

블록암호 (1)

2022년 9월 21일 수요일

정보보호

충남대학교 정보보호연구실 허강준

중요공지!!!

- kheo@islab.work 으로 메일 보내지 마세요
 - 나는 답장을 보냈는데 못받았다는 얘기가 속출하고 있습니다...
 - 분명 스팸으로 들어갔을거 같긴 한데...
 - 아무튼!
- knowledge@o.cnu.ac.kr (학교메일) 로 보내주시고
 - 이것도 영 못미덥다 싶으면 knowledge@patche.me (개인메일)로 보내주세요
 - 둘 다 폰에 실시간으로 알림 뜨니까 제가 못 볼일도 없을거여요

Enigma Revisited

- <https://piotte13.github.io/enigma-cipher/>

PLUG SETTING: [10 PAIRS ONLY]

RESET MACHINE: RESET

RING SETTING: << 0 >> << 0 >> << 0 >>

[10 PAIRS ONLY]

<< B >>

<< III >>

<< II >>

<< I >>

ETW

PlugBoard

<< B >>

<< III >>

<< II >>

<< I >>

ETW

PlugBoard

<< B >>

<< III >>

<< II >>

<< I >>

ETW

PlugBoard

<< B >>

<< III >>

<< II >>

<< I >>

ETW

PlugBoard

<< B >>

<< III >>

<< II >>

<< I >>

ETW

PlugBoard

<< B >>

<< III >>

<< II >>

<< I >>

ETW

PlugBoard

<< B >>

<< III >>

<< II >>

<< I >>

ETW

PlugBoard

<< B >>

<< III >>

<< II >>

<< I >>

ETW

PlugBoard

<< B >>

<< III >>

<< II >>

<< I >>

ETW

PlugBoard

<< B >>

<< III >>

<< II >>

<< I >>

ETW

PlugBoard

<< B >>

<< III >>

<< II >>

<< I >>

ETW

PlugBoard

<< B >>

<< III >>

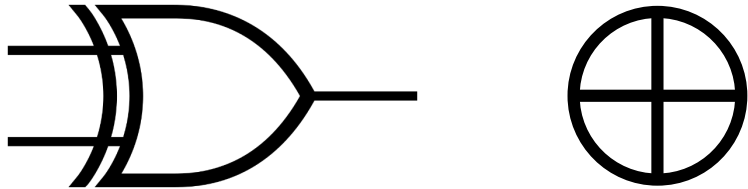
<< II >>


<< I >>

ETW

XOR (Exclusive-OR)

- Back to the 논리회로...
 - 같으면 0, 다르면 1...
 - 없으면 붙이고 있으면 없애고?



	0	1
0	0	1
1	1	0

XOR (Exclusive-OR)

- 암호에 사용해 볼까?

평문: 10110011 11110000 00001111 11101110 11111111

암호키: 10000001 10000001 10000001 10000001 10000001

암호문: 00110010 01110001 10001110 01101111 01111110

암호키: 10000001 10000001 10000001 10000001 10000001

평문: 10110011 11110000 00001111 11101110 11111111

XOR (Exclusive-OR)

- One Time Pad

- 평문의 길이와 키 길이가 같은 암호
- 절! 대! 못 뚫는다!

평문:	10110011	11110000	00001111	11101110	11111111
암호키:	10000001	11111111	11110000	11001100	00001111
암호문:	00110010	00001111	11111111	00100010	11110000
암호키:	10000001	11111111	11110000	11001100	00001111
평문:	10110011	11110000	00001111	11101110	11111111

- 이렇게나 강력한데... 왜 대중화 되지 않았을까

XOR (Exclusive-OR)

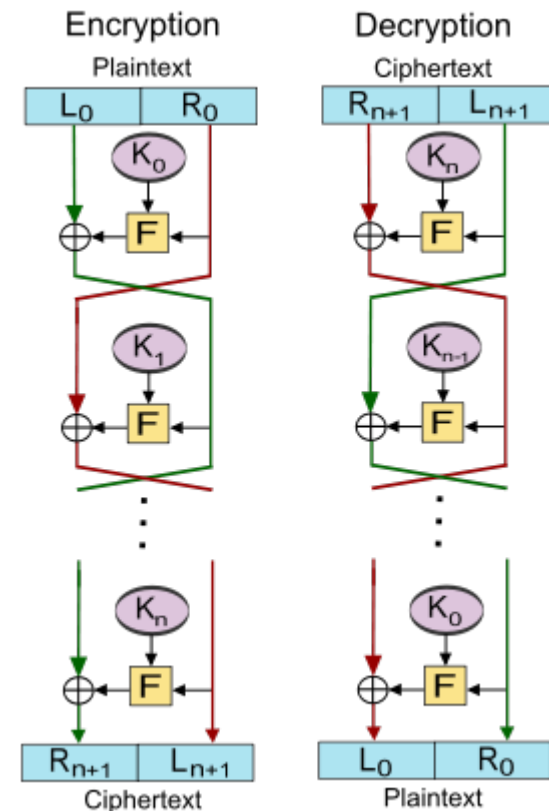
- One Time Pad
 - 생각해봅시다...
 - 평문과 같은 길이의 키를 안전하게 교환할 수 있다면?
 - 왜 평문을 직접 교환하지 않고?
- 하지만 정말로 강력한데 써먹을 방법이 좀 없을까?

대칭키 암호 – 블록 암호 알고리즘

- 대칭키 (Symmetric Key)
 - 암호화-복호화에 사용하는 키가 서로 같다!
 - 비대칭키(e.g. 공개키 암호) 와 구분
- 블록 암호 알고리즘
 - DES (Data Encryption Standard) -- 오늘의 주제
 - AES (Advanced Encryption Standard)

DES – Data Encryption Standard

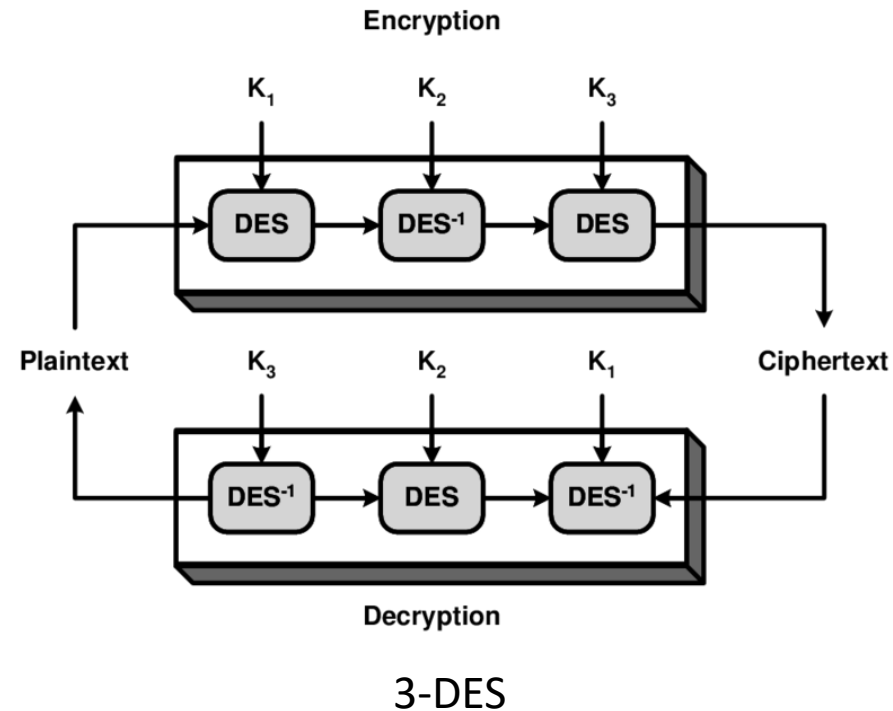
- 1975년 개발, 1977년 표준화
- Feistel 구조 알고리즘
- 56비트 키, 64비트 블록
 - 실제로는 64비트 키... 지만 오류 검출용 패리티 8비트 포함
- 지금은 사용 X
 - 56비트를 누구 코에 붙여?
 - 발전된 암호 해독 기술들(차분 분석, 선형 분석, ...)
 - 그래서...



Feistel 구조 예시

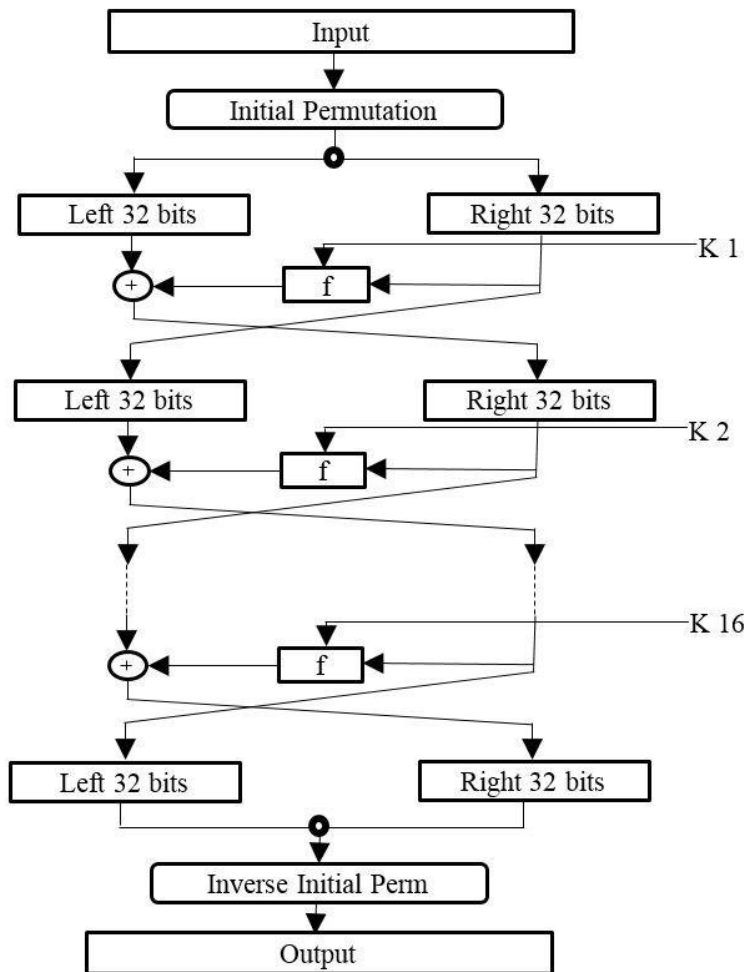
DES – Data Encryption Standard

- 112비트 (혹은 168비트) 로 확장한 3-DES
 - 는 그저 DES를 3개 붙였을 뿐
 - 이 역시 **사용하면 안됨!**



DES – Data Encryption Standard

- 재미로 알아보자



DES – Data Encryption Standard

- Permutation: 평문 → 재배치
 - Initial Permutation (IP)

Input

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64



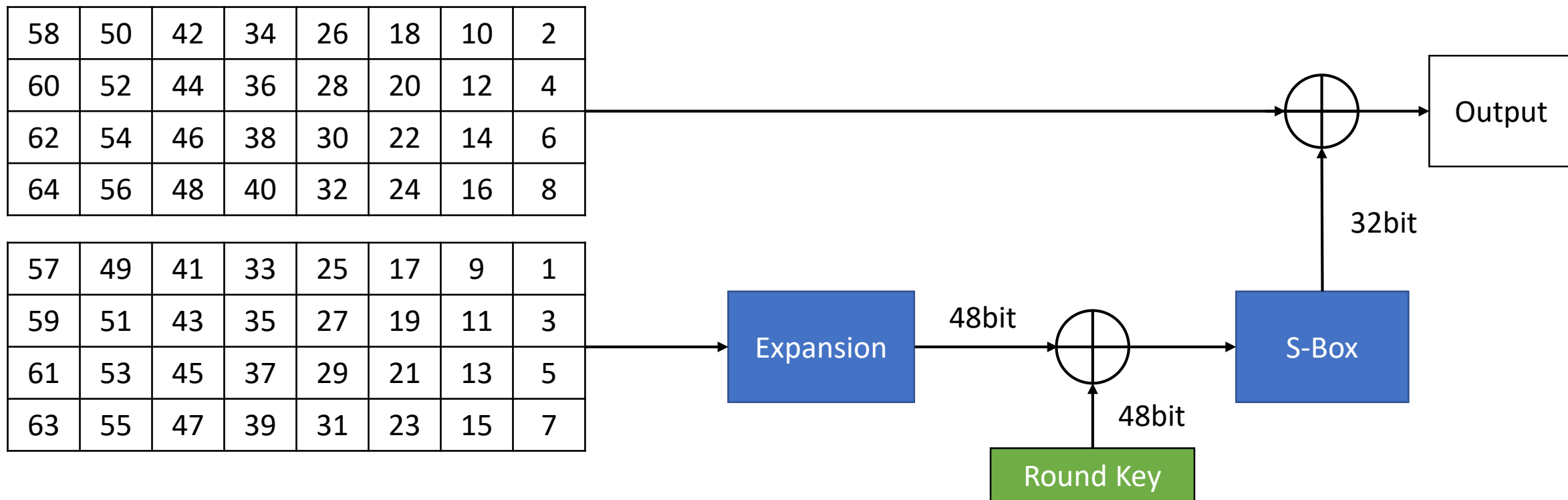
IP(Input)

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

DES – Data Encryption Standard

- 라운드 함수

- 입력을 반으로 32비트씩 (L, R) 뚝 자르고 번갈아가며 투입
- DES는 16라운드까지 있음!
- 라운드마다 바뀌는 키가 계속 변함



DES – Data Encryption Standard

- 라운드 함수 (Expansion)
 - 32비트 입력 → 48비트로 확장

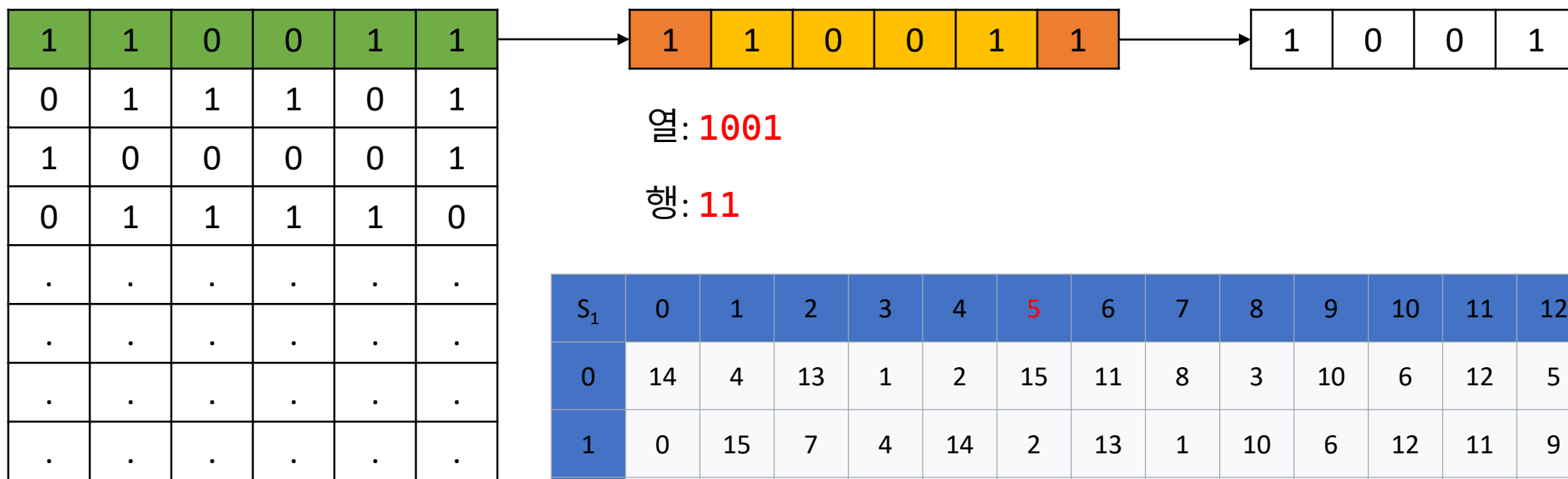
1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32



32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

DES – Data Encryption Standard

- 라운드 함수 (S-Box, 총 8개)
 - 6비트씩 선택 → 4비트 선택 → 총 32비트로 축소

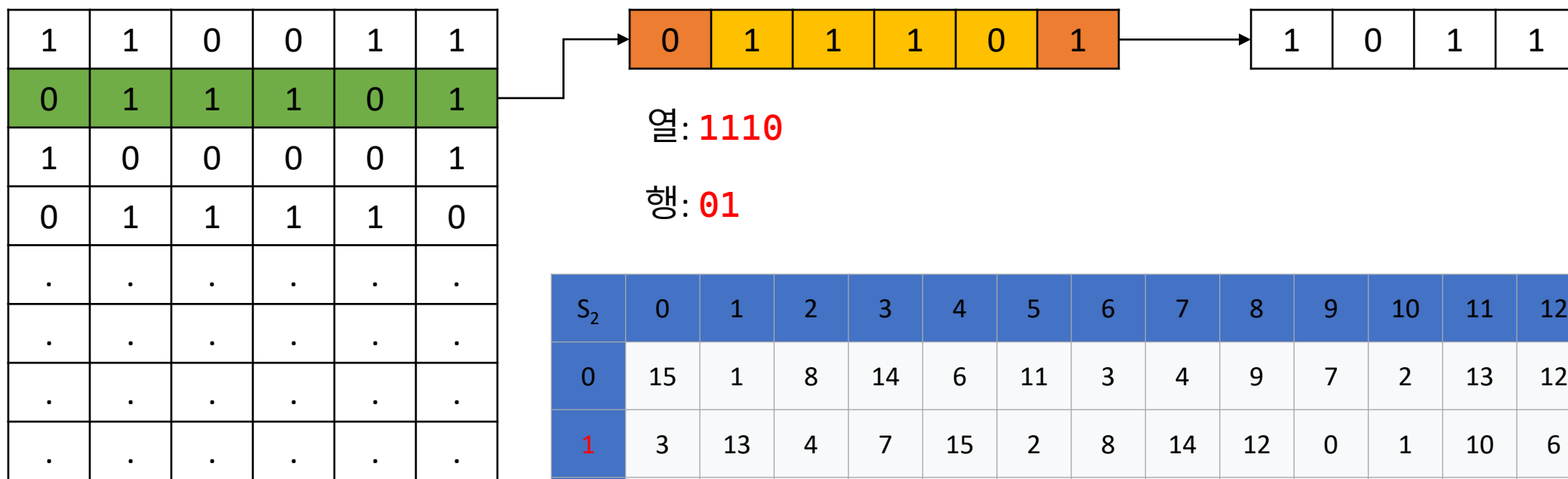


S ₁	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

https://en.wikipedia.org/wiki/DES_supplementary_material

DES – Data Encryption Standard

- 라운드 함수 (S-Box, 총 8개)
 - 6비트씩 선택 → 4비트 선택 → 총 32비트로 축소

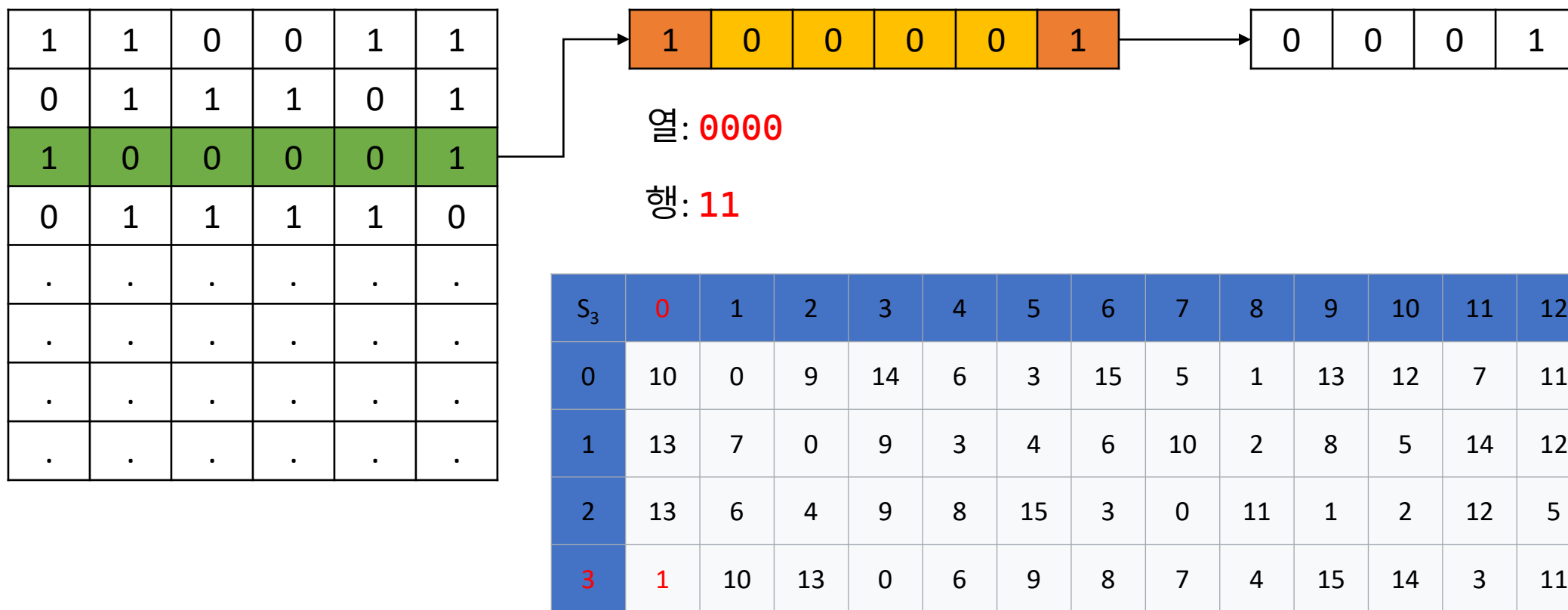


S ₂	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

https://en.wikipedia.org/wiki/DES_supplementary_material

DES – Data Encryption Standard

- 라운드 함수 (S-Box, 총 8개)
 - 6비트씩 선택 → 4비트 선택 → 총 32비트로 축소



https://en.wikipedia.org/wiki/DES_supplementary_material

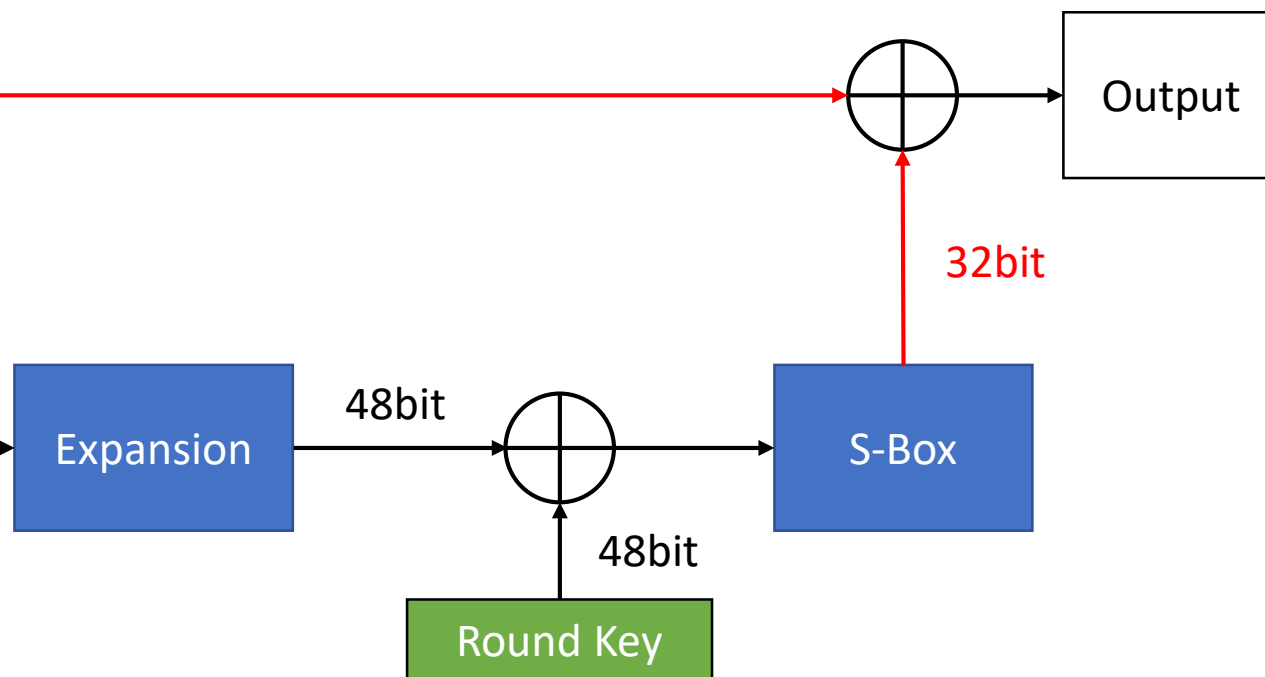
DES – Data Encryption Standard

- 라운드 함수

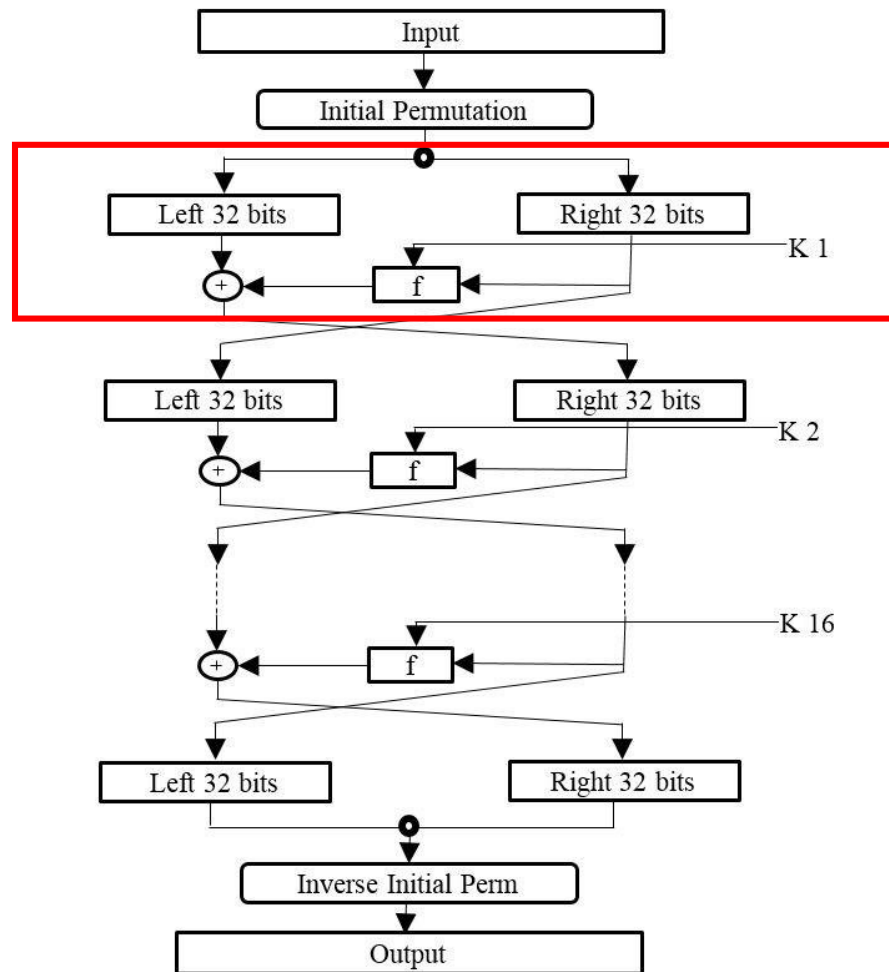
- 입력을 반으로 32비트씩 (L, R) 뚝 자르고 번갈아가며 투입
- DES는 16라운드까지 있음!
- 라운드마다 바뀌는 키가 계속 변함

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8

57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

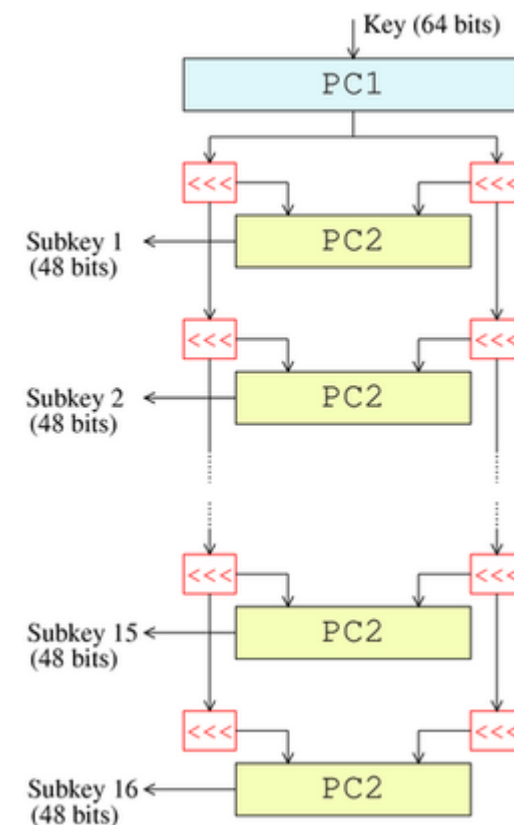


DES – Data Encryption Standard



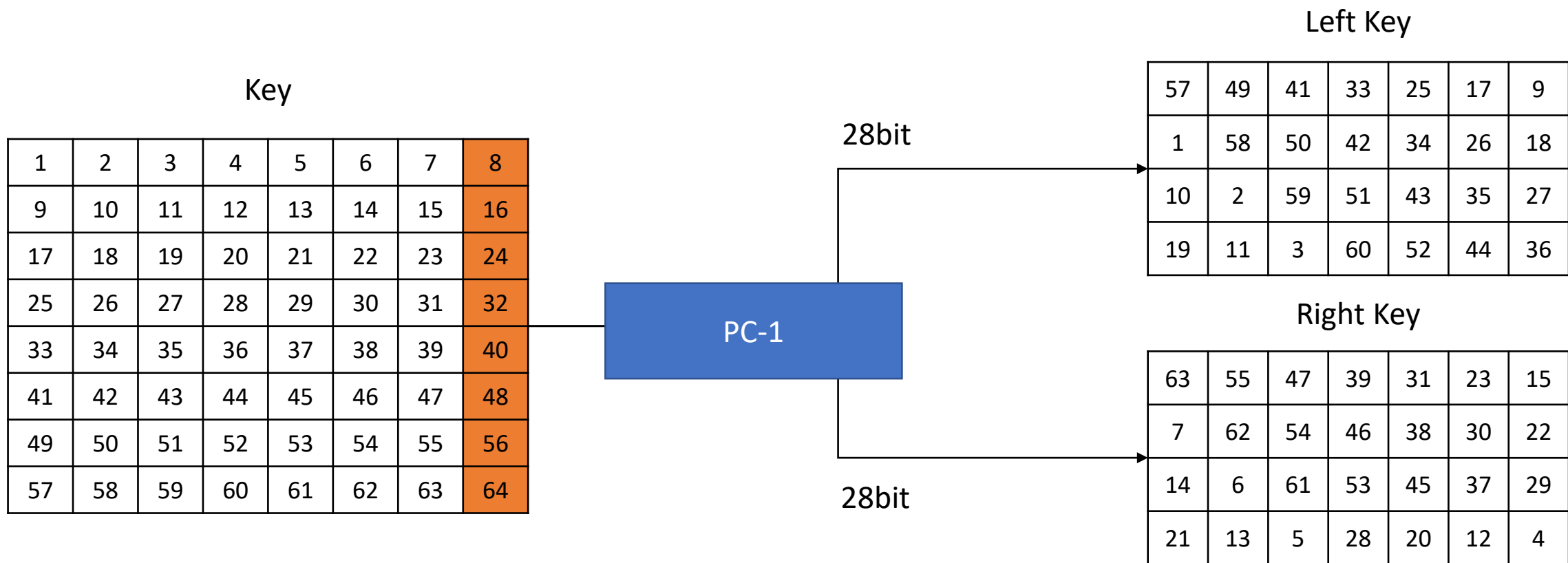
DES – Data Encryption Standard

- 라운드 키 (K1~K16) 생성
 - 키 스케줄링 (Key Scheduling) 이라고도 불러요
 - 각 라운드 함수마다 사용되는 보조 키 (Subkey)



DES – Data Encryption Standard

- 라운드 키 (K1~K16) 생성
 - 64비트 키 입력 → 56비트 선택 (PC-1) → 반으로 뚝!



DES – Data Encryption Standard

- 라운드 키 (K1~K16) 생성
 - 매 라운드마다 Key Rotation 실행 (1, 2, 9, 16 라운드)

Left Key							Right Key						
57	49	41	33	25	17	9	63	55	47	39	31	23	15
1	58	50	42	34	26	18	7	62	54	46	38	30	22
10	2	59	51	43	35	27	14	6	61	53	45	37	29
19	11	3	60	52	44	36	21	13	5	28	20	12	4

Left Key							Right Key						
36	57	49	41	33	25	17	4	63	55	47	39	31	23
9	1	58	50	42	34	26	5	7	62	54	46	38	30
18	10	2	59	51	43	35	22	14	6	61	53	45	37
27	19	11	3	60	52	44	29	21	13	5	28	20	12



DES – Data Encryption Standard

- 라운드 키 (K1~K16) 생성
 - 매 라운드마다 Key Rotation 실행 (그 외)

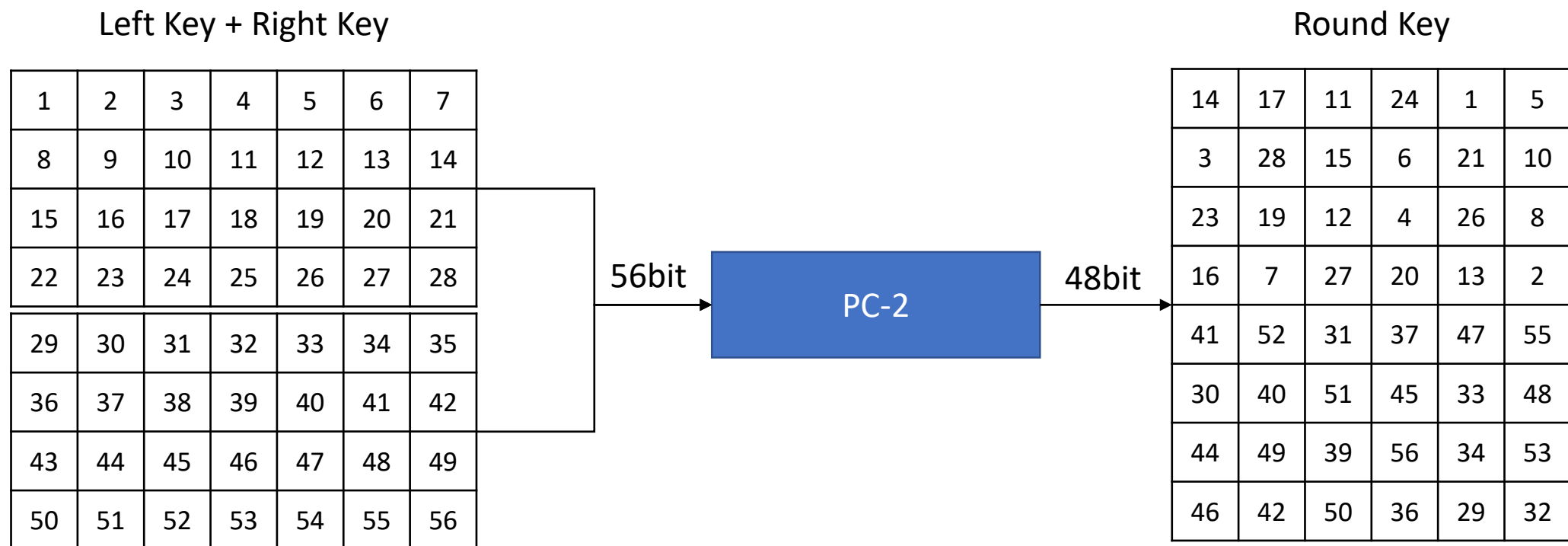
Left Key							Right Key						
57	49	41	33	25	17	9	63	55	47	39	31	23	15
1	58	50	42	34	26	18	7	62	54	46	38	30	22
10	2	59	51	43	35	27	14	6	61	53	45	37	29
19	11	3	60	52	44	36	21	13	5	28	20	12	4

Left Key							Right Key						
44	36	57	49	41	33	25	12	4	63	55	47	39	31
17	9	1	58	50	42	34	23	15	7	62	54	46	38
26	18	10	2	59	51	43	30	22	14	6	61	53	45
35	27	19	11	3	60	52	37	29	21	13	5	28	20

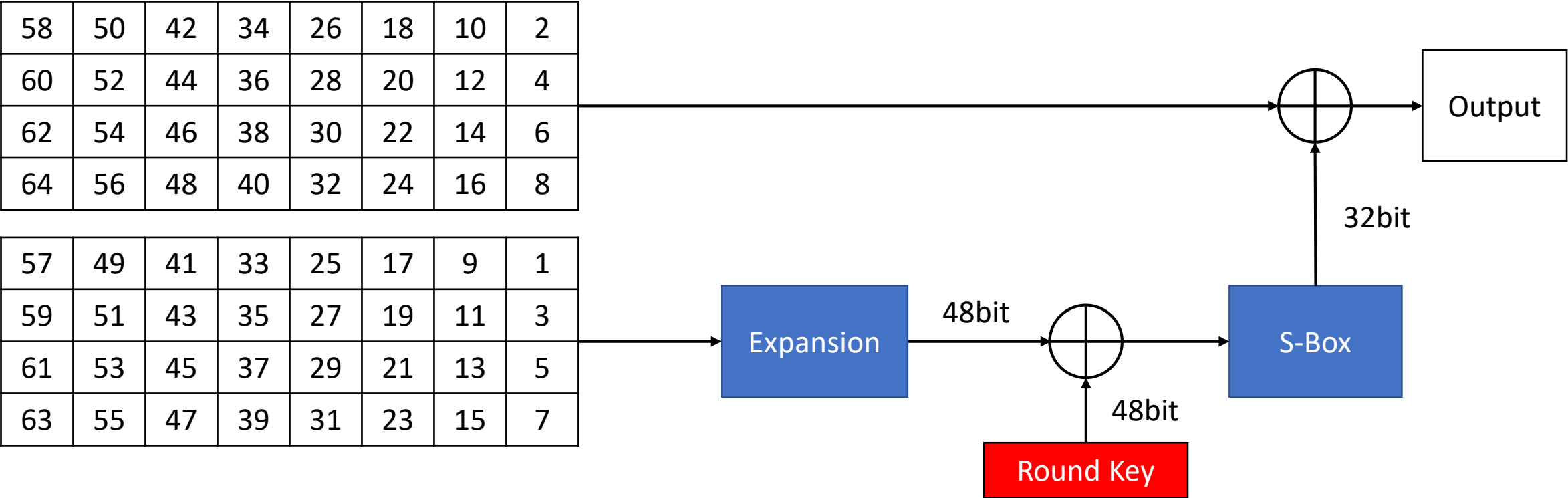


DES – Data Encryption Standard

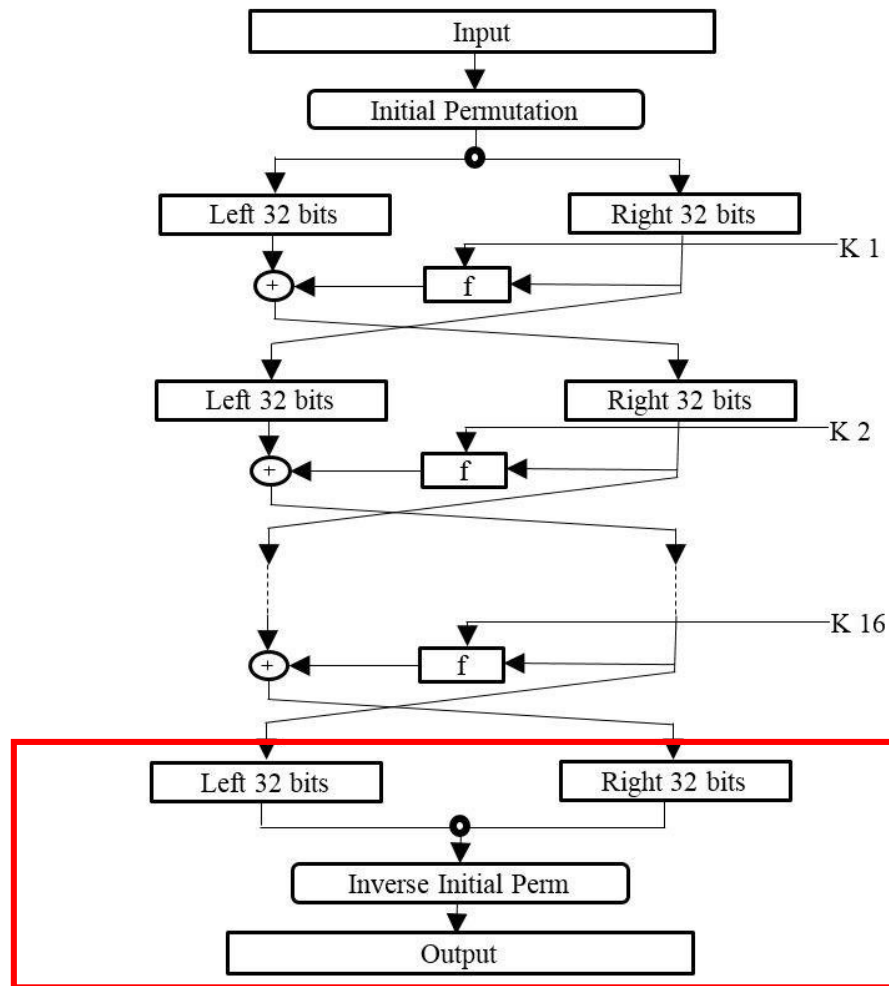
- 라운드 키 (K1~K16) 생성
 - 합친 뒤 한번 더 섞기 (PC-2) → 48비트 라운드 키



DES – Data Encryption Standard



DES – Data Encryption Standard



DES – Data Encryption Standard

- Final Permutation(FP)
 - 처음 Permutation(IP)의 역연산(Inverse)

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32

33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

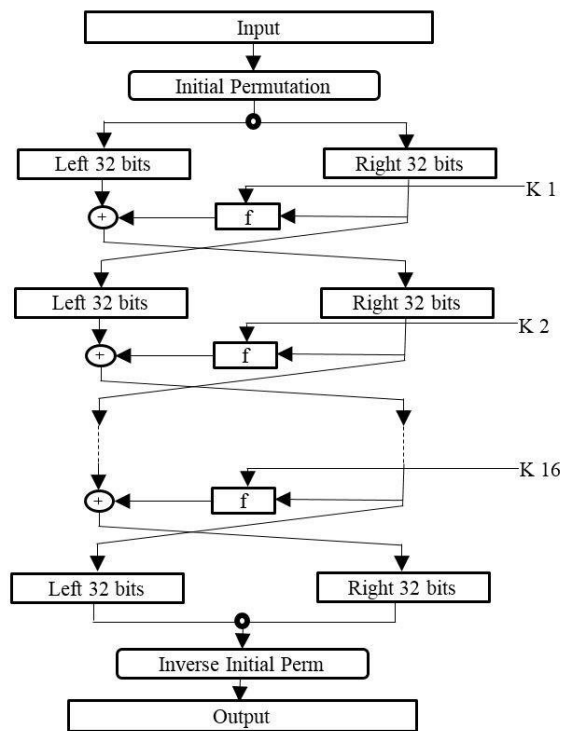


“DES 암호화 결과”

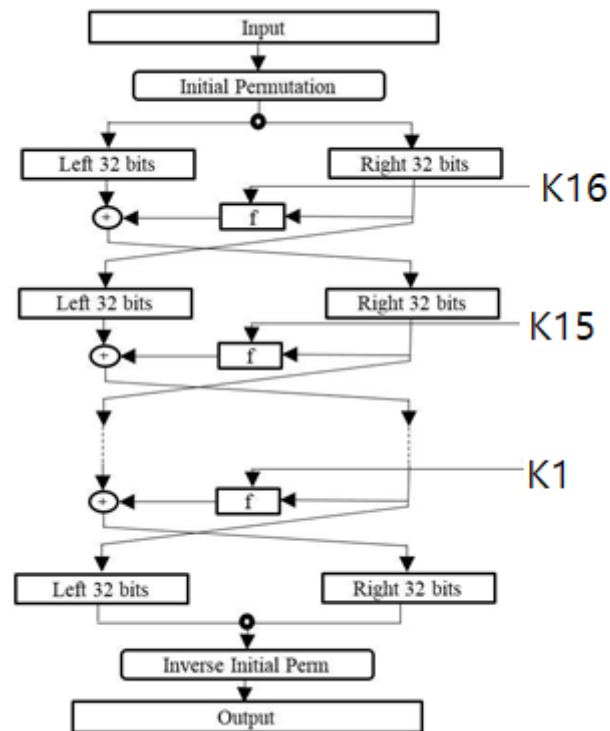
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

DES – Data Encryption Standard

- 복호화?
 - 라운드 키를 거꾸로 집어넣으면 된다



DES 암호화



DES 복호화

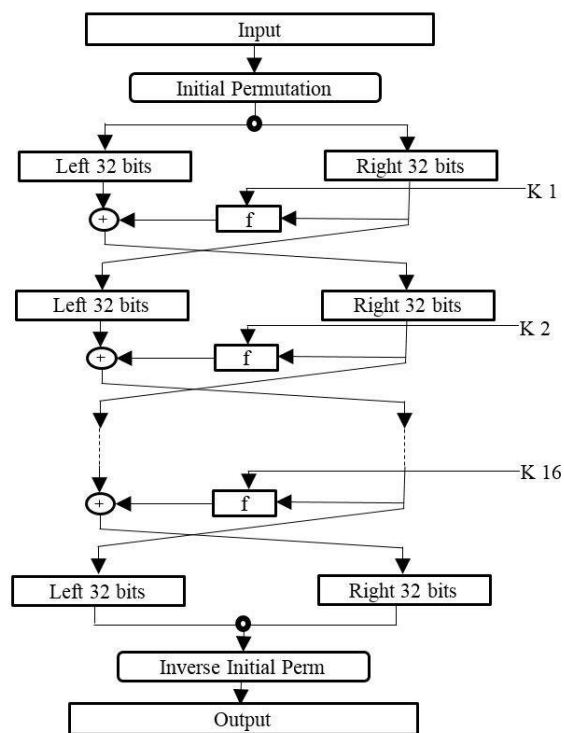
DES – Data Encryption Standard

- 의문점

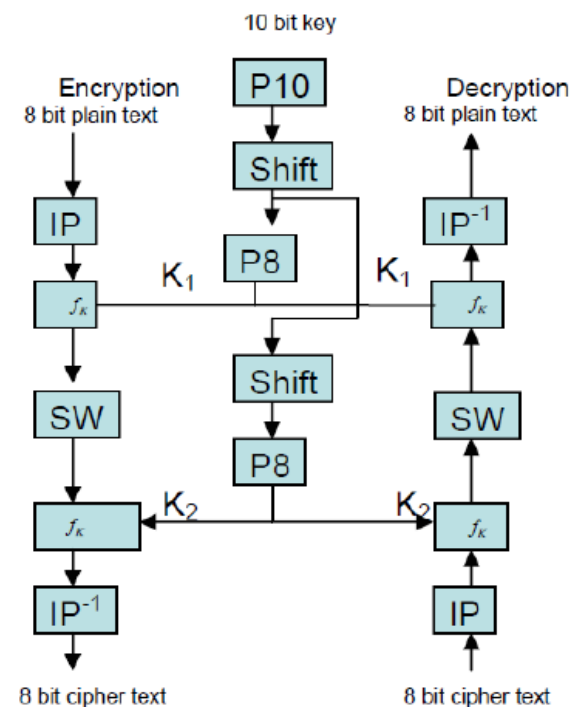
- DES나 AES는 길이 상관 없이 전부 암호화 가능
- 그런데 왜 알고리즘에서는 블록 크기 만큼만 암호화 하지?
 - 블록 크기보다 작거나 큰 데이터는 어떻게 암호화 하지?
- 다음주 이론 시간 (or) 다다음주 실습 때 설명
 - 선행학습을 위한 키워드: 운용 모드, 패딩

- S-DES (Simplified DES) 구현하기

<https://github.com/CNUCSE-InformationSecurity-2022-Fall/Simplified-DES>



DES



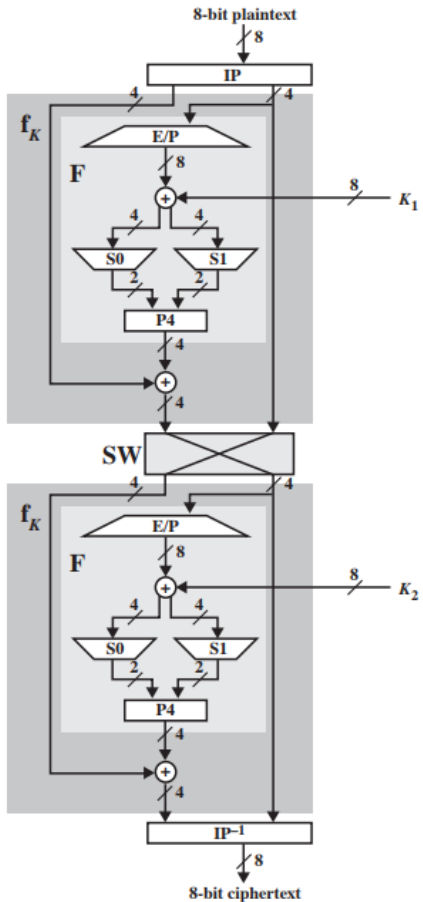
S-DES

- DES vs S-DES

	DES	S-DES
블록 크기	64비트	8비트
키 길이	56비트	10비트
라운드 수	16라운드	2라운드

과제

- 구현해야 할 부분
 - `sdes(text, key, mode)`



```
...  
sdes: encrypts/decrypts plaintext or ciphertext.  
mode determines that this function do encryption or decryption.  
    MODE_ENCRYPT or MODE_DECRYPT available.  
...  
def sdes(text: bytearray, key: bytearray, mode) -> bytearray:  
    result = bytearray()  
  
    # Place your own implementation of S-DES Here  
  
    return result
```


과제

- 미리 구현된 부분들
 - 각종 Permutation 용 P-Box, S-Box들
 - 키 스케줄러 (라운드 키 생성기) → 리스트 형태로 리턴
 - 라운드 함수

```
'''
schedule_keys: generate round keys for round function
returns array of round keys.
keep in mind that total rounds of S-DES is 2.
'''
def schedule_keys(key: bytearray) -> list[bytearray]:
    round_keys = []
    permuted_key = bytearray()

    for i in P10:
        permuted_key.append(key[i])

    permuted_key_left = permuted_key[0:5]
    permuted_key_right = permuted_key[5:10]

    for i in range(1, 3):
        # shift for each round
        # round 1: shift 1, round 2: shift 2
        # shifting will be accumulated for each rounds
        permuted_key_left = permuted_key_left[i:] + permuted_key_left[0:i]
        permuted_key_right = permuted_key_right[i:] + permuted_key_right[0:i]

        # merge and permute with P8
        merge_permutation = permuted_key_left + permuted_key_right
        round_key = bytearray()

        for j in P8:
            round_key.append(merge_permutation[j])

        round_keys.append(round_key)

    return round_keys
```

```
'''
round: round function
returns the output of round function
'''
def round(text: bytearray, round_key: bytearray) -> bytearray:
    # implement round function
    expanded = bytearray()
    for i in EP:
        expanded.append(text[i])
    expanded ^= round_key

    # S0
    s0_row = expanded[0:4]
    s0_sel_row = (s0_row[0] << 1) + s0_row[3]
    s0_sel_col = (s0_row[1] << 1) + s0_row[2]
    s0_result = ba_util.int2ba(S0[s0_sel_row][s0_sel_col], length=2)

    # S1
    s1_row = expanded[4:8]
    s1_sel_row = (s1_row[0] << 1) + s1_row[3]
    s1_sel_col = (s1_row[1] << 1) + s1_row[2]
    s1_result = ba_util.int2ba(S1[s1_sel_row][s1_sel_col], length=2)

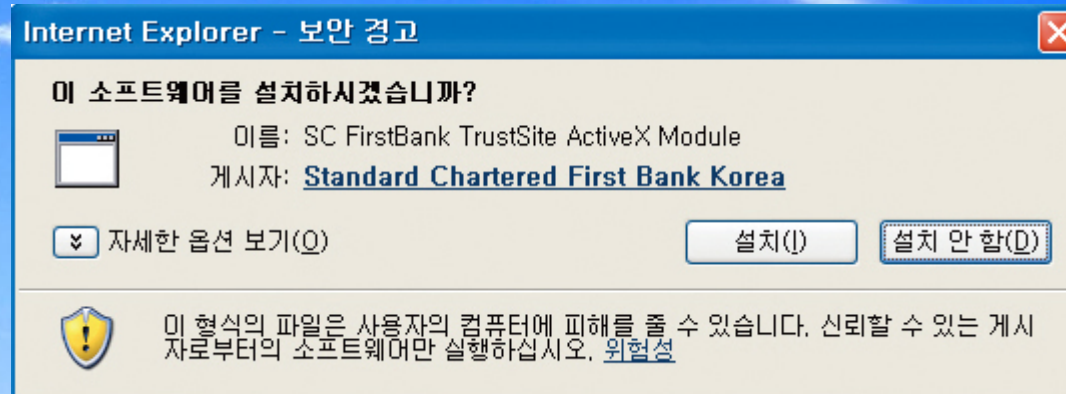
    pre_perm4 = s0_result + s1_result

    result = bytearray()
    for i in P4:
        result.append(pre_perm4[i])

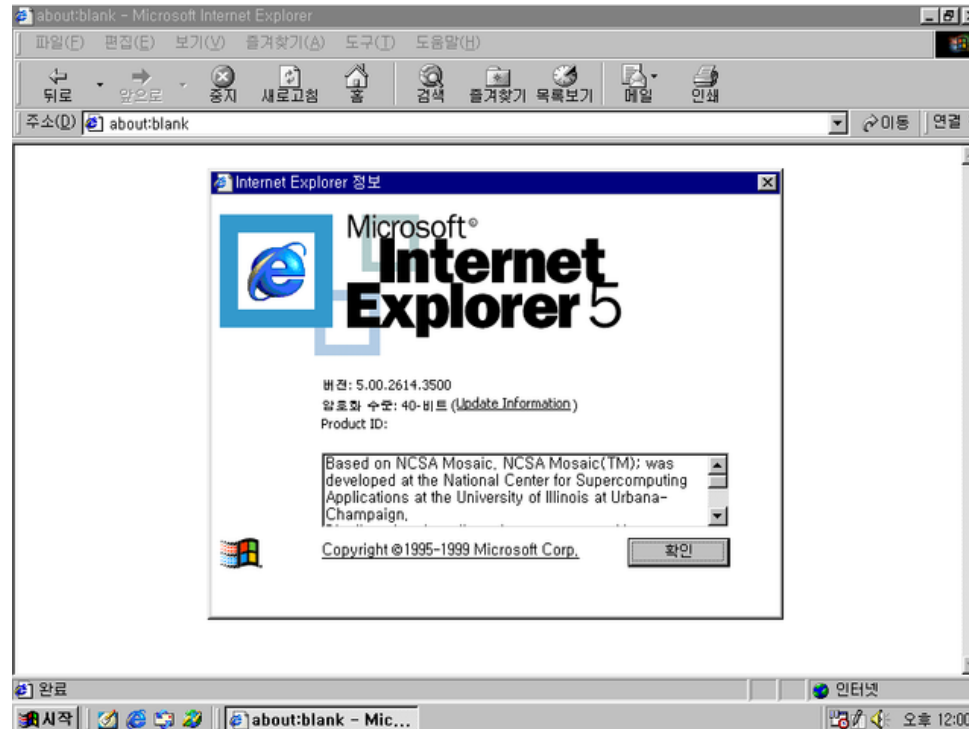
    return result
```

- bitarray 패키지 설치 필요 (PIP에서 설치)
 - `pip install bitarray`
- 학부생 때 재미삼아 구현했던 DES/S-DES
 - <https://github.com/CNUCSE-InformationSecurity-2022-Fall/Simplified-DES/blob/main/reference/des.py> (Python)
 - <https://github.com/CNUCSE-InformationSecurity-2022-Fall/Simplified-DES/blob/main/reference/sdes.hpp> (C++)

보안을 위해 Internet Explorer가 이 사이트에서 사용자의 컴퓨터로 ActiveX 컨트롤을 설치하는 것을 차단했습니다. 옵션을 보려면 여기를 클릭하십시오. X

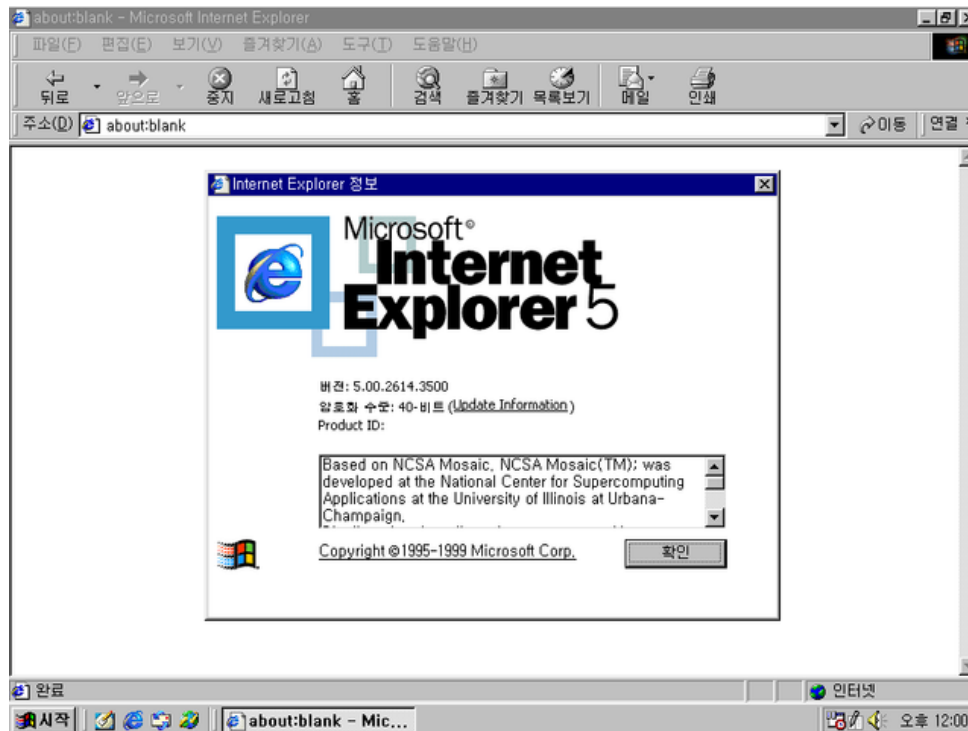


- 때는 세기 말, 인터넷뱅킹좀 해볼까 싶었다. 그런데...

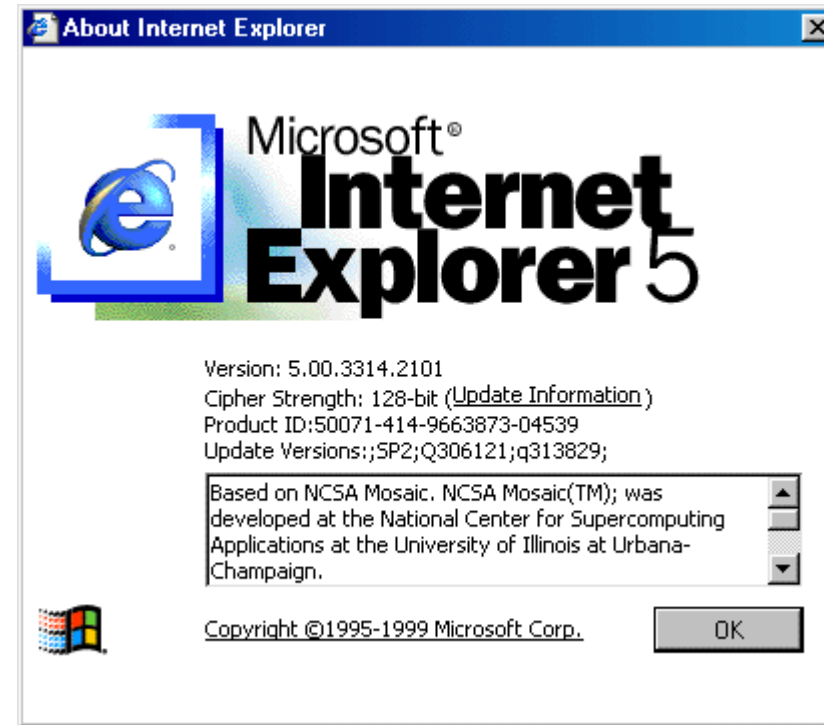


- $2^{40} = 1,099,511,627,776$ (약 1조)

- 고강도 암호 기술은 전략물자
- 美 정부의 수출 통제로 인해 수출판 IE의 보안 통신은 없는 수준



한국판 IE 5.0



Meanwhile in United States...

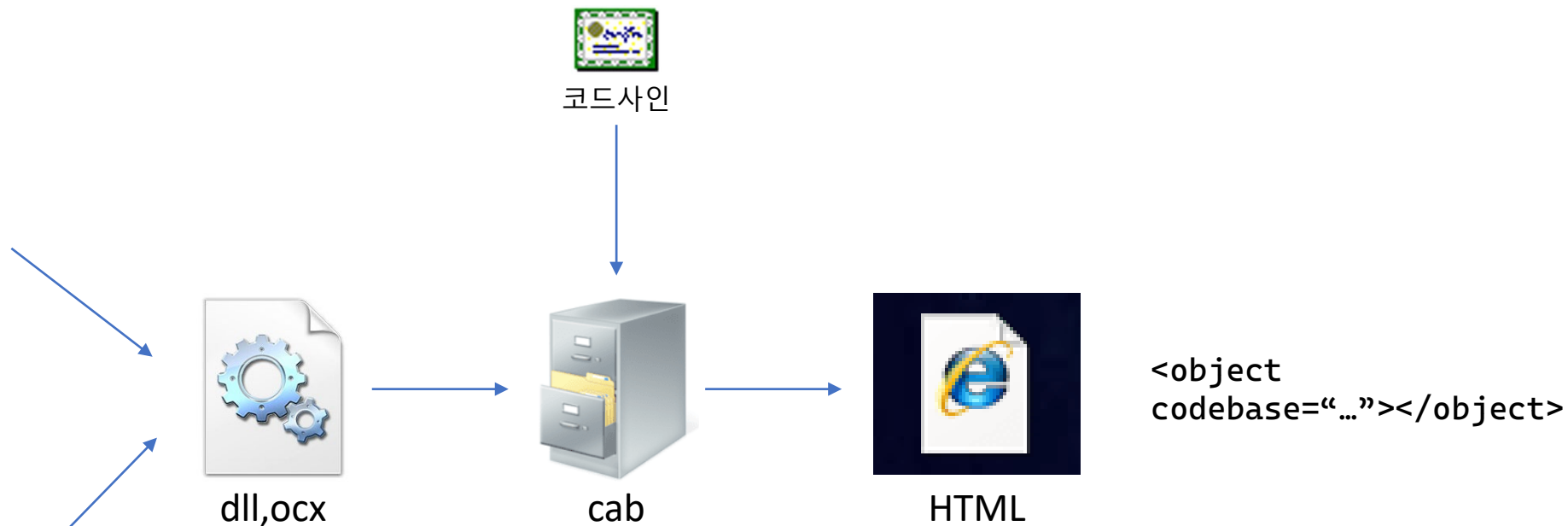
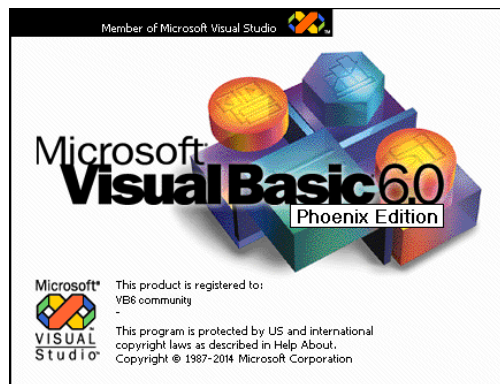
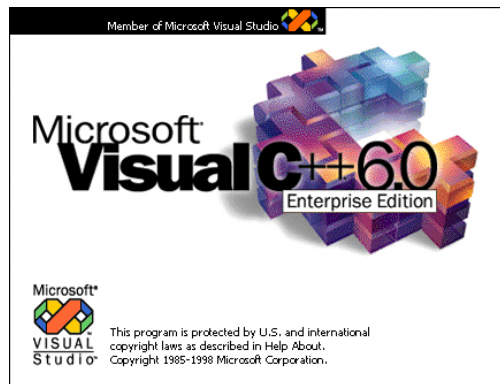
보안을 위해 Internet Explorer가 이 사이트에서 사용자의 컴퓨터로 ActiveX 컨트롤을 설치하는 것을 차단했습니다. 옵션을 보려면 여기를 클릭하십시오. x

KISA (주) 한국정보보호진흥원 (연) KISA 한국인터넷진흥원
KOREA INTERNET & SECURITY AGENCY

- 그래서 새로운 암호 알고리즘을 만들었다! (SEED)
 - 그런데 비표준이네? → 브라우저에서 지원을 안하네?
 - 어떻게 실행하지?



“그 티브 X”



- 다행히(?) 웹 표준 유행에 따라 Active X는 퇴출

MTO 머니투데이  구독

[단독] '액티브X' 없애라 했더니... "새 프로그램 설치"

입력 2014.12.29. 오전 5:45 · 수정 2014.12.29. 오전 9:26  기사원문

김평화 기자 >

 76

 541



가



[머니투데이 김평화 기자][정부, 온라인쇼핑몰 업계에 연말까지 액티브-X 대체 방화벽프로그램 설치 통보]

앞으로 온라인 쇼핑몰을 이용하려면 액티브-X(Active-X)가 아닌 다른 종류의 보안프로그램을 설치해야 한다. 최근 미래창조과학부와 금융위원회가 온라인쇼핑몰 업체들을 소집해 연말까지 액티브-X 시스템을 모두 없애라며 요구한 조건이다. 일단 액티브-X만 아니면 된다는 것이어서 '눈 가리고 아웅'식 정책에 업계의 불만이 속출하고 있다.

28일 관계부처와 업계에 따르면 미래부와 금융위는 최근 온라인 쇼핑몰 업체들과 회의를 열고 이달 31일까지 액티브-X를 모두 없애라고 업체측에 통보했다. 대신 내년부터 온라인 쇼핑몰 이용자는 액티브-X 대신 확장자 이름이 '.exe'인 새로운 방화벽 프로그램을 설치해야 한다.

- 인줄 알았죠?

- 더 골때리는 보안 플러그인 (EXE)



설치



서비스 등록
(메모리 상주, 자동시작)

실행



Web Server

보안 기능이 구현된
로컬 웹서버 (프록시)

• 더 골때리는 보안 플러그인 (EXE)



보안 기능 실행 API 요청
(키보드 보안, 공인인증서 등...)

실행 결과를 JSON
등으로 송신

Web Server

Task Manager

File Options View

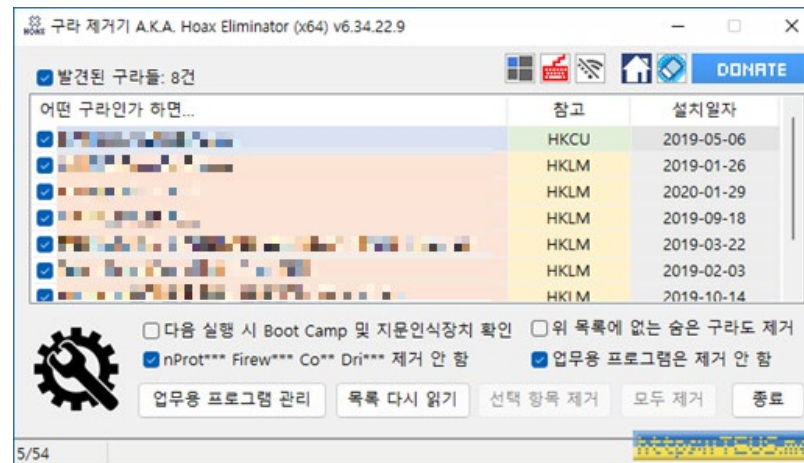
Processes Performance App history Startup Users Details Services

Name	PID	Status	User name	CPU	Memory (ac...	UAC virtualizati...
	6368	Running	SYSTEM	00	316 K	Not allowed
	6192	Running	SYSTEM	00	284 K	Not allowed
	11072	Running	LOCAL SER...	00	4,444 K	Not allowed
	22716	Running	patche	00	28,416 K	Disabled
	6948	Running	patche	00	9,192 K	Disabled
	56888	Running	SYSTEM	01	10,308 K	Not allowed
	6200	Running	SYSTEM	00	660 K	Not allowed
	52748	Running	patche	00	148 K	Disabled
	45380	Running	patche	00	156 K	Disabled
	26672	Running	patche	00	144 K	Disabled
	26836	Running	patche	00	1,964 K	Disabled
	2884	Running	SYSTEM	00	80 K	Not allowed
	11396	Running	SYSTEM	00	4,872 K	Not allowed
	6160	Running	SYSTEM	00	336 K	Not allowed
	44364	Running	patche	00	9,972 K	Disabled
	14416	Running	patche	00	124 K	Disabled
	33152	Running	patche	00	54,744 K	Disabled
	25944	Running	SYSTEM	00	6,448 K	Not allowed

^ Fewer details End task

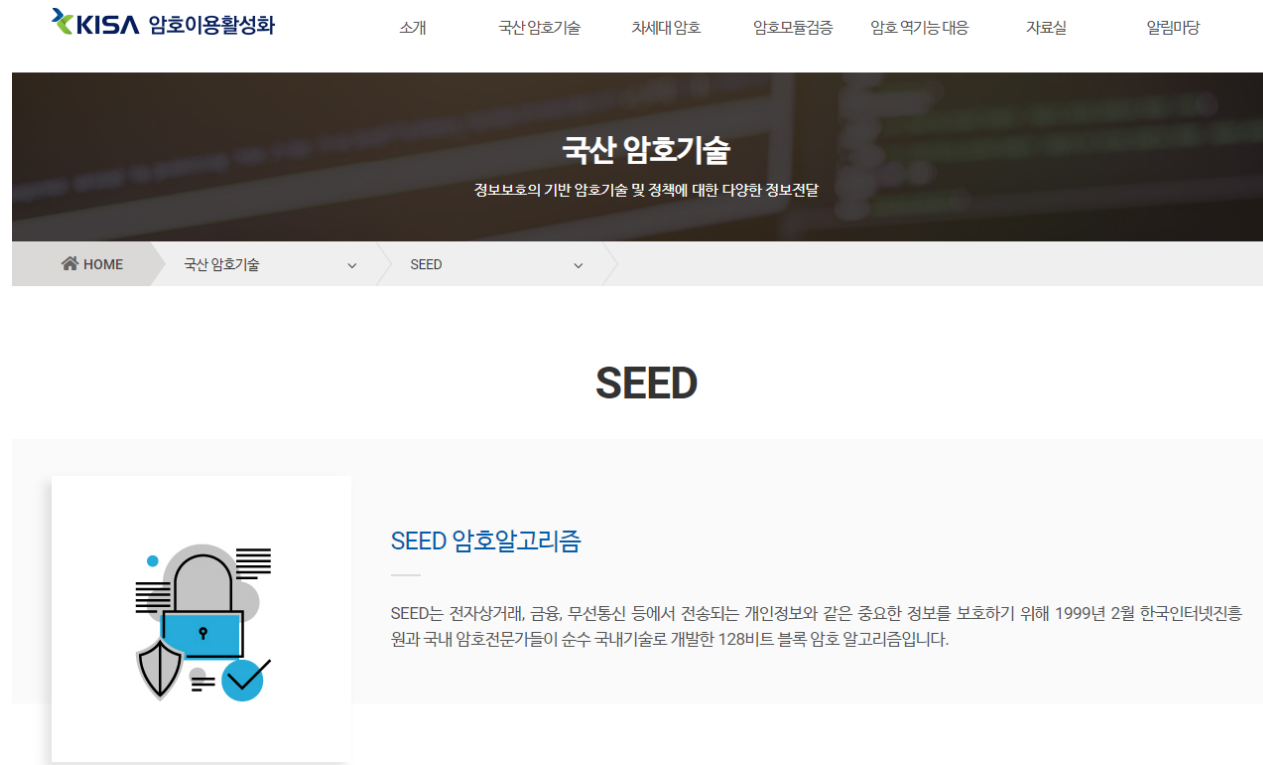
시스템 최고 권한(NT AUTHORITY\SYSTEM)으로
실행 중인 “보안” 프로그램

- 시대적 배경에 따라 도입되었다지만...
- 보안을 저해하는 “보안” 프로그램
 - 컴퓨터도 느려지게 한다
 - 싹 다 날려주는 구라제거기 (<https://teus.me>)



- 다른 이야기는 인증서 때 더 풀기로 하고...

- 참고로 SEED 외에도 ARIA, HIGHT, LSH 등 여러 국산 암호가 개발
- <https://seed.kisa.or.kr/kisa/algorithm/EgovSeedInfo.do>
 - 설계 문서, 레퍼런스 소스 등 열람 가능



주차	실습 주제	과제	날짜
1	오리엔테이션 & 썰풀기	과제를 위한 GitHub 설정	9/7
2	카이사르&비즈네르 암호	ENIGMA	9/14
3	XOR과 블록암호	Simplified DES 구현하기	9/21
4	여러가지 블록암호	블록암호를 이용하여 암호통신기 완성하기	9/28
5	블록암호 운용모드	S-DES-CBC, S-DES-ECB 구현하기	10/5
6	RSA	RSA 구현하기, 저강도 RSA 크랙하기	10/12
7	해시	암호통신기에 무결성 검증 기능 추가하기	10/19
8	중 간 고 사 (10/24)		공강
9	메세지 인증코드(MAC)	HMAC 구현하기	11/2
10	디지털 서명	사실인증서 생성 및 프로그램 코드 서명	11/9
11	하이브리드 암호	하이브리드 기반 암호 통신기	11/16
12	난수	시드값 추측을 이용한 암호문 크랙	11/23
13	블록체인과 머클 트리	머클트리 구현하기	11/30
14	TLS와 PGP(GPG)	GPG를 이용하여 암호 메일 보내기	12/7
15	기 말 고 사 (12/12)		종강

질문?

- 없으면 자리에서 일어나셔도 좋습니다 :)
- **대 학 원 입 학 문 의**는 언제나 환영
 - 블록체인, Web 3, 해킹 관심있거나 유경험자 우대

입학문의

- 류재철 교수님 (jcryou [at] cnu.ac.kr)
- 허강준 조교 (knowledge [at] o.cnu.ac.kr)