Practical private key cryptography

Gurevich Lev

Saint Petersburg State University

31st March 2005

Outline ciphers

Outline ciphers

Outline DES

Outline DES

Outline attacks

5 Properties important for security

6 Types of attacks

- Exhausitive key search
- Differential analysis
- Linear analysis

Outline attacks

5. Properties important for security

6. Types of attacks
   - Exhausitive key search
   - Differential analysis
   - Linear analysis

Part I

Block ciphers

Concept

The main concept: Alice and Bob have given key K in common knowledge.

Alice enciphering message using K, and Bob deciphering message using K.

Noone who doesn't know K can read message

Concept

The main concept: Alice and Bob have given key K in common knowledge.
Alice enciphering message using K, and Bob deciphering message using K.
Noone who doesn't know K can read message

Concept

The main concept: Alice and Bob have given key K in common knowledge.
Alice enciphering message using K, and Bob deciphering message using K.
Noone who doesn't know K can read message

Formal definition

Definition

Block cipher is a function $E : \{0,1\}^k \times \{0,1\}^l \longrightarrow \{0,1\}^l$ that takes two inputs, a k-bit key K and l-bit "plaintext" M and returns l-bit "ciphertext" $C = E(K, M)$.

We can also define $E_K = E(K, M)$ function for each key $K \in \{0,1\}^k$. For each K it must be *permutation*. Let E_K^{-1} be an inverse permutation. And now define $E^{-1}(K, C) = E_K^{-1}(C)$.

Requirements

Requiments for block cipher

- Must be public fully specified algorithm
- Both E and E^{-1} should be easy computable
- Computation of key based on known plaintext-ciphertext pairs must be computationaly difficult

Requirements

Requiments for block cipher

- Must be public fully specified algorithm
- Both E and E^{-1} should be easy computable
- Computation of key based on known plaintext-ciphertext pairs must be computationaly difficult

Requirements

Requiments for block cipher

- Must be public fully specified algorithm
- Both E and E^{-1} should be easy computable
- Computation of key based on known plaintext-ciphertext pairs must be computationaly difficult

Task

Typicaly in chiphers length of one block are very short (64 or 128 bits). In practice we want to encipher much longer texts. To do this one uses a block cipher in some mode of operations. In further if we have a text x with length multiple of l we will denote i'th l-bit block as x[i].

Electronic codebook mode

The obvious one.

Enciphering

Algorithm E_K (M[1],...,M[n])
for i=1,..,n do C[i]$\longleftarrow$ E_K(M[i])
Return C[1]...C[n]

Deciphering

Algorithm D_K(C[1],...,C[n])
for i=1,..,n do M[i]$\longleftarrow$ E_K^{-1}(C[i])
Return M[1]...M[n]

Cipher-block chaining mode

It uses initial vector IV, which can be choosen at random for each new message.This method is widely used in practice.

Enciphering

Algorithm E_K(IV,M[1],...,M[n])
C[0] $\longleftarrow$ IV
for i=1,..,n do C[i] $\longleftarrow$ E_K(M[i] $\oplus$ C[i-1])
Return C[0]C[1]...C[n]

Denciphering

Algorithm D_K(C[0]C[1],...,C[n])
for i=1,..,n do M[i] $\longleftarrow$ $E_K^{-1}(C[i]) \oplus C[i-1]$
Return M[1]...M[n]

Counter mode

It uses initial auxiliary integer value $IV \in [0; 2^{l-1}]$ In following operation $+$ is considered as $+$ done modulo 2^l and $BS(j)$ is representation of j as l-bit string.

Enciphering

Algorithm $E_K(IV, M[1], ..., M[n])$
for i=1,..,n do $C[i] \longleftarrow M[i] \oplus E_K(BS(IV + i))$
Return BS(IV)C[1]...C[n]

Denciphering

Algorithm $D_K(BS(IV)C[1], ..., C[n])$
for i=1,..,n do $M[i] \longleftarrow C[i] \oplus E_K(BS(IV + i))$
Return $M[1]...M[n]$

Counter mode

Note that in this case we don't need to have E^{-1} . In fact we even didn't require that E_K is permutation. Other advantage against CBC is parallelizable.

Part II

DES

What is DES

DES - Data Encription Standart is the quintessentianal block cipher. It's most used block cipher by now days. Every time you use ATM you are using DES. He is remarkably secure.

History

Developed by IBM as part of Lucifier project. Adopted by NBS, ANSI, American Banking Association.

Algorithm

Algorithm is designed to encipher and decipher blocks consisting of 64 bits under control of 64-bit key.

Enciphering

Algorithm $E(M, K)$
Making initial permutation $PM \longleftarrow IP(M)$
$PM = L_0, R_0$
for i=1,..,16 do
$L_i \longleftarrow R_{i-1};$
$R_i \longleftarrow L_{i-1} \oplus f(R_{i-1}, K_i)$
od;
$Preoutputblock \longleftarrow L_{16}, R_{16}$ Return IP^{-1} (Preoutputblock)

Key schedule

We have a key schedule function $K_n = KS(n, KEY)$, which takes an integer n and 64bit key and yields 48bit permuted selection of bits from KEY.

The cipher function

The cipher function is function $f(R, K)$ takes 48 bits of key and 32 bits block as input and yields 32 bits block as output, which must be unique defined by R

- First of all it computates $R1 = E(R)$, where E takes 32 bits block and yields 48 bits block.
- Then $R2 = R1 \oplus K$ - only key dependent operation in all cipher :)
- Then breaks R2 into 8 6 bits parts $R2_1, ..., R2_8$ and applies S_i functions to $R2_i$.
- Collecting return values of S functions to block than permutes it with special permutation P and return it.

The cipher function

The cipher function is function $f(R, K)$ takes 48 bits of key and 32 bits block as input and yields 32 bits block as output, which must be unique defined by R

- First of all it computates $R1 = E(R)$, where E takes 32 bits block and yields 48 bits block.
- Then $R2 = R1 \oplus K$ - only key dependent operation in all cipher :)
- Then breaks R2 into 8 6 bits parts $R2_1, ..., R2_8$ and applies S_i functions to $R2_i$.
- Collecting return values of S functions to block than permutes it with special permutation P and return it.

The cipher function

The cipher function is function $f(R, K)$ takes 48 bits of key and 32 bits block as input and yields 32 bits block as output, which must be unique defined by R

- First of all it computates $R1 = E(R)$, where E takes 32 bits block and yields 48 bits block.
- Then $R2 = R1 \oplus K$ - only key dependent operation in all cipher :)
- Then breaks R2 into 8 6 bits parts $R2_1, \dots, R2_8$ and applies S_i functions to $R2_i$.
- Collecting return values of S functions to block than permutes it with special permutation P and return it.

The cipher function

The cipher function is function $f(R, K)$ takes 48 bits of key and 32 bits block as input and yields 32 bits block as output, which must be unique defined by R

- First of all it computates $R1 = E(R)$, where E takes 32 bits block and yields 48 bits block.
- Then $R2 = R1 \oplus K$ - only key dependent operation in all cipher :)
- Then breaks R2 into 8 6 bits parts $R2_1, ..., R2_8$ and applies S_i functions to $R2_i$.
- Collecting return values of S functions to block than permutes it with special permutation P and return it.

S-boxes

Definition

S-box is a function which takes 6 bits block as an input and yields 4 bits block as an output. It's defined as follows.

- It takes first and last bit of input and consider them as a number A in range 0 to 3.
- Then it takes other 4 bits and consider them as a number B in range 0 to 15.
- Then it takes a table, defined for each S_i where $i \in \{1..8\}$ in DES standard. It's size is 4×16 . It yields number on intersection of A'th row and B'th column

S-boxes

Definition

S-box is a function which takes 6 bits block as an input and yields 4 bits block as an output. It's defined as follows.

- It takes first and last bit of input and consider them as a number A in range 0 to 3.
- Then it takes other 4 bits and consider them as a number B in range 0 to 15.
- Then it takes a table, defined for each S_i where $i \in \{1..8\}$ in DES standard. It's size is 4×16 . It yields number on intersection of A'th row and B'th column

S-boxes

Definition

S-box is a function which takes 6 bits block as an input and yields 4 bits block as an output. It's defined as follows.

- It takes first and last bit of input and consider them as a number A in range 0 to 3.
- Then it takes other 4 bits and consider them as a number B in range 0 to 15.
- Then it takes a table, defined for each S_i where $i \in \{1..8\}$ in DES standard. It's size is 4×16 . It yields number on intersection of A'th row and B'th column

Algorithm

Deciphering

Algorithm $D(M, K)$

Making initial permutation $PM \longleftarrow IP^{-1}(M)$

$PM = L_{16}, R_{16}$

for i=16,..,1 do

$R_{i-1} \longleftarrow L_i;$

$L_{i-1} \longleftarrow R_i \oplus f(L_i, K_i)$

od;

$Preoutputblock \longleftarrow L_0, R_0$ Return IP (Preoutputblock)

Note

It's important to note that all permutations, S-function tables, key schedule etc are part of standard, and strength of the algorithm crucial depends on their definitions.

Part III

Attacks

Properties important for security

Following two properties are essential for security.

- Linearity
- Number of rounds

Types of attacks

All known (to me :)) DES attacks are based on known plaintext concept. There are 2 different types:

- Choosen plaintext attack
- Given plain text attack

Types of attacks

All known (to me :)) DES attacks are based on known plaintext concept. There are 2 different types:

- Choosen plaintext attack
- Given plain text attack

Types of attacks

Exhausitive key search iterates over the key space and trying to encipher given ciphertext. If it finds key on which result of this operation is equal to given ciphertext, it checks it on other pair, and if it's right again yields key. It needs only two given plain text.

Differential Cryptoanalysis

Differential cryptoanalysis is a type of choosen plaintext attacks. It analyses the effect of differences in plaintexts on the differences on resultant chipher text. It can assign probabilities to different key candidates, and find the most probable key. Base point - analysis how do the bits of output of s-box changes after after changing input bits.

Differential Cryptoanalysis results

While differential cryptoanalysis shows good results on reduced DES on full 16-round DES it is slower than exhausitive key search.

Linear Cryptoanalysis

Block ciphers commonly uses non-linear operations in their schedule. In DES only non-linear operation is S-boxes. But S-boxes has more in common with linear functions than one would expect if they were chosen completely in random

Idea

IDEA

One approximate non-linear S-box using linear expression:

$$\left(\bigoplus_{i\in\{1..64\}} P^{(i)}\right) \oplus \left(\bigoplus_{j\in\{1..64\}} C^{(j)}\right) = \bigoplus_{k\in\{1..56\}} K^{(k)}(1)$$

This is not true, in general, but it holds with probability $p \neq \frac{1}{2}$

The magnitude

$$\epsilon = |p - \frac{1}{2}|$$

represents effectiveness of our approximation.

Goal is to find effective approximation.

Idea

IDEA

One approximate non-linear S-box using linear expression:

$$(\bigoplus_{i\in\{1..64\}} P^{(i)}) \oplus (\bigoplus_{j\in\{1..64\}} C^{(j)}) = \bigoplus_{k\in\{1..56\}} K^{(k)}(1)$$

This is not true, in general, but it holds with probability $p \neq \frac{1}{2}$

The magnitude

$$\epsilon = |p - \frac{1}{2}|$$

represents effectiveness of our approximation.

Goal is to find effective approximation.

Idea

IDEA

One approximate non-linear S-box using linear expression:

$$(\bigoplus_{i\in\{1..64\}} P^{(i)}) \oplus (\bigoplus_{j\in\{1..64\}} C^{(j)}) = \bigoplus_{k\in\{1..56\}} K^{(k)}(1)$$

This is not true, in general, but it holds with probability $p \neq \frac{1}{2}$

The magnitude

$$\epsilon = |p - \frac{1}{2}|$$

represents effectiveness of our approximation.

Goal is to find effective approximation.

Getting one bit information about key

Algorithm

T:=#of plain texts (out of N) such a left side of (1) is equal to 0

if $T > \frac{N}{2}$

THEN guess $\bigoplus K^{(k)} = 0$ (when $p > \frac{1}{2}$) or 1 (otherwise)

ELSE guess $\bigoplus K^{(k)} = 1$ (when $p > \frac{1}{2}$) or 0 (otherwise)

Getting one bit information about key

To get more information about key we can use another expression

$$(\bigoplus_{i\in\{1..64\}} P^{(i)}) \oplus (\bigoplus_{j\in\{1..64\}} C^{(j)}) \oplus (\bigoplus_{m\in\{1..32\}} f(C, K_{16})^{(m)}) =$$

$$\bigoplus_{k\in\{1..56\}} K^{(k)} (2)$$

where K_{16} is a posible key candidate It's intuitively understood that if we take wrong candidate probability that (2) holds will be much less different from $1/2$ than in case of right candidate.

Getting multiple bits of key

Algorithm

FOREACH subkey candidate K^i of K DO
T^i:=#of plain texts (out of N) such a left side of (2) is equal to 0
OD
$T_{max} = max\{T_i\}$
$T_{min} = min\{T_i\}$
IF $|T_{max} - \frac{N}{2}| > |T_{min} - \frac{N}{2}|$
THEN adopt key candidate corresponding T_{max} and guess $\bigoplus K^{(k)} = 0$ (when $p > \frac{1}{2}$) or 1 (otherwise)
ELSE adopt key candidate corresponding T_{min} and guess $\bigoplus K^{(k)} = 1$ (when $p > \frac{1}{2}$) or 0 (otherwise)

Results of linear this algorithm

Using algorithm given above we can break 16 rounds DES using 2^{47} given plaintext-ciphertext pairs. This method retrieves 14 key bits, 42 remaining bits having to be found using exhaustive key search.

Results of linear cryptoanalysis

In 1994 Matsui built an algorithm which breaks DES using $2^4$3 known plaintext-ciphertext pairs. Then he made computational experiment, and break DES in 50 days (40 of which used to generate keys), using 12 computers. This algorithm used 2 14-round DES linear expressions.

Conclusion

Private key criptography needs such powerful tools as DES cipher. By now, no practical attack on DES was not succeed, but it feeling it's age. Now there is several alternatives to DES (AES, 3DES) which differents from DES in number of rounds and block length. Any way main principles of their design are similar to DES. That's why approachs to analysis of such algorithms are very important.

Thank you