Seminar: Cryptography

Private key cryptography

Lev Gurevich

July 26, 2005

Abstract

Private key cryptography is an approach to private information transfer based on idea of shared secret key. It is one of the most common used in practice because of its performance and its simplicity in implementation.

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

In this article this concept is briefly described. Then DES cipher, the most popular standard in private key cryptography, and the ways of his cracking is considered.

Contents

1 Block ciphers

Block cipher is a central notion in private key cryptography.

Definition 1 *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 $E^{-1}(K,C) = E_K^{-1}(C)$. It was a formal defenition, but in practice we need a block cipher to satisfy a set of requirements. It is:

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

It is obvious that all this requirements are dictated by security reasons.

Typicaly in chiphers length of one block is 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. We will consider a string as a sequence of little blocks of date and work with each block separately. The following are examples of such modes.
In further if we have a text x with length multiple of l we will denote i'th l-bit block as x[i].

1.1 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]
```

Denciphering

```
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]
```

1.2 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]
```

1.3 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 $M[1]...M[n]$

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. Another advantage against CBC is parallelizable.

2 DES

In 1972, the National Institute of Standards and Technology (called the National Bureau of Standards at the time) decided that a strong cryptographic algorithm was needed to protect non-classified information. The algorithm was required to be cheap, widely available, and very secure. NIST envisioned something that would be available to the general public and could be used in a wide variety of applications. So they asked for public proposals for such an algorithm. In 1974 IBM submitted the Lucifer algorithm, which appeared to meet most of NIST's design requirements. NIST enlisted the help of the National Security Agency to evaluate the security of Lucifer. At the time many people distrusted the NSA due to their extremely secretive activities, so there was initially a certain degree of skepticism regarding the analysis of Lucifer. One of the greatest worries was that the key length, originally 128 bits, was reduced to just 56 bits, weakening it significantly. The NSA was also accused of changing the algorithm to plant a "back door" in it that would allow agents to decrypt any information without having to know the encryption key. But these fears proved unjustified and no such back door has ever been found. The modified Lucifer algorithm was adopted by NIST as a federal standard on November 23, 1976. Its name was changed to the Data Encryption Standard (DES). The algorithm specification was published in January 1977, and with the official backing of the government it became a very widely employed algorithm in a short amount of time. Unfortunately, over time various shortcut attacks were found that could significantly reduce the amount of time needed to find a DES key by brute force. And as computers became progressively faster and more powerful, it was recognized that a 56-bit key was simply not large enough for high security applications. As a result of these serious flaws, NIST abandoned their official endorsement of DES in 1997 and began work on a replacement, to be called the Advanced Encryption Standard (AES). Despite the growing concerns about its vulnerability, DES is still widely used by financial services and other industries worldwide to protect sensitive on-line applications. To highlight the need for stronger security than a 56-bit key can offer, RSA Data Security has been sponsoring a series of DES cracking contests since early 1997. In 1998 the Electronic Frontier Foundation won the RSA DES Challenge II-2 contest by breaking DES in less than 3 days. EFF used a specially developed computer called the DES Cracker, which was developed for under 250,000 dollars. The encryption chip that powered the DES Cracker was capable of processing 88 billion keys per second. More recently, in early 1999, Distributed. Net used the DES Cracker and a worldwide network of nearly 100,000 PCs to win the RSA DES Challenge III in a record breaking 22 hours and 15 minutes. The DES Cracker and PCs combined were testing 245 billion keys per second when the correct key was found. In addition, it has been shown that for a cost of one million dollars a dedicated hardware device can be built that can search all possible DES keys in about 3.5 hours. This just serves to illustrate that any organization with moderate resources can break through DES with very little effort these days. This algorithm is designed to encipher and decipher blocks consisting of 64 bits under control of 64-bit key. Actual key length is only 56 bits. 8 bits are used as a control of correctness. This algorithm is very simple and uses only simple bit operations (permutations, XORs) so it is not based on any complexity theory problems as most of public key cryptography algorithms do. Data encryption (cryptography) is utilized in various applications and environments. The specific utilization of encryption and the implementation of the DES will be based on many factors particular to the computer system and its associated components. In general, cryptography is used to protect data while it is being communicated between two points or while it is stored in a medium vulnerable to physical theft. Communication security provides protection to data by enciphering it at the transmitting point and deciphering it at the receiving point. File security provides protection to data by enciphering it when it is recorded on a storage medium and deciphering it when it is read back from the storage medium. In the first case, the key must be available at the transmitter and receiver simultaneously during communication. In the second case, the key must be maintained and accessible for the duration of the storage period. FIPS 171 provides approved methods for managing the keys used by the algorithm specified in this standard. Full description of this standard can be found on http://www.itl.nist.gov/fipspubs/fip46-2.htm

```
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)
```

As you can see algorithm above on each step uses 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. This function is specified by table in algorithm specification. 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 $R2_i$.
- Collects return values of S functions to block than permutes it with special permutation P and return it.

The most powerful element in this algorithm is S-box.

Definition 2 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*

Deciphering is a reflection of enciphering algorithm in inverse order. Formally it looks like:

```
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)

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. In fact there was a several modification of DES based on other tables and they has shown a better results then initial algorithm but they did not puss such amount of certification as DES do.

3 Attacks

To understand attacks on DES you need first to understand which properties make algorithm secure or not. Following two properties are essential for security.

- Linearity how for algorithm from linear function (existence of nice linear approximation)
- Number of rounds how obvious function is algorithm from its input bits

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

- Chosen plaintext attack
- Given plain text attack

3.1 Exhaustive key search

The most obvious type of attack is exhaustive key search. It iterates over the key space and trying to encipher given cipher text. 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. The probability that found key is wrong is neglible. The advantage of this attack is that only two given plain text are needed. But complexity of this algorithm is not acceptable especially in case of long key.

3.2 Differential crypto analysis

Another approach is differential crypto analysis. It is a type of choosen plaintext attacks. It analyses an effect of the differences in plaintexts on the differences on resultant cipher text. On each step it can assign probabilities to different key candidates, and find the most probable key. The base point of differential crypto analysis is analysis how do the bits of output of s-box changes after after changing input bits. Used in this analysis technique is very complicated and outside the scope of this document. While differential crypto analysis shows good results on reduced DES on full 16-round DES it is slower than exhausitive key search, because when number of rounds is big enough cipher becomes an almost random function of its bits.

3.3 Linear crypto analysis

On linear weakness of DES linear crypto analysis approach is based. Block ciphers commonly uses non-linear operations in their schedule. In DES only non-linear operations are S-boxes. But S-boxes has much more in common with linear functions than one would expect if they were chosen completely in random The basic idea of linear analysis is as following: *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.

Using this idea following algorithm extracts one bit of 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)
```

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. But in practice we want to find as much bits of key as its possible. For this purposes following evolution of previous algorithm can be used:

```
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)
```

Using algorithm given above we can break 16 rounds DES using 2^{47} given plaintext-cipher text pairs. This method retrieves 14 key bits, 42 remaining bits having to be found using exhausitive key search. In 1994 Matsui built an algorithm which breaks DES using 2^43 known plaintext-cipher text 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.

4 Conclusion

Private key cryptography needs such powerful tools as DES cipher. By now, no practical attack on DES has been succeeding, but it feels it's age. Now there are 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 the approachs to analysis of such algorithms are very important.