Introduction to Practical Cryptography

Cryptanalysis

Agenda

- **Overview**
- Block Ciphers:
  - Linear
  - Differential
  - Other Attacks
  - Statistical Analysis
- Stream Ciphers
- General
  - Side Channel Attacks

Overview

- What is cryptanalysis?
- Theory
- distinguish from random
- Less work than exhaustive search, even if not practical 2^127 vs 2^100
- Practical – recover key bits, determine plaintext/ciphertext bits

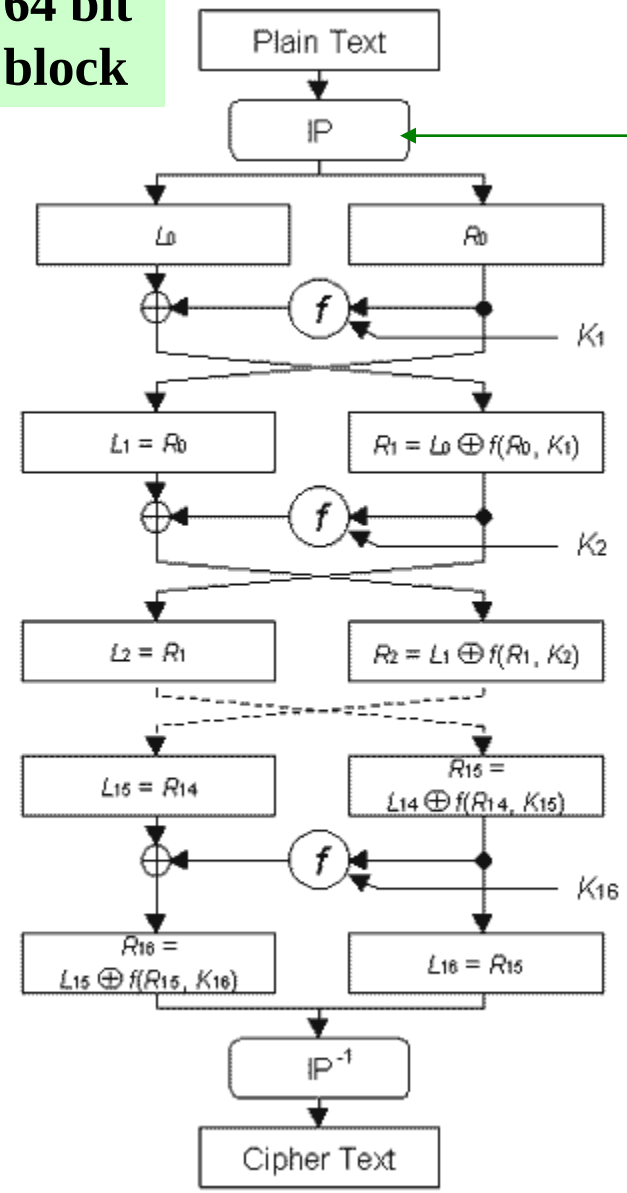
Agenda

- Overview
- **Block Ciphers:**
  - Linear
  - Differential
  - Other Attacks
  - Statistical Analysis
- Stream Ciphers
- General
  - Side Channel Attacks

Differential and Linear Cryptanalysis Origins

- Differential cryptanalysis originally defined on DES
- Eli Biham and Adi Shamir, Differential Cryptanalysis of the Data Encryption Standard, Springer Verlag, 1993.
- Linear cryptanalysis first defined on Feal by Matsui and Yamagishi, 1992.
- Matsui later published a linear attack on DES.

64 bit block



initial, final permutations

16 round Feistel network

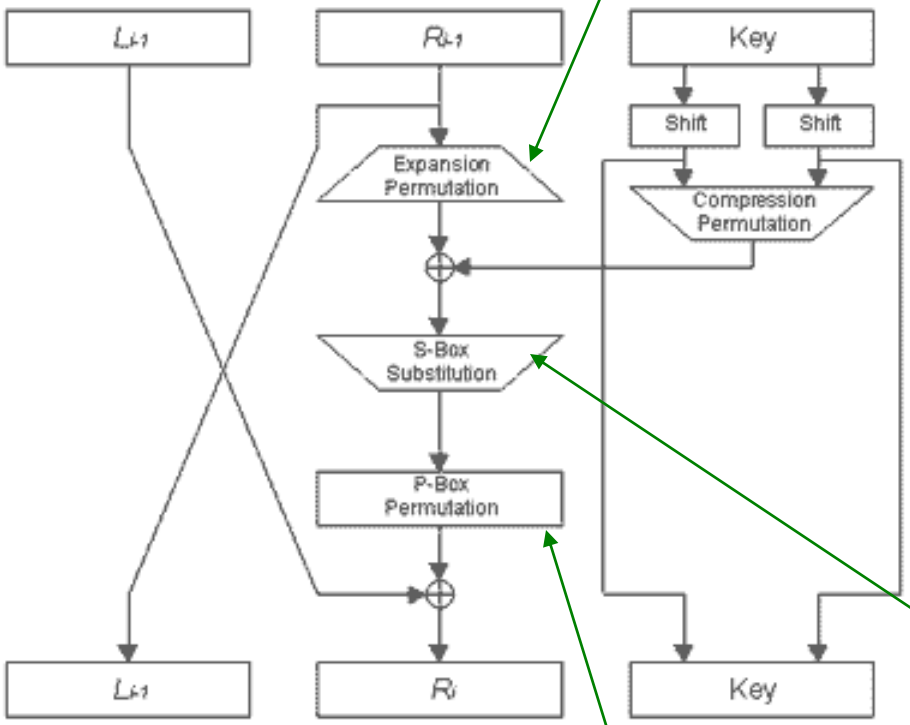
56 bit key

Decryption same as encryption with round keys used in reverse order.

DES images downloaded from
http://www.chipcenter.com/eexpert/jleiseboer/jleiseboer023.html
(original source unknown)

DES

Right half expanded from 32 to 48 bits.
Some of the 32 bits are input to 2 S-Boxes.



Round key

- **Rotate each half of 56 bit key, select 48 bits.**
- **Rotation is 1 or 2 bits, depends on round.**
- **Each key bit used in ≈ 14 rounds not in same position.**

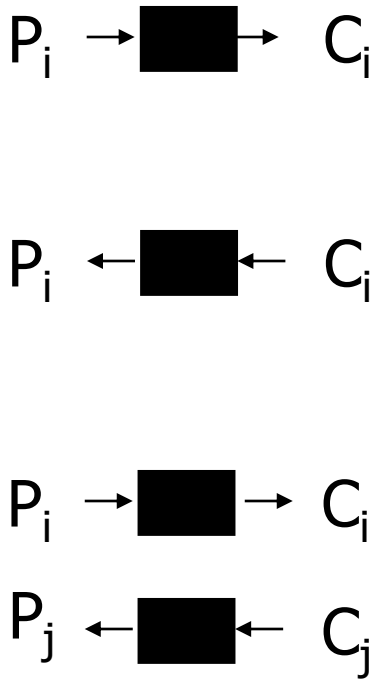
S-Box outputs permuted

8 S-Boxes
6 bit input
4 bit output

Impacts linear and differential cryptanalysis

Plaintext, Ciphertext Queries

- Ciphertext only
- Known plaintext: have set of plaintext, ciphertext pairs (P_1,C_1), (P_2,C_2) … (P_i,C_i):
- Chosen Plaintext:
  - Choose P_i's, receive C_i's
- Chosen Ciphertext:
  - Choose C_i's , receive P_i's
- Chosen Plaintext – Chosen Ciphertext:
  - Choose P_i's and C_j's, receive C_i's and P_j's



Plaintext, Ciphertext Queries

Given queries (P_1,C_1), (P_2,C_2) … (P_i,C_i):

- Adaptive Chosen Plaintext:
  - Input P_i, receive C_i, choose P_{i+1} …

- Adaptive Chosen Ciphertext:
  - Input C_i, receive P_i, choose C_{i+1} …

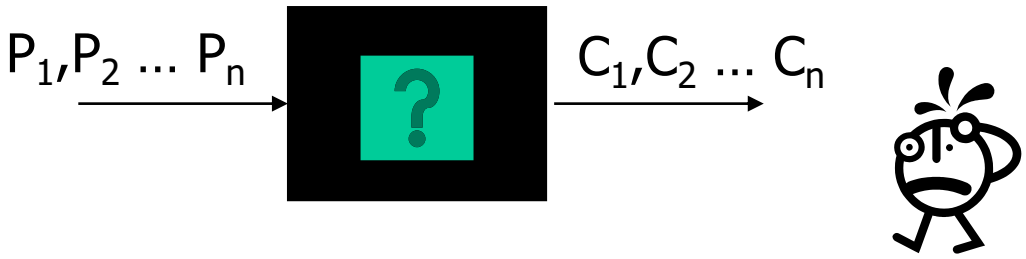
- Adaptive Chosen Plaintext – Adaptive Chosen Ciphertext:
  - Input a P_i receive C_i or input C_i receive P_i then choose next query

Attack Categories – Other

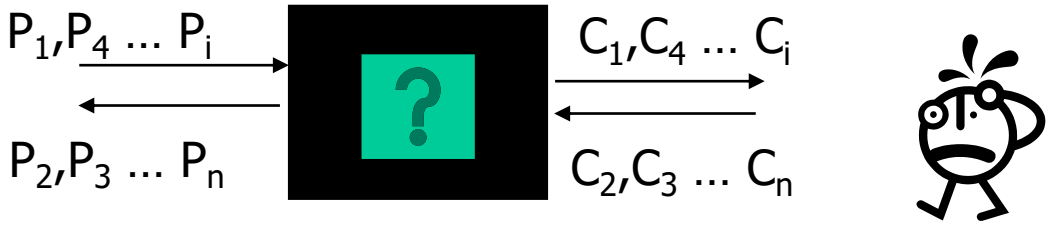
related keys – adversary chooses relation between keys, but not keys themselves, and obtains plaintext, ciphertext pairs

Recall PRP, SPRP

- Box contains either the block cipher or a random permutation
- Pseudorandom permutation (PRP): Attacker cannot make polynomial many adaptive chosen plaintext or adaptive chosen ciphertext queries (but not both) and determine contents of box with probability ½ + *e* for non-negligible *e* > 0.



- Strong PRP (SPRP): same idea as PRP, but can make queries in both directions



Attack Bounds

- If an attack holds with probability $\leq 2^{-x}$
- $x > 0$
- Block size b
- If $x \geq b$, need $\geq 2^{b}$ plaintexts

Agenda

- Overview
- **Block Ciphers:**
  - **Linear**
  - Differential
  - Other Attacks
  - Statistical Analysis
- Stream Ciphers
- General
  - Side Channel Attacks

Linear Cryptanalysis

Notation

P = plaintext

p_i = i^{th} bit of P

C = Ciphertext

c_i = i^{th} bit of C

K = Key (initial or expanded)

k_i = i^{th} bit of K

$\oplus_{i=1,n}\ p_i = p_1 \oplus p_2 \oplus \quad \ldots. \oplus p_n$

X,Y,Z are subsets of bits (notation on next slide only)

Linear Cryptanalysis

Attack Overview

Obtain linear approximation(s) of the cipher relating P,K,C

$$\oplus_{i \in X,} \; \mathbf{p_i} \oplus_{j \in Y} \mathbf{c_j} = \oplus_{g \in Z} \mathbf{k_g}$$

which occur with probability pr = ½ + e for max bias $-½ \leq e_i \leq ½$.

Encrypt random P's to obtain C's and compute k_g's.

Known plaintext attack

Guess remaining key bits via exhaustive search.

Example – Single S-Box

P_2P_1 \ K_2K_1	00	01	10	11
00	10	11	00	01
01	11	00	01	10
10	00	01	10	11
11	01	10	11	00

(P,C) pairs

(a) $00 \rightarrow 00$

(b) $01 \rightarrow 01$

(c) $10 \rightarrow 10$

In each pair

$P_1 \oplus C_1 = 0$

$P_2 \oplus C_2 = 0$

Considering only relationships between 1 input bit,1 output bit and 1 key bit:

(1) $\Pr(P_1 \oplus C_1 = K_1) = 1$

(2) $\Pr(P_2 \oplus C_2 = K_1) = 5/8$

(3) $\Pr(P_2 \oplus C_2 = K_2) = 3/8$

For all other triples of P_i, C_i, K_i

$\Pr(P_i \oplus C_i = K_i) = ½$

Use (1) and (3) to determine the key.

Can determine K_1 from one (P,C) by (1)

$P_1 \oplus C_1 = 0 = K_1$

One $P_2 \oplus C_2 = 0$ is not enough to infer K_2 is 1

Additional (P,C)'s needed

(3) returns 0, implying K_2 is 1.

Guess key = 10

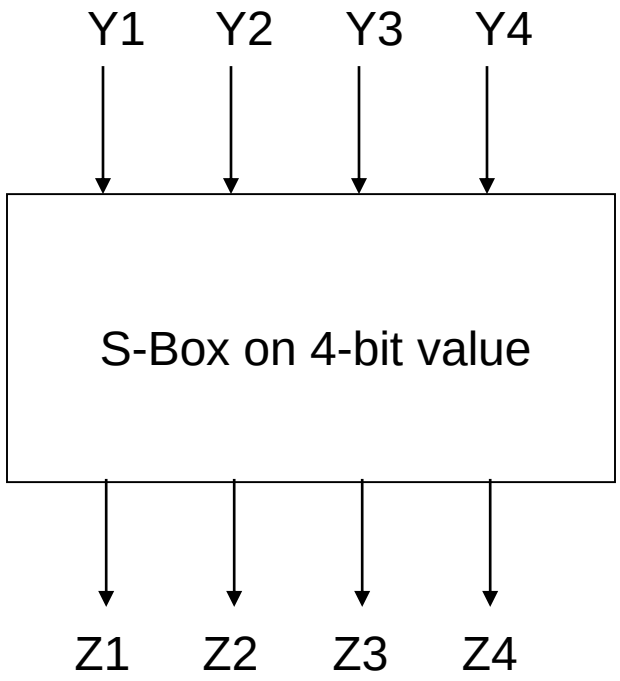
Example S-Box

Input:Output (4 bits, in hex)

0:E
1:4
2:D
3:1
4:2
5:F
6:B
7:8
8:3
9:A
A:6
B:C
C:5
D:9
E:0
F:7

S-Box Example from Tutorial on Linear and Differential Crypt. Tutorial, H. Heys, Memorial U. of of Newfoundland

Example S-Box



Y2 ⊕ Y3 = Z1 ⊕ Z3 ⊕ Z4 in 12 of the 16 input, output pairs
12/16 = ½ + ¼ and the bias is ¼
Y1 ⊕ Y4 = Z2 in ½ of the pairs, so there is no bias
Y3 ⊕ Y4 = Z1 ⊕ Z4 in 2 of the 16 pairs, so the bias is -3/8
2/16 = ½ -3/8

Finding Linear Relationships

General form of linear relationship:

$a1Y1 \oplus a2Y2 \oplus a3Y3 \oplus a4Y4$

$=$

$b1Z1 \oplus b2Z2 \oplus b3Z3 \oplus b4\ Z4$

$ai, bi \in \{0,1\}$

Summarize all equations in a table

Only need to do once – upfront work

Finding Linear Relationships

b1b2b3b4

a1a2a3a4	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	-2	-2	0	0	-2	6	2	2	0	0	2	2	0	0
2	0	0	-2	-2	0	0	-2	-2	0	0	2	2	0	0	-6	2
3	0	0	0	0	0	0	0	0	2	-6	-2	-2	2	2	-2	-2
4	0	2	0	-2	-2	-4	-2	0	0	-2	0	2	2	-4	2	0
5	0	-2	-2	0	-2	0	4	2	-2	0	4	-2	0	-2	-2	0
6	0	2	-2	4	2	0	0	2	0	-2	2	4	-2	0	0	-2
7	0	-2	0	2	2	-4	2	0	-2	0	2	0	4	2	0	2
8	0	0	0	0	0	0	0	0	-2	2	2	-2	2	-2	-2	6
9	0	0	-2	-2	0	0	-2	-2	-4	0	-2	2	0	4	2	-2
A	0	4	-2	2	-4	0	2	-2	2	2	0	0	2	2	0	0
B	0	4	0	-4	4	0	4	0	0	0	0	0	0	0	0	0
C	0	-2	4	-2	-2	0	2	0	2	0	2	4	0	2	0	2
D	0	2	2	0	-2	4	0	2	-4	-2	2	0	2	0	0	2
E	0	2	2	0	-2	-4	0	2	-2	0	0	-2	-4	2	-2	0
F	0	-2	4	-2	-2	0	2	0	0	-2	4	-2	-2	0	2	0

# of times equation holds: a1Y1 ⊕ a2Y2 ⊕ a3Y3 ⊕ a4Y4 = b1Z1 ⊕ b2Z2 ⊕ b3Z3 ⊕ b4 Z4

Finding Linear Relationships

- “a” value of E: a1 =1, a2 = 1, a3 = 1, a4 = 0
- “b” value of 1: b1 = 0, b2 = 0, b3 = 0, b4 = 1
- Row E, Column 1 has a value of 2
- Bias is 2/16 = 1/8
- Probability X1⊕ X2 ⊕ X3 = Y4 is ½ + 1/8 = 5/8

Piling-Up Lemma

Matsui

- Know $\Pr(V_i = 0) = \frac{1}{2} + e_i$
- $\Pr(V_1 \oplus V_2 \oplus \ldots \oplus V_n = 0) = \frac{1}{2} + 2^{n-1} \prod_{i=1}^{n} e_i$
- V_i's are independent random variables
- e_i is the bias $-\frac{1}{2} \leq e_i \leq \frac{1}{2}$

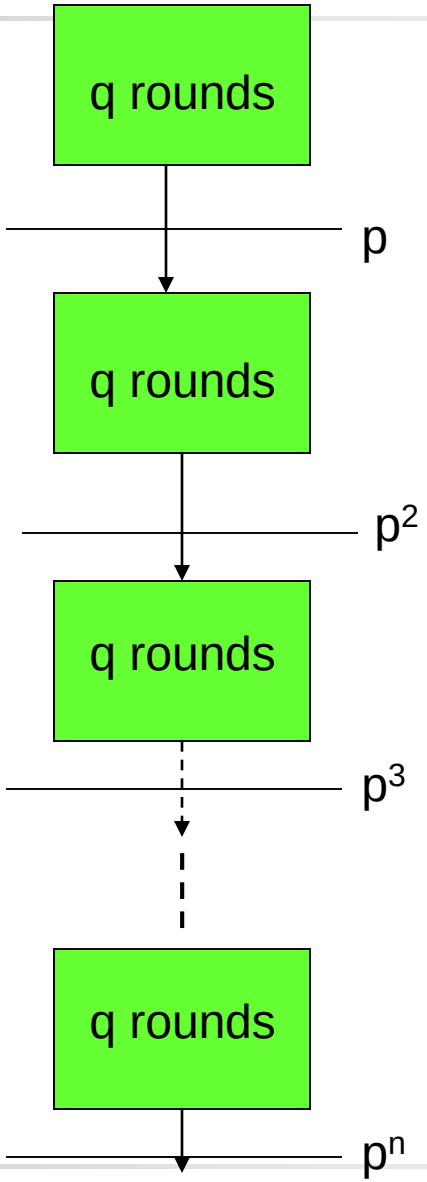
Use to combine linear equations if view each as independent random variable

Finding Linear Relationships

- Apply same process used for S-Box to other steps within the round function
- Determine equations for entire round
- Incorporate whitening (if any) into equations

Linear Bounds

- Bound a linear equation holds across q rounds: $0 < p \leq 1$
- Cipher has nq rounds
- Estimate upper bound $\leq p^n$
- 2^b possible plaintexts
- $\leq 2^b/p^n$ satisfy equations
- Round key bits, output of a round/input to next round not independent
- If $p^n \leq 2^{-b}$,, no attack



Applying an Attack

When attacking the cipher, try to determine key bits for first or last round, then repeat attack on reduced round version of the cipher

DES has 16 rounds, find round key for 1st or last round, repeat attack for 15 round version …

If same expanded key bits used in multiple rounds, fill in round key bits as they become known

Linear Cryptanalysis DES

- Determined linear approximations via exhaustive search
  - First for S-Boxes
  - Then extended to round function and multiple rounds.
- Approximations
  - 5 good approximations for initial key bits with bias *e* ranging from ≈ 0.031 to 0.218
  - Examples,
    - 1st round: $\oplus_{i \in X}\ fo_{i,1} \oplus p_{15} = k_{22}$ X = {7,18,24,29} with probability 19%
    - Last round: $\oplus_{i \in X}\ fo_{i,16} \oplus fin_{15,16} = k_{22}$ X = {7,18,24} with probability 66%
  - 1 approximation for round key bits with $e = O(2^{-3})$.
  - Others with $e = O(2^{-5})$ to $O(2^{-30})$

fin_{ij} = i^{th} bit of input of round function in j^{th} round
fo_{ij} = i^{th} bit of output of round function in j^{th} round

Linear Cryptanalysis DES

- Plaintext Attack
  - Found 14 key bits.
  - Remaining 42 key bits found by exhaustive search.
  - 8 rounds required 2^{21} P's with 96% success.
  - 16 rounds required 2^{47} P's with 96% success
- Ciphertext Only Attack
  - Found 7 key bits.
  - Assumed some p_is were 0 to have equations of C, K only.
  - 8 rounds required 2^{37} C's with 78% success, assumed 1 p_i is 0
  - 16 rounds required 1.82×2^{53} C's with 78% success, assumed 5 p_i's are 0.

Linear Bounds AES

- 4 rounds $\leq 2^{-75}$
- 8 rounds $\leq 2^{-150}$ exponent > 128 so don't need to estimate all 10 rounds

Agenda

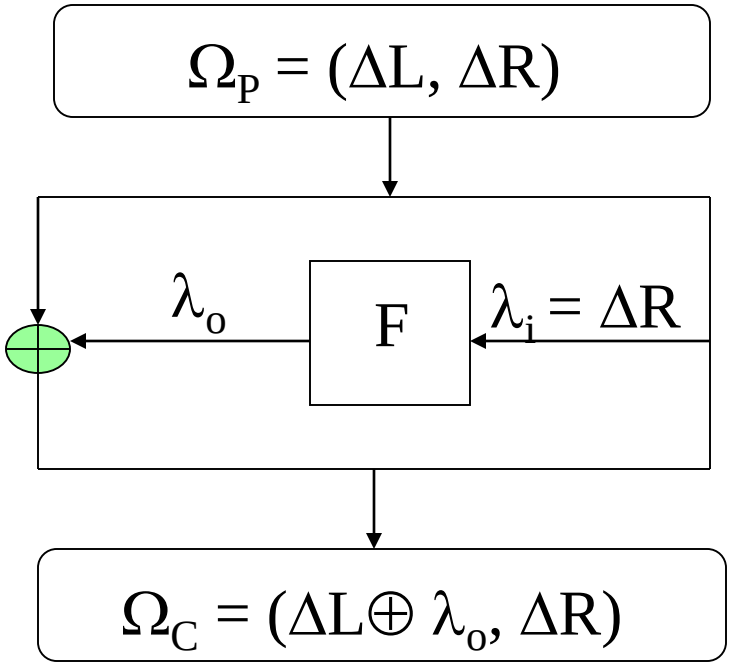
- Overview
- **Block Ciphers:**
  - Linear
  - **Differential**
  - Other Attacks
  - Statistical Analysis
- Stream Ciphers
- General
  - Side Channel Attacks

Differential Cryptanalysis

Notation

- P = plaintext
- C = ciphertext
- (P1,P2) = plaintext pair
- (C1,C2) = ciphertext pair
- $\Delta P = P1 \oplus P2$
- $\Delta C = C1 \oplus C2$
- Characteristic: $\Omega = (\lambda_{i1},\lambda_{o1},\lambda_{i2},\lambda_{o2},\ldots.\lambda_{ir},\lambda_{or})$
  - λ_{ij}= $\oplus$ of inputs to round j
  - λ_{oj}= $\oplus$ of outputs from round j
  - If pr_j= probability λ_{oj} occurs given λ_{ij}
  - then probability of $\Omega = \Pi\ pr_j$'s (upper bound)

Example: 1 round Ω's



If $\Delta R = 0$ then

$\lambda_o = 0$

$\Omega_C = (\Delta L, 0)$

with probability 1.

First round of any Feistel network does not assist in preventing differential crypt.

If ΔR = 60 00 00 00 then

λ_o = 00 80 82 00

$\Omega_C = (\Delta L \oplus$ 00 08 82 00,
60 00 00 00)

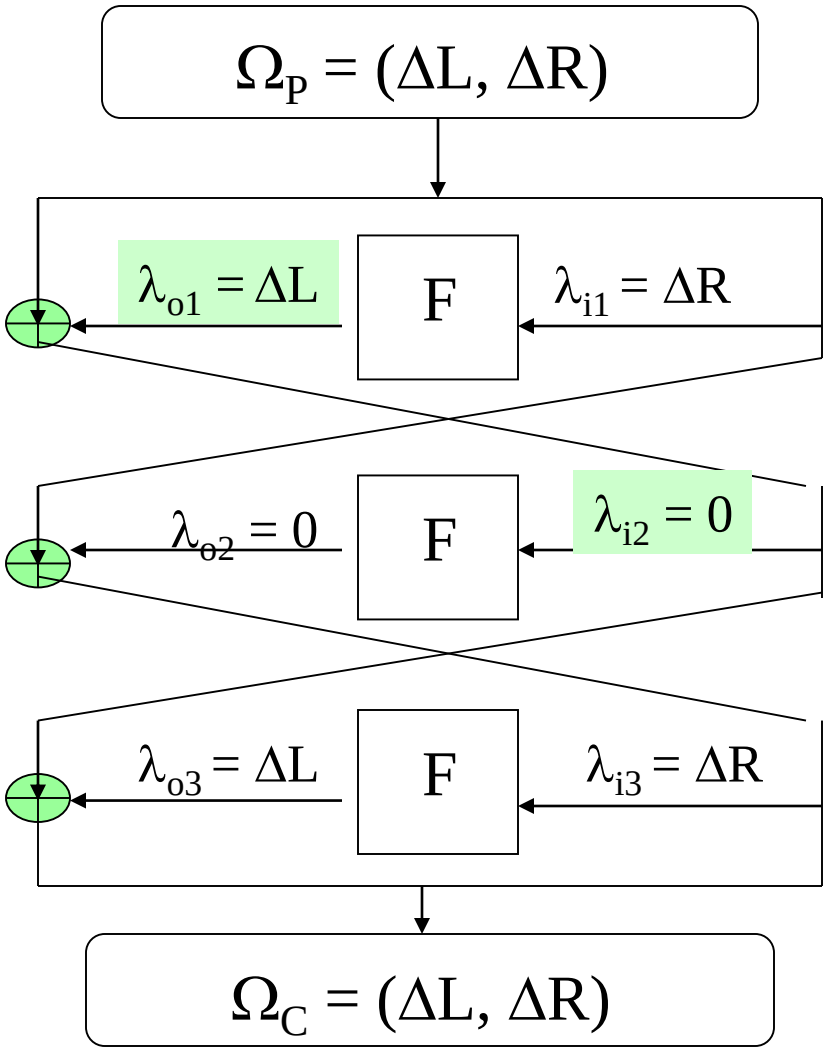
with probability 14/64.

DES without initial and final permutations.

Finding Characteristics

- Process similar to that used in linear crypt example
- Enumerate all cases
- Only need to do once – one time upfront work

Differential Cryptanalysis - DES



3 round Ω with $\Delta P = \Delta C$

Probability $(14/64)^2 \approx 0.048$

14/64

Want output of first F to cancel ΔL

1

14/64 Same Δ as input to first F

Differential Cryptanalysis

Attack Overview

- Find Ω with non-negligible probability.
  - Minimal key bits to guess, but allow guessing those in last (or first) round.
  - Exhaustive search to find best Ω's.
- Determine key bits of last round:
  - Choose pairs (P1,P2) such that ΔP provides λ_{i1} .
  - Decrypt ciphertext with key guess for last round
  - Count # of (C1,C2) pairs such that match characterstic
  - Assume correct key bits is guess with highest count.
  - Eliminate last round and attack the reduced cipher.
- Can also work from 1st round:
  - Choose pairs (C1,C2) such that $\Delta C = \lambda_{or}$
  - Determine key bits in 1st round.

Finding Ω's

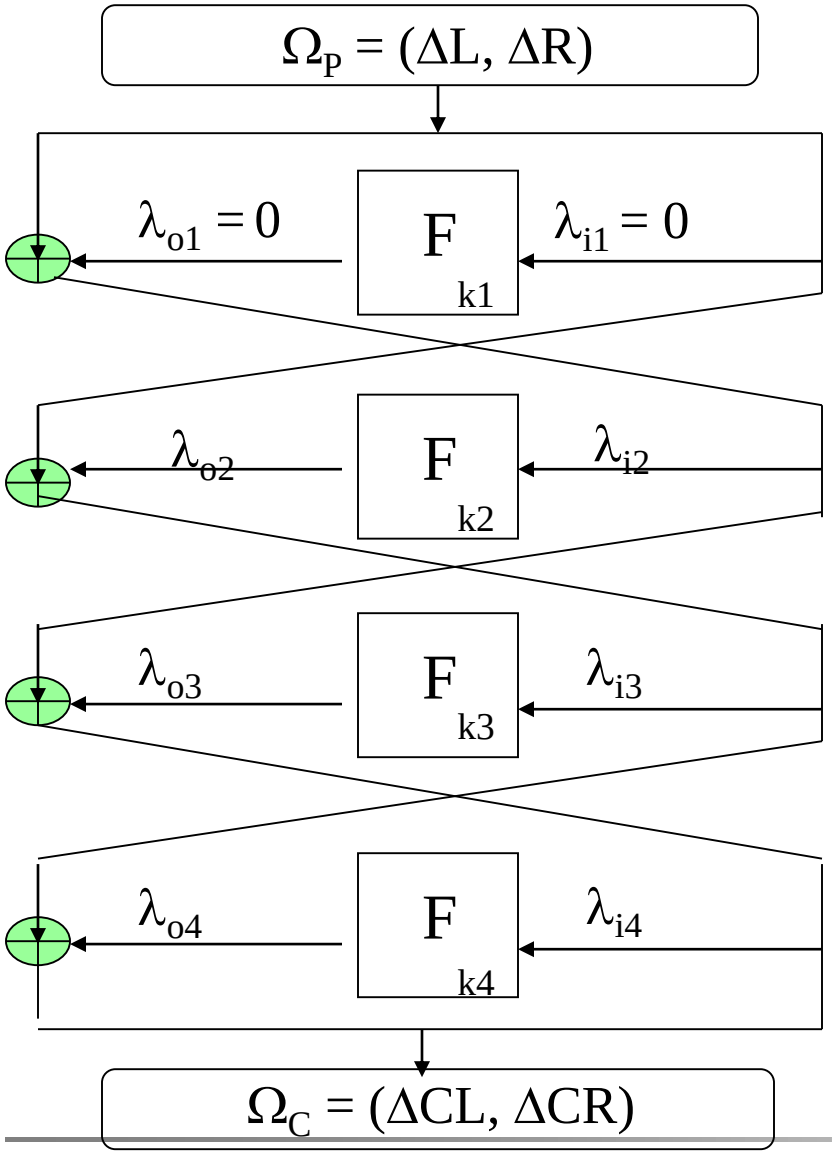
Manually created distribution tables for input ⊕'s and output ⊕'s for each S-Box.

	Output ⊕					
Input ⊕	…	2	3	4	5	…
2	…	0	8	0	4	…
3	…	2	2	10	6	…

Segment of distribution table for DES S-Box 0

If input ⊕ is 2, output ⊕ is 5, 4 possible keys.

Differential Cryptanalysis - DES



4 round Ω

Ω_P with

$\Delta L = 20\ 00\ 00\ 00$

$\Delta R = 00\ 00\ 00\ 00$

Then

$\lambda_{o1} = 00\ 00\ 00\ 00$

$\lambda_{i2} = \Delta L = 20\ 00\ 00\ 00$

λ_{i2} affect only 1st S-Box so 28 bits of λ_{o2} are 0.

$\lambda_{o4} = \lambda_{i3} \oplus \Delta CL$

$= \lambda_{i1} \oplus \lambda_{o2} \oplus \Delta CL$

$= \lambda_{o2} \oplus \Delta CL$

know all but 4 bits of λ_{o2}

Know right halves of ciphertexts,

$\Rightarrow$ know inputs into 4th round.

λ_{i4} : at most 11 non zero bits

ΔCR varies amongst pairs.

Differential Cryptanalysis

Number of Plaintexts

Use $m = c/pr(\Omega)$ plaintext pairs, for some small $c > 0$.

Chosen Plaintext: Select m pairs that satisfy ΔP.

Known Plaintext: have set of P's, but did not choose them, so need to find pairs satisfying ΔP.

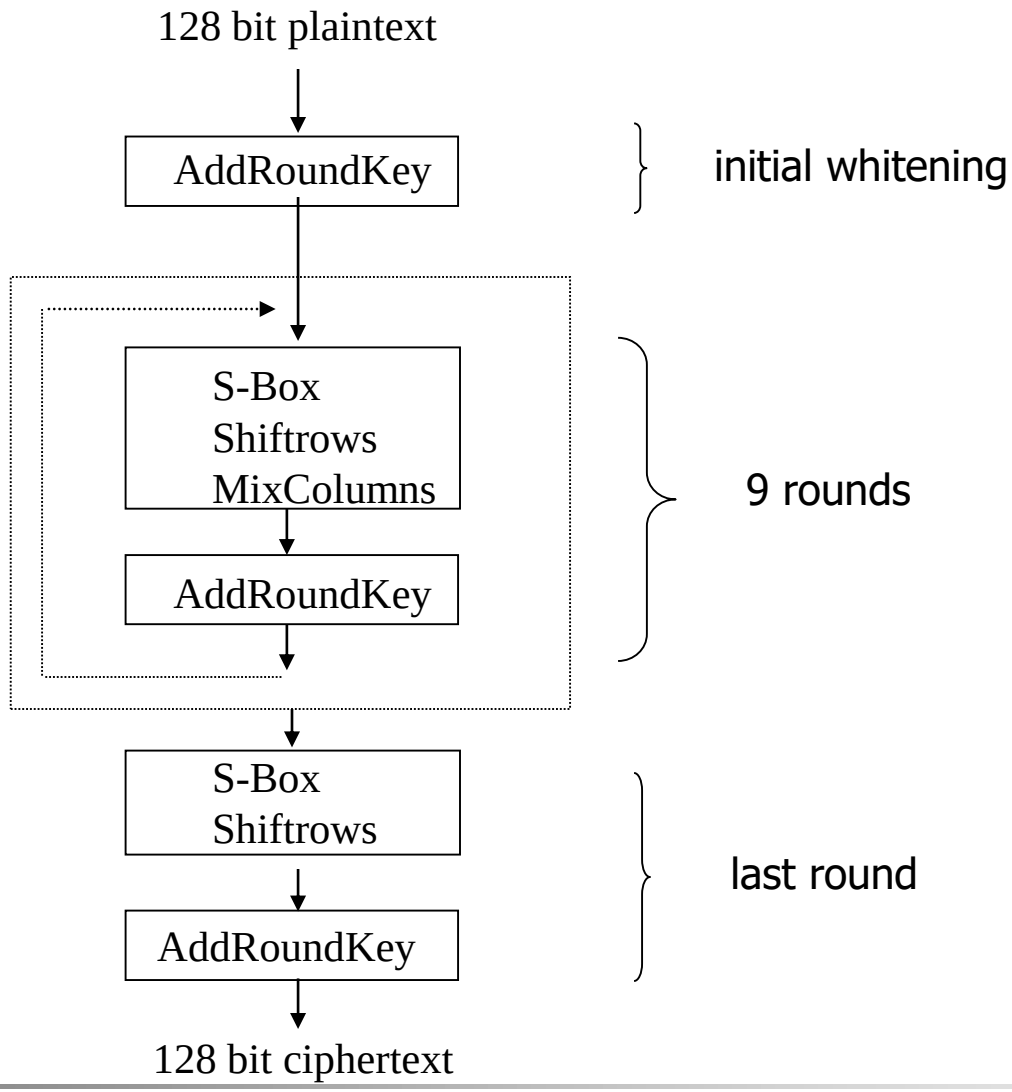
- $2^{|P|/2}(2m)^{½}$ plaintexts required
  - Can form ½ $(2^{|P|/2}(2m)^{½})^2 = 2^{|P|}m$ pairs.
  - $2^{|P|}$ possible ΔP's.
  - $2^{|P|}m / 2^{|P|} = m$ pairs on average create each ΔP.
- If > # of possible P's, attack not possible.

Differential Cryptanalysis - DES

Any reduced round version of DES is breakable via a known plaintext attack faster than via exhaustive key search.

# Rounds	# Chosen Plaintexts	# Known Plaintexts
4	2^3	2^{33}
6	2^8	2^{36}
8	2^{14}	2^{38}
9	2^{24}	2^{44}
11	2^{31}	2^{47}
13	2^{39}	2^{52}
16	2^{47}	2^{55}

AES – 128 bit block



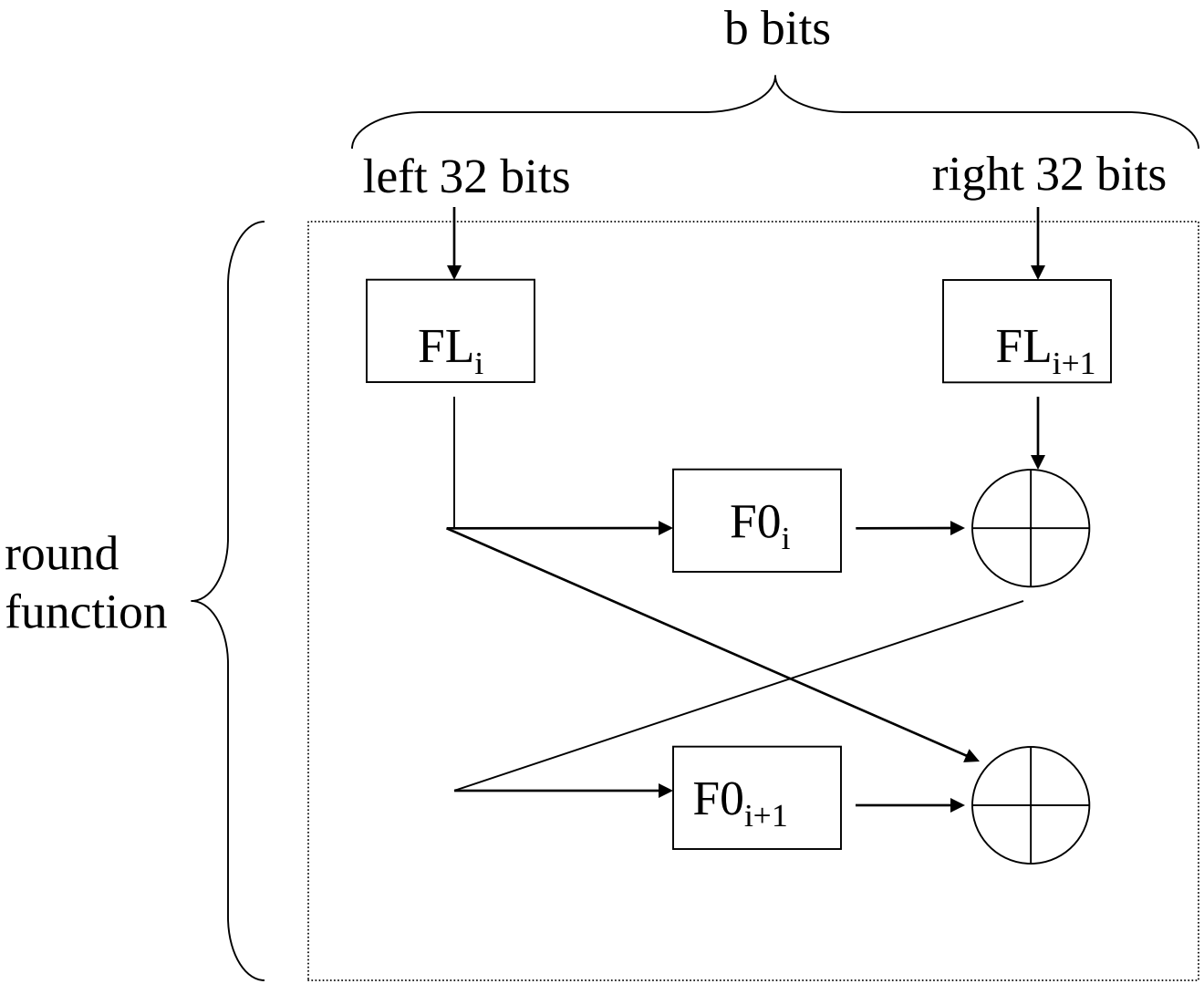
AES Differentials

- AES: each non-zero byte in delta input to a round contributes 2^{-6} or 2^{-7} to probability of output difference.
  - If difference input to a round is 0 except in one byte, probability specific difference occurs in output of the round is $\leq 2^{-6}$
  - If difference input to a round is 0 except in two bytes, probability specific difference occurs in output of the round is $\leq 2^{-12}$
- Entirely due to the S-Box – other steps in round do not impact differential probability

AES Differentials

- 2 round bound: $\leq 2^{-24}$
- 4 round bound: $\leq 2^{-96}$ small enough to eliminate differential attack over 10 rounds

MISTY1 Round



MISTY1

- Each application of the F0 function contributes $\leq 2^{-7}$ to the probability
- So if non-zero difference into exactly one application of the F0 function in a round, the probability a specific difference occurs in the round's output is $\leq 2^{-7}$
- So if non-zero difference into exactly one application of the F0 function in a round, the probability a specific difference occurs in the round's output is $\leq 2^{-14}$
- At least one F0 function in a round must have a non-zero input difference. Therefore, lose upper bound on a differential is 2^{-56} (2^{-7} over each of 8 rounds).

Agenda

- Overview
- **Block Ciphers:**
  - Linear
  - Differential
  - **Other Attacks**
  - Statistical Analysis
- Stream Ciphers
- General
  - Side Channel Attacks

Differential Variations

- Impossible Differential
  - Differential characteristic occurs with probability 0
  - Eliminate values for key bits
- Partial Differential
  - Block size b bits, consider differential in < b bits
- Higher Order Differentials
- Boomerang Attack and variations

Boomerang Attack

- P,P'.Q,Q' are plaintexts
- C,C',D,D' are the corresponding ciphertexts
- Cipher is a series of rounds
- E = encryption function
- View E as a composition of two functions E_0,E_1
  - for example, if E consists of n rounds, E_0 is the first n_0 rounds, E_1 is the remaining $n-n_0$ rounds
  - $E(P) = E_1(E_0(P))$

Boomerang Attack

- Characteristic for E_0 : $\Delta \rightarrow \Delta^*$
- Characteristic for E_1^{-1}: $\nabla \rightarrow \nabla^*$

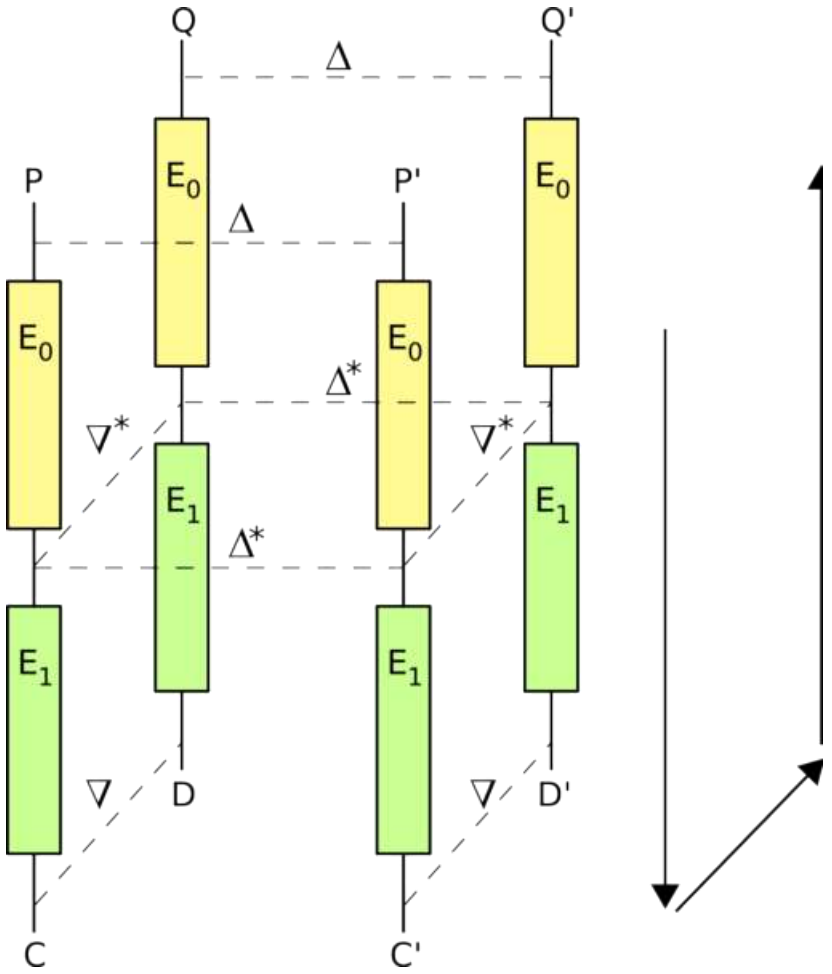
Want to choose plaintexts such that

- $P \oplus P'$ produces $\Delta \rightarrow \Delta^*$
- $P \oplus Q$ produces $\nabla \rightarrow \nabla^*$
- $P' \oplus Q'$ produces $\nabla \rightarrow \nabla^*$

Then show

- $D \oplus D'$, $Q \oplus Q'$ corresponds to $\Delta^* \rightarrow \Delta$ for E_0^{-1}

Bommerang Attack



Boomerang Attack

$E_0(Q) \oplus E_0(Q')$

$= E_0(Q) \oplus E_0(Q') \oplus E_0(P) \oplus E_0(P) \oplus E_0(P') \oplus E_0(P')$

$= [E_0(P) \oplus E_0(P')] \oplus [E_0(P) \oplus E_0(Q)] \oplus [E_0(P') \oplus E_0(Q')]$

$= [E_0(P) \oplus E_0(P')] \oplus [E_1^{-1}(C) \oplus E_1^{-1}(D)] \oplus [E_1^{-1}(C') \oplus E_1^{-1}(D')]$

$= \Delta^* \oplus \nabla^* \oplus \nabla^*$

$= \Delta^*$

Boomerang Attack

Find characteristic that holds for E_0 and one that holds for E_1

Generate pairs using chosen plaintext –chosen ciphertext queries:

- $P' = P \oplus \Delta$
- Request P,P' be encrypted to get C, C'
- $D = C \oplus \nabla$
- $D' = C' \oplus \nabla$
- Request D, D' be decrypted to get Q,Q'

Key Schedules

- Designed to be efficient
  - Rekeying (example network applications handling multiple data streams)
  - Key (not expanded key) may be stored by application or entered each time cipher is applied – cost of key expansion incurred
- Tradeoff – complete lack of randomness in expanded key bits

Key Schedules

Assistance in guessing key bits in any attack

- AES: expanded key bits that are XOR of two other bits
- MISTY1, Camellia: same expanded key bit used in multiple locations
- RC6 : more difficult – no obvious equation relating expanded key bits

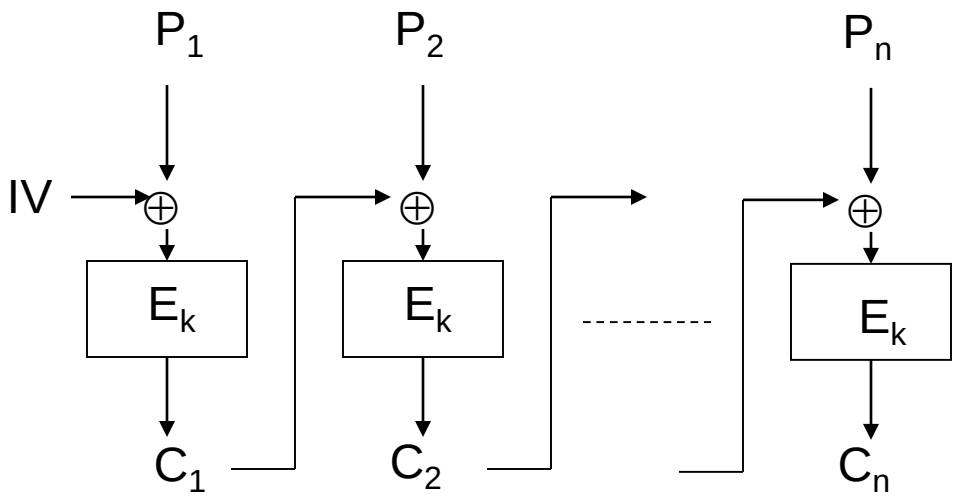
Related Keys

- Attacker specifies relationship between two keys, but not actual keys
- Get plaintext, ciphertext pairs for each key
- Try to determine round keys
- Example: Slide attack
- AES can have two keys K1, K2 such that K2 is K1 slid one round. i.e. expanded key bits of round 1 when using K1 = those for round 2 in expanded key bits of K2
  - S-box and XOR with a constant step prevents “sliding” more than one round

Other Attacks

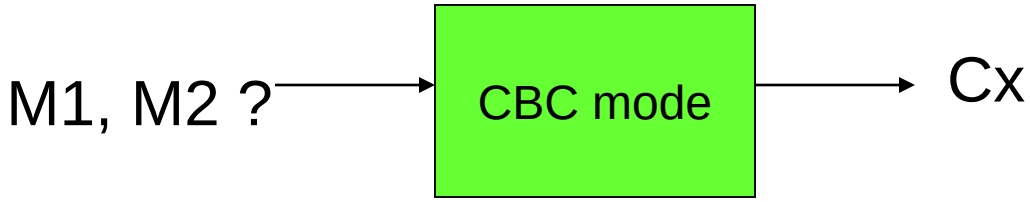
- Blockwise Adaptive Attack
- Non-linear (algebraic) Cryptanalysis
- Square Attack – named for attack on block cipher Square – a predecessor to Rijndael

Reminder: CBC Mode



Blockwise Adaptive

- Consider a block cipher and CBC mode
- Environment where see ciphertext from plaintext block i before having to input plaintext block i+1
- M1,M2,M3 are three distinct 2b-bit plaintexts.
- Know one of M1 and M2 was encrypted. Ciphertext, Cx



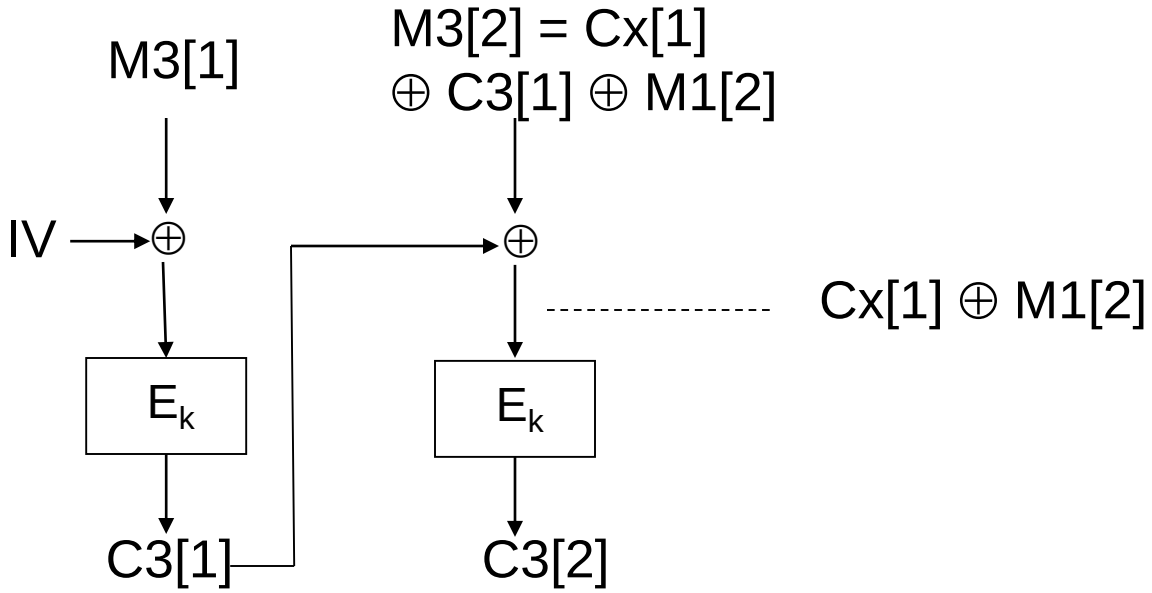
- Can form M3 to determine if it is M1 or M2.

Blockwise Adaptive

- M3: for first block send an arbitrary b-bit bits, receive the ciphertext, C3[1]
- Generate the next b bits of M3 by XORing the first block from Cx, C3[1] and M1[2]

Notation: X[i] = i^{th} block of X

Blockwise Adaptive



C3[2] = Cx[2] if Cx is the encryption of M1
C3[2] ≠ Cx[2] if Cx is the encryption of M2.

Side Channel Analysis

- Differential Fault Analysis – late 1990's
- Timing Analysis – late 1990's
- Power Analysis – late 1990's
- Memory Access – 2005

- Applied to Public and Private Key Algorithms
  - Public key cipher: attempt to learn non-public parameters
  - Block ciphers: attempt to learn input/output of internal rounds and/or expanded key bits.

Agenda

- Overview
- **Block Ciphers:**
  - Linear
  - Differential
  - Other Attacks
  - **Statistical Analysis**
- Stream Ciphers
- General
  - Side Channel Attacks

Statistical Tests

- Sixteen tests performed on eight sets of data for each cipher.
  - **Do not prove cipher is secure**
  - **Failing a test indicates a weakness**
  - NIST AES competition finalists: > 96.33% of cases passing
- What if cipher fails a test?
  - Some relationship between P,C,K – but don't know exactly what
  - Example, key with a 1 in bit j may be prone to produce ciphertext with more 0's than 1's.

Statistical Tests

- **Frequency (Monobit):** are proportions of 0's and 1's in the bit sequence close enough to ½ .
- **Frequency within a Block:** Frequency test applied to fixed-sized blocks within the bit sequence.
- **Runs:** The number of runs (sequence of all 0's or all 1's) in the bit sequence is determined.
- **Longest Run of Ones within a Block:** The longest run of 1's within a block is determined.
- **Binary Matrix Rank:** 32-by-32 matrices are created from the bit sequence and their ranks computed. Determines if any linear dependence among fixed-length segments of bits within the sequence.
- **Discrete Fourier Transform:** determines if there are repetitive patterns in the bit sequence.
- **Non-overlapping Template Matching:** counts the number of times a m-bit pattern occurs in the bit sequence using a sliding window. The window slides 1 bit when no match and slides m bits when a match occurs so a bit will be involved in at most one match for a given pattern. Ex. m = 9
- **Overlapping Template Matching:** same as the previous test except that the window always slides 1 bit.

Statistical Tests

- **Maurer's Universal Statistical:** determines if the bit sequence can be compressed based on the number of bits between occurrences of a pattern.
- **Lempel-Ziv Compression:** determines how much a bit sequence can be compressed based on the number of distinct patterns.
- **Linear Complexity:** Berlekamp-Massey algorithm is applied to a 1000 bit sequence to determine a linear feedback shift register that produces the sequence. The length of the LFRS indicates if the sequence is sufficiently random.
- **Serial:** The number of times each 2^m bit pattern occurs is determined, for some integer m.
- **Approximate Entropy:** The number of times each 2^m and each 2^(m+1) bit pattern is determined, for some integer m.
- **Cumulative Sums:** cumulative sum of the bits is computed for each position in the sequence. The sum is computed by adding -1 for each bit that is 0 and adding 1 for each bit that is 1.
- **Random Excursions:** number of times the cumulative sum crosses zero is determined.
- **Random Excursions Variant:** number of times the cumulative sum is a particular value is determined.

Data Sets

- **Plaintext Avalanche:** key is fixed random value. Random plaintexts. The data tested is the XOR of the encrypted plaintext and the encryption of the plaintext with the ith bit flipped. This is repeated for i = 1 to b+y and for all plaintexts.
- **Key Avalanche:** plaintext of all zeroes. Random keys. The data tested is the XOR of the plaintext encrypted with a random key and the plaintext encrypted with the random key with the ith bit flipped. This is repeated for i= 1 to 128 and for all keys.
- **Plaintext-Ciphertext Correlation:** Random keys and random plaintexts. The data tested consisted of the ciphertext XORed with the plaintext, for all plaintexts and all keys.
- **High Density Keys:** same as the low density keys except keys of all 1's and keys with a single 0 are used instead of all 0's and a single 1.

Data Sets

- **CBC Mode:** Random keys, random plaintexts and an IV of all 0's. For each key, the plaintexts are encrypted using CBC mode.
- Low Density Plaintext: Random keys. For each key, a plaintext block of all 0's and every plaintext block containing exactly one 1 are encrypted.
- **Low Density Keys:** Random plaintext blocks. Each plaintext is encrypted with a key of all 0's and every key containing a single 1.
- **High Density Plaintext:** same as the low density plaintexts except plaintexts of all 1's and plaintexts with a single 0 are used instead of all 0's and a single 1.

Agenda

- **Overview**
- Block Ciphers:
  - Linear
  - Differential
  - Statistical Analysis
- **Stream Ciphers**
- General
  - Side Channel Attacks

Cryptanalysis

Single LSRF can easily be broken: Berlekamp-Massey algorithm

Correlation attack

- Keystream generator G consisting of a set of LFSRs and a nonlinear function
- Adversary knows G and some keystream segments
- Try to relate output bits to output of one or more LFSRs
- Exhaustive search over possible states of LFSRs in G
  - n LFSRs,
  - 2^{l_i} -1 possible initial states for ith LFSR
  - Π $(2^{l_i}$ -1) i = 1 to n combinations
- If each LSFR is correlated to the keystream;
  - Σ $(2^{l_i}$ -1) i = 1 to n combinations (guess 1st LSRF then hold constant, guess 2nd LSFR …)

Cryptanalysis

Information available to attacker, same idea as with block ciphers:

- Ciphertext only
- Plaintext, ciphertext pairs
  - Known plaintext from standard header information in network protocols, file formats
- Chosen, adaptive versions

Cryptanalysis

Distinguishing attacks

- Distinguish keystream from random bits
- Statistical tests
- Does not imply the cipher can be broken in practice

Side channel analysis

- Timing analysis
- Differential Fault
- Memory – are keystream bits and internal state available

Berlekamp-Massey Algorithm

Why LFSR alone is not sufficient

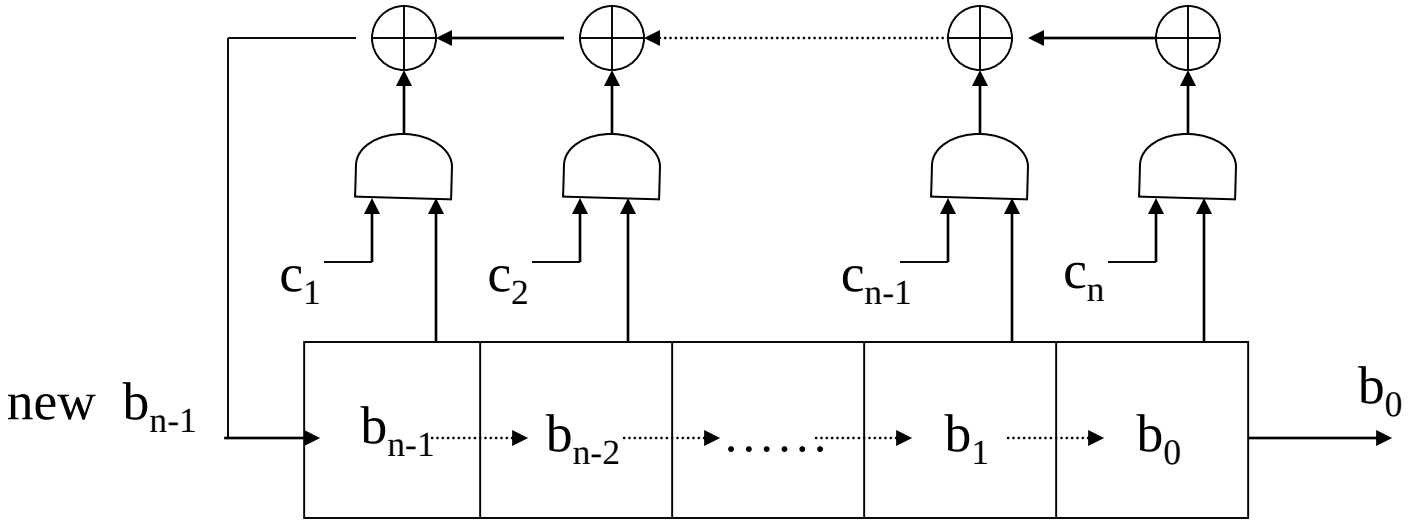
Given a bit sequence, $s^n = s_0 s_1 s_2 \ldots s_{n-1,}$, finds corresponding LFSR.

Initialize LFSR guess.

Walk through s^n, comparing to next output from LFSR.

- If $(N+1)^{st}$ term of LFSR = s_N ,LFSR generates s_N
- Else modify LFSR
- $O(n^2)$ work

LFRS Polynomial Representation



$$1 + c_1x + c_2x^2 + \ldots\ldots\ldots\ldots c_{n-1}x^{n-1} + c_nx^n$$

use b_{n-j} as x^j value

$$\text{new } b_{n-1} = c_1b_{n-1} + c_2b_{n-2} + \ldots\ldots c_{n-1}b_1 + c_nb_0$$

Berlekamp-Massey Algorithm

Input: $s^n = s_0 s_1 s_2 \ldots s_{n-1}$
C(x) = 1; L = 0; m = -1; B(x) = 1; N = 0; // initialize
while (N < n) {
    d = $(s_N + \sum_{i=1,L} c_i\, s_{N-i}) \bmod 2$; // next discrepancy

    if (d == 1) { // update LFSR
        T(x) = C(x); C(x) = C(x) + B(x)*x^{N-m};
        if L ≤ N/2 {
            L = N+1-L; m = N; B(x) = T(x);
        }
    }
    ++N;
}
return(L,C);

C(X) = polynomial representation of LFSR
c_i's = coefficients of C.
L = linear complexity of LFSR

Berlekamp-Massey Example

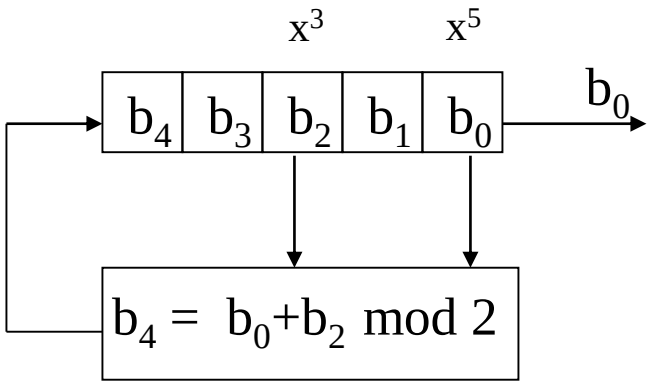
Given:

$s^n = 001101110$, n = 9

Output:

Polynomial: $1+x^3+x^5$

Can determine:



Initial state 00110
01100: 0 ⊕ 1, move in 1, output 0
10110: 0 ⊕ 1, move in 1, output 0
11011: 1 ⊕ 0, move in 1, output 1
11101: 1 ⊕ 1, move in 0, output 1
01110: 0 ⊕ 1, move in 1, output 0
10111: etc …

values at end of each while loop iteration

s_n	d	T(x)	C(x)	L	m	B(x)	N
-	-	-	1	0	-1	1	0
0	0	-	1	0	-1	1	1
0	0	-	1	0	-1	1	2
1	1	1	$1+x^3$	3	2	1	3
1	1	$1+x^3$	$1+x+x^3$	3	2	1	4
0	1	$1+x+x^3$	$1+x+x^2+x^3$	3	2	1	5
1	1	$1+x+x^2+x^3$	$1+x+x^2$	3	2	1	6
1	0	$1+x+x^2+x^3$	$1+x+x^2$	3	2	1	7
1	1	$1+x+x^2$	$1+x+x^2+x^5$	5	7	$1+x+x^2$	8
0	1	$1+x+x^2+x^5$	$1+x^3+x^5$	5	7	$1+x+x^2$	9

N = 4, L = 3: $C(x) = c_1s_2 + c_2s_1 + c_3s_0 = 1*1+0*0+1*0 = 1$

$s_4 = 0 \neq C(x)$, so set d = 1

Cryptanalysis

Attacks on non-FSR designs

- Depends on components
- Analyze function for relations between
  - Keystream and key or initial state
  - Keystream bits

Guessing subset of unknown bits used internally to determine state

Cellular Encryption

History of poor algorithm choice

A5/1

A5/3

Don’t create algorithms without understanding requirements and attacks

- Last October received an email from rep on standards committee: if we tweak A5, will it work, need answer in a day

A5/1

- Used in Global System for Mobil Communications (GSM)
- Example of a cipher manufacturers tried to keep secret, it was leaked and also reversed engineered within 5 years
- A5/2 – weaker cipher used in some countries due to export rules
- GSM phone conversations are sent as sequences of *frames*.
- One 228 bit frame is sent every 4.6 milliseconds: 114 bits for the communication in each direction.
- A5/1 produces 228 bits to XOR with the frame
- Initialized using a 64-bit key combined with a publicly-known 22-bit frame number.
- In some GSM implementations, 10 key bits are fixed at zero - effective key length is 54 bits.
- A5/1 is based around a combination of three LFSRs with irregular clocking.

A5/1

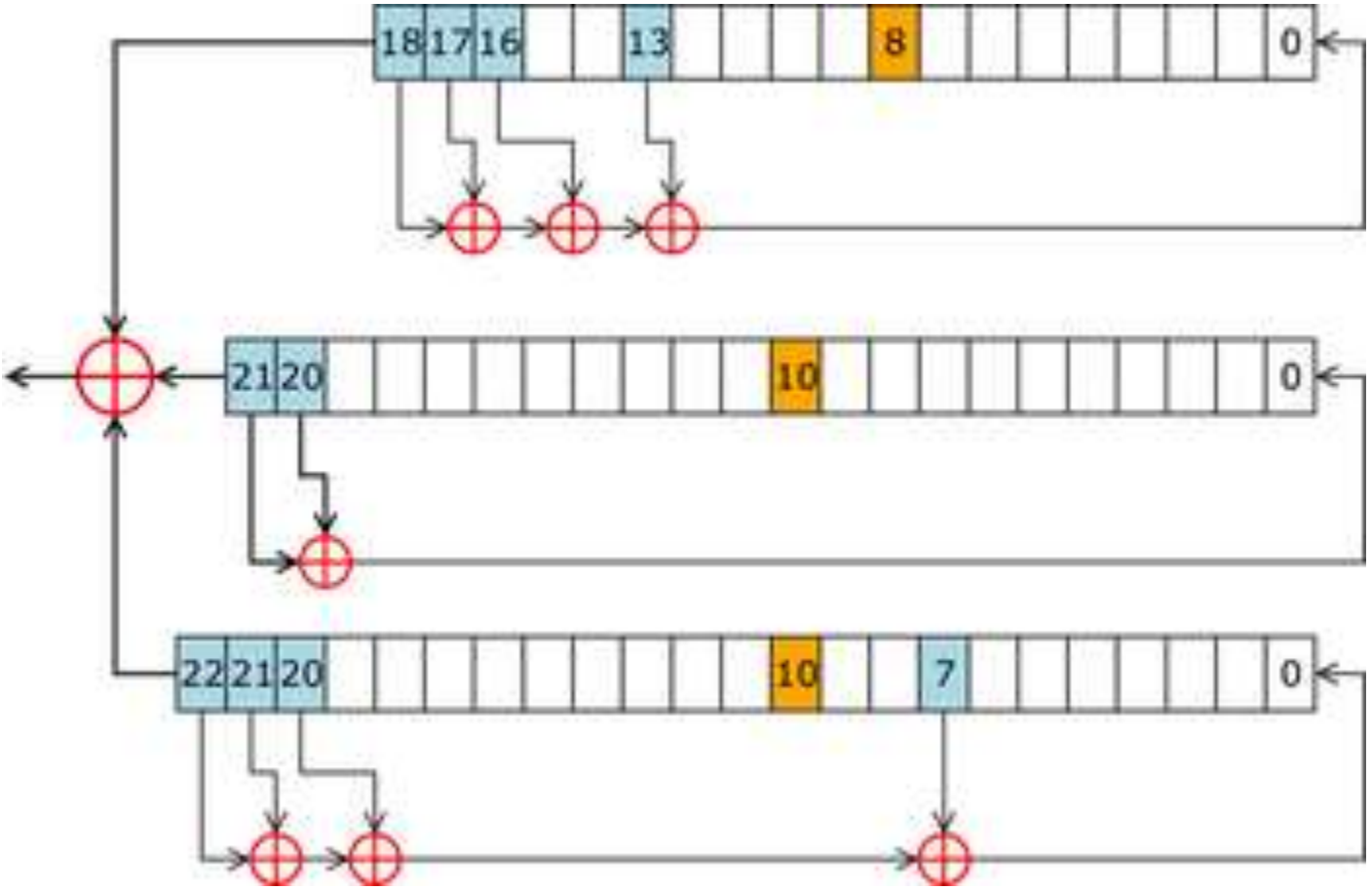


Image from Wikipedia

A5/1 LFSRs

- 19 bits
  - $x^{19} + x^5 + x^2 + x + 1$
  - clock bit 8
  - tapped bits: 13, 16, 17, 18
- 22 bits
  - $x^{22} + x + 1$
  - clock bit 10
  - tapped bits 20, 21
- 23 bits
  - $x^{23} + x^{15} + x^2 + x + 1$
  - clock bit 10
  - tapped bits 7, 20, 21, 22

- Least significant bit numbered 0
- Tapped bits of each LFSR are XORed to create value of next 0 bit.
- Output bits of the three LFSRs are XORed to form the keystream bit

A5/1

- Each cycle, look at the three clock bits. The majority value, c_m, is determined.
- In each LFSR, if the clock bit matches c_m, the registers are clocked.
- In each cycle, 2 or 3 LFSRs will be clocked.

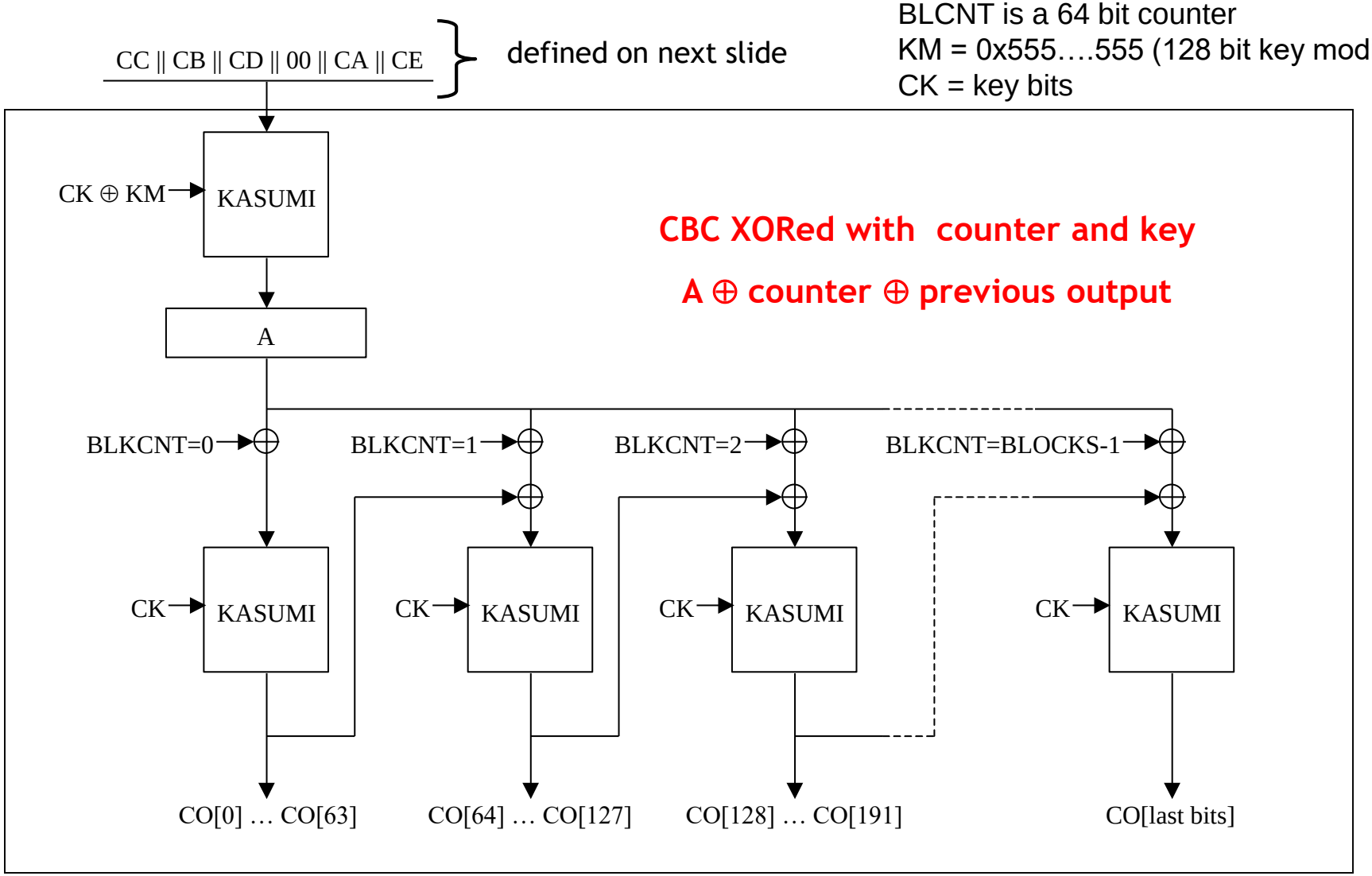
A5/1 Initialization

- Registers set to all 0's
- Incorporate the key and frame number:
  - For 64 cycles, the key is mixed in by XORing the ith key bit with the least significant bit of each register
  - For 22 cycles, the 22 bit frame value is mixed in – same as with key value
  - Normal clocking used
- 100 cycles are run using the majority clocking, the output is discarded
- End result is the initial state

A5/1

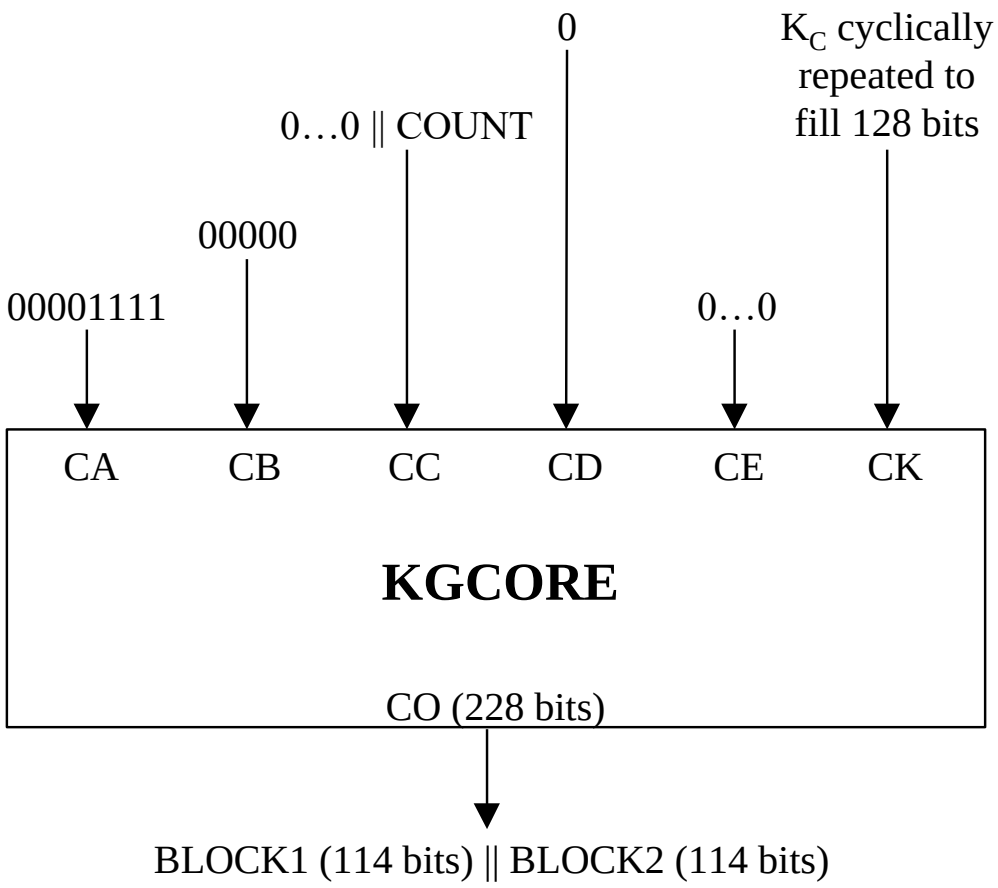
- Three short LFSRs
- Not many tap bits to guess

A5/3 Core


CC || CB || CD || 00 || CA || CE
defined on next slide
BLCNT is a 64 bit counter
KM = 0x555….555 (128 bit key modifier)
CK = key bits
CK ⊕ KM
KASUMI
A
CBC XORed with counter and key
A ⊕ counter ⊕ previous output
BLKCNT=0
BLKCNT=1
BLKCNT=2
BLKCNT=BLOCKS-1
CK
KASUMI
CO[0] … CO[63]
CO[64] … CO[127]
CO[128] … CO[191]
CO[last bits]

A5/3 GSM

K_c = key



http://www.gsmworld.com/using/algorithms/docs/a5_3_and_gea3_specifications.pdf

Agenda

- Overview
- Block Ciphers:
  - Linear
  - Differential
  - Statistical Analysis
- Stream Ciphers
- **General**
  - **Side Channel Attacks**

Side Channel Analysis

- Time
  - Does the number of CPU cycles depend on exact values used in the operation? ex. RSA exponent
  - Memory access – do exact values impact tables used, time to read from a table and/or number of memory accesses? ex. AES using tables of 32-bit values
- Acoustics
  - Impacted by operations or exact values used?
- Memory
  - Can intermediate values be read from memory by another process?

Timing – Toy Example

k: array of n key bytes

d: 16 byte data

Suppose encryption is a series of n rounds

n = 16;

d = plaintext;

```
for (i=0; i < n; ++i) {
   d = f(d,k[i]);  // do something to the data with k, but
                   // whose time does not depend on k
   d[i] = d[i] int(k[i]) mod 256; // alter one byte, time depends on k
}
```

Timing – Toy Example

What if use a table lookup instead?

table(a,b): function retrieves table a, entry b

```
d = plaintext;
x = 0;
for (i=0; i < n; ++i) {
  // do something to the data with k where time does not depend on k
     d = f(d,k[i]);
  // memory lookup - was table already in cache?
  // (k[i] same as a previous key byte)
     x= table(k[i], d[i]);
}
```

Timing and Power Analysis

- P. Kocher, Timing Attacks on Implementations of RSA, DH, DSS and Other Systems, Crypto 1996.
- A. Shamir and E. Tromer, Acoustic Cryptanalysis on Nosy People and Noisy Machines, 2004 presentation
- J. Kelsey, B. Schneier, D. Wagner and C. Hall, Side Channel Cryptanalysis of Product Ciphers. Journal of Computer Science, 8(2-3),pages 141-158, 2000. (DES, IDEA, RC5 used in examples)
- Companies, ex. Riscure, sell software for performing timing analysis on smart cards.

Differential Fault

- Induce faults into the device
- Observe outputs without the fault and with the fault
- Example: radiation
- Exact fault introduced likely to be unknown
- Assumes device can be tampered with – chips may be designed to stop working if tampered with, enclosures such as wire mesh
- Less practical then timing attacks
- Public Key Ciphers: Boneh, Denillo and Lipton, On the Importance of Checking Cryptographic Protocols for Faults. Eurocrypt 1997.
- Private Key Ciphers: Biham and Shamir, Differential Fault Analysis of Secret Key Cryptosystems, Technion CS Technical Report 1997.

Memory

- Process accessing same memory (cache) used by the cipher may obtain information
- Used to attack AES (specific OS, implementation)
- If attacker can perform the attack, there are greater security concerns about the system.
- Osvik, Shamir, Tromer, Cache Attacks and Countermeasures, the Case of AES. CT-RSA 2006.