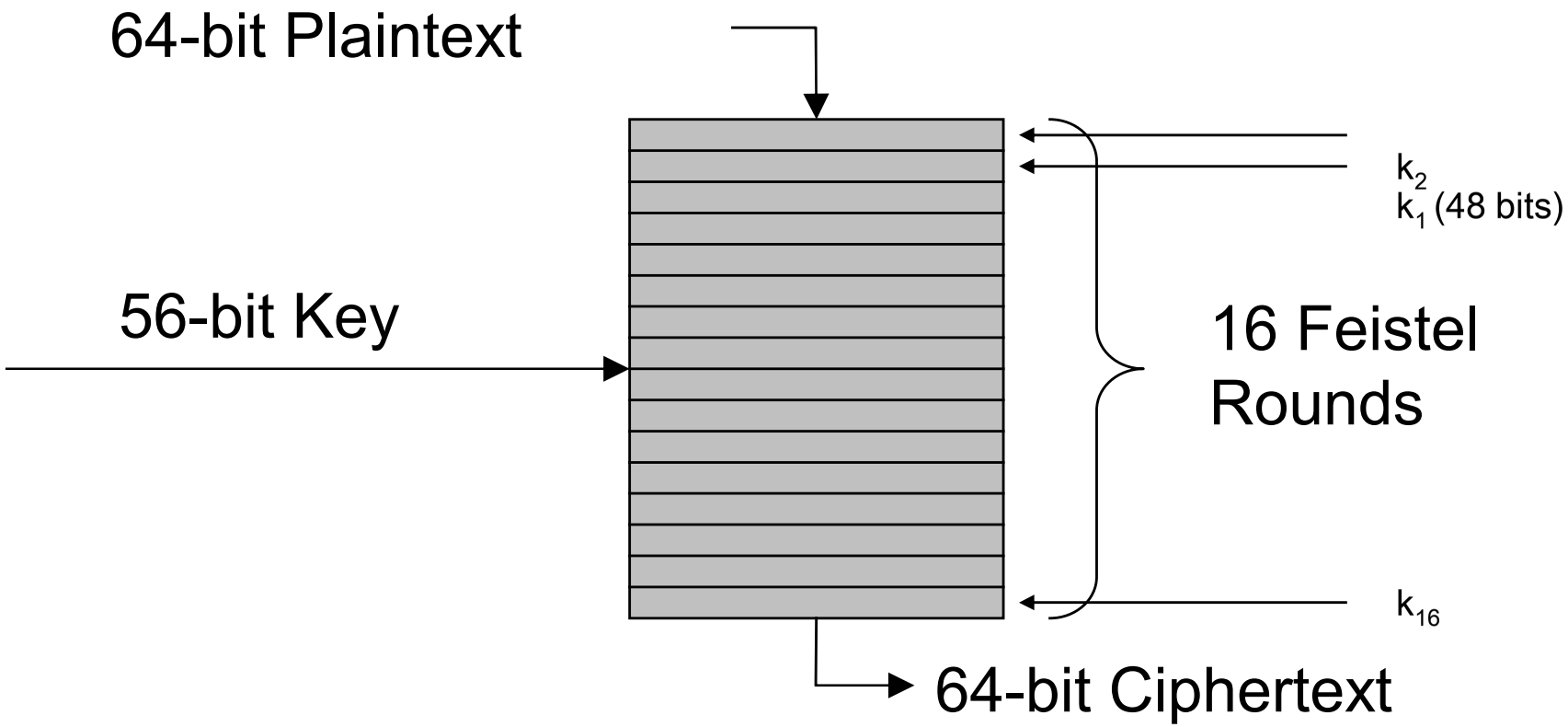
  $c_1 = f_{11}(K)p_1 + f_{12}(K)p_2 + \ldots + f_{1n}(K)p_n$

  $c_2 = f_{21}(K)p_1 + f_{22}(K)p_2 + \ldots + f_{2n}(K)p_n$

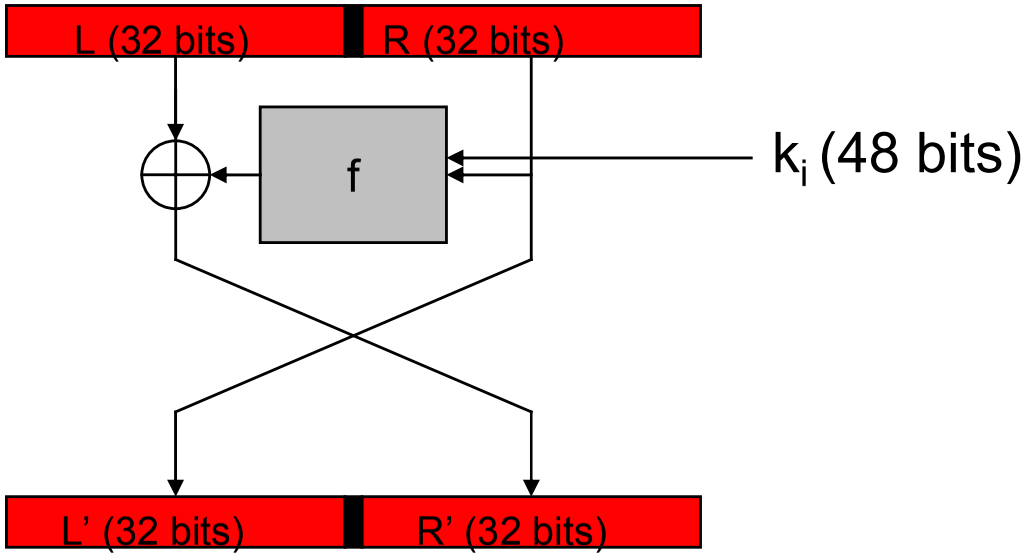
  … … … …

  $c_m = f_{m1}(K)p_1 + f_{m2}(K)p_2 + \ldots + f_{mn}(K)p_n$
- Even a weak round function can yield a strong Feistel cipher if iterated sufficiently.
  - Provided it's non-linear

# DES


64-bit Plaintext
56-bit Key
$k_2$
$k_1$ (48 bits)
16 Feistel Rounds
$k_{16}$
64-bit Ciphertext

# DES Round




F(K,X)= non-linear function

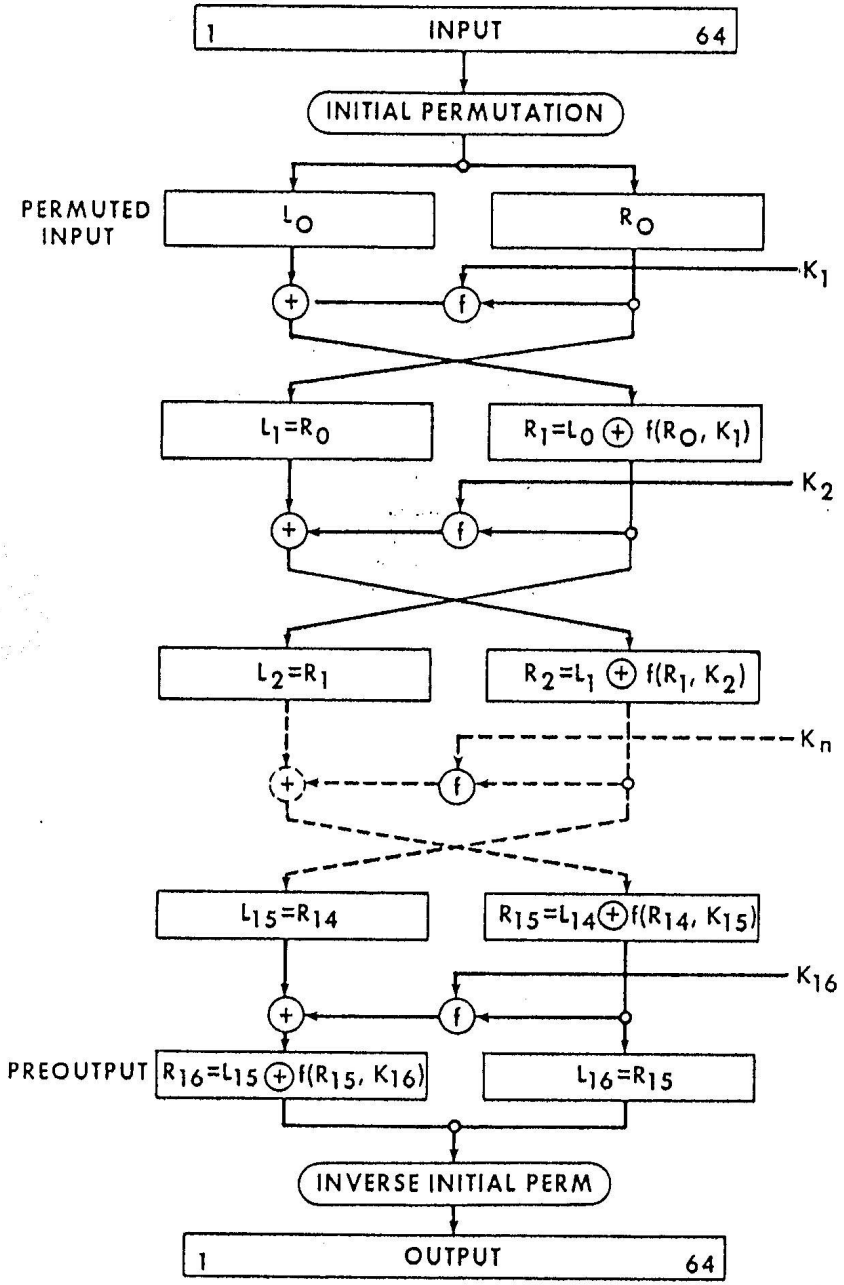



Figure 5.1. Electronic Codebook (ECB) Mode—Enciphering Computation.

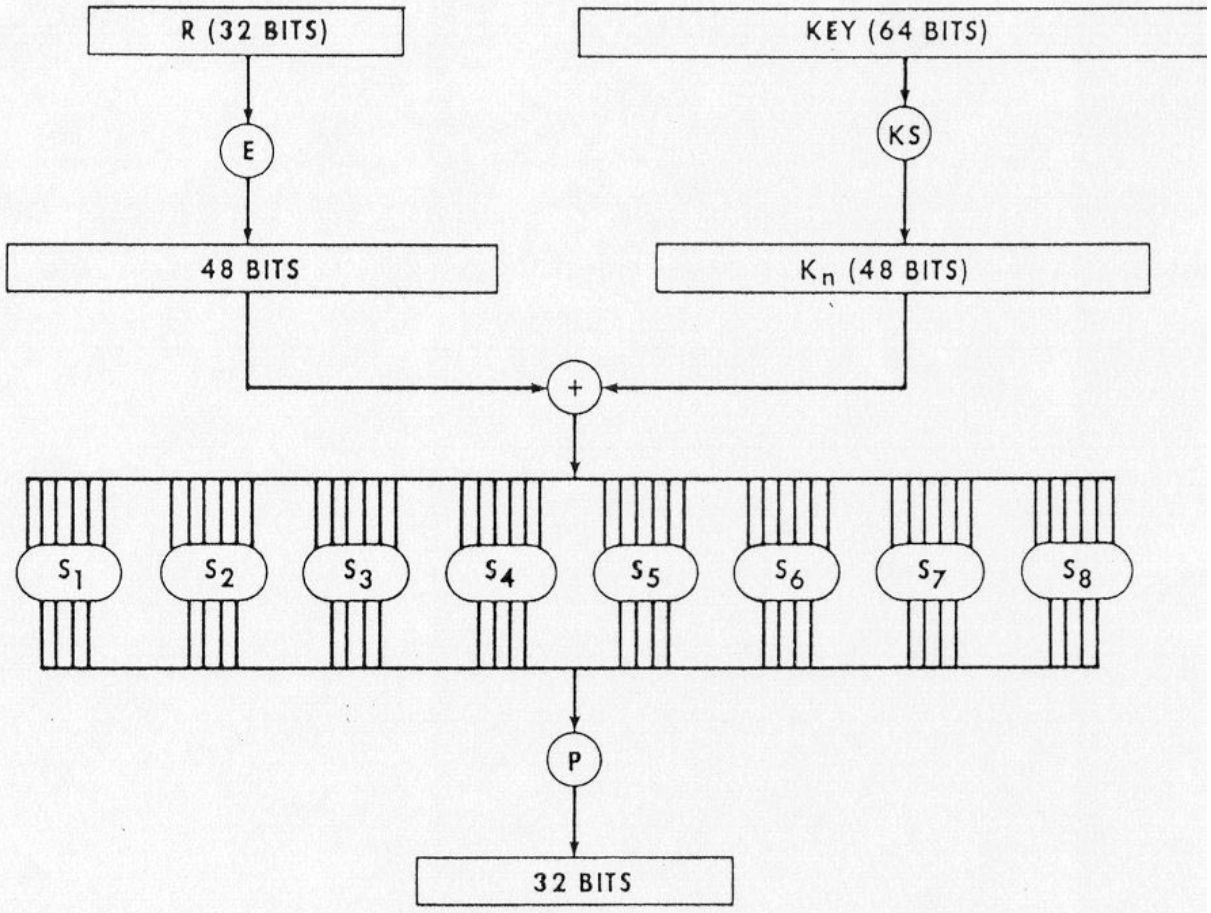



**Figure 5.2.** Electronic Codebook (ECB) Mode—Calculation of f(R,K).

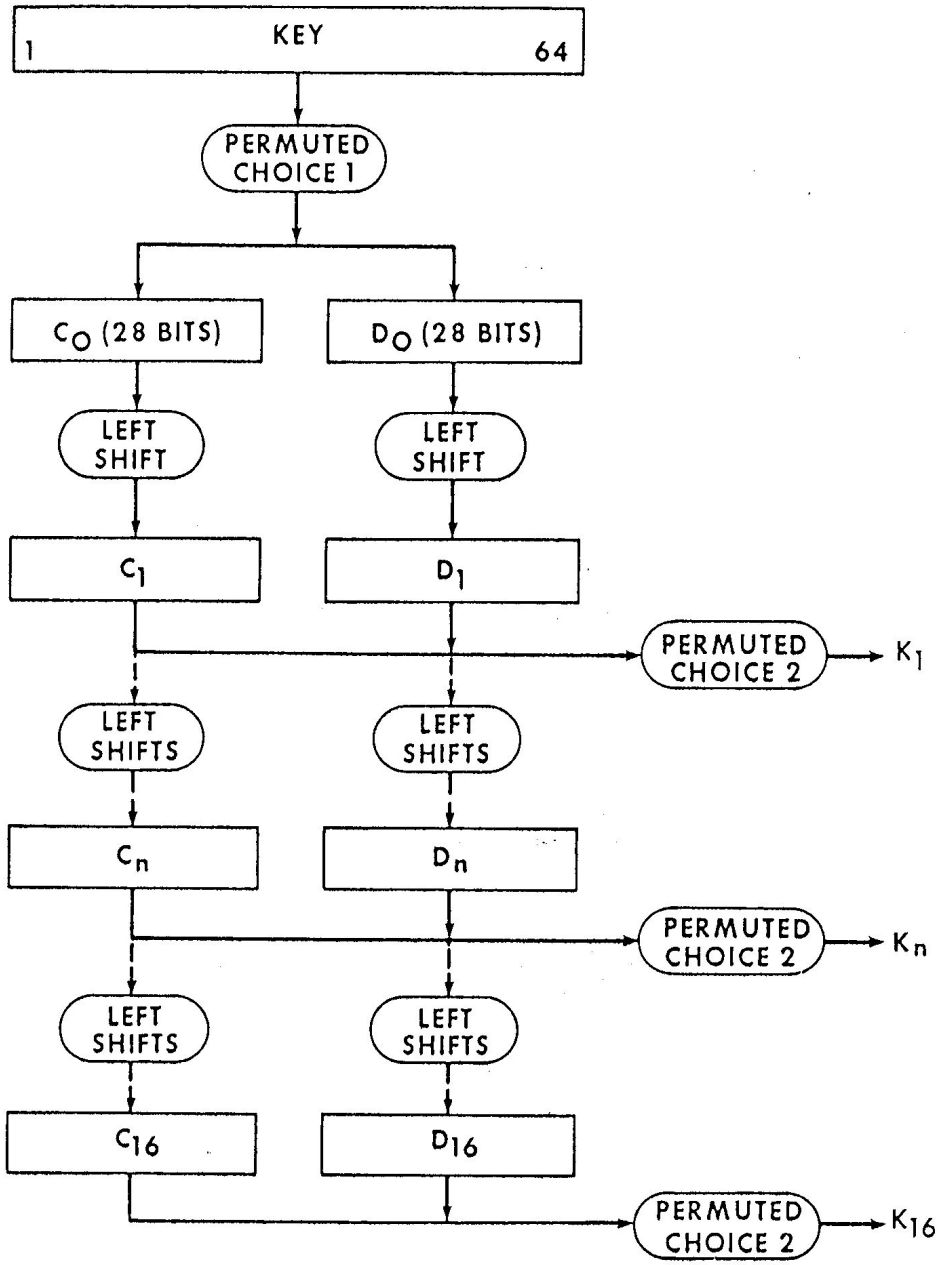



Figure 5.3. Electronic Codebook (ECB) Mode—Key Schedule (KS) Calculation.

# DES Described Algebraically

$\sigma_i(L,R) = (L \oplus f(E(R) \oplus k_i), R)$

$k_i$ is 48 bit sub-key for round i and

$f(x) = P(S_1 S_2 S_3 \ldots S_8(x))$. Here, each S –box operates on 6 bit quantities and outputs 4 bit quantities. P permutes the resulting 32 output bits.

$\tau(L,R) = (R,L)$.

Each round (except last) is $\tau \sigma_i$. Note that $\tau \tau = \tau^2 = 1 = \sigma_i \sigma_i = \sigma_i^2$.

Full DES is:

$DES_K(x) = IP^{-1} \sigma_{16} \tau \ldots \sigma_3 \tau \sigma_2 \tau \sigma_1 IP(x)$

So its inverse is

$DES_K^{-1}(x) = IP^{-1} \sigma_1 \tau \ldots \sigma_{14} \tau \sigma_{15} \tau \sigma_{16} IP(x)$

# DES Key Schedule

$C_0D_0 = PC_1(K)$

$C_{i+1} = \text{LeftShift}(\text{Shift}_i, C_i)$, $D_{i+1} = \text{LeftShift}(\text{Shift}_i, D_i)$,

$K_i = PC_2(C_i \,||D_i)$

$\text{Shift}_i = <1,2,2,2,2,2,2,1,2,2,2,2,2,2,1,1>$

Note: Irregular Key schedule protects against related key attacks. [Biham, New Types of Cryptanalytic Attacks using Related Keys, TR-753, Technion]

# DES Key Schedule

```
pc1[64]
  57 49 41 33 25 17 09 01 58 50 42 34 26 18 10 02
  59 51 43 35 27 19 11 03 60 52 44 36 63 55 47 39
  31 23 15 07 62 54 46 38 30 22 14 06 61 53 45 37
  29 21 13 05 28 20 12 04 00 00 00 00 00 00 00 00

pc2[48]
  14 17 11 24 01 05 03 28 15 06 21 10 23 19 12 04
  26 08 16 07 27 20 13 02 41 52 31 37 47 55 30 40
  51 45 33 48 44 49 39 56 34 53 46 42 50 36 29 32
```

# DES Key Schedule

Key schedule round 1

```
10 51 34 60 49 17 33 57  2  9 19 42  3 35 26 25 44 58 59
 1 36 27 18 41
22 28 39 54 37  4 47 30  5 53 23 29 61 21 38 63 15 20 45
14 13 62 55 31
```

Key schedule round 2

```
 2 43 26 52 41  9 25 49 59  1 11 34 60 27 18 17 36 50 51
58 57 19 10 33
14 20 31 46 29 63 39 22 28 45 15 21 53 13 30 55  7 12 37
 6  5 54 47 23
```

# DES Data

```
S4 (hex)
   7 d e 3 0 6 9 a 1 2 8 5 b c 4 f
   d 8 b 5 6 f 0 3 4 7 2 c 1 a e 9
   a 6 9 0 c b 7 d f 1 3 e 5 2 8 4
   3 f 0 6 a 1 d 8 9 4 5 b c 7 2 e

S5 (hex)
   2 c 4 1 7 a b 6 8 5 3 f d 0 e 9
   e b 2 c 4 7 d 1 5 0 f a 3 9 8 6
   4 2 1 b a d 7 8 f 9 c 5 6 3 0 e
   b 8 c 7 1 e 2 d 6 f 0 9 a 4 5 3

S6 (hex)
   c 1 a f 9 2 6 8 0 d 3 4 e 7 5 b
   a f 4 2 7 c 9 5 6 1 d e 0 b 3 8
   9 e f 5 2 8 c 3 7 0 4 a 1 d b 6
   4 3 2 c 9 5 f a b e 1 7 6 0 8 d
```

# DES Data

S7 (hex)

```
4 b 2 e f 0 8 d 3 c 9 7 5 a 6 1
d 0 b 7 4 9 1 a e 3 5 c 2 f 8 6
1 4 b d c 3 7 e a f 6 8 0 5 9 2
6 b d 8 1 4 a 7 9 5 0 f e 2 3 c
```

S8 (hex)

```
d 2 8 4 6 f b 1 a 9 3 e 5 0 c 7
1 f d 8 a 3 7 4 c 5 6 b 0 e 9 2
7 b 4 1 9 c e 2 0 6 a d f 3 5 8
2 1 e 7 4 a 8 d f c 9 0 3 5 6 b
```

E

```
32  1  2  3  4  5
 4  5  6  7  8  9
 8  9 10 11 12 13
12 13 14 15 16 17
16 17 18 19 20 21
20 21 22 23 24 25
24 25 26 27 28 29
28 29 30 31 32  1
```

Note: DES can be made more secure against linear attacks by changing the order of the S-Boxes: Matsui, On Correlation between the order of S-Boxes and the Strength of DES. Eurocrypt,94.

# S Boxes as Polynomials over GF(2)

```
1,1:
    56+4+35+2+26+25+246+245+236+2356+16+15+156+14+146+145+13+135+134+1346+1
    345+13456+125+1256+1245+123+12356+1234+12346

1,2:
    C+6+5+4+45+456+36+35+34+346+26+25+24+246+2456+23+236+235+234+2346+1+15+
    156+134+13456+12+126+1256+124+1246+1245+12456+123+1236+1235+12356+
    1234+12346

1,3:
    C+6+56+46+45+3+35+356+346+3456+2+26+24+246+245+236+16+15+145+13+1356+13
    4+13456+12+126+125+12456+123+1236+1235+12356+1234+12346

1,4: C+6+5+456+3+34+346+345+2+23+234+1+15+14+146+135+134+1346+1345+1256+
     124+1246+1245+123+12356+1234+12346
```

# DES Data

P

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 16 | 7 | 20 | 21 | 29 | 12 | 28 | 17 | 1 | 15 | 23 | 26 | 5 | 18 | 31 | 10 |

| 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 8 | 24 | 14 | 32 | 27 | 3 | 9 | 19 | 13 | 30 | 6 | 22 | 11 | 4 | 25 |

Note on applying permutations: For permutations of bit positions, like P above, the table entries consisting of two rows, the top row of which is "in order" means the following. If t is above b, the bit at b is moved into position t in the permuted bit string. For example, after applying P, above, the most significant bit of the output string was at position 16 of the input string.

# Homework

Review DES description by reading

http://www.aci.net/kalliste/des.htm or

http://www.itl.nist.gov/fipspubs/fip46-2.htm

You need to know DES well for a future class as well as for problem number 5.

# DES Data

```
S1 (hex)
   e 4 d 1 2 f b 8 3 a 6 c 5 9 0 7
   0 f 7 4 e 2 d 1 a 6 c b 9 5 3 8
   4 1 e 8 d 6 2 b f c 9 7 3 a 5 0
   f c 8 2 4 9 1 7 5 b 3 e a 0 6 d

S2 (hex)
   f 1 8 e 6 b 3 4 9 7 2 d c 0 5 a
   3 d 4 7 f 2 8 e c 0 1 a 6 9 b 5
   0 e 7 b a 4 d 1 5 8 c 6 9 3 2 f
   d 8 a 1 3 f 4 2 b 6 7 c 0 5 e 9

S3 (hex)
   a 0 9 e 6 3 f 5 1 d c 7 b 4 2 8
   d 7 0 9 3 4 6 a 2 8 5 e c b f 1
   d 6 4 9 8 f 3 0 b 1 2 c 5 a e 7
   1 a d 0 6 9 8 7 4 f e 3 b 5 2 c
```

# Cryptographic Hashes

A cryptographic hash ("CH") is a "one way function," h, from all binary strings (of arbitrary length) into a fixed block of size n (called the size of the hash) with the following properties:

1. Given y=h(x) it is infeasible to calculate a x' $\neq$x such that y=h(x'). ("One way," "non-invertibility" or "pre-image" resistance). Functions satisfying this condition are called One Way Hash Functions (OWHF)
2. Given u, it is infeasible to find w such that h(u)=h(w). (weak collision resistance, 2nd pre-image resistance).
3. It is infeasible to find u, w such that h(u)=h(w). (strong collision resistance). Note 3→2. Functions satisfying this condition are called Collision Resistant Functions (CRFs).

- Just like Symmetric ciphers ratio of work factor for computation of hash vs work factor to break hash should be very high.
- Adversary has complete information on computing hash and (obviously) can compute as many hashes from the target as she wants.

# Observations on Cryptographic Hashes

- Hashes are a strong “checksum”
- OWHF and CRF conditions make CHs satisfy many of the properties of “random functions”
  - Small changes should create large changes (otherwise the pre-image of near neighbors are near neighbors making collisions easy to find)
  - Small input changes should be statistically unrelated (uncorrelated) to changes in a subset of the hash bits
  - Analysis of CHs very similar to Symmetric Cipher techniques

Popular practical cryptographic hashes

- MD4, MD5 (now “broken”)
- SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 (last 4 are “SHA-2”)
- RIPEMD

# Observations

- Collision Resistance → $2^{nd}$ pre-image resistance
- Let $f(x)= x^2-1 \pmod p$.
  - f(x) acts like a random function but is not a OWHF since square roots are easy to calculate mod p.
- Let $f(x)= x^2 \pmod{pq}$.
  - f(x) is a OWHF but is neither collision nor $2^{nd}$ pre-image resistant
- If either $h_1(x)$ or $h_2(x)$ is a CRHF so is $h(x)= h_1(x) \,||\, h_2(x)$
- MDC+signature & MAC+unknown Key require all three properties
- Ideal Work Factors:

| **Type** | **Work** | **Property** |
|---|---|---|
| OWHF | $2^n$ | Pre-image<br>$2^{nd}$ Pre-image |
| CRHF | $2^{n/2}$ | Collision |
| MAC | $2^t$ | Key recovery, computational resistance |

# What are Hash Functions Good for?

- Modification Detection Codes (MDCs): This is a strong checksum (integrity check). Sometimes called “unkeyed” hashes.
- Message Authentication Code (MACs): If shared secret is part of the hash, two parties can determine authenticated integrity with CHs. Called “keyed hashes”.
- Message Digests (MDs): Encrypting (with private key) the CH of a message (its MD) acts as a certification that the message was “approved” by possessor of private key. This is called a Digital Signature. [Note: you could “sign’ the whole message rather than the hash but this would take oodles of time by comparison.]

# What are Hash Functions Good for?

- Uniquely and securely identifies bit streams like programs. Hash is strong name for program.
- Entropy mixing:  Since CHs are random functions into fixed size blocks with the properties of random functions, they are often used to “mix” biased input to produce a “seed” for a psuedo-random number generator.
- Password Protection: Store salted hash of password instead of password (Needham).
- Bit Committment

# One-Way Functions

Hashes come from two basic classes of one-way functions

- Mathematical
  - Multiplication: $Z=X \bullet Y$
  - Modular Exponentiation: $Z = Y^X \pmod n$ (Chaum vP Hash)
- Ad-hoc (Symmetric cipher-like constructions)
  - Custom Hash functions (MD4, SHA, MD5, RIPEMD)

# Chaum-vanHeijst-Pfitzmann Compression Function

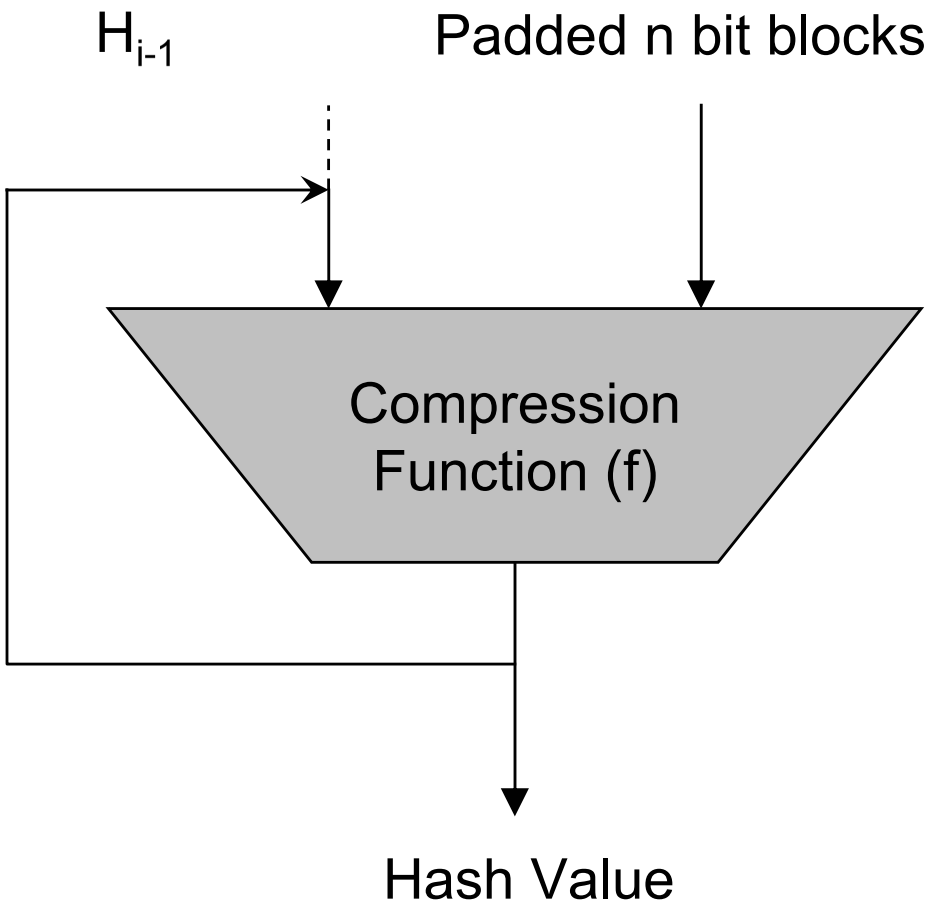
- Suppose p is prime, q=(p-1)/2 is prime, a is a primitive root in $F_p$, b is random.
- g: $\{1,2,\ldots,q-1\}^2 \rightarrow \{1,2,\ldots,p-1\}$, q=(p-1)/2 by:
  - $g(s, t) = a^s b^t \pmod p$
- Not used in practice: too slow.
- Reduction to discrete log:

  Suppose g(s, t)= g(u, v) can be found. Then $a^s b^t \pmod p = a^u b^v \pmod p$.

  So $a^{s-u} \pmod p = b^{v-t} \pmod p$. Let $b= a^x \pmod p$. Then $(s-u)=x(y-t) \pmod{p-1}$.

  But p-1= 2q so we can solve for x, thus determining the discrete log of b.

# Merkle/Damgard Construction

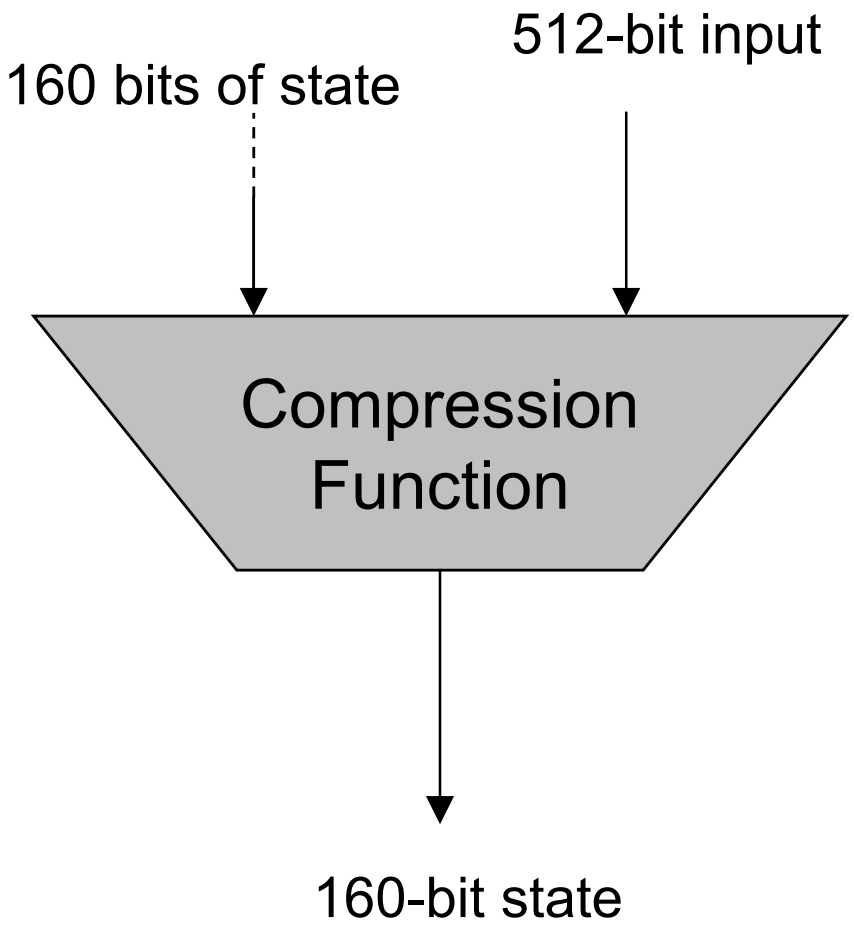



Input: $x=x_1||\ldots||x_t$
Input is usually padded

$H_0$= IV
$H_i= f(H_{i-1}, x_i)$
$h(x)= g(h_t)$

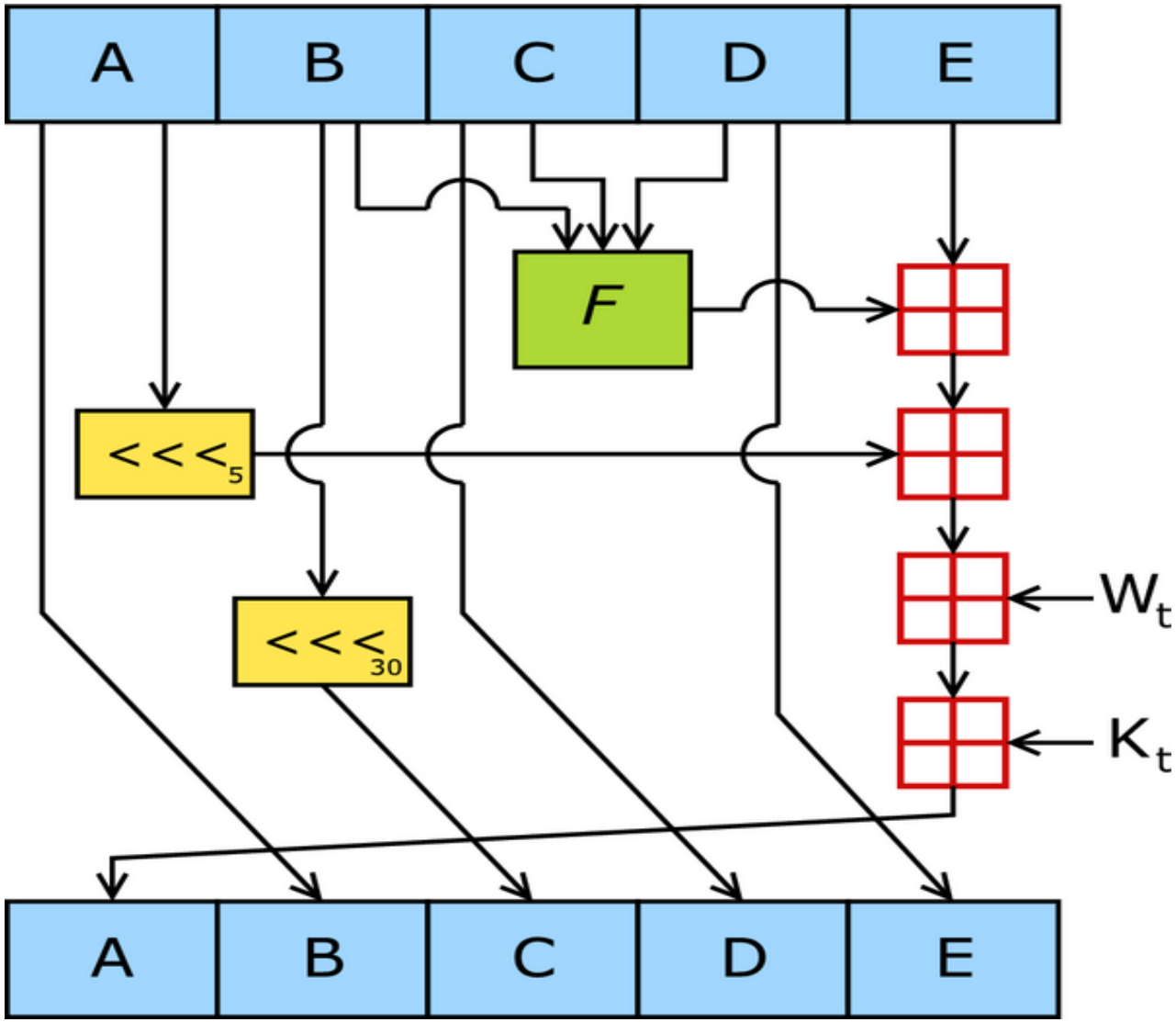
Graphic by Josh Benaloh

# A Cryptographic Hash: SHA-1




Slide by Josh Benaloh

# A Cryptographic Hash: SHA-1




Picture from Wikipedia

# Padding

- Standard technique
  - Let last message block have k bits. If k=n, make a new block and set k= 0.
  - Append a 1 to last block leaving r=n-k-1 remaining bits in block.
  - If r>=64, append r-64 0s then append bit length of input expressed as 64 bit unsigned integer
  - If r<64, append n-r 0's (to fill out block), append n-64 0's at beginning of next block then append bit length of input expressed as 64 bit unsigned integer

# Technique for CHs from Block Ciphers

Let input be $x= x_1 \| x_2 \| \ldots \| x_t$ where each $x_i$ is n bits long. Let g be a function taking an n bit input to an m bit input. Let E(k, x) be a block cipher with m bit keyspace and n bit block. Let $H_0= IV$.

Construction 1

$H_i= E(g(H_{i-1}), x_i) \oplus H_{i-1}$

Construction 2

$H_i= E(x_i, H_{i-1}) \oplus H_{i-1}$

Construction 3

$H_i= E(g(H_{i-1}), x_i) \oplus x_i \oplus H_{i-1}$

Note: Because of collisions n should be >64. Ideally, m=n and g= id. DES with n= 64 is too small. AES with n=m=128 is better.

# Birthday Attacks

- Probability of collision determined by “Birthday Paradox” calculation:
  - $(1-1/n)\ (1-2/n)\ \ldots\ (1-(k-1)/n) = (n!/k!)/n^k$
  - Probability of collision is >.5 when $k^2 > n$.
  - Need $2^{80}$ blocks for SHA.
  - $1+x \leqq e^x$, $\Pi_{i=1}^{i=k}\ (1-i/n) \leqq e^{-k(k-1)/(2n)}$

# Attacks on Cryptographic Hashes

- Berson (1992) using differential cryptanalysis on 1 round MD-5.
- Boer and Bosselaers (1993), Pseudo collision in MD5.
- Dobbertin (1996), Collisions in compression function. Attacks inspired RIPEMD proposal.
- Biham and Chen (2004), Collisions in SHA-0.
- Chabaud and Joux (2004), Collisions in SHA-0 .
- Wang, Feng, Lai, Yu, (2004), MD4, MD5, RIPEMD
- Wang et al, (2004, 2005), SHA-1

- SHA-1 has stood up best: best known theoretical attack (11/05) requires $2^{63}$ operations.

# MACs using Hashes

- Prefix and suffix attacks
- Hash($k_1$, Hash($k_2$, m))
- Hash(k|p|m|k)

- $HMAC_K(x)$= SHA-1(K⊕opad || SHA-1(K ⊕ipad )||x)

# Winnowing and Chaffing (Rivest)

- Want to send 1001. Pick random stream ($m_i$) and embed message at positions (say) 3, 7, 8 14 MAC each packet ($mm_i$).
- Make sure MAC is correct only in message positions

# Homework 1-Question 1

Encrypt the following message using a Vigeniere cipher with direct standard alphabets. Key: JOSH.

```
All persons born or naturalized in the United States, and
subject to the jurisdiction thereof, are citizens of the
United States and of the state wherein they reside. No
state shall make or enforce any law which shall abridge
the privileges or immunities of citizens of the United
States; nor shall any state deprive any person of life,
liberty, or property, without due process of law; nor deny
to any person within its jurisdiction the equal protection
of the laws.
```

Upper case only! Turn plain and cipher text into 5 letter groups.

Calculate the index of coincidence of the plaintext and ciphertext.

Break the ciphertext into 4 columns. What is the index of coincidence of each column?

# Homework 1-Question 1 (addendum)

```
ALLPE RSONS BORNO RNATU RALIZ EDINT HEUNI TEDST ATESA NDSUB JECTT
OTHEJ URISD ICTIO NTHER EOFAR ECITI ZENSO FTHEU NITED STATE SANDO
FTHES TATEW HEREI NTHEY RESID ENOST ATESH ALLMA KEORE NFORC EANYL
AWWHI CHSHA LLABR IDGET HEPRI VILEG ESORI MMUNI TIESO FCITI ZENSO
FTHEU NITED STATE SNORS HALLA NYSTA TEDEP RIVEA NYPER SONOF LIFEL
IBERT YORPR OPERT YWITH OUTDU EPROC ESSOF LAWNO RDENY TOANY PERSO
NWITH INITS JURIS DICTI ONTHE EQUAL PROTE CTION OFTHE LAWS
```

# Homework 1-Question 2

Break the Vigeniere based ciphertext below. Plaintext and ciphertext alphabets are direct standard.

What is the key length? What is the key?

If the key length is k, how long a corresponding plain, ciphertext sequence be given to solve? Can you give an upper bound on the pure ciphertext length needed?

```
IGDLK MJSGC FMGEP PLYRC IGDLA TYBMR KDYVY XJGMR TDSVK ZCCWG ZRRIP
UERXY EEYHE UTOWS ERYWC QRRIP UERXJ QREWQ FPSZC ALDSD ULSWF FFOAM
DIGIY DCSRR AZSRB GNDLC ZYDMM ZQGSS ZBCXM OYBID APRMK IFYWF MJVLY
HCLSP ZCDLC NYDXJ QYXHD APRMQ IGNSU MLNLG EMBTF MLDSB AYVPU TGMLK
MWKGF UCFIY ZBMLC DGCLY VSCXY ZBVEQ FGXKN QYMIY YMXKM GPCIJ HCCEL
PUSXF MJVRY FGYRQ
```

# Homework 1-Question 3

Consider a message source M(x) with the following distribution:

M: P(x=0)= p

M: P(x=1)= q, with p+q=1

and a one time pad selected from distribution P(x)

P: P(x=0)= ½

P: P(x=1)= ½

Consider the ciphertext formed by "xoring" the message m with the pad p, so that c= m⊕p

What is the ciphertext distribution C?

Calculate H(M), H(P), H(C).

Calculate: I(M|C)= H(M)-H(M|C)

Suppose, P: P(x=0)=3/4 and P(x=1)=1/4. What is the ciphertext distribution and H(M), H(P), H(C) and I(M|C) now.

# Homework 1-Question 4

Calculate the output of the first two rounds of DES with input message 0x313233435363738 and key 0x00abcdefabcdef89. The input to round 1 (after initial permutation) and the first 2 round keys are given below.

For fixed key, DES is a permutation on $2^{64}$ letters.

Approximately how many such permutations are there? (Hint: Use Stirling's approximation.)

Compare this to the size of the key space for DES.

```
Round 01 Key:    01001111 01010111 00000111 10111001 01011100 10101011
Round 02 Key:    00101111 00100111 01101001 00111011 01111111 00100100

Input to DES:    00110100 00110011 00110010 00110001
                 00111000 00110111 00110110 00110101
Round 1 Input:   00000000 11111111 11100001 10101010
                 00000000 11111111 00010000 01100110
```

# Homework 1-Question 5 (Extra Credit)

Given a one rotor machine, M, depicted below with equation $C^i R^{-1} C^{-i} U C^{-i} R C^i (p)=c$ with C, U, R below.

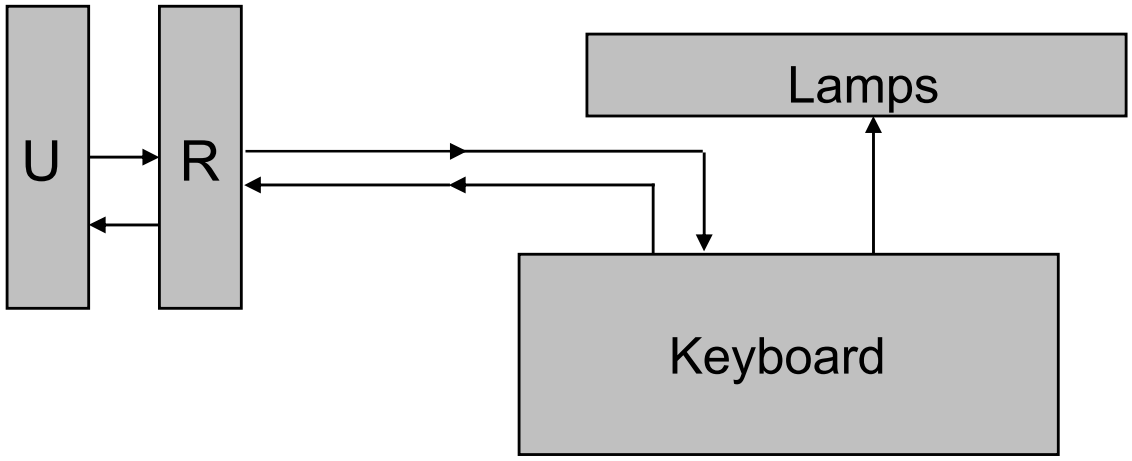
R: `ABCDEFGHIJKLMNOPQRSTUVWXYZ`

`EKMFLGDQVZNTOWYHXUSPAIBRCJ`

U: `ABCDEFGHIJKLMNOPQRSTUVWXYZ`

`YRUHQSLDPXNGOKMIEBFZCWVJAT`

C: The cyclic permutation A→B, B→C, etc



# Homework 1-Question 5 (cont)

Calculate the ciphertext derived from the plaintext: `HELLOWORLD`

What do you think the index of coincidence of the ciphertext from a 50 letter message is?

If the key was the "starting position," i, of M, how many letters of corresponding plain/cipher text would you need to find the key? How many ciphertext only letters?

The following may be helpful:

$R^{-1}$ `ABCDEFGHIJKLMNOPQRSTUVWXYZ`

`UWYGADFPVZBECKMTHXSLRINQOJ`

# General Modern References


Blake, Seroussi, and Smart, Elliptic Curves in Cryptography, Cambridge

Bressoud and Wagon, Computational Number Theory. Key Press.

Bach and Shallit, Algorithmic Number Theory.

Berlekamp, Algebraic Coding Theory. Reprinted by Aegean Park Press.

Biham and Shamir, Differential Cryptanalysis of DES. Springer.

Boneh, Twenty Years of attacks on RSA. Notices AMS.

Buchmann, Introduction to Cryptography. Springer.

Cohen, A Course in Computational Algebraic Number Theory. Springer.

Damgard, Lectures on Data Security. Springer.

Golumb, Shift Register Sequences. Reprinted by Aegean Park Press.

Koblitz, A Course in Number Theory and Cryptography. Springer.

Koblitz, Algebraic Aspects of Cryptography. Springer.

Konheim, Cryptography: A Primer. Wiley.

# General Modern References


Landau, DES, AES, Survey article. Notices AMS.

MacWilliams et. al., Theory of Error Correcting Codes. North Holland.

Menezes, van Oorshot, Vanstone, Handbook of Applied Cryptography. (Online: http://www.cacr.math.uwaterloo.ca/hac/). CRC Press.

Rhee, Cryptography and Secure Communications.

Rivest, Class notes on Security and Crypto online. (web.mit.edu).

Schneier, Applied Cryptography. Wiley.

Simovits, The DES: Documentation and Evaluation. Aegean Park Press.

Stinson, Cryptography: Theory and Practice. CRC Press.

Welch, Codes and Cryptography. Oxford.


Web sites: www.rsa.com, www.counterpane.com, www.iacr.org has loads of preprints.

# End Paper

- Done