Cryptography 111

Attack on Titan

Table of Contents

自古OP多劇透

- Key recovery attack on
  - Classical Cipher
  - Side channel
  - Old Modern Cipher
  - Old Modern Hashes
- Factorization
- Discrete Logarithm

Goal

現在，我的手中抓住了未來

Key Recovery

謎已經全部解開了

Given plaintext and ciphertext, a key-recovery attack is an adversary's attempt to recover the cryptographic key of an encryption scheme.

Some General Tools 少這個…就很不方便 ...

Brute Force

爆搜雖可恥，但有用

Solving with $$

你想用錢來收買我嗎！？
這是對我的侮辱！我本想這麼大
聲斥責他，但錢實在是太多了

```c
// gcc main.c -fopenmp -O3
#pragma omp parallel for num_threads(32)
for (uint64_t i = 0; i < (1ULL<<32); i++) {
    if (hash(i) == target) {
        printf("%llu\n", i);
        exit(0);
    }
}
```

Birthday Paradox

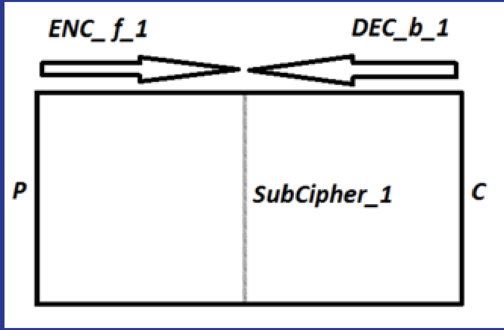
現在想起來
那也是理所當然的事情

- The goal of the attack is to find two different inputs x_1, x_2 such that $f(x_1) = f(x_2)$.
- The function f has H outcomes.
- Expected to find after evaluating the function for $1.25\sqrt{H}$ times.

Meet In The Middle

此時此刻，他不是一個人在戰鬥

- $c = E_k(m) = E_{0k_0}\left(E_{1k_1}(m)\right)$
- $\Rightarrow D_{0k_0}(c) = E_{1k_1}(m)$
- Complexity is $O(2N)$ not $O(N^2)$.



Toy Classics 戰鬥力只有5的渣渣

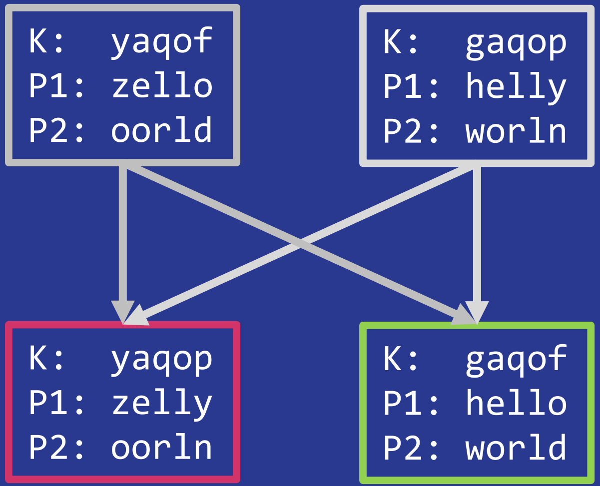
Our Target

懶惰可是人類的天性呢

```python
# Vigenère cipher
def encrypt(plain, key):
    N, ksz = len(charset), len(key)
    return ''.join(charset[(c + key[i % ksz]) % N]
        for i, c in enumerate(plain))
```

Diffusion

快看他畫風和我們不一樣



Genetic Algo.

不就是一塊石頭麼，
看我用鋼彈把它推回去

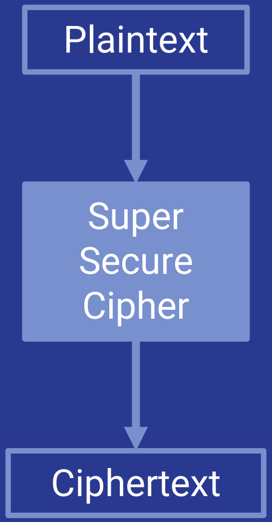
- Initial population
    - Randomly generate N candidates
- Selection with fitness function
    - Select best N candidates
- Generate second generation
    - Crossover / Mutation

[LAB] Classical Cipher

Side Channel 不要跟他硬拼，試著切他中路

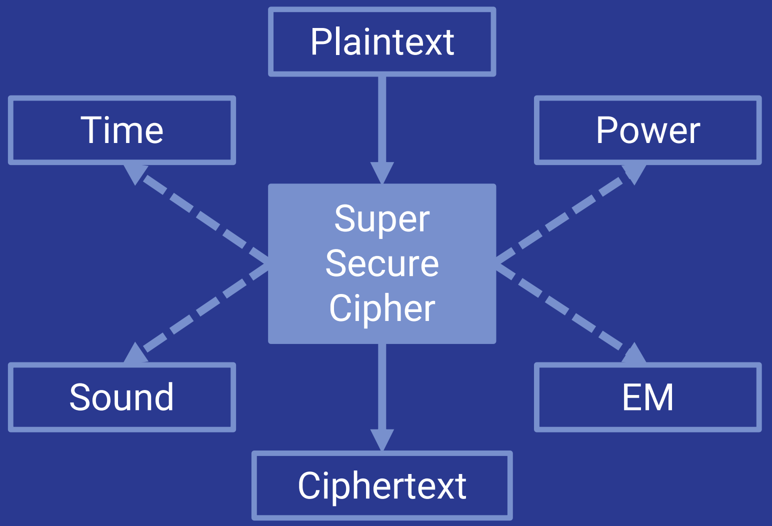
Side Channel

不要跟他硬拼，試著切他中路



Side Channel

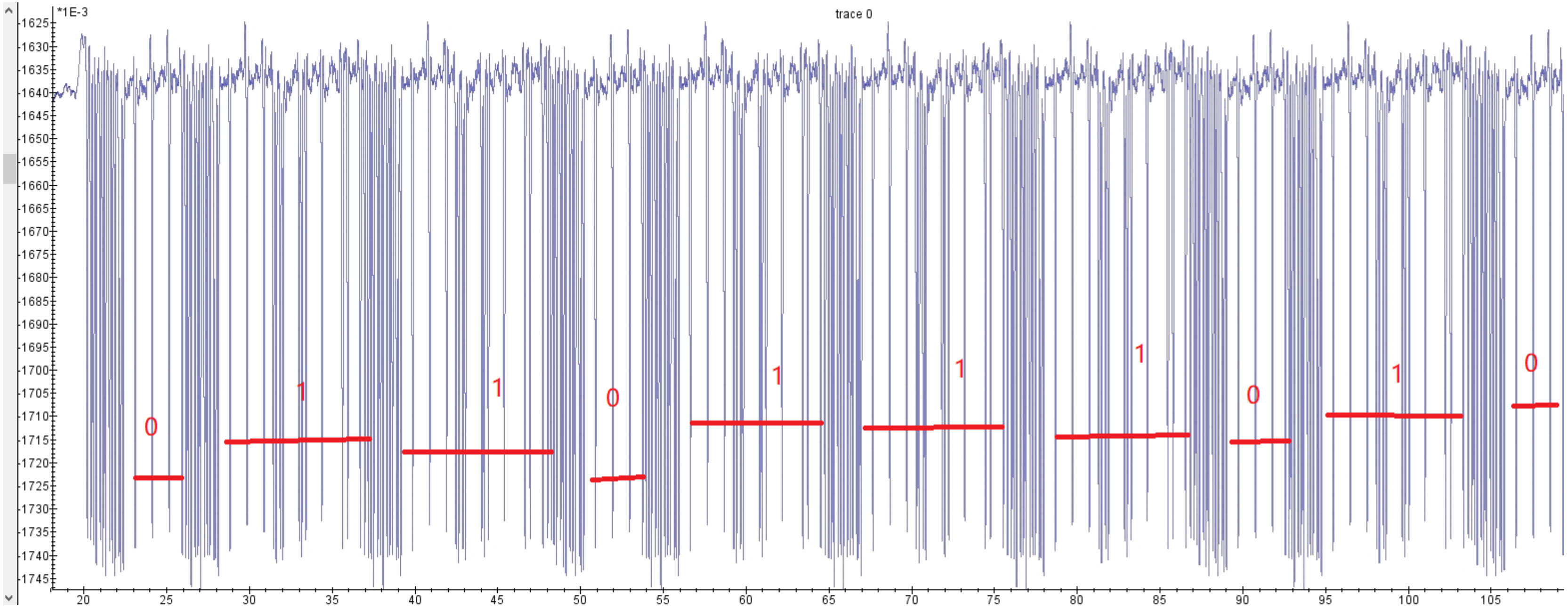
不要跟他硬拼，試著切他中路



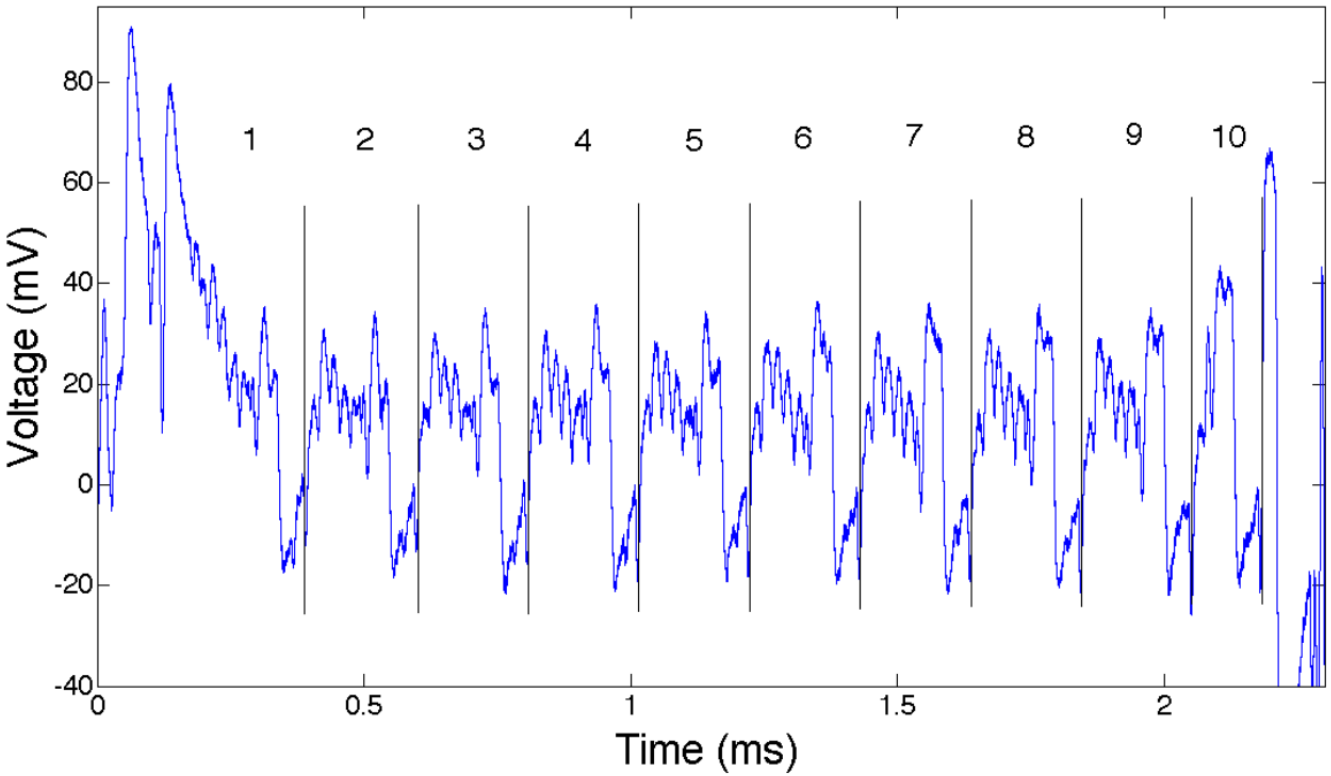
Simple Power Analysis

前方高能反應

Variations in power consumption occur as the device performs different operations.



Simple Power Analysis 前方高能反應



AES Power trace

相同的招式對聖鬥士是沒用的

Side-Channel Attacks on the Yubikey 2 One-Time Password Generator

Differential Power Analysis

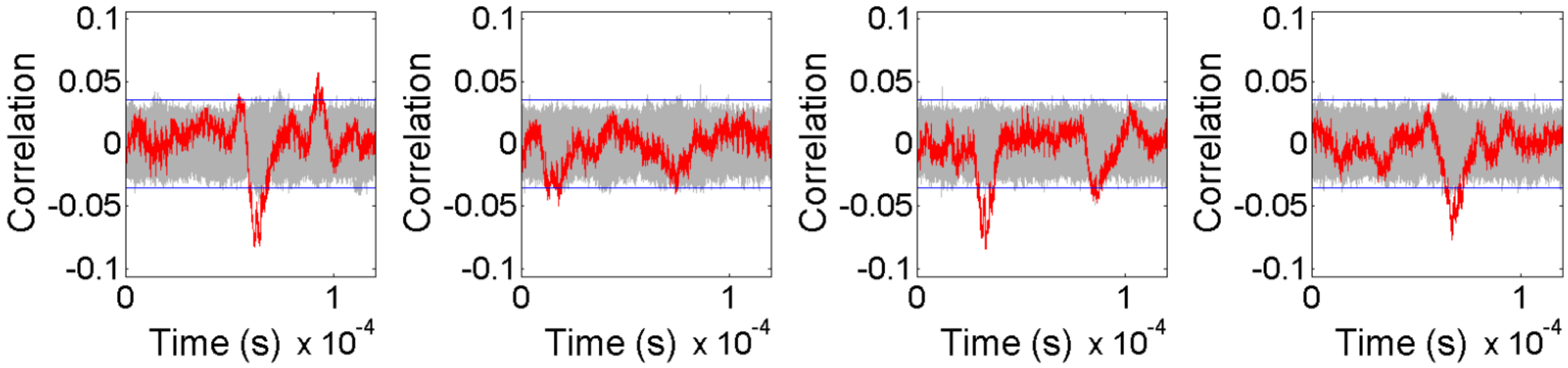
難道藏了我所不知道的武器？

- No timing difference
- 1 consume more power than 0
- But noise is much larger than that tiny power difference.

Differential Power Analysis

今天的風兒好喧囂啊

- Statistics to the rescue
- Split power traces to two groups
- Group 1 is expected to consume more power than group 2
- Compare their average / median
- Check correlation if split to more than two groups



Side-Channel Attacks on the Yubikey 2 One-Time Password Generator

Differential Power Analysis 燃えろ！俺の魂！

[LAB] Differential Power Analysis

Weaker Goal

可是那一天，我有了新的想法

Distinguish

這味道……是說謊的味道。

Given plaintext (possibly chosen by attacker) and a message, the attacker can tell whether the message is its corresponding ciphertext or just a random string.

With probability larger than ½.

Gimme the Key!

計画通り😌

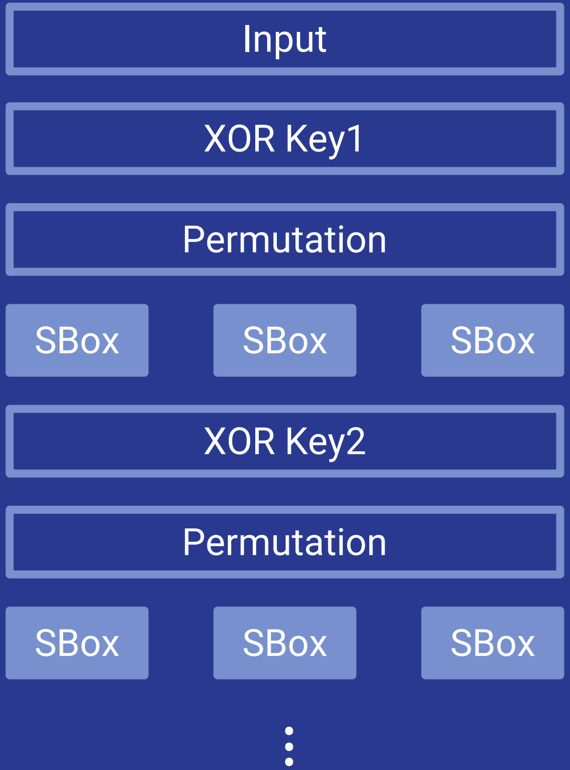
- $E_k(m) \coloneqq E_{1k_1}\left(E_{0k_0}(m)\right)$
- An algorithm to distinguish $E_{0k_0}(m)$ from random oracle.
- Decrypt $E_k(m)$ using all possible k_1 and check whether the output is $E_{0k_0}(m)$ or not.

Cryptoanalysis

看啊，你的死兆星在天上閃耀

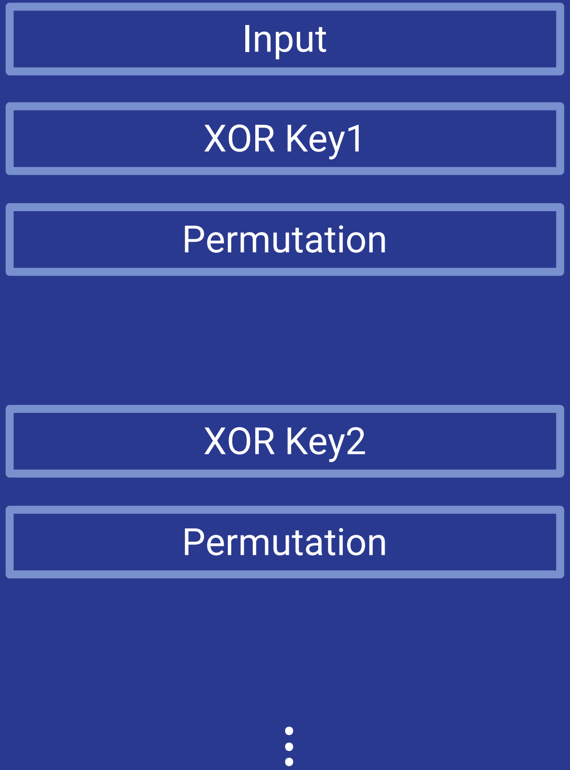
Our Target

警察叔叔，就是這個人



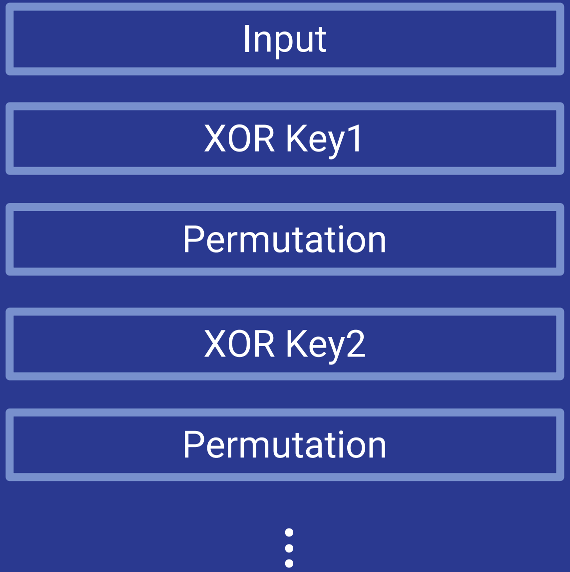
Without SBox

如果去掉就是神作了



Without SBox

如果去掉就是神作了



$$E_k(m) = P(m) + Key'$$

Distinguishing oracle:
$E_k(m) + P(m)$ are all the same

Linear

你為什麼不問問神奇海螺呢？

Linear approximation of SBox

abc	00	01	10	11
0	1	0	0	0
1	0	1	1	1

Linear Equation $o = b + c + 1$ holds with probability of $\frac{6}{8}$

Linear

你為什麼不問問神奇海螺呢？

Linear approximation of SBox

X_1	X_2	X_3	X_4	Y_1	Y_2	Y_3	Y_4	$X_2 \oplus X_3$	$Y_1 \oplus Y_3 \oplus Y_4$	$X_1 \oplus X_4$	Y_2	$X_3 \oplus X_4$	$Y_1 \oplus Y_4$
0	0	0	0	1	1	1	0	0	0	0	1	0	1
0	0	0	1	0	1	0	0	0	0	1	1	1	0
0	0	1	0	1	1	0	1	1	0	0	1	1	0
0	0	1	1	0	0	0	1	1	1	1	0	0	1
0	1	0	0	0	0	1	0	1	1	0	0	0	0
0	1	0	1	1	1	1	1	1	1	1	1	1	0
0	1	1	0	1	0	1	1	0	1	0	0	1	0
0	1	1	1	1	0	0	0	0	1	1	0	0	1
1	0	0	0	0	0	1	1	0	0	1	0	0	1
1	0	0	1	1	0	1	0	0	0	0	0	1	1
1	0	1	0	0	1	1	0	1	1	1	1	1	0
1	0	1	1	1	1	0	0	1	1	0	1	0	1
1	1	0	0	0	1	0	1	1	1	1	1	0	1
1	1	0	1	1	0	0	1	1	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1	0	1	0
1	1	1	1	0	1	1	1	0	0	0	1	0	1

A Tutorial on Linear and Differential Cryptanalysis

Linear

你為什麼不問問神奇海螺呢？

Calculate probability bias table for all equations of input and equations of output

$$f(x;a,b) = ax + bS(x)$$

$$v_{ij} = \sum_{k=1}^{2^n} f(x_k;a_i,b_j) - \frac{2^n}{2}$$

$$\Pr(f(x;a_i,b_j) = 1) = \frac{1}{2} + \frac{v_{ij}}{2^n}$$

Linear

你為什麼不問問神奇海螺呢？

Calculate probability bias table for all equations of input and equations of output

Input Sum	Output Sum: 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

A Tutorial on Linear and Differential Cryptanalysis

Linear Path

今天我生日て

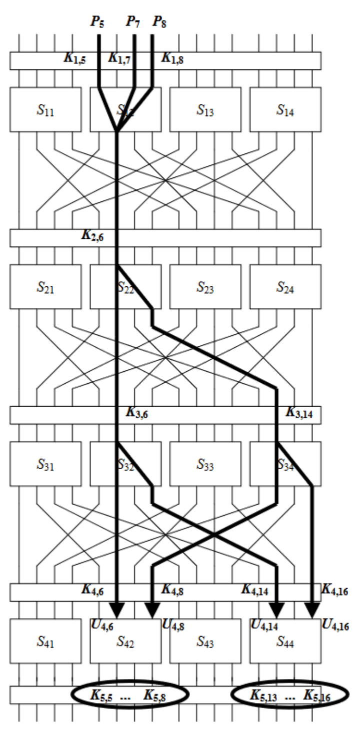
Let’s assume that we are sooooooo lucky that all the approximations we choose hold respect to our input.

Linear Path

今天我生日て

Chaining different approximation to get full cipher linear approximation

$$P_5 + P_7 + P_8$$
$$+ U_{4,6} + U_{4,8} \dots$$
$$+ K' = 0$$



A Tutorial on Linear and Differential Cryptanalysis

[HW] Linear Cryptoanalysis

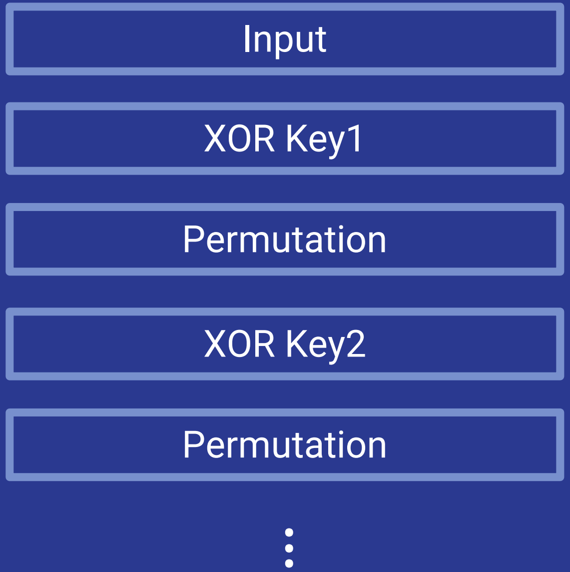
Differential

有奇怪的東西混進去了

- We can get ciphertext of any plaintext we choose.
- What will the cipher output if we send x and $x + \Delta$?

Without SBox

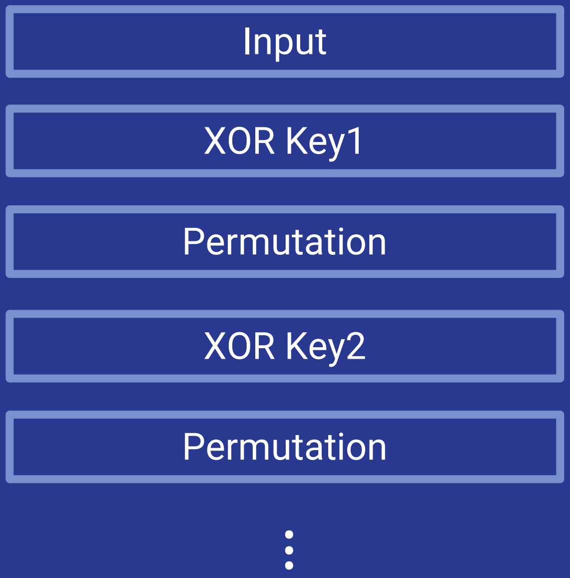
有奇怪的東西混進去了



$$E_k(m) = P(m) + Key'$$
$$E_k(m + \Delta) = P(m + \Delta) + Key'$$
$$P(m + \Delta) = P(m) + P(\Delta)$$
$$E_k(m + \Delta) = E_k(m) + P(\Delta)$$

Without SBox

有奇怪的東西混進去了



Distinguishing oracle:
Check if $E_k(m + \Delta) = E_k(m) + \Delta'$

Differential

有奇怪的東西混進去了

Approximate SBox's differences instead of function value

X	Y	ΔY		
		$\Delta X = 1011$	$\Delta X = 1000$	$\Delta X = 0100$
0000	1110	0010	1101	1100
0001	0100	0010	1110	1011
0010	1101	0111	0101	0110
0011	0001	0010	1011	1001
0100	0010	0101	0111	1100
0101	1111	1111	0110	1011
0110	1011	0010	1011	0110
0111	1000	1101	1111	1001
1000	0011	0010	1101	0110
1001	1010	0111	1110	0011
1010	0110	0010	0101	0110
1011	1100	0010	1011	1011
1100	0101	1101	0111	0110
1101	1001	0010	0110	0011
1110	0000	1111	1011	0110
1111	0111	0101	1111	1011

A Tutorial on Linear and Differential Cryptanalysis

Interpolation

人之亡死來招乃我但，
喜討又愛可然雖

- Pure cipher / KN Cipher
- Feistel cipher structure
- $F_k(x) = (x + k)^3$
- Cube function is provably secure against conventional linear and differential cryptoanalysis

Interpolation

人之亡死來招乃我但，
喜討又愛可然雖

- The cipher is just a polynomial…
- $E_k(m) = a_{279}m^{279} + a_{278}m^{278}$ …
- Construct its coefficient using our favorite linear algebra

Hash Collision

太相似的話就會有版權問題

- Goal: find A, B s.t. $H(A) = H(B)$
- $m := (m_1, m_2, \dots, m_n)$
- $H(m) :=$
  $S(A \dots (A(A(IV, m_1), m_2), \dots))$
- Merkle damgard:
  - S = Identity, A = round function
- Sponge Construction:
  - S = Squeeze, A = Absorb

$$s_0 = IV,\ s_i = A(s_{i-1}, m_i),\ \Delta_i' = A(s_{i-1}, m_i + \Delta_i) - s_i$$

$$\begin{aligned} H(m + \Delta) &= S(A \ldots (A(A(s_0, m_1 + \Delta_1), m_2 + \Delta_2), \ldots)) \\ &= S(A \ldots (A(s_1 + \Delta_1', m_2 + \Delta_2), \ldots)) \\ &= S(A \ldots (s_2 + \Delta_2', \ldots)) \\ &= S(s_n + \Delta_n') \\ &= output + \Delta_o' \end{aligned}$$

If $\Delta_o' = 0$, we found a collection!
Two block pair (i.e. $m = (m_1, m_2)$) is a good choice. (e.g. MD5)

Differential Path

神說你還不能死在這裡

Sufficient cond.

圍繞著你的世界，
比你想像的要溫柔一些

- Different from ciphers, we have access to all the constants and intermediate outputs.
- We could derive some sufficient conditions that makes the difference holds (with high probability).
- $c_{1,7} = 0,\ c_{1,8} = b_{1,8},\ \ldots$

Brute Force

我本來不想用這一招的…

- If we brute force for a input that satisfy all conditions, the complexity is about $O(2^{\#cond})$
- We have hundreds of condition for MD5…

Take MD5 as an example, we can generate intermediate value based on conditions and reconstruct our input.

```
Q[ 1]=Q[ 0]+RL(F(Q[ 0],Q[-1],Q[-2])+Q[-3]+x[ 0]+0xd76aa478, 7); 0 c.
Q[ 2]=Q[ 1]+RL(F(Q[ 1],Q[ 0],Q[-1])+Q[-2]+x[ 1]+0xe8c7b756,12); 0 c.
Q[ 3]=Q[ 2]+RL(F(Q[ 2],Q[ 1],Q[ 0])+Q[-1]+x[ 2]+0x242070db,17); 17 c.
Q[ 4]=Q[ 3]+RL(F(Q[ 3],Q[ 2],Q[ 1])+Q[ 0]+x[ 3]+0xc1bdceee,22); 21 c.
Q[ 5]=Q[ 4]+RL(F(Q[ 4],Q[ 3],Q[ 2])+Q[ 1]+x[ 4]+0xf57c0faf, 7); 32 c.
Q[ 6]=Q[ 5]+RL(F(Q[ 5],Q[ 4],Q[ 3])+Q[ 2]+x[ 5]+0x4787c62a,12); 32 c.
Q[ 7]=Q[ 6]+RL(F(Q[ 6],Q[ 5],Q[ 4])+Q[ 3]+x[ 6]+0xa8304613,17); 32 c.
Q[ 8]=Q[ 7]+RL(F(Q[ 7],Q[ 6],Q[ 5])+Q[ 4]+x[ 7]+0xfd469501,22); 29 c.
Q[ 9]=Q[ 8]+RL(F(Q[ 8],Q[ 7],Q[ 6])+Q[ 5]+x[ 8]+0x698098d8, 7); 28 c.
Q[10]=Q[ 9]+RL(F(Q[ 9],Q[ 8],Q[ 7])+Q[ 6]+x[ 9]+0x8b44f7af,12); 18 c.
Q[11]=Q[10]+RL(F(Q[10],Q[ 9],Q[ 8])+Q[ 7]+x[10]+0xffff5bb1,17); 19 c.
Q[12]=Q[11]+RL(F(Q[11],Q[10],Q[ 9])+Q[ 8]+x[11]+0x895cd7be,22); 15 c.
Q[13]=Q[12]+RL(F(Q[12],Q[11],Q[10])+Q[ 9]+x[12]+0x6b901122, 7); 14 c.
Q[14]=Q[13]+RL(F(Q[13],Q[12],Q[11])+Q[10]+x[13]+0xfd987193,12); 15 c.
Q[15]=Q[14]+RL(F(Q[14],Q[13],Q[12])+Q[11]+x[14]+0xa679438e,17); 9 c.
Q[16]=Q[15]+RL(F(Q[15],Q[14],Q[13])+Q[12]+x[15]+0x49b40821,22); 6 c.
```

Modification 真不想承認啊，這是我太過年輕而犯下的錯

For latter parts, where we don't have enough freedom on input to control intermediate output, we have to modify previous intermediate output.

This technique could generate a message that satisfy all conditions up to Q[24] in MD5.

```
Q[ 2]=Q[ 1]+RL(F(Q[ 1],Q[ 0],Q[-1])+Q[-2]+x[ 1]+0xe8c7b756,12); 0 c.
Q[13]=Q[12]+RL(F(Q[12],Q[11],Q[10])+Q[ 9]+x[12]+0x6b901122, 7); 14 c.
Q[14]=Q[13]+RL(F(Q[13],Q[12],Q[11])+Q[10]+x[13]+0xfd987193,12); 15 c.
Q[15]=Q[14]+RL(F(Q[14],Q[13],Q[12])+Q[11]+x[14]+0xa679438e,17); 9 c.
Q[16]=Q[15]+RL(F(Q[15],Q[14],Q[13])+Q[12]+x[15]+0x49b40821,22); 6 c.

Q[17]=Q[16]+RL(G(Q[16],Q[15],Q[14])+Q[13]+x[ 1]+0xf61e2562, 5); 5 c.
```

Modification 真不想承認啊，這是我太過年輕而犯下的錯

When the complexity goes too high for modification, we leave all other conditions to be fulfilled randomly.

```
Q[23]=Q[22]+RL(G(Q[22],Q[21],Q[20])+Q[19]+x[15]+0xd8a1e681,14); 2 c. sat.
Q[24]=Q[23]+RL(G(Q[23],Q[22],Q[21])+Q[20]+x[ 4]+0xe7d3fbc8,20); 1 c. sat.

.......... Here is the point of verification (POV) .................

Q[25]=Q[24]+RL(G(Q[24],Q[23],Q[22])+Q[21]+x[ 9]+0x21e1cde6, 5);
Q[26]=Q[25]+RL(G(Q[25],Q[24],Q[23])+Q[22]+x[14]+0xc33707d6, 9);
                                        ...
Q[35]=Q[34]+RL(H(Q[34],Q[33],Q[32])+Q[31]+x[11]+0x6d9d6122,16); 1 c.
                                        ...
Q[48]=Q[47]+RL(H(Q[47],Q[46],Q[45])+Q[44]+x[ 2]+0xc4ac5665,23); 1 c.
Q[49]=Q[48]+RL(I(Q[48],Q[47],Q[46])+Q[45]+x[ 0]+0xf4292244, 6); 1 c.
Q[50]=Q[49]+RL(I(Q[49],Q[48],Q[47])+Q[46]+x[ 7]+0x432aff97,10); 1 c.
Q[51]=Q[50]+RL(I(Q[50],Q[49],Q[48])+Q[47]+x[14]+0xab9423a7,15); 1 c.
```

Point of Verification 收了可觀的小費後，酒館老闆小聲道

Tunneling

我已經用了二次啦

- Given an input which satisfied all conditions before PoV, we want to have an algorithm that generate more inputs with little effort.

If we trying to modifying Q[9], we need to fix conditions before PoV

```
Q[ 8]=Q[ 7]+RL(F(Q[ 7],Q[ 6],Q[ 5])+Q[ 4]+x[ 7]+0xfd469501,22);
Q[ 9]=Q[ 8]+RL(F(Q[ 8],Q[ 7],Q[ 6])+Q[ 5]+x[ 8]+0x698098d8, 7);
Q[10]=Q[ 9]+RL(F(Q[ 9],Q[ 8],Q[ 7])+Q[ 6]+x[ 9]+0x8b44f7af,12);
Q[11]=Q[10]+RL(F(Q[10],Q[ 9],Q[ 8])+Q[ 7]+x[10]+0xffff5bb1,17);
Q[12]=Q[11]+RL(F(Q[11],Q[10],Q[ 9])+Q[ 8]+x[11]+0x895cd7be,22);
Q[13]=Q[12]+RL(F(Q[12],Q[11],Q[10])+Q[ 9]+x[12]+0x6b901122, 7);
Q[14]=Q[13]+RL(F(Q[13],Q[12],Q[11])+Q[10]+x[13]+0xfd987193,12);

Q[19]=Q[18]+RL(G(Q[18],Q[17],Q[16])+Q[15]+x[11]+0x265e5a51,14);
Q[22]=Q[21]+RL(G(Q[21],Q[20],Q[19])+Q[18]+x[10]+0x02441453, 9);
Q[24]=Q[23]+RL(G(Q[23],Q[22],Q[21])+Q[20]+x[ 4]+0xe7d3fbc8,20);

.......... Here is the point of verification (POV) .................

Q[25]=Q[24]+RL(G(Q[24],Q[23],Q[22])+Q[21]+x[ 9]+0x21e1cde6, 5);
Q[28]=Q[27]+RL(G(Q[27],Q[26],Q[25])+Q[24]+x[ 8]+0x455a14ed,20);
```

Q9 Tunnel 我已經用了二次啦

If we add extra conditions that Q[10][i] = 0, Q[11][i] = 1

```
Q[ 8]=Q[ 7]+RL(F(Q[ 7],Q[ 6],Q[ 5])+Q[ 4]+x[ 7]+0xfd469501,22);
Q[ 9]=Q[ 8]+RL(F(Q[ 8],Q[ 7],Q[ 6])+Q[ 5]+x[ 8]+0x698098d8, 7);
Q[10]=Q[ 9]+RL(F(Q[ 9],Q[ 8],Q[ 7])+Q[ 6]+x[ 9]+0x8b44f7af,12);
Q[11]=Q[10]+RL(                Q[ 8] +Q[ 7]+x[10]+0xffff5bb1,17);
Q[12]=Q[11]+RL(         Q[10]        +Q[ 8]+x[11]+0x895cd7be,22);
Q[13]=Q[12]+RL(F(Q[12],Q[11],Q[10])+Q[ 9]+x[12]+0x6b901122, 7);
Q[14]=Q[13]+RL(F(Q[13],Q[12],Q[11])+Q[10]+x[13]+0xfd987193,12);

Q[19]=Q[18]+RL(G(Q[18],Q[17],Q[16])+Q[15]+x[11]+0x265e5a51,14); 2 c.(+1s.)
Q[22]=Q[21]+RL(G(Q[21],Q[20],Q[19])+Q[18]+x[10]+0x02441453, 9); 1 c.
Q[24]=Q[23]+RL(G(Q[23],Q[22],Q[21])+Q[20]+x[ 4]+0xe7d3fbc8,20); 1 c.

.......... Here is the point of verification (POV) ................

Q[25]=Q[24]+RL(G(Q[24],Q[23],Q[22])+Q[21]+x[ 9]+0x21e1cde6, 5);
Q[28]=Q[27]+RL(G(Q[27],Q[26],Q[25])+Q[24]+x[ 8]+0x455a14ed,20);

F(X,Y,Z) = XY + X’Z
```

Q9 Tunnel 我已經用了二次啦

We can modify Q[9] for different PoV result.

```
Q[ 8]=Q[ 7]+RL(F(Q[ 7],Q[ 6],Q[ 5])+Q[ 4]+x[ 7]+0xfd469501,22);
Q[ 9]=Q[ 8]+RL(F(Q[ 8],Q[ 7],Q[ 6])+Q[ 5]+x[ 8]+0x698098d8, 7);
Q[10]=Q[ 9]+RL(F(Q[ 9],Q[ 8],Q[ 7])+Q[ 6]+x[ 9]+0x8b44f7af,12);
Q[11]=Q[10]+RL(             Q[ 8] +Q[ 7]+x[10]+0xffff5bb1,17);
Q[12]=Q[11]+RL(       Q[10]       +Q[ 8]+x[11]+0x895cd7be,22);
Q[13]=Q[12]+RL(F(Q[12],Q[11],Q[10])+Q[ 9]+x[12]+0x6b901122, 7);
Q[14]=Q[13]+RL(F(Q[13],Q[12],Q[11])+Q[10]+x[13]+0xfd987193,12);

Q[19]=Q[18]+RL(G(Q[18],Q[17],Q[16])+Q[15]+x[11]+0x265e5a51,14); 2 c.(+1s.)
Q[22]=Q[21]+RL(G(Q[21],Q[20],Q[19])+Q[18]+x[10]+0x02441453, 9); 1 c.
Q[24]=Q[23]+RL(G(Q[23],Q[22],Q[21])+Q[20]+x[ 4]+0xe7d3fbc8,20); 1 c.

.......... Here is the point of verification (POV) ................

Q[25]=Q[24]+RL(G(Q[24],Q[23],Q[22])+Q[21]+x[ 9]+0x21e1cde6, 5);
Q[28]=Q[27]+RL(G(Q[27],Q[26],Q[25])+Q[24]+x[ 8]+0x455a14ed,20);

F(X,Y,Z) = XY + X'Z
```

Q9 Tunnel 我已經用了二次啦

Factoring 徒手拆鋼彈

Quadratic Sieve

你們兩個，乾脆交往算啦

- The second fastest method
- Fermat's factorization:
  - $a^2 - b^2 = 0 \mod n$
  - $(a+b)(a-b) = 0 \mod n$
  - $O(\sqrt{N})$ if searching a, b directly

We defined a factor base: $p \coloneqq \{p_0, p_1, p_2, \ldots, p_{\pi(B)}\} \coloneqq \{2, 3, 5, \ldots\}$
And then we finding some integers $r \;\; s.t. \;\; (r^2 \mod N) = \prod_i^{\pi(B)} p_i^{e_{ri}}$

Notice that
$r^2 s^2 = (rs)^2 = \prod_i^{\pi(B)} p_i^{e_{ri}} \prod_i^{\pi(B)} p_i^{e_{si}} = \prod_i^{\pi(B)} p_i^{e_{ri}+e_{si}}$. (mod N removed for simplicity)

After collecting enough pair of r and its corresponding e,
our goal (finding $a^2 = b^2$) can be changed to
find a linear combination of e vectors such that all elements are even.

Gaussian elimination under $GF(2)$ rocks.

Quadratic Sieve 你們兩個，乾脆交往算啦

To find some integers $r \quad s.t. \quad (r^2 \mod N) = \prod_i^{\pi(B)} p_i^{e_{ri}}$, we want:

1. $(r^2 \mod N)$ is small that it's more likely to be fully factorized with our base.
2. r^2 is larger then N that we won't got trivial relations.
3. We also need a fast algorithm to test whether it is fully factorizable.

Choose $(r^2 \mod N) = f(r) = \left(x + \lceil \sqrt{N} \rceil\right)^2 - N$, with small x.
Condition 1, 2 are satisfied.

Notice that $(r + kp)^2 = r^2 + 2rkp + k^2p^2 \equiv r^2 \pmod p$
We can solve $f(r) \equiv 0 \pmod p$ first, got two root α & β.
Then mark all $f(\alpha + kp)$ and $f(\beta + kp)$ has factor p.

Quadratic Sieve – Sieving 你們兩個，乾脆交往算啦

Elliptic Curve

是擅長分解的朋友！すごーい！

- Elliptic Curve factorization method (ECM)
- The third fastest method
- Great for removing small factors

Define a group EC_n with random elliptic curve under modulo $n = pq$,
It is actually a direct product of group $EC_p \times EC_q$.

P is a non-trivial point on EC_n, and P_p is its corresponding point on EC_p.

Assuming that order of EC_p is B-smooth, $[k]P_p = \infty$, $[k]P$ is undefined,
where k is $\prod_i^{\pi(B)} p_i$ (i.e. product of small primes.)
It also means when calculating the point, it's slope will be u/v with $v \,\%\, p = 0$.

Now if $\gcd(v, n) = p$, then we find it.

If $[k]P$ is well defined, which means both EC_p, EC_q aren't smooth, try again.

Elliptic Curve Method 是擅長分解的朋友！すごーい！

Discrete logarithm

Brute force

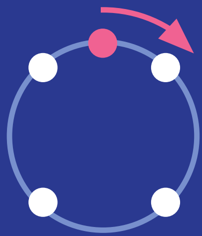
我們的戰鬥才剛剛開始

- Given y, find $g^x = y$ where g, y are elements of a multiplicative finite group, $x \in R$.
- N is the order of the group.
- Brute force is $O(N)$.

Baby/Giant step

我終於...
終於踏出了第一步

- The baby-step giant-step is a meet-in-the-middle algorithm for computing the discrete logarithm.
- Complexity is $O(\sqrt{N})$.
- $yg^{\sqrt{n}a} = g^b$



Pollard's rho

この瞬間を待っていたんだ！

- Reduced to collision finding:
  - $g^{\alpha}y^{\beta} = g^{A}y^{B}$
- Deterministic random walk based on last value:
  - $g^{\alpha_{i+1}}y^{\beta_{i+1}} = f(g^{\alpha}y^{\beta})$
- Collision with histories:
  - We entered a loop in $O(\sqrt{N})$

Pollard's rho

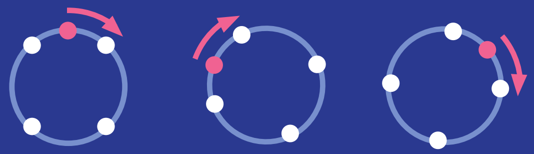
この瞬間を待っていたんだ！

- Floyd's cycle-finding algorithm
  - $g^{\alpha_{i+1}} y^{\beta_{i+1}} = f\left(g^{\alpha_i} y^{\beta_i}\right)$
  - $g^{\alpha_{2i+2}} y^{\beta_{2i+2}} = f\left(f\left(g^{\alpha_{2i}} y^{\beta_{2i}}\right)\right)$
  - $g^{\alpha_{i+1}} y^{\beta_{i+1}} = g^{\alpha_{2i+2}} y^{\beta_{2i+2}}$ in $O(\sqrt{\frac{\pi n}{2}})$

Pohlig–Hellman

佛山一個能打的都沒有！

- Order of the group N has k factors $N = \prod_i^k n_i$
- Solve $g^{(x \bmod n_i)N/n_i} = y^{N/n_i}$
- Reconstruct with CRT



[Lab] Pohlig-Hellman

Index calculus

夏亞，你算計我！夏亞！

- Sub exponential complexity
- Prerequisite: a factor base, an efficient factor algorithm in underlying group

We defined a factor base: $p \coloneqq \{p_0, p_1, p_2, \ldots, p_{\pi(B)}\} \coloneqq \{-1, 2, 3, 5, \ldots\}$

And then we finding some integers $r \;\; s.t. \;\; g^r = \prod_i^{\pi(B)} p_i^{e_{ri}}$

Notice that

$g^r g^s = g^{r+s} = \prod_i^{\pi(B)} p_i^{e_{ri}} \prod_i^{\pi(B)} p_i^{e_{si}} = \prod_i^{\pi(B)} p_i^{e_{ri}+e_{si}}.$

After collecting enough pair of r and its corresponding e,

A linear transformation to standard basis gives $\log(p_i)$.

Finally, find $s \;\; s.t. \;\; g^s y = \prod_i^{\pi(B)} p_i^{e_{si}} \quad \Rightarrow \quad x = -s + \sum_i^{\pi(B)} e_{si} \log(p_i)$

Index calculus 夏亞，你算計我！夏亞！