In the name of Allah

Slide Attacks with Probability 1

Shahsavaran

Contents

- Slide cryptanalysis in a nutshell
- A brief description of Blowfish
- Slide cryptanalysis of a blowfish variant [BirWag]
- Slide cryptanalysis of a blowfish variant (with success prob of 1)
- Application of the new idea to non-Feistel algorithms
- Biham related-key attack(with success prob of 1)

Slide Cryptanalysis

- E is a block cipher. E_k the encryption function with key k
- Preconditions of slide attack:
  1. E_k can be written as the composition of sevral identical functions

$$E_k(P) = (F_{K'} \circ F_{K'} \circ \ldots F_{K'})(P)$$

  2. having 2 known-plaintext equations

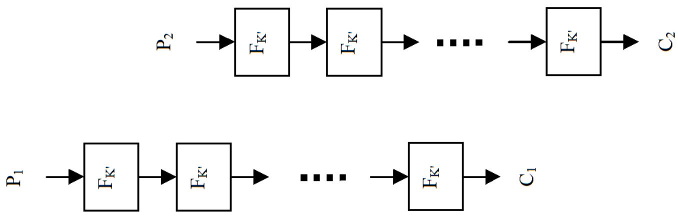
$$F_K'(x_1)=y_1$$
$$F_K'(x_2)=y_2$$

the key k' can be "*easily*" recovered

Slide Cryptanalysis

- Slide attack :
  1. Look for a pair of plntxts (P1,P2) st $F_K'(P_1)=P_2$

→ $F_K'(C_1)=C_2$



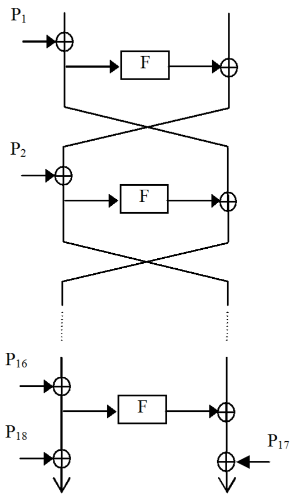
Slide Cryptanalysis

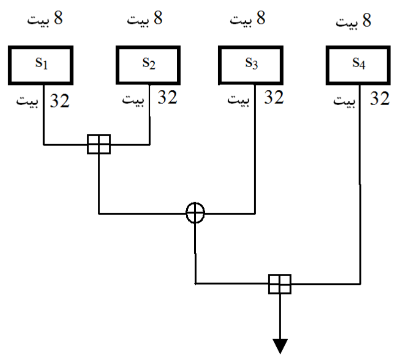
- Slide attack (cont'd) :
  2. $F_K'(P_1)=P_2$ & $F_K'(C_1)=C_2$ are expected to reveal k'.

- How to find (P1,P2) st $F_K'(P_1)=P_2$?
  - Common approach: gather almost $2^{n/2}$ known pairs(Pi,Ci).
  - Birthday paradox →with good prob, there is at least 1 pair (P1,P2) st $F_K'(P_1)=P_2$
  - The existence of such pairs is probabilistic.

- Our method guarantees the existence of such a pair with prob=1, for some special cases.

Blowfish





Slide cryptanalysis of Blowfish – Pi's [BirWag]

- Consider Blowfish – Pi's
- Consider the eq F(x)=y

$$[(s_1(x_1) + s_2(x_2)) \oplus s_3(x_3)] + s_4(x_4) = y$$

- This eq gives 32 bits of info about the sbox entries.
- Every slide pair → 2x32=64 bits of info.
- 1024 sbox entries in total . 2^9=512 slide pair needed.

Slide cryptanalysis of Blowfish – Pi's [BirWag]

- Consider Blowfish – Pi's
- How to find a slide pair?
- Consider the 2 sets {X_i | i=1,2,…,2^{16}} and {Y_j | j=1,2,…,2^{16}}
- Birthday paradox→ with prob~0.5 , there is X_i & Y_j st $F(R) \oplus X_i = Y_j$
- → $F_1(R,X_i)=(Y_j,R)$: a slide pair.

Slide cryptanalysis of Blowfish – Pi's

- Consider Blowfish – Pi's
- The previous approach for obtaining a slide pair is probabilistic.
- How to find a slide pair with prob 1?
- The idea is to use chosen plaintexts instead of known plaintexts.
- (R,X_i) & (Y_j,R) are a slide pair iff $F(R) \oplus X_i = Y_j$ or $X_i \oplus Y_j = F(R)$
- F(R) is constant, so if we choose X_i & Y_j st $X_i \oplus Y_j$ spans all 32bit values, one will equal F(R).

Slide cryptanalysis of Blowfish – Pi's

- Consider Blowfish – Pi's
- One such set of texts is $X_i = a_i || 0_{16}$ and $Y_j = 0_{16} || a_j$ where $a_i = i$. $X_i \oplus Y_j = a_i || a_j$
- When i&j span the set $\{1,2,\ldots,2^{16}\}$ independently, $a_i || a_j$ will span the set $\{1,2,\ldots,2^{32}\}$
- So for one (i,j) : $X_i \oplus Y_j = F(R)$
- NOTE : we are sure there is one slide pair, but we can't tell it from other pairs.
- the process of identifying the slide pair is as before.

Slide cryptanalysis of Blowfish – Pi's

- Consider Blowfish – Pi's
- for each R there is one slide pair with certainty.
- we need 512 slide pairs
  - change R and each time use the above-mentioned type of texts.

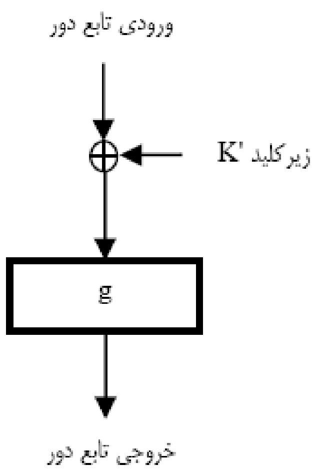
extension to nonFeistel algorithms

- g
  - arbitrary
  - unKeyed
  - invertible

- (p,p') is a slide pair iff

$p' = g(p \oplus K')$

or $g^{-1}(p') \oplus p = K'$



extension to nonFeistel algorithms

- 2n= block lengths
- put $p_i=a_i||0_n$ and $g^{-1}(p'_j)=0_n||a_j$ where ‹a_i=i for i=0, 1, …, 2^n-1 $g^{-1}(p'_j)\oplus p_i = a_i||a_j$
- When i&j span the set {1,2,…,2^n} independently, $a_i||a_j$ will span the set {1,2,…,2^{2n}}
- so $p_i=a_i||0_n$ and $p'_j=g(0_n||a_j)$
- Data complexity is exactly 2^n+2^n

extension to nonFeistel algorithms

- the key combining operation is not restricted to simple xor.
- e.g. for $p=(p_1,p_2,p_3,p_4)$ and $K'=(k'_1,k'_2,k'_3,k'_4)$
  - $p+K'=((p_1+ k'_1)\bmod 2^{32}, (p_2+ k'_2)\bmod 2^{32}, (p_3+ k'_3)\bmod 2^{32}, (p_4+ k'_4)\bmod 2^{32})$
- (p,p') is a slide pair iff $p' = g(p + K')$

or $g^{-1}(p') - p = K'$

extension to nonFeistel algorithms

- put
  $$p_{i,j} = (-a_i, -a_j, 0_{32}, 0_{32})$$
  $$p'_{m,n} = g(0_{32}, 0_{32}, a_m, a_n)$$

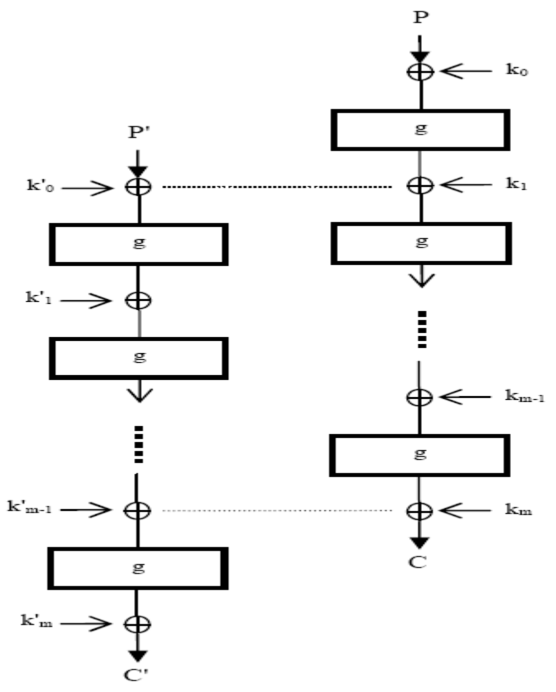
- where $a_i = i$ for $i=0, 1, \ldots, 2^{32}-1$

$$g^{-1}(p'_{m,n}) - p_{i,j} = (a_i, a_j, a_m, a_n)$$

Biham’s related-key attack

- Applicable to algorithms whose
  - i-th rnd func is the same as the 1st rnd func
  - (i+1)-th rnd func is the same as the 2nd rnd func
  - ...

  for some small i (1 or 2)
  - plus for each key K with subkeys $k_0, k_1, \ldots$ there is a related-key K’ with subkeys $k'_0, k'_1, \ldots$ st $k'_0 = k_i$, $k'_1 = k_{i+1}$, …



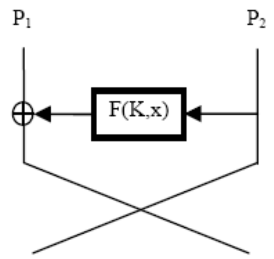
Biham's related-key attack

- If $P' = g(P \oplus k_0)$ then $C' = g(C) \oplus k'_m$ and these two relations are expected to reveal info about subkeys.
- The problem is to find (P,P') st $P' = g(P \oplus k_0)$
- Traditional method : appeal to birthday paradox
  - probabilistic in nature
- Our method can guarantee the existence of (P,P') with prob 1

Biham's related-key attack

- 1- Feistel type rnd func
  - F : absolutely arbitrary
- We have to find (P_1, P_2) and

(P'_1, P'_2) st $$P'_1 = P_2$$ $$F(K, P_2) \oplus P_1 = P'_2$$



Biham's related-key attack

- 1- Feistel type rnd func
- put $$P_1^{(i)} = a_i || 0_n$$ $$P'^{(j)}_2 = 0_n || a_j$$

then $$P_1^{(i)} \oplus P'^{(j)}_2 = a_i || a_j$$

and for each constant R, $F(K, R) = P_1^{(i)} \oplus P'^{(j)}_2$ occurs for some i&j.

Biham's related-key attack

- 1- Feistel type rnd func
- So a possible candidate is

$$(P_1^{(i)}, P_2) = (a_i||0_n, R)$$

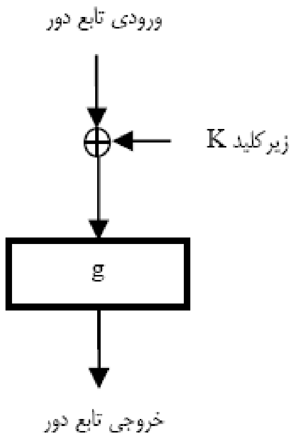
$$(P_1', P'^{(j)}_2) = (R, 0_n||a_j)$$

where a_i=i for i=0, 1, …, 2^n-1 (2n= block len)

- the combining operation need NOT be restricted to xor.

Biham's related-key attack

- 1- nonFeistel type rnd func
- g as before (SPN a special case)
- the key combining op need NOT be restricted to xor.
- previous solution works



$$P' = g(P \oplus K)$$

Biham’s related-key attack

- Conclusion : The idea works well for i=1 and with some restriction on the round structure.
- Avenue for future research :
  - find similar methods for i=2 .

Biham’s related-key attack

Any Question?