1. 보안의 3대 요소 CIA

(1) Confidentiality(기밀성) : 인가받지 않은 누군가가 민감한 정보를 볼 수 없어야 한다.
(2) Integrity(무결성) : 인가받지 않은 누군가가 함부로 정보를 변경할 수 없어야 한다.
(3) Availability(가용성) : 시스템(서비스)가 원하는 시점에 제공될 수 있어야 한다.

2. Beyond CIA

(1) Cryptography(암호화) : 시스템에 로그인한 사용자가 진짜 사용자 맞는지 확인
(2) Protocol(통신규약) : replay attack을 방지하기 위한 안전한 네트워크 인증 프로토콜 필요
(3) Access Control(접근제어) : 인증+권한부여
(4) Software(소프트웨어) : 암호화, 통신규약, 접근제어 구현
(5) OS(운영체제) : 크고 복잡한 소프트웨어

3. crypto-

(1) cryptography : 암호화
(2) cryptanalysis : 암호해독
(3) cryptology : 암호학
(4) crypto- : 위 세가지를 다 의미

4. Kerckhoff's principle

암호화 알고리즘은 공개되있으나 key는 비밀이다.

5. Exhaustive Key Search

가능한 모든 key를 시도해보는 방법

6. 암호 시스템 안전한가? 안전하지 않은가?

가능한 공격 방법이 exhaustive key search뿐이면 안전하고,
그것보다 저렴한 비용의 shortcut attack이 존재한다면 안전하지 않다.

7. Substitution Cipher

(1) Shift by 3(Caesar's Cipher)
(2) Shift by n : Exhaustive key search
(3) Any Permutation of letters : 26!경우의 수는 시간 걸리므로 통계적 분석 활용

8. One-time Pad

(1) Encryption : Plaintext xor Key = Ciphertext
(2) Decryption : Ciphertext xor Key = Plaintext
(3) 조건 : Key Stream이 랜덤이어야 하고 한번만 사용되어야 한다.
(4) 안전한가? : 아마도 안전하다
(5) 실용적인가? : 실용성 0

9. One-time Pad 활용 사례

(1) Project VENONA
(2) Zimmerman Telegram
(3) Election of 1876
=> 같은 key 반복 사용

10. Confusion vs Diffusion

(1) Confusion : Substitution, One-Time Pad -> Plaintext에 포함된 글자가 Ciphertext에 없음
(2) Diffusion : Permutation, Double Transposition : Plaintext에 포함된 글자가 Ciphertext에 있음

11. Ciphertext Only(암호문 공격)

(1) 언제 : Ciphertext를 알고 있을 때
(2) 목적 : 알고 있는 Ciphertext에 대한 Plaintext를 얻는게 목적
(3) 방법 : Exhaustive key search, 통계적 분석

12. Known Plaintext(알려진 평문 공격)

(1) 언제 : 몇 가지 Plaintext에 대해 Ciphertext를 가지고 있을 때
(2) 목적 : 원하는 Plaintext에 대한 Ciphertext를 얻는게 목적
(3) Ciphertext only보다 주어진 정보가 많아 공격자가 더 선호하는 공격 방법이다.

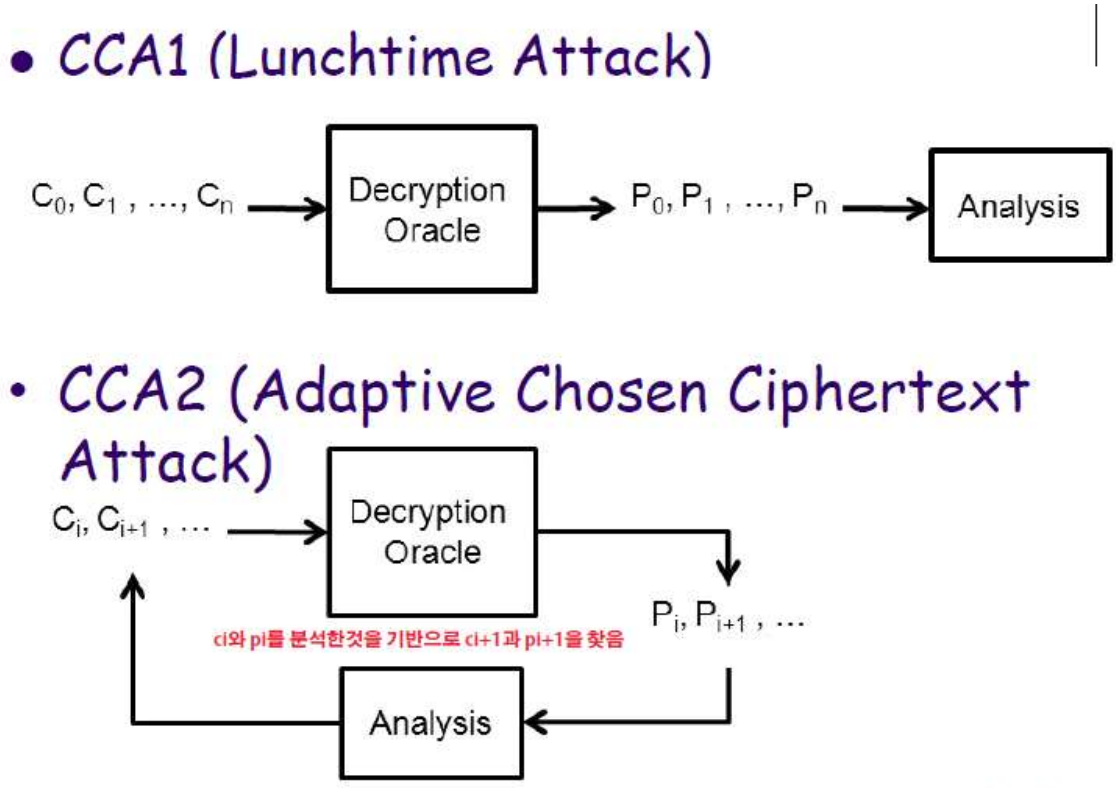
13. Chosen Plaintext(선택된 평문 공격)

(1) 언제 : 원하는 Plaintext에 대해 Ciphertext를 알 수 있을 때
(2) 목적 : Sender가 보낸 Plaintext에 대한 Ciphertext를 얻는게 목적
(3) 자신이 선택한 t개의 Plaintext에 대한 Ciphertext 수집이 먼저 필요
(4) 현실적으로 가능하다.

14. Chosen Ciphertext(선택된 암호문 공격)

(1) 언제 : 원하는 Ciphertext에 대해 Plaintext를 알 수 있을 때
(2) 목적 : Key를 알아내는 것이 목적
(3) Lunchtime attack : 이전에 선택한 Ciphertext가 다음에 선택할 Ciphertext에 영향 X
(4) Adaptive Chosen-Ciphertext : 이전에 선택한 Ciphertext가 다음에 선택할 Ciphertext에 영향 O

15. Lunchtime attack(CCA1) vs Adaptive Chosen-Ciphertext(CCA2)



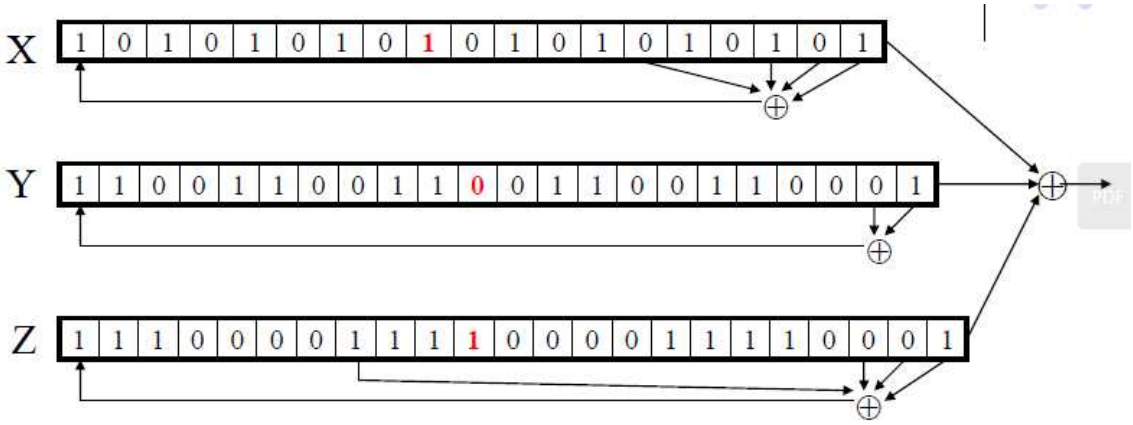
16. Stream Cipher

(1) One-Time Pad와 비슷(XOR을 통한 암호화, 복호화)
(2) 짧은 key가 주어지고 이를 Plaintext길이에 맞게 long keystream으로 확장
(3) One-Time Pad와 다르게 안전하지 않다.(key가 message보다 짧기 때문)
(4) Confusion Only
(5) 종류 : A5/1, RC4

17. A5/1

(1) bit 단위의 key
(2) 초기화를 위한 64bit의 초기 key 필요(x-19 + y-22 + z-23)
(3) m=maj(x8, y10, z10) : 0이 많으면 0, 1이 많으면 1
(4) x8=m -> t = x13 xor x16 xor x17 xor x18
(5) y10=m -> t = y20 xor y21
(6) z10=m -> t = z7 xor z20 xor z21 xor z22
(7) keystream bit : x18 xor y21 xor z22
(8) x, y, z 중 적어도 두 개의 값이 항상 바뀜
(9) 하드웨어 기반

(ex)



m = maj(x8, y10, z10) = maj(1, 0, 1) = 1

key stream bit = 0 xor 1 xor 0 = 1

18. RC4

(1) byte 단위의 key

(2) 256 bytes의 lookup table 기반 (0~255의 permutation이 lookup table에 저장)

(3)

```
for i = 0 to 255
        S[i] = i
        K[i] = key[i (mod N)]
next i
j = 0
for i = 0 to 255
        j = (j + S[i] + K[i]) mod 256
        swap(S[i], S[j])
next j
i = j = 0
```

```
i = (i + 1) mod 256
j = (j + S[i]) mod 256
swap(S[i], S[j])
t = (S[i] + S[j]) mod 256
```

(4) **keystreamByte = S[t]**

(5) 첫 256bytes는 버리고 그 다음 byte부터 사용하는 것이 안전하다.

(6) 소프트웨어 기반

19. Block Cipher

(1) 고정된 길의의 block 단위로만 암호화, 복호화

(2) Confusion, Diffusion

(3) 동일한 round 반복

(4) 이전 round output이 다음 round input이 된다.

(5) 종류 : Feistel Cipher, Substitution-Permutation Networks

(6) 기반 암호화 알고리즘 : DES, AES

20. Feistel Cipher

(1) 입력으로 주어진 Plaintext를 반으로 잘라 사용

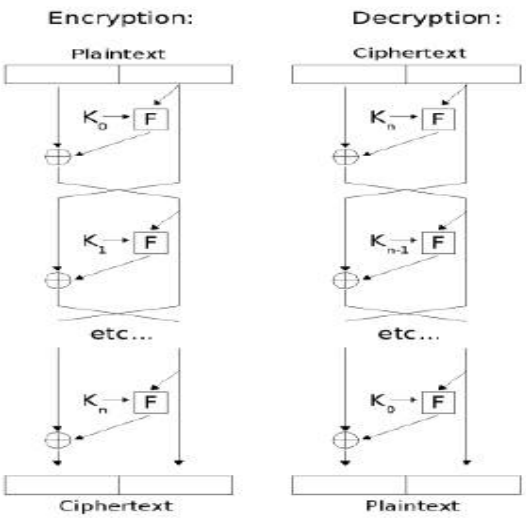
(2) Encryption : Li = Ri-1 / Ri=Li-1 xor F(Ri-1, Ki)

(3) Decryption : Ri-1 = Li / Li-1 = Ri xor F(Ri-1, Ki)

(4) F함수는 어떤 함수든 가능

(5) 안전한 F함수 설계는 필요

(6) 기반 암호화 알고리즘 : DES



Feistel Cipher

21. Substitution-Permutation(SP) Networks

(1) Substitution -> Confusion, Permutation -> Diffusion
(2) 기반 암호화 알고리즘 : AES
(3) F함수는 역함수가 존재해야 한다.
(4) Encryption 알고리즘과 Decryption 알고리즘이 다르다.

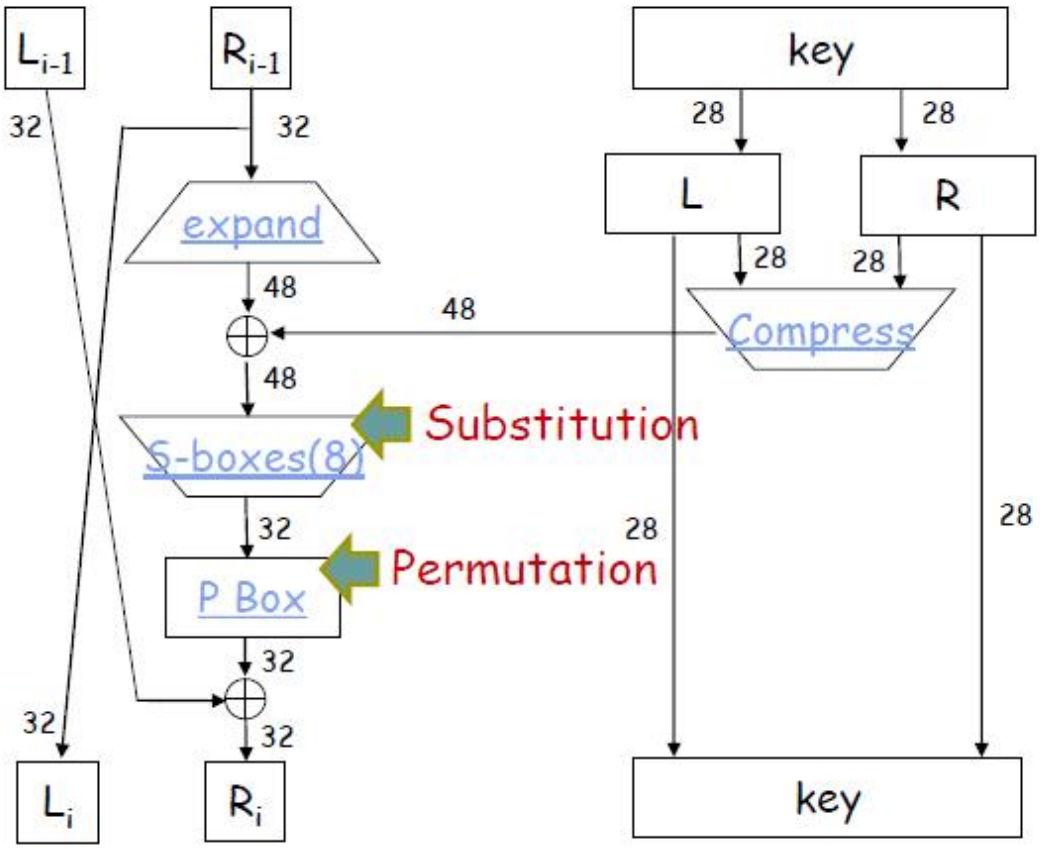
22. Feistel Cipher vs SP networks

(1) Feistel Cipher의 장점
동일한 알고리즘으로 암호화, 복호화 가능
F함수 제약이 없다

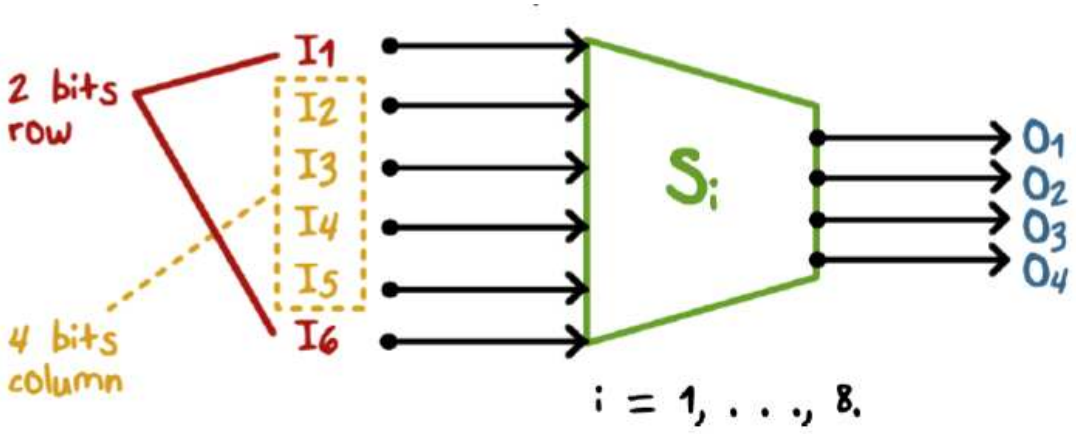
(2) SP networks 장점
입력으로 주어진 데이터 모두가 바뀌므로 안전하다.

23. Data Encryption Standard(DES)

(1) Feistel Cipher 기반
(2) S-Box에 보안이 집중되있다.(S-box 제외 모두 linear -> 공격 하기 쉬움)
(3) One Round of DES



(4) S-box



(5) S-box Example

- 8 "substitution boxes" or S-boxes
- Each S-box maps 6 bits to 4 bits
- S-box number 1

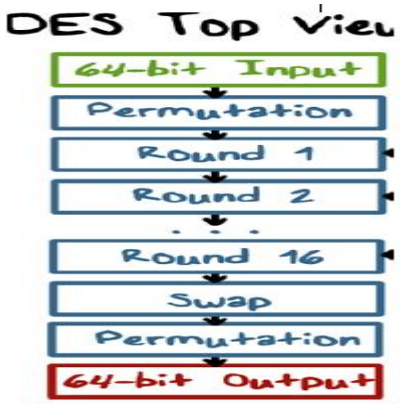
Input: 011011
Output: ??

input bits (0,5) ↓ input bits (1,2,3,4)

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	1110	0100	1101	0001	0010	1111	1011	1000	0011	1010	0110	1100	0101	1001	0000	0111
01	0000	1111	0111	0100	1110	0010	1101	0001	1010	0110	1100	1011	1001	0101	0011	1000
10	0100	1111	0001	1100	1011	0111	1110	1010	0101	0110	1000	0000	0011	1101	0010	1001
11	1111	1100	1000	0010	0111	1101	0001	0110	1011	0011	1110	0101	0000	1010	1001	0100

Output : 0101

(6)



첫 permutation과 마지막 permutation끼리 상쇄 /

첫 permutation, swap, 마지막 permutation은 보안상 의미X

(7) 안전한가? 안전하다. 하지만, key 길이가 짧은게 단점

24. Double DES(C = E(E(P, K), K))

(1) 여전히 key길이는 56bit이므로 짧다.
(2) Meet in the middle attack이 가능하다

25. Meet in the middle attack

CPA를 활용한 공격
(1) 특정한 Plaintext P를 선택
(2) 가능한 모든 Ki에 대해 C'=E(Ki, P)를 구한다.
(3) 1에서 선택한 P에 대한 Ciphertext C를 얻는다.
(4) 가능한 모든 K에 대해 C''=D(K, C)를 구한다.
(5) C'=C''인 곳의 Ki, K가 Double DES에 활용된 Key다.
(6) Meet in the middle attack은 미리 계산해 놓을 수 있어 시간 소요가 적다.
-> shortcut attack 존재 => 안전하지 않다.

26. Triple DES(C=E(D(E(P, K1),K2),K1))

(1) single DES와의 backward 호환을 위해 E,D,E 사용
(2) 112bit key 길이로 충분

27. AES

(1) 3DES는 너무 느리기 때문에 AES의 입지가 좋다.
(2) SP Networks 기반
(3) 4bytes = 1 word
(4) 10round -> 11개의 round key 필요
(5) 10 round -> key 128bit / 12 round -> key 192 bit / 14 round -> key 256 bit
(6) 4 Functions : 1 Permutation + 3 Substitution /
3 Layers : Linear, Nonlinear and Key addition
(7) Permutation : ShiftRow(Linear)
(8) Substitution : ByteSub(Nonlinear), MixColumn(Nonlinear), AddRoundKey(Key addition)

28. Byte Substitution

(1) Confusion
(2) S-Box Lookup Table을 기반으로 Substitution을 수행한다.

(ex)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

HEX 19는 HEX D4로 치환된다.

29. Shift Row

(1) Diffusion

(2) 각 행에 대해 행-1만큼 left shift

(ex) 1행은 0번 left shift / 2행은 1번 left shift ....

30. Mix Columns

(1) Confusion & Diffusion

(2) 마지막 round에서는 제외된다.

(3) 행렬 곱

31. Add Round Key

(1) Confusion

(2) State s와 round key 두 table의 같은 bit 끼리 XOR하여 새로운 state를 생성한다.