# Lecture 4: Striving for Confusion

Structures have been found in DES that were undoubtedly inserted to strengthen the system against certain types of attack. Structures have also been found that appear to weaken the system.

*Lexar Corporation, "An Evalution of the DES", 1976.*

CS588: Security and Privacy
University of Virginia
Computer Science

David Evans
http://www.cs.virginia.edu/~evans

# Menu

- Projects
- Enigma Continued
- Block Ciphers



# Operation

- Day key (distributed in code book)
- Each message begins with message key ("randomly" choosen by sender) encoded using day key
- Message key sent twice to check
- After receiving message key, re-orient rotors according to key



# Letter Permutations

Symmetry of Enigma:
if $E_{pos}(x) = y$ we know $E_{pos}(y) = x$
Given message openings

| | | |
|---|---|---|
| `DMQ VBM` | $E_1(m_1) = D$ | $E_4(m_1) = V$ |
| `VON PUY` | $E_1(m_2) = V$ | $E_4(m_2) = P$ |
| `PUC FMQ` | | |

With enough message openings, we can build complete cycles for each position pair:

$E_1E_4$ = (DVPFKXGZYO) (EIJMUNQLHT) (BC) (RW) (A) (S)

Note: Cycles must come in pairs of equal length
(Examples in Code Book had pairs of unequal length)

## Composing Involutions

- $E_1$ and $E_4$ are involutions ($x \rightarrow y \Rightarrow y \rightarrow x$)
- Without loss of generality, we can write:

$E_1$ contains $(a_1 a_2)$ $(a_3 a_4)$ … $(a_{2k-1} a_{2k})$
$E_2$ contains $(a_2 a_3)$ $(a_4 a_5)$ … $(a_{2k} a_1)$

| $E_1$ | $E_2$ |
|---|---|
| $a_1 \leftrightarrow a_2$ | $a_2 \leftrightarrow x = a_3$ or $x = a_1$ |
| $a_3 \leftrightarrow a_4$ | $a_4 \leftrightarrow x = a_5$ or $x = a_1$ |



## Rejewski's Theorem

$E_1$ contains $(a_1 a_2)$ $(a_3 a_4)$ … $(a_{2k-1} a_{2k})$
$E_4$ contains $(a_2 a_3)$ $(a_4 a_5)$ … $(a_{2k} a_1)$

$E_1 E_4$ contains $(a_1 a_3 a_5 \ldots a_{2k-1})$
$(a_{2k} a_{2k-2} \ldots a_4 a_2)$

- The product of two involutions consists of pairs cycles of the same length
- For cycles of length *n*, there are *n* possible factorizations



## Factoring Permutations

$E_1E_4$ = (DVPFKXGZYO) (EIJMUNQLHT) (BC) (RW) (A) (S)

(A) (S) = (AS) o (SA)
(BC) (RW) = (BR)(CW) o (BW)(CR)
or = (BW)(RC) o (WC) (BR)



## How many factorizations?

(DVPFKXGZYO) (EIJMUNQLHT)

| $E_1$ | $E_2$ |
|---|---|
| $D \leftrightarrow a_2$ | $a_2 \leftrightarrow V$ |
| $V \leftrightarrow a_4$ | $a_4 \leftrightarrow P$ |

Once we guess $a_2$ everything else must follow!
So, only *n* possible factorizations for an *n*-letter cycle

Total to try = 2 * 10 = 20

$E_2E_5$ and $E_3E_6$ likely to have about 20 to try also

$\Rightarrow$ About $20^3$ (8000) factorizations to try
(still too many in pre-computer days)

## Luckily…

- Operators picked guessable message keys (“cillies”)
  - Identical letters
  - Easy to type (e.g., QWE)
- If we can guess $P_1 = P_2 = P_3$ (or known relationships) can reduce number of possible factorizations
- If we’re lucky – this leads to $E_1 \ldots E_6$



## 1939

- Early 1939 – Germany changes scamblers and adds extra plugboard cables, stop double-transmissions
  - Poland unable to cryptanalyze
- July 1939 – Rejewski invites French and British cryptographers
  - It is actually breakable
  - Gives England replica Enigma machine constructed from plans



## Bletchley Park

- Alan Turing leads British effort to crack Enigma
- Use cribs (“WETTER” transmitted every day at 6am)
- Still needed to brute force check ~1M keys.
- Built “bombes” to automate testing
- How many people worked on breaking Enigma? 30,000 people worked at Bletchley Park on breaking Enigma – 100,000 for Manhattan Project



## Enigma Cryptanalysis

- Relied on combination of sheer brilliance, mathematics, espionage, operator errors, and hard work
- Huge impact on WWII
  - Britain knew where German U-boats were
  - Advance notice of bombing raids
  - But...keeping code break secret more important than short-term uses

## End of classical ciphers

*A billion billion is a large number, but it's not that large a number.*
— Whitfield Diffie



## Goals of Cipher: Diffusion and Confusion

- Claude Shannon [1945]
- Diffusion:
  - Small change in *plaintext*, changes lots of *ciphertext*
  - Statistical properties of plaintext hidden in ciphertext
- Confusion:
  - Statistical relationship between *key* and *ciphertext* as complex as possible
- So, need to design functions that produce output that is diffuse and confused



## Block Ciphers

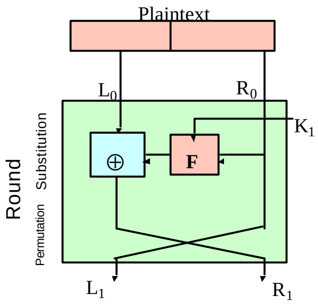
- Stream Ciphers
  - Encrypts small (bit or byte) units one at a time
- Block Ciphers
  - Encrypts large chunks (64 bits) at once
- Ciphers we have seen so far:
  - Changing one letter of message only changes one letter of ciphertext
  - There were classical ciphers that had some diffusion: Vigenère autokey, Hill cipher (2-letter chunks)



## Ideal Block Cipher

- 64 bit blocks
- $2^{64}$ possible plaintext blocks, must have at least $2^{64}$ corresponding ciphertext blocks
  - There are $2^{64}!$ possible mappings
- Why not just create a random mapping?
  - Need a $2^{64}$ * 64-bit table $\approx 10^{21}$ bits
  - $14 quadrillion
  - Need to distribute new table if compromised
- Approximate ideal random mapping using components controlled by a key

## Feistel Cipher Structure




$L_0$ = left half of plaintext
$R_0$ = right half of plaintext

$L_i = R_{i-1}$
$R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$

$C = R_n \| L_n$
n is number of rounds
(undo last permutation)



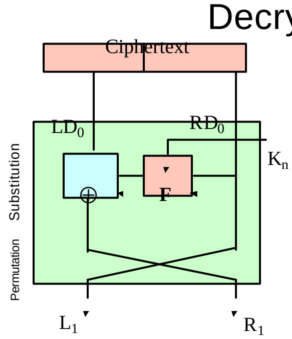
## One Round Feistel

$L_i = R_{i-1}$
$R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$

$E(L_0 \| R_0)$:
$L_1 = R_0$
$R_1 = L_0 \oplus F(R_0, K_1))$
$C = R_1 \| L_1 = L_0 \oplus F(R_0, K_1)) \| R_0$



## Decryption




$LD_0$ = left half of ciphertext
$RD_0$ = right half of ciphertext

$LD_i = RD_{i-1}$
$RD_i = LD_{i-1}$
$\oplus F(RD_{i-1}, \mathbf{K_{n-i+1}})$

$P = RD_n \| LD_n$
n is number of rounds



## Decryption

$LD_i = RD_{i-1}$
$RD_i = LD_{i-1} \oplus F(RD_{i-1}, K_{n-i+1})$

$D(L_0 \oplus F(R_0, K_1)) \| R_0)$
$LD_0 = L_0 \oplus F(R_0, K_1)$ $\quad RD_0 = R_0$
$LD_1 = R_0$
$RD_1 = LD_0 \oplus F(RD_0, K_1)$
$= L_0 \oplus F(R_0, K_1) \oplus F(RD_0, K_1))$
$= L_0$
$P = RD_1 \| LD_1 = L_0 \| R_0$

***Yippee!***

## Multiple Rounds

- The entire round is a function:
  $f_K (L \| R) = R \| L \oplus F (R, K))$
  $swap (L \| R) = R \| L$
- $E = swap \circ swap \circ f_{K_r} \circ swap \circ f_{K_{r-1}} \circ \ldots \circ f_{K_2} \circ swap \circ f_{K_1}$
- $D = f_{K_1} \circ swap \circ f_{K_2} \circ \ldots \circ f_{K_{r-1}} \circ swap \circ f_{K_r} \circ swap \circ swap$



## Decryption

$$\begin{aligned} &swap (f_K (swap (f_K (L \| R)) \\ &= swap (f_K (swap (R \| L \oplus F (R, K)))) \\ &= swap (f_K (L \oplus F (R, K) \| R)) \\ &= swap (R \| (L \oplus F (R, K)) \oplus F (R, K)) \\ &= swap (R \| L) = L \| R \end{aligned}$$

So $swap \circ f_K$ its own inverse!



## F

- What are the requirements on F?
  - For decryption to work: none!
  - For security:
    - Hide patterns in plaintext
    - Hide patterns in key
    - Coming up with a good F is hard



## DES

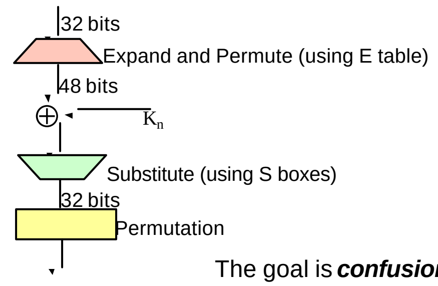
- NIST (then NBS) sought standard for data security (1973)
- IBM's Lucifer only reasonable proposal
- Modified by NSA
  - Changed S-Boxes
  - Reduced key from 128 to 56 bits
- Adopted as standard in 1976
- More bits have been encrypted using DES than any other cipher

# DES Algorithm

- Feistel cipher with added initial permutation
- Complex choice of F
- 16 rounds
- 56-bit key, shifts and permutations produce 48-bit subkeys for each round



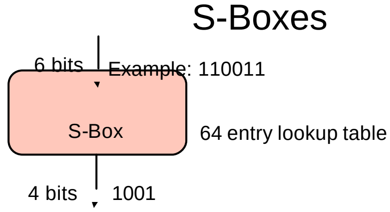
# DES's F




The goal is ***confusion***!



# S-Boxes




Critical to security
NSA changed choice of S-Boxes
Only *non-linear* step in DES

E(11) ≠ E(01) + E(10)



# DES Avalanche

```
Input:     ...............................................................*    1
Permuted:  .......................................*........................    1
Round 1:   .......*........................................................    1
Round 2:   .*..*...*.....*........................*........................    5
Round 3:   .*..*.*.**..*.*.*.*....**.....**.*..*...*.....*.................   18
Round 4:   ..*.*****.*.*****.*.*......*.....*..*.*.**..*.*.*.*....**.....**   28
Round 5:   *...**..*.*...*.*.*.*...*.***..*..*.*****.*.*****.*.*......*....   29
Round 6:   ...*..**.....*.*..**.*.**...*..**...**..*.*...*.*.*.*...*.***..*   26
Round 7:   *****...***....**...*..*.*..*......*..**.....*.*..**.*.**...*..*
Round 8:   *.*.*.*.**.....*.*.*...**.*...*******...***....**...*..*.*..*...
Round 9:   ***.*.***...**.*.****.....**.*..*.*.*.*.**.....*.*.*...**.*...**
Round 10:  *.*..*.*.**.*..*.**.***.**.*...****.*.***...**.*.****.....**.*..
Round 11:  ..******......*..******....*....*.*..*.*.**.*..*.**.***.**.*...*
Round 12:  *..***....*...*.*.*.***...****....******.....*..******.....*....
Round 13:  **..*....*..******...*........*.*..***....*...*.*.*.***...****..
Round 14:  *.**.*....*.*....**.*...*..**.****..*....*..******...*........*.
Round 15:  **.*....*.*.*...*.**.*..*.*.**.**.**.*....*.*....**.*...*..**.**
Round 16:  .*..*.*..*..*.**....**..*..*..****.*....*.*.*...*.**.*..*.*.**.*
Output:    ..*..**.*.*...*....***..***.**.*...*..*..*.*.*.**.*....*.*.*.**.
```

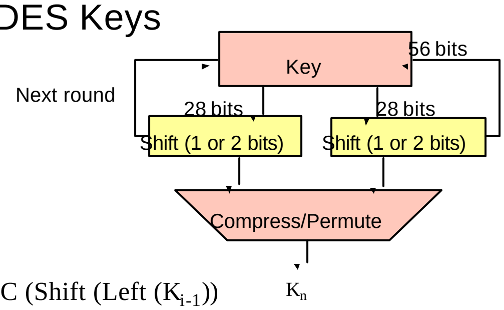
Source: *Willem de Graaf*, http://www-groups.dcs.st-and.ac.uk/~wdg/slides/node150.html

# Key Schedule

- Need 16 48-bit keys
  - Best security: just use 16 independent keys
  - 768 key bits
- 56-bit key used (64 bits for parity checking)
  - Produce 48-bit round keys by shifting and permuting



# DES Keys




$K_i$ = PC (Shift (Left ($K_{i-1}$))
|| Shift (Right ($K_{i-1}$)))

Are there any weak keys?



# Is DES a perfect cipher?

- No: more messages than keys
- Even for 1 64-bit block

  $2^{64}$ messages > $2^{56}$ keys



# Attacking DES: Brute Force

- Key is 56 bits
- $2^{56} = 7.2 * 10^{16}$ = 72 quadrillion
- Try 1 per second = 9 Billion years to search entire space
- Distributed attacks
  - Steal/borrow idle cycles on networked PCs
  - Search half of key space with 100000 PCs * 1M keys/second in 25 days

## Cracking DES

90B keys per second
Cost < $250K (in 1998)
56 hours to solve RSA DES Challenge



## Brute Force Attacks

- RSA DES challenges:
  - 1997: 96 days (using 70,000 machines)
  - Feb 1998: 41 days (distributed.net)
  - July 1998: 56 hours (custom hardware)
  - January 1999: 22 hours (EFF + distributed.net)
    - 245 Billion keys per second
- NSA can probably crack DES routinely (but they won't admit it)



## Charge

- Next time:
  - Better than brute force DES attacks
  - 3-DES
  - Modes of Operation
- Find your project teammates
- Start thinking about projects