(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2021-0037945
(43) 공개일자 2021년04월07일

(51) 국제특허분류(Int. Cl.)
H04L 9/06 (2006.01) *H04L 9/00* (2006.01)
(52) CPC특허분류
H04L 9/0631 (2013.01)
H04L 9/003 (2013.01)
(21) 출원번호 10-2019-0120487
(22) 출원일자 2019년09월30일
심사청구일자 2019년09월30일

(71) 출원인
국민대학교산학협력단
서울특별시 성북구 정릉로 77 (정릉동, 국민대학교)
고려대학교 산학협력단
서울특별시 성북구 안암로 145, 고려대학교 (안암동5가)
(72) 발명자
김한기
서울특별시 성북구 정릉로 77 국민대학교 과학관 317호
김종성
서울특별시 성북구 정릉로 77 국민대학교 수학과 휘랑관 706호
(뒷면에 계속)
(74) 대리인
특허법인영비

전체 청구항 수 : 총 20 항

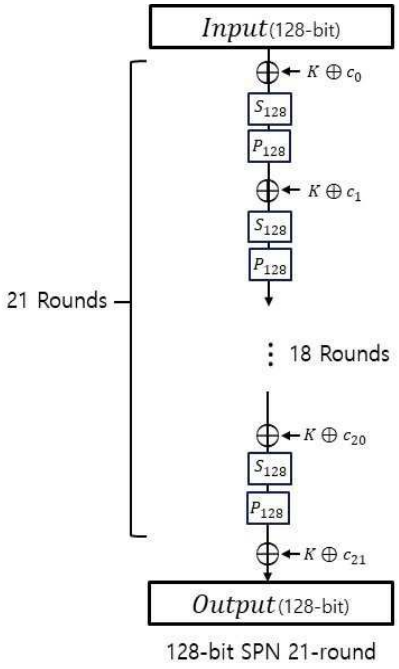
(54) 발명의 명칭 **부채널 공격 대응이 용이한 128비트 경량 블록 암호화 방법 및 이를 이용한 장치**

(57) 요 약

본 개시서에는 암호학적으로 데이터를 처리하는 방법 및 이를 이용한 장치가 개시된다. 구체적으로, 본 개시서에 따른 방법에 의하면, 소정 라운드 횟수인 자연수 M에 대하여, 제1 내지 제M+1 라운드 키를 생성하고 생성된 상기 제1 내지 제M+1 라운드 키를 이용한 암호화로써 평문으로부터 암호문을 산출하는바, 구체적으로, 마스터 키

(뒷면에 계속)

대 표 도 - 도4



128-bit SPN 21-round

를 생성하고, 상기 마스터 키와 라운드별 상수에 대해 XOR을 포함한 연산을 수행한 결과값으로서 제1 내지 제M+1 라운드 키를 생성하며, 상기 라운드별 상수는 라운드마다 상이하게 정해진다. 상기 평문으로부터 상기 암호문을 산출함은, 상기 제1 라운드 키를 이용하여 상기 평문으로부터 제1 중간값을 산출하고, n=1, ..., M에 대하여, 순차적으로 (i) 제n 중간값에 치환 연산(substitution)을 포함한 연산을 적용한 결과에 비트 간 위치변환 연산(bit permutation)을 포함한 연산을 적용하거나 상기 제n 중간값에 상기 비트 간 위치변환 연산을 포함한 연산을 적용한 결과에 상기 치환 연산을 포함한 연산을 적용하여 제n 결과값을 산출하는 단계 및 (ii) 제n+1 라운드 키를 이용하여 제n 결과값으로부터 제n+1 중간값을 산출하는 단계를 반복 수행함으로써 이루어지고, 그 중 제M+1 중간값은 암호문으로 산출된다.

(52) CPC특허분류
H04L 2209/04 (2013.01)
H04L 2209/122 (2013.01)
H04L 2209/24 (2013.01)

(72) 발명자
한동국
서울특별시 노원구 중계로14나길 25 삼성아파트 106동 1501호
김기윤
서울특별시 성북구 정릉로 77 국민대학교 과학관 317호
전용진
서울특별시 성북구 정릉로 77 국민대학교 과학관 317호
홍석희
서울특별시 노원구 덕릉로73길 28 양지대림아파트 206동 2304호

명 세 서

청구범위

청구항 1

데이터를 처리하여 암호화하는 방법에 있어서,

소정 라운드 횟수인 자연수 M에 대하여,

제1 내지 제M+1 라운드 키를 생성하는 단계; 및

생성된 상기 제1 내지 제M+1 라운드 키를 이용한 암호화로써 평문으로부터 암호문을 산출하는 단계

를 포함하는 데이터 암호화 방법.

청구항 2

제1항에 있어서,

상기 제1 내지 제M+1 라운드 키를 생성하는 단계는,

마스터 키를 생성하는 단계; 및

상기 마스터 키와 라운드별 상수에 대해 XOR을 포함한 연산을 수행한 결과값으로서 상기 제1 내지 제M+1 라운드 키를 생성하는 단계를 포함하고,

상기 라운드별 상수는 라운드마다 상이하게 정해진 것을 특징으로 하는 데이터 암호화 방법.

청구항 3

제1항에 있어서,

상기 평문으로부터 상기 암호문을 산출하는 단계는,

(a) 상기 제1 라운드 키를 이용하여 상기 평문으로부터 제1 중간값을 산출하는 단계; 및

(b) n=1, ..., M에 대하여, 순차적으로 (i) 제n 중간값에 치환 연산(substitution)을 포함한 연산을 적용한 결과에 비트 간 위치변환 연산(bit permutation)을 포함한 연산을 적용하거나 상기 제n 중간값에 상기 비트 간 위치변환 연산을 포함한 연산을 적용한 결과에 상기 치환 연산을 포함한 연산을 적용하여 제n 결과값을 산출하는 단계 및 (ii) 제n+1 라운드 키를 이용하여 제n 결과값으로부터 제n+1 중간값을 산출하는 단계를 반복 수행하는 단계

를 포함하되,

제M+1 중간값이 암호문인, 데이터 암호화 방법.

청구항 4

제3항에 있어서,

상기 치환 연산은,

128비트 평문에 대하여 적용되는 128비트 치환 연산 S_{128}이고,

상기 128비트 평문을 가로쓰기로 오른쪽 상단에서 왼쪽 하단까지 8×16의 비트열로 표현한 경우, 16개 열들 각각의 상단 열로부터 하단 열까지 취한 비트들(x_0, x_1, ..., x_7)로 구성된 16개의 8비트 값들 각각에 제2 치환 연산 S_8을 적용하는 연산인 것을 특징으로 하는 데이터 암호화 방법.

청구항 5

제4항에 있어서,

상기 제2 치환 연산 S_8은,

제3 치환 연산 S_4의 3회 반복을 포함하는 파이스텔(Feistel) 구조로 구현 가능하면서 256개 원소로 구성된 8비트 치환 테이블(substitution table)을 이용하는 LUT(룩업 테이블; lookup table) 방식으로 구현 가능한 이중성을 가지는 것을 특징으로 하는 데이터 암호화 방법.

청구항 6

제5항에 있어서,

상기 제2 치환 연산 S_8은 차분도가 16과 같거나 그보다 작고, 비선형도가 96과 같거나 그보다 큰 동시에 DBN이 3인 것을 특징으로 하는 데이터 암호화 방법.

청구항 7

제5항에 있어서,

상기 치환 테이블은,

입력값의 왼쪽 4비트를 행의 상단으로부터 하단까지 열거하고 오른쪽 4비트를 열의 왼쪽으로부터 오른쪽까지 열거하여 상기 입력값에 대응하는 출력값을 나타내면,

Left (high-order) 4-bit \ Right (low-order) 4-bit	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0x00	0xCF	0xD5	0xF7	0x57	0x3A	0x2B	0x8E	0x92	0xAF	0x68	0x17	0xE7	0x75	0xBF	0x4A
1	0xF8	0x6F	0xA8	0x5C	0x81	0xB8	0x72	0x0B	0x41	0x28	0x39	0x9D	0x1C	0xD6	0xCC	0xE8
2	0x5E	0xC2	0x6D	0x94	0x80	0xB9	0x34	0x4F	0x19	0x7C	0xA1	0x06	0xEA	0x2D	0xD4	0xF6
3	0x77	0x87	0x48	0xDB	0x26	0x59	0x93	0xAE	0xB6	0x1A	0x05	0x66	0xCD	0xE9	0xF5	0x3F
4	0x7D	0x18	0xC9	0x64	0x88	0x45	0xB2	0x98	0x32	0xAA	0x0F	0xFA	0xE2	0xDF	0x58	0x27
5	0xA5	0x8B	0x9B	0x53	0xD0	0x7E	0x67	0x04	0x4E	0x35	0xFD	0xC6	0x13	0xE5	0x20	0x8D
6	0xB3	0x99	0x62	0xA7	0x43	0xEC	0x3B	0x56	0x0A	0x73	0xFC	0xC7	0xD9	0x22	0x85	0x11
7	0xE3	0xDE	0x16	0x69	0x96	0x0D	0xC0	0x30	0x78	0xB0	0xA4	0x8A	0x29	0x40	0x55	0xF3
8	0x24	0x14	0x91	0xE1	0xF9	0x6E	0xC1	0x31	0x44	0x89	0x7B	0x51	0xB4	0xD3	0x07	0xA0
9	0xFE	0x82	0x08	0x36	0x23	0xD8	0x74	0xE6	0x47	0x61	0x9A	0x52	0xB7	0x1B	0xCA	0xA3
A	0x8F	0x2A	0xF0	0x9F	0x7A	0x50	0xA6	0x63	0x12	0xE4	0x49	0xDA	0xC4	0xBA	0x37	0x09
B	0x79	0xB1	0x46	0x60	0x8C	0xEF	0x38	0x9C	0x15	0x25	0xAD	0xDD	0xC3	0x5F	0xFB	0x0E
C	0x76	0x86	0x21	0xBC	0xAC	0xDC	0x5B	0x6B	0xED	0x42	0x9E	0xF1	0x1E	0x3C	0xCE	0x01
D	0x54	0xF2	0xEE	0x8D	0x2E	0x02	0x1D	0xD7	0x95	0x6C	0xAB	0x33	0xC5	0xBB	0x71	0x4D
E	0xFF	0x83	0x4C	0x70	0xA9	0x5D	0x97	0x0C	0x1F	0x3D	0x2C	0xEB	0x65	0xC8	0xD2	0xB5
F	0xA2	0xCB	0xD1	0x7F	0xF4	0x3E	0x2F	0x03	0x10	0x84	0x4B	0xBE	0x6A	0x5A	0x90	0xE0

S_8

와 같이 정해지는 것을 특징으로 하는 데이터 암호화 방법.

청구항 8

제5항에 있어서,

상기 제3 치환 연산 S_4은,

상기 S4에 입력되는 값의 최하위 비트인 제1 비트 내지 상기 입력되는 값의 최상위 비트인 제4 비트에 대하여,

제4 비트와 제1 비트를 AND 연산한 결과값과 제3 비트를 XOR 연산한 결과값을 다시 제3 비트에 배정(assign)하는 제1 단계;

제3 비트와 제4 비트를 XOR 연산한 결과값을 다시 제4 비트에 배정하는 제2 단계;

제2 비트와 제1 비트를 XOR 연산한 결과값을 다시 제1 비트에 배정하는 제3 단계;

제2 비트와 제1 비트를 OR 연산한 결과값과 제3 비트를 XOR 연산한 결과값을 다시 제3 비트에 배정하는 제4 단계;

제2 비트와 제4 비트를 XOR 연산한 결과값을 임시(temp) 비트에 배정하는 제5 단계;

제1 비트와 제3 비트를 OR 연산한 결과값과 제4 비트를 XOR 연산한 결과값을 다시 제4 비트에 배정하는 제6 단계;

제4 비트와 제3 비트를 AND 연산한 결과값과 제1 비트를 XOR 연산한 결과값을 제2 비트에 배정하는 제7 단계; 및

상기 임시 비트의 값을 제1 비트에 배정하는 제8 단계

를 수행함으로써 구현되는 것을 특징으로 하는 데이터 암호화 방법.

청구항 9

제5항에 있어서,

상기 S4에 입력되는 값의 최하위 비트인 제1 비트 내지 상기 입력되는 값의 최상위 비트인 제4 비트에 대하여,

제4 비트와 제1 비트를 AND 연산한 결과값과 제3 비트를 XOR 연산한 결과값을 다시 제3 비트에 배정(assign)하는 제1 단계;

제3 비트와 제4 비트를 XOR 연산한 결과값을 다시 제4 비트에 배정하는 제2 단계;

제2 비트와 제1 비트를 XOR 연산한 결과값을 다시 제1 비트에 배정하는 제3 단계;

제2 비트와 제1 비트를 OR 연산한 결과값과 제3 비트를 XOR 연산한 결과값을 다시 제3 비트에 배정하는 제4 단계;

제4 비트와 제2 비트를 XOR 연산한 결과값을 다시 제2 비트에 배정하는 제5' 단계;

제1 비트와 제3 비트를 OR 연산한 결과값과 제4 비트를 XOR 연산한 결과값을 다시 제4 비트에 배정하는 제6 단계; 및

제4 비트와 제3 비트를 AND 연산한 결과값과 제1 비트를 XOR 연산한 결과값을 다시 제1 비트에 배정하는 제7' 단계를 수행하고,

제1 비트와 제2 비트의 값을 서로 뒤바꾸는(swap) 제8' 단계를 더 수행하거나 하드웨어 배선(wiring)을 통하여 제1 비트와 제2 비트의 값이 서로 뒤바뀌어 산출됨으로써 구현되는 것을 특징으로 하는 데이터 암호화 방법.

청구항 10

제3항에 있어서,

상기 비트 간 위치변환 연산은,

행 전환(row conversion) 및 행별 로테이션(rotation)을 포함하는 연산으로써 구현되는 것을 특징으로 하는 데이터 암호화 방법.

청구항 11

데이터를 처리하여 복호화하는 방법에 있어서,

소정 라운드 횟수인 자연수 M에 대하여,

제1 내지 제M+1 라운드 키를 생성하는 단계; 및

생성된 상기 제1 내지 제M+1 라운드 키를 이용한 복호화로써 암호문으로부터 평문을 산출하는 단계

를 포함하는 데이터 복호화 방법.

청구항 12

제11항에 있어서,

상기 암호문으로부터 상기 평문을 산출하는 단계는,

(a) n=M, M-1, ..., 1에 대하여, 순차적으로 (i) 제n+1 라운드 키를 이용하여 제n+1 중간값으로부터 제n 결과값을 산출하는 단계 및 (ii) 제n 결과값에 치환 연산(substitution)을 포함하는 연산을 적용한 결과에 비트 간 위치변환 연산(bit permutation)의 역연산을 포함하는 연산을 적용하거나 상기 제n 결과값에 상기 비트 간 위치변환 연산의 상기 역연산을 포함하는 연산을 적용한 결과에 상기 치환 연산을 포함하는 연산을 적용하여 제n 중간값을 산출하는 단계를 반복 수행하는 단계로서, 상기 암호문은 제M+1 중간값인, 단계; 및

(b) 상기 제1 라운드 키를 이용하여 제1 중간값으로부터 평문을 산출하는 단계

를 포함하는 데이터 복호화 방법.

청구항 13

제12항에 있어서,

상기 치환 연산은,

128비트 평문에 대하여 적용되는 128비트 치환 연산 S_{128}이고,

상기 128비트 평문을 가로쓰기로 오른쪽 상단에서 왼쪽 하단까지 8×16의 비트열로 표현한 경우, 16개 열들 각각의 상단 열로부터 하단 열까지 취한 비트들(x_0, x_1, ..., x_7)로 구성된 16개의 8비트 값들 각각에 제2 치환 연산 S_8을 적용하는 연산인 것을 특징으로 하는 데이터 복호화 방법.

청구항 14

제13항에 있어서,

상기 제2 치환 연산 S_8은,

제3 치환 연산 S_4의 3회 반복을 포함하는 파이스텔(Feistel) 구조로 구현 가능하면서 256개 원소로 구성된 8비트 치환 테이블(substitution table)을 이용하는 LUT(룩업 테이블; lookup table) 방식으로 구현 가능한 이중성을 가지는 것을 특징으로 하는 데이터 복호화 방법.

청구항 15

입출력 가능한 연산 장치로 하여금, 제1항 내지 제14항 중 어느 한 항의 방법을 수행하도록 구현된 명령어(instructions)를 포함하는, 기계 판독 가능한 비일시적 기록 매체에 저장된, 프로그램 코드.

청구항 16

데이터를 처리하여 암호화하는 장치에 있어서,

평문을 획득하고 상기 평문으로부터 산출된 암호문을 출력하는 입출력부; 및

소정 라운드 횟수인 자연수 M에 대하여, 제1 내지 제M+1 라운드 키를 생성하는 프로세스 및 생성된 상기 제1 내지 제M+1 라운드 키를 이용한 암호화로써 평문으로부터 암호문을 산출하는 프로세스를 수행하는 프로세서

를 포함하는 데이터 암호화 장치.

청구항 17

제16항에 있어서,

상기 프로세서가 상기 제1 내지 제M+1 라운드 키를 생성하는 프로세스는,

마스터 키를 생성하는 프로세스; 및

상기 마스터 키와 라운드별 상수에 대해 XOR을 포함한 연산을 수행한 결과값으로서 상기 제1 내지 제M+1 라운드 키를 생성하는 프로세스를 포함하고,

상기 라운드별 상수는 라운드마다 상이하게 정해진 것을 특징으로 하는 데이터 암호화 장치.

청구항 18

제16항에 있어서,

상기 평문으로부터 상기 암호문을 산출하는 프로세스는,

(a) 상기 제1 라운드 키를 이용하여 상기 평문으로부터 제1 중간값을 산출하는 프로세스; 및

(b) n=1, ..., M에 대하여, 순차적으로 (i) 제n 중간값에 치환 연산(substitution)을 포함한 연산을 적용한 결과에 비트 간 위치변환 연산(bit permutation)을 포함한 연산을 적용하거나 상기 제n 중간값에 상기 비트 간 위치변환 연산을 포함한 연산을 적용한 결과에 상기 치환 연산을 포함한 연산을 적용하여 제n 결과값을 산출하는 프로세스 및 (ii) 제n+1 라운드 키를 이용하여 제n 결과값으로부터 제n+1 중간값을 산출하는 프로세스를 반복 수행하는 프로세스

를 포함하되,

제M+1 중간값이 암호문인 것을 특징으로 하는 데이터 암호화 장치.

청구항 19

데이터를 처리하여 복호화하는 장치에 있어서,

암호문을 획득하고 상기 암호문으로부터 산출된 평문을 출력하는 입출력부; 및

소정 라운드 횟수인 자연수 M에 대하여, 제1 내지 제M+1 라운드 키를 생성하는 프로세스 및 생성된 상기 제1 내지 제M+1 라운드 키를 이용한 복호화로써 암호문으로부터 평문을 산출하는 프로세스를 수행하는 프로세서

를 포함하는 데이터 암호화 장치.

청구항 20

제19항에 있어서,

상기 평문으로부터 상기 암호문을 산출하는 프로세스는,

(a) n=M, M-1, ..., 1에 대하여, 순차적으로 (i) 제n+1 라운드 키를 이용하여 제n+1 중간값으로부터 제n 결과값을 산출하는 프로세스 및 (ii) 제n 결과값에 치환 연산(substitution)을 포함한 연산을 적용한 결과에 비트 간 위치변환 연산(bit permutation)의 역연산을 포함한 연산을 적용하거나 상기 제n 결과값에 상기 비트 간 위치변환 연산의 상기 역연산을 포함한 연산을 적용한 결과에 상기 치환 연산을 포함한 연산을 적용하여 제n 중간값을 산출하는 프로세스를 반복 수행하는 프로세스로서, 상기 암호문은 제M+1 중간값인, 프로세스; 및

(b) 상기 제1 라운드 키를 이용하여 제1 중간값으로부터 평문을 산출하는 프로세스

를 포함하는 데이터 복호화 장치.

발명의 설명

기 술 분 야

[0001] 본 개시서에는 암호학적으로 데이터를 처리하는 방법 및 이를 이용한 장치가 개시된다.

[0002] 구체적으로, 본 개시서에 따른 방법에 의하면, 소정 라운드 횟수인 자연수 M에 대하여, 제1 내지 제M+1 라운드 키를 생성하고 생성된 상기 제1 내지 제M+1 라운드 키를 이용한 암호화로써 평문으로부터 암호문을 산출하는바, 구체적으로, 마스터 키를 생성하고, 상기 마스터 키와 라운드별 상수에 대해 XOR을 포함한 연산을 수행한 결과값으로서 제1 내지 제M+1 라운드 키를 생성하며, 상기 라운드별 상수는 라운드마다 상이하게 정해진다. 상기 평문으로부터 상기 암호문을 산출함은, 상기 제1 라운드 키를 이용하여 상기 평문으로부터 제1 중간값을 산출하고, n=1, ..., M에 대하여, 순차적으로 (i) 제n 중간값에 치환 연산(substitution)을 포함한 연산을 적용한 결과에 비트 간 위치변환 연산(bit permutation)을 포함한 연산을 적용하거나 상기 제n 중간값에 상기 비트 간 위치변환 연산을 포함한 연산을 적용한 결과에 상기 치환 연산을 포함한 연산을 적용하여 제n 결과값을 산출하는 단계 및 (ii) 제n+1 라운드 키를 이용하여 제n 결과값으로부터 제n+1 중간값을 산출하는 단계를 반복 수행함으

로써 이루어지고, 그 중 제M+1 중간값은 암호문으로 산출된다.

배 경 기 술

[0003] 스마트 카드 및 IC(integrated circuit) 카드는 사용자에 관한 보안 정보를 포함하고, 사용자의 보안 정보가 유출되는 것을 방지하기 위하여 그 보안 정보를 암호문으로 만들어서 송수신할 필요가 있다. 송신측에서는 평문을 암호화하여 암호문을 만들어 송신하고, 수신측에서는 수신된 암호문을 복호화하여 평문을 획득한다.

[0004] 암호화 및 복호화 연산은 소프트웨어로 구현되는 때에 속도가 느리기 때문에 스마트 카드와 같은 장치에 적용하기 위하여 하드웨어로 구현하는 경우가 많고, 이를 위한 블록 암호 기법에는 DES(Data Encryption Standard), AES(Advanced Encryption Standard), SEED, ARIA, SM4 등이 있다.

[0005] 특히, 근래 사물 인터넷(IoT; Internet of Things) 환경의 대부분의 기기들은 제한된 리소스(메모리, 속도, 하드웨어 GE 등)를 가지기 때문에 암호학적 안전성을 요구하는 시스템에서 사용될 경우 리소스 부하가 적은 경량 블록 암호를 사용하기를 선호한다.

[0006] 하지만 경량 블록 암호라고 하더라도 부채널 분석에 대한 안전성을 가져야 하는 환경에서 사용되는 기기에 적용될 때에는 부채널 분석 방지 마스킹 기법을 적용하여 구현하여야 한다. 부채널 분석 방지 기술은 부채널을 통하여 수집되는 정보인 전력, 전자기파를 무작위적으로 나타나게 하거나 균일하게 나타나게 함으로써 부채널 분석을 어렵게 하는 것이다.

[0007] 그런데, 부채널 분석 대응 마스킹에 추가적으로 필요한 리소스는 경량 블록 암호를 사용하는 의미를 퇴색시키며, 더 높은 부채널 안전성을 요구하는 환경일수록 블록 암호 구현에 더욱 많은 리소스를 필요로 하게 된다. 따라서 부채널 안전성을 요구하는 시스템에서 사용되는 IoT 환경에서도 리소스 부담을 가지지 않고 사용할 수 있는 경량 블록 암호의 필요성이 대두된다.

[0008] 종래 경량 블록 암호 기법 가운데 RECTANGLE은 4×16의 64비트 블록(state)을 가지는 블록 암호로서, 본 개시서에 따른 블록 암호와 유사하게 비트슬라이스 구현이 가능하다. 하지만, RECTANGLE은 암호학적 안전성이 비교적 약한 4비트 S-박스를 이용하고 있기 때문에 64비트 블록을 채용한 블록 암호임에도 본 개시서에 따라 128비트 블록을 사용하는 블록 암호의 라운드 수인 21보다 많은 25개의 라운드를 사용하여 암호학적 안전성을 확보한다. 이와 같은 라운드 수의 증가는 블록 암호 실행 시간에 큰 영향을 미치기 때문에 더 적은 라운드 수를 가진다면 더 빠른 데이터 암호화 및 복호화가 가능할 것이다.

[0009] 또한, 본 개시서에 따른 블록 암호에 채용된 SPN(substitution-permutation network) 구조에 있어 선형 함수로서 비트 간 위치변환 연산(bit-permutation)만을 사용하는 블록 암호의 경우에는 선형 함수에서 차분 특성과 선형 특성의 확산이 크지 않은 단점이 있는바, 확산력(diffusion power)의 척도로서 아래와 같이 정의되는 DBN(differential branch number)이 2가 아닌 3인 성질은 차분 공격에 대한 안전성 확보에 큰 영향을 미치는 이점으로 작용한다.

수학식 1

[0010] DBN Differential Branch Number of an S-box defined as $\min_{a,b\neq a}(wt(a\oplus b)+wt(S(a)\oplus S(b)))$.

[0011] {여기에서 wt는 해밍 가중치(Hamming weight)를 지칭한다.}

[0012] 종래의 대표적인 경량 블록 암호 기법인 PRESENT의 경우 비트 자리바꿈과 함께 DBN이 3인 S-박스를 채용하고 있기는 하지만 암호학적 안전성이 상대적으로 약한 4비트 S-박스를 사용하고 있으므로 안전성을 확보하기 위하여 31개 라운드라는 많은 라운드 수를 채용하고 있다.

[0013] 따라서 본 발명자는 위 종래의 기술들의 단점을 극복하기 위한 수단으로서, 차분도(differentiality) 및 비선형도(non-linearity)가 아래와 같이 정의될 때, 차분도가 16 이하이며, 비선형도가 96 이상인 동시에 DBN이 3인 8비트 S-박스를 이용한 경량 블록 암호 기법을 개시하고자 한다.

수학식 2

[0014] $$\text{Differentiality} \quad \max_{\Delta\alpha\neq 0,\Delta\beta} \#\{x \in \mathbb{F}_2^n | S(x) \oplus S(x \oplus \Delta\alpha) = \Delta\beta\}.$$

수학식 3

[0015] $$\text{Non-linearity} \quad 2^{n-1} - 2^{-1} \times \max_{\lambda_\alpha,\lambda_\beta\neq 0} |\Phi(\lambda_\alpha,\lambda_\beta)|, \text{ where } \Phi(\lambda_\alpha,\lambda_\beta) = \sum_{x\in\mathbb{F}_2^n} (-1)^{\lambda_\beta \bullet S(x)\oplus\lambda_\alpha\bullet x}.$$

선행기술문헌

특허문헌

[0016] (특허문헌 0001) 한국공개특허 제10-2008-0080175(2008.09.02)호

비특허문헌

[0017] (비특허문헌 0001) [1] Zhang, Wentao, et al. "RECTANGLE: a bit-slice lightweight block cipher suitable for multiple platforms." Science China Information Sciences 58.12 (2015): 1-15.

(비특허문헌 0002) [2] Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C. PRESENT: An Ultra-Lightweight Block Cipher, CHES 2007, LNCS 4727, pp. 450-466.

발명의 내용

해결하려는 과제

[0018] 본 개시서의 일 실시 예는 경량성을 만족하면서도 확산력을 확보할 수 있는 블록 암호 기법을 채용한 암호화 방법 및 장치를 제공하는 것을 목적으로 한다.

[0019] 구체적으로, 본 개시서의 일 실시 예는 역연산이 자기 자신과 같은 치환 테이블(substitution table) 또는 룩업 테이블(lookup table)을 채용하고, XOR, OR, AND, 행 전환 및 왼쪽 로테이션 연산만을 포함함으로써 비선형성을 가지면서도 연산 리소스가 절감되는 개선된 경량 블록 암호 기법을 제공하는 것을 목적으로 한다.

[0020] 또한, 본 개시서의 일 실시 예는 병렬 연산에도 적합한 경량 블록 암호 기법을 제공하는 것도 그 일 목적으로 한다.

과제의 해결 수단

[0021] 상기한 바와 같은 본 발명의 목적을 달성하고, 후술하는 본 발명의 특징적인 효과를 실현하기 위한 본 발명의 특징적인 구성은 하기와 같다.

[0022] 본 발명의 일 태양(aspect)에 따르면, 데이터를 처리하여 암호화하는 방법이 제공되는바, 그 방법은, 소정 라운드 횟수인 자연수 M에 대하여, 제1 내지 제M+1 라운드 키를 생성하는 단계; 및 생성된 상기 제1 내지 제M+1 라운드 키를 이용한 암호화로써 평문으로부터 암호문을 산출하는 단계를 포함한다.

[0023] 유리하게는, 상기 제1 내지 제M+1 라운드 키를 생성하는 단계는, 마스터 키를 생성하는 단계; 및 상기 마스터 키와 라운드별 상수에 대해 XOR을 포함한 연산을 수행한 결과값으로서 상기 제1 내지 제M+1 라운드 키를 생성하

는 단계를 포함하고, 상기 라운드별 상수는 라운드마다 상이하게 정해질 수 있다.

[0024] 바람직하게는, 상기 평문으로부터 상기 암호문을 산출하는 단계는, (a) 상기 제1 라운드 키를 이용하여 상기 평문으로부터 제1 중간값을 산출하는 단계; 및 (b) n=1, ..., M에 대하여, 순차적으로 (i) 제n 중간값에 치환 연산(substitution)을 포함한 연산을 적용한 결과에 비트 간 위치변환 연산(bit permutation)을 포함한 연산을 적용하거나 상기 제n 중간값에 상기 비트 간 위치변환 연산을 포함한 연산을 적용한 결과에 상기 치환 연산을 포함한 연산을 적용하여 제n 결과값을 산출하는 단계 및 (ii) 제n+1 라운드 키를 이용하여 제n 결과값으로부터 제n+1 중간값을 산출하는 단계를 반복 수행하는 단계를 포함할 수 있고, 여기에서 제M+1 중간값은 암호문이다.

[0025] 본 발명의 다른 태양에 따르면, 데이터를 처리하여 복호화하는 방법이 제공되는바, 그 방법은, 소정 라운드 횟수인 자연수 M에 대하여, 제1 내지 제M+1 라운드 키를 생성하는 단계; 및 생성된 상기 제1 내지 제M+1 라운드 키를 이용한 복호화로써 암호문으로부터 평문을 산출하는 단계를 포함한다.

[0026] 바람직하게는, 상기 암호문으로부터 상기 평문을 산출하는 단계는, 상기 암호문을 제M+1 중간값으로 표현하면, (a) n=M, M-1, ..., 1에 대하여, 순차적으로 (i) 제n+1 라운드 키를 이용하여 제n+1 중간값으로부터 제n 결과값을 산출하는 단계 및 (ii) 제n 결과값에 치환 연산(substitution)을 포함한 연산을 적용한 결과에 비트 간 위치변환 연산(bit permutation)의 역연산을 포함한 연산을 적용하거나 상기 제n 결과값에 상기 비트 간 위치변환 연산의 상기 역연산을 포함한 연산을 적용한 결과에 상기 치환 연산을 포함한 연산을 적용하여 제n 중간값을 산출하는 단계를 반복 수행하는 단계; 및 (b) 상기 제1 라운드 키를 이용하여 제1 중간값으로부터 평문을 산출하는 단계를 포함할 수 있다.

[0027] 본 발명의 또 다른 태양에 따르면, 본 개시서에 따른 데이터 암호화 방법 및 데이터 복호화 방법 중 적어도 하나를 수행하도록 구현된 명령어(instructions)를 포함하는, 기계 판독 가능한 비일시적 기록 매체에 저장된, 컴퓨터 프로그램도 제공된다.

[0028] 본 발명의 다른 일 태양에 따르면, 데이터를 처리하여 암호화하는 장치가 제공되는바, 그 장치는, 평문을 획득하고 상기 평문으로부터 산출된 암호문을 출력하는 입출력부; 및 소정 라운드 횟수인 자연수 M에 대하여, 제1 내지 제M+1 라운드 키를 생성하는 프로세스 및 생성된 상기 제1 내지 제M+1 라운드 키를 이용한 암호화로써 평문으로부터 암호문을 산출하는 프로세스를 수행하는 프로세서를 포함한다.

[0029] 본 발명의 또 다른 일 태양에 따르면, 데이터를 처리하여 복호화하는 장치가 제공되는바, 그 장치는, 암호문을 획득하고 상기 암호문으로부터 산출된 평문을 출력하는 입출력부; 및 소정 라운드 횟수인 자연수 M에 대하여, 제1 내지 제M+1 라운드 키를 생성하는 프로세스 및 생성된 상기 제1 내지 제M+1 라운드 키를 이용한 복호화로써 암호문으로부터 평문을 산출하는 프로세스를 수행하는 프로세서를 포함한다.

발명의 효과

[0030] 본 개시서의 예시적인 실시 예에 따른 암호화 방법 및 복호화 방법은 비트슬라이스 구현이 가능한 8비트 치환 테이블과 왼쪽 로테이션 연산만을 이용하여 설계되었으므로 IoT 환경에서의 경량 구현에 적합할 뿐만 아니라 최소한의 비선형 연산만으로 구현이 가능하기 때문에 비교적 적은 추가 리소스로도 부채널 분석 대응 마스킹 기법의 적용이 가능한 효과가 있다.

도면의 간단한 설명

[0031] 본 발명의 실시 예의 설명에 이용되기 위하여 첨부된 아래 도면들은 본 발명의 실시 예들 중 단지 일부일 뿐이며, 본 발명의 기술분야에서 통상의 지식을 가진 사람(이하 "통상의 기술자"라 함)에게 있어서는 발명에 이르는 추가 노력 없이 이 도면들에 기초하여 다른 도면들이 얻어질 수 있다.

도 1은 본 개시서에 따라 데이터를 처리하여 암호화하는 방법(이하 "데이터 암호화 방법"이라 함) 및/또는 데이터를 처리하여 복호화하는 방법(이하 "데이터 복호화 방법"이라 함)을 수행하는 연산 장치의 예시적 구성을 개략적으로 도시한 개념도이다.

도 2는 본 개시서의 데이터 암호화 방법과 데이터 복호화 방법에 이용되는 연산 장치의 하드웨어 또는 소프트웨어 구성요소를 도시한 예시적 블록도이다.

도 3은 본 개시서의 데이터 암호화 방법을 예시적으로 나타낸 흐름도이다.

도 4는 본 개시서의 일 실시 예에 따른 데이터 암호화 방법을 개념적으로 나타낸 블록도이다.

도 5는 본 개시서의 데이터 암호화 방법 및 데이터 복호화 방법에서 이용되는 치환 연산(substitution) S_{128}을 128비트 평문에 대하여 적용하는 방식을 개념적으로 나타낸 도면이다.

도 6은 본 개시서에 따른 치환 연산(substitution)의 일 실시 예로서 128비트 평문에 대하여 적용되는 예시적인 128비트 치환 연산(S_{128})을 도출하는 8비트 치환 테이블을 그 치환 테이블의 입력값의 왼쪽 4비트를 행의 상단으로부터 하단까지 열거하고 오른쪽 4비트를 열의 왼쪽으로부터 오른쪽까지 열거하여 상기 입력값에 대응하는 출력값을 나타낸 도면이다.

도 7은 본 개시서의 일 실시 예에서 제3 치환 연산 S_4을 이용하여 S_8을 구현한 파이스텔(Feistel) 구조를 개념적으로 나타낸 도면이다.

도 8은 도 7에 나타난 파이스텔 구조에 포함되는 제3 치환 연산 S_4을 입력값(input)과 출력값(output)의 쌍으로 표현한 테이블을 나타낸 것이다.

도 9a 및 도 9b는 본 개시서의 일 실시 예에서 제3 치환 연산 S_4을 2가지의 비트슬라이스 구현으로 도시한 것이다.

도 10은 본 개시서의 데이터 암호화 방법 및 데이터 복호화 방법에서 이용되는 비트 간 위치변환 연산 P_{128}을 128비트 평문에 대하여 적용하는 방식을 개념적으로 나타낸 도면이다.

발명을 실시하기 위한 구체적인 내용

[0032] 후술하는 본 발명에 대한 상세한 설명은, 본 발명의 목적들, 기술적 해법들 및 장점들을 분명하게 하기 위하여 본 발명이 실시될 수 있는 특정 실시 예를 예시로서 도시하는 첨부 도면을 참조한다. 이들 실시 예는 통상의 기술자가 본 발명을 실시할 수 있도록 상세히 설명된다.

[0033] 본 발명의 상세한 설명 및 청구항들에 걸쳐, '포함하다'라는 단어 및 그 변형은 다른 기술적 특징들, 부가물들, 구성요소들 또는 단계들을 제외하는 것으로 의도된 것이 아니다. 또한, '하나' 또는 '한'은 하나 이상의 의미로 쓰인 것이며, '또 다른'은 적어도 두 번째 이상으로 한정된다.

[0034] 또한, 본 발명의 '제1', '제2' 등의 용어는 하나의 구성요소를 다른 구성요소로부터 구별하기 위한 것으로서, 순서를 나타내는 것으로 이해되지 않는 한 이들 용어들에 의하여 권리범위가 한정되어서는 아니 된다. 예를 들어, 제1 구성요소는 제2 구성요소로 명명될 수 있고, 이와 유사하게 제2 구성요소도 제1 구성요소로 명명될 수 있다.

[0035] 어떤 구성요소가 다른 구성요소에 "연결되어" 있다고 언급된 때에는 그 다른 구성요소에 직접 연결될 수도 있지만 중간에 다른 구성요소가 개재할 수도 있다고 이해되어야 할 것이다. 반면에 어떤 구성요소가 다른 구성요소에 "직접 연결되어" 있다고 언급된 때에는 중간에 다른 구성요소가 존재하지 않는 것으로 이해되어야 할 것이다. 한편, 구성요소들 간의 관계를 설명하는 다른 표현들, 즉, "~사이에"와 "바로 ~사이에" 또는 "~에 이웃하는"과 "~에 직접 이웃하는" 등도 마찬가지로 해석되어야 한다.

[0036] 각 단계들에 있어서 식별부호(예를 들어, a, b, c 등)는 설명의 편의를 위하여 사용된 것으로 식별부호는 논리상 필연적으로 귀결되지 않는 한 각 단계들의 순서를 설명하는 것이 아니며, 각 단계들은 명기된 순서와 다르게 일어날 수 있다. 즉, 각 단계들은 명기된 순서와 동일하게 일어날 수도 있고 실질적으로 동시에 수행될 수도 있으며, 반대의 순서로 수행될 수도 있다.

[0037] 통상의 기술자에게 본 발명의 다른 목적들, 장점들 및 특성들이 일부는 본 설명서로부터, 그리고 일부는 본 발명의 실시로부터 드러날 것이다. 아래의 예시 및 도면은 실례로서 제공되며, 본 발명을 한정하는 것으로 의도된 것이 아니다. 따라서, 특정 구조나 기능에 관하여 본 명세서에 개시된 상세 사항들은 한정하는 의미로 해석되어서는 아니되고, 단지 통상의 기술자가 실질적으로 적합한 임의의 상세 구조들로써 본 발명을 다양하게 실시하도록 지침을 제공하는 대표적인 기초 자료로 해석되어야 할 것이다.

[0038] 더욱이 본 발명은 본 명세서에 표시된 실시 예들의 모든 가능한 조합들을 망라한다. 본 발명의 다양한 실시 예는 서로 다르지만 상호 배타적일 필요는 없음이 이해되어야 한다. 예를 들어, 여기에 기재되어 있는 특정 형상, 구조 및 특성은 일 실시 예에 관련하여 본 발명의 사상 및 범위를 벗어나지 않으면서 다른 실시 예로 구현될 수 있다. 또한, 각각의 개시된 실시 예 내의 개별 구성요소의 위치 또는 배치는 본 발명의 사상 및 범위

를 벗어나지 않으면서 변경될 수 있음이 이해되어야 한다. 따라서, 후술하는 상세한 설명은 한정적인 의미로서 취하려는 것이 아니며, 본 발명의 범위는, 적절하게 설명된다면, 그 청구항들이 주장하는 것과 균등한 모든 범위와 더불어 첨부된 청구항에 의해서만 한정된다. 도면에서 유사한 참조부호는 여러 측면에 걸쳐서 동일하거나 유사한 기능을 지칭한다.

[0039] 본 명세서에서 달리 표시되거나 분명히 문맥에 모순되지 않는 한, 단수로 지칭된 항목은, 그 문맥에서 달리 요구되지 않는 한, 복수의 것을 아우른다. 또한, 본 발명을 설명함에 있어, 관련된 공지 구성 또는 기능에 대한 구체적인 설명이 본 발명의 요지를 흐릴 수 있다고 판단되는 경우에는 그 상세한 설명은 생략한다.

[0040] 이하, 통상의 기술자가 본 발명을 용이하게 실시할 수 있도록 하기 위하여, 본 발명의 바람직한 실시 예들에 관하여 첨부된 도면을 참조하여 상세히 설명하기로 한다.

[0041] 도 1은 본 개시서에 따른 데이터 암호화 방법 및/또는 데이터 복호화 방법을 수행하는 연산 장치의 예시적 구성을 개략적으로 도시한 개념도이다.

[0042] 도 1을 참조하면, 본 발명의 일 실시 예에 따른 연산 장치(100)는, 입출력부(110) 및 프로세서(120)를 포함하며, 상기 입출력부(110)를 통하여 외부와 직간접적으로 데이터 신호를 송수신할 수 있다.

[0043] 구체적으로, 상기 연산 장치(100)는, 전형적인 컴퓨터 하드웨어(예컨대, 컴퓨터 프로세서, 메모리, 스토리지, 입력 장치 및 출력 장치, 기타 기존의 연산 장치의 구성요소들을 포함할 수 있는 장치; 라우터, 스위치 등과 같은 전자 통신 장치; 네트워크 부착 스토리지(NAS; network-attached storage) 및 스토리지 영역 네트워크(SAN; storage area network)와 같은 전자 정보 스토리지 시스템)와 컴퓨터 소프트웨어(즉, 연산 장치로 하여금 특정의 방식으로 기능하게 하는 명령어들)의 조합을 이용하여 원하는 시스템 성능을 달성하는 것일 수 있다.

[0044] 이와 같은 연산 장치의 입출력부(110)는 요청과 응답을 송수신할 수 있는바, 상기 요청은 암호화의 대상인 평문 데이터 신호의 입력만으로, 상기 응답은 그 평문 데이터 신호로부터 얻은 암호문의 출력만으로 구성될 수 있고(암호화의 경우), 반대로 상기 요청이 복호화의 대상인 암호문 데이터 신호의 입력만으로, 상기 응답은 그 암호문 데이터 신호로부터 얻은 평문의 출력만으로 구성될 수도 있다(복호화의 경우).

[0045] 물론 상기 요청과 상기 응답이 오류 정정(error correction)과 모드 변환(암호화 모드, 복호화 모드 등) 등의 적절한 제어를 위한 다른 신호를 포함할 수 있음은 물론이다. 또한, 일 예시로서 그러한 요청과 응답이 동일한 TCP(transmission control protocol) 세션(session)에 의하여 이루어질 수도 있고, UDP(user datagram protocol) 데이터그램(datagram)으로서 송수신될 수도 있을 것이지만 이에 한정되지 않는다.

[0046] 또한, 연산 장치의 프로세서(120)는 MPU(micro processing unit), CPU(central processing unit), GPU(graphics processing unit), NPU(neural processing unit) 또는 TPU(tensor processing unit), 캐시 메모리(cache memory), 데이터 버스(data bus) 등의 하드웨어 구성을 포함할 수 있다. 또한, 범용 컴퓨팅 장치인 경우 운영체제, 특정 목적을 수행하는 애플리케이션의 소프트웨어 구성을 더 포함할 수도 있다.

[0047] 도 2는 본 개시서의 데이터 암호화 방법과 데이터 복호화 방법에 이용되는 연산 장치의 하드웨어 또는 소프트웨어 구성요소를 도시한 예시적 블록도이다.

[0048] 도 2를 참조하면, 연산 장치(100)는 데이터 획득부(210), 암호화·복호화부(220) 및 결과 출력부(230)를 포함할 수 있다.

[0049] 데이터 획득부(210)는 본 개시서에 따른 방법이 적용되는 평문 또는 암호문을 획득하도록 구성되고, 결과 출력부(230)는 본 개시서에 따른 방법을 수행한 결과로 산출되는 암호문 또는 평문을 출력하도록 구성되는바, 도 2에 도시된 개별 모듈들은, 예컨대, 연산 장치(100)에 포함된 입출력부(110)나 프로세서(120), 또는 상기 입출력부(110) 및 프로세서(120)의 연동에 의하여 구현되거나 데이터 획득부(210), 결과 출력부(230) 각각이 연산 장치(100)의 입력 단자 또는 출력 단자 자체인 것으로 구현될 수도 있다.

[0050] 암호화·복호화부(220)는 연산부, 제어부, 및 이들과 빠르게 소통되는 레지스터 및/또는 외부 메모리를 포함하는 다목적의 범용 마이크로프로세서, 마이크로컨트롤러, 임베디드 마이크로컨트롤러, 프로그래머블 디지털 신호 프로세서 또는 기타 프로그래머블 장치로 된 프로세서에 의하여 실현될 수 있으나, 본 개시서에 따른 방법들을 수행하도록 설계된 주문형 집적회로(application specific integrated circuit; ASIC), 프로그래머블 게이트 어레이(programmable gate array), 프로그래머블 어레이 로직(Programmable Array Logic; PAL) 또는 전자 신호들을 처리하기 위해 구성될 수 있는 임의의 다른 장치 또는 장치들의 조합으로 된 프로세서에 의해서도 실시될 수도 있다. 요컨대, 본 개시서의 방법을 수행하는 하드웨어는 범용 컴퓨터 및/또는 전용 연산 장치 또는 특정

연산 장치 또는 특정 연산 장치의 특별한 모습 또는 구성요소를 포함할 수 있다.

[0051] 암호화·복호화부(220)는 후술하는 LUT 방식에 따른 룩업 테이블을 비일시적 또는 일시적으로 저장함으로써 프로세서(222)가 이를 연산에 이용할 수 있게 하는 룩업 테이블 보유부(226)를 더 포함할 수 있고, 역시 후술하는 비트슬라이스 구현에 따라 룩업 테이블 보유부(226)를 생략할 수도 있다. 비트슬라이스 구현은 병렬 연산이 더 용이한바 이를 위한 다수의 프로세서(222, ..., 224)가 마련될 수 있다.

[0052] 도 3은 본 개시서의 데이터 암호화 방법을 예시적으로 나타낸 흐름도이고, 도 4는 본 개시서의 일 실시 예에 따른 데이터 암호화 방법을 개념적으로 나타낸 블록도이다.

[0053] 도 3을 참조하면, 본 개시서에 따른 데이터 암호화 방법은, 먼저, 암호화부(220)가, 소정 라운드 횟수인 자연수 M에 대하여 제1 내지 제M+1 라운드 키를 생성하는 단계(S100)를 포함한다. 아래에 설명되는 본 개시서에 따른 실시 예에서 M은 21이다.

[0054] 제1 내지 제M+1 라운드 키(r_0, r_1, ..., r_M)를 생성하는 단계(S100)는, 도 4에 개념적으로 나타난 바와 같은 마스터 키(K)를 생성하는 단계(S120), 및 상기 마스터 키와 라운드별 상수(c_0, c_1, ..., c_M)를 XOR 연산한 결과값으로서 상기 제1 내지 제M+1 라운드 키($r_i=K\oplus c_i$)를 생성하는 단계(S140)를 포함할 수 있다.

[0055] 여기에서 마스터 키(K)는 무작위적(randomly)으로 생성될 수 있고, 라운드별 상수는 라운드마다 상이하게 정해질 수 있다. 예를 들어, i=0, ..., M에 대하여 라운드별 상수는 $c_i=i$일 수 있다.

[0056] 다음으로, 본 개시서에 따른 암호화 방법은, 평문이 획득되면, 암호화부(220)가, 상기 제1 내지 제M+1 라운드 키를 이용한 암호화로써 평문으로부터 암호문을 산출하는 단계(S200)를 더 포함한다.

[0057] 상기 평문으로부터 암호문을 산출하는 단계(S200)는, 상기 제1 라운드 키를 이용하여 상기 평문(P)으로부터 제1 중간값을 산출하는 단계(S220); 및 n=1, ..., M에 대하여, 순차적으로 (i) 제n 중간값에 치환 연산(substitution)을 적용한 결과에 비트 간 위치변환 연산(bit permutation)을 적용하거나 상기 제n 중간값에 상기 비트 간 위치변환 연산을 적용한 결과에 상기 치환 연산을 적용하여 제n 결과값을 산출하는 단계(S242) 및 (ii) 제n+1 라운드 키를 이용하여 제n 결과값으로부터 제n+1 중간값을 산출하는 단계(S244)를 반복 수행하는 단계(S240; S242, S244)를 포함할 수 있다.

[0058] 예컨대, 단계(S220)에서 제1 중간값은 $M_0 = P\oplus r_0$으로 산출될 수 있는바, 도 4에 구체적으로 예시된 바와 같다.

[0059] 도 4를 참조하여 단계(S240)를 설명하면, 제n 중간값(M_{n-1})에 치환 연산(substitution)(S_{128})을 적용한 결과에 비트 간 위치변환 연산(bit permutation)(P_{128})을 적용하여 제n 결과값을 산출할 수 있고, 이 제n 결과값에 제n+1 라운드 키를 XOR 연산한 결과로 나온 값을 제n+1 중간값이라고 하는데, 이는 다시 n을 1만큼 증가시켜 반복 수행됨을 이해할 수 있다. 도 4의 예시에서는 총 21번의 라운드가 반복되고, 마지막 제M+1 중간값(도 4에서는 제M 결과값에 제M+1 라운드 키인 $K\oplus c_{21}$을 XOR 연산한 결과값)이 암호문(도 4에서 output(128-bit)로 나타남)이 된다.

[0060] 구체적으로, 상기 치환 연산은, 128비트 평문에 대하여 적용되는 128비트 치환 연산 S_{128}일 수 있다.

[0061] 도 5는 본 개시서의 데이터 암호화 방법 및 데이터 복호화 방법에서 이용되는 이 치환 연산 S_{128}을 128비트 평문에 대하여 적용하는 방식을 개념적으로 나타낸 도면이다.

[0062] 도 5를 참조하면, 상기 치환 연산은 상기 128비트 평문을 가로쓰기로 오른쪽 상단에서 왼쪽 하단까지 8×16의 비트열로 표현한 경우, 16개 열들 각각의 상단 열로부터 하단 열까지 취한 비트들(x_0, x_1, ..., x_7)로 구성된 16개의 8비트 값들 각각에 8비트의 제2 치환 연산 S_8을 적용하는 것과 동치(equivalent)이다.

[0063] 특히, 본 개시서에 따른 제2 치환 연산 S_8은 16×16 = 256개 원소로 구성된 8비트 치환 테이블(substitution table)을 이용하는 LUT(룩업 테이블; lookup table) 방식으로 구현 가능하면서도 동시에 제3의 4비트 치환 연산 S_4의 3회 반복을 포함하는 파이스텔(Feistel) 구조의 비트슬라이스 구현도 가능한 이중성을 가진다.

[0064] 이러한 특성을 만족시키는 제2 치환 연산 S_8의 일 예시는 도 6에 8비트 치환 테이블로 표현된 것과 같다.

[0065] 구체적으로, 도 6은 본 개시서에 따른 예시적인 128비트 치환 연산(S_{128})을 도출하는 제2 치환 연산 S_8을 8비트 치환 테이블로 표현한 것을 그 치환 테이블의 입력값의 왼쪽 4비트를 행의 상단으로부터 하단까지 열거하고 오른쪽 4비트를 열의 왼쪽으로부터 오른쪽까지 열거함으로써 상기 입력값에 대응하는 출력값을 나타낸 도면이다.

[0066] 이 제2 치환 연산 S_8은 3개 라운드를 가진 파이스텔(3-round Feistel) 구조에 동일한 4비트 S-박스(S-box), 즉, 제3 치환 연산 S_4를 세 번 사용하여 설계한 8비트 S-박스로서, 대합적인(involutive) 성질을 가진다. 즉, S_8은 S_8의 역함수와 동일하기 때문에 암호화와 복호화 알고리즘에서 사용하는 치환 연산 S_{128}을 별도로 구현할 필요가 없으므로 경량성의 장점을 가진다.

[0067] 도 7은 본 개시서의 일 실시 예에서 제3 치환 연산 S_4을 이용하여 S_8을 구현한 파이스텔 구조를 개념적으로 나타낸 도면이다. 제2 치환 연산 S_8에 입력되는 값의 최하위 비트 x_0가 x[0]으로, 최상위 비트 x_7가 x[7]로 표현되어 있다.

[0068] 한편, 이 파이스텔(Feistel) 구조에 포함되는 4비트 S-박스, 즉, 상기 제3 치환 연산 S_4을 입력값(input)과 출력값(output)의 쌍으로 표현한 테이블은 도 8로 도시되어 있다.

[0069] 도 9a 및 도 9b는 본 개시서의 일 실시 예에서 제3 치환 연산 S_4을 2가지의 비트슬라이스 구현으로 도시한 것인바, 도 9a는 소프트웨어에 최적화된 비트슬라이스 구현을 나타내고, 도 9b는 하드웨어에 최적화된 비트슬라이스 구현을 나타낸다.

[0070] 도 9a를 참조하면, 상기 제3 치환 연산은, 상기 S4에 입력되는 값의 최하위 비트인 제1 비트(x[0]) 내지 상기 입력되는 값의 최상위 비트인 제4 비트(x[3])에 대하여, 제4 비트와 제1 비트를 AND 연산한 결과값과 제3 비트(x[2])를 XOR 연산한 결과값을 다시 제3 비트에 배정(assign)하는 제1 단계; 제3 비트와 제4 비트를 XOR 연산한 결과값을 다시 제4 비트에 배정하는 제2 단계; 제2 비트(x[1])와 제1 비트를 XOR 연산한 결과값을 다시 제1 비트에 배정하는 제3 단계; 제2 비트와 제1 비트를 OR 연산한 결과값과 제3 비트를 XOR 연산한 결과값을 다시 제3 비트에 배정하는 제4 단계; 제2 비트와 제4 비트를 XOR 연산한 결과값을 임시(temp) 비트(T)에 배정하는 제5 단계; 제1 비트와 제3 비트를 OR 연산한 결과값과 제4 비트를 XOR 연산한 결과값을 다시 제4 비트에 배정하는 제6 단계; 제4 비트와 제3 비트를 AND 연산한 결과값과 제1 비트를 XOR 연산한 결과값을 제2 비트에 배정하는 제7 단계; 및 상기 임시 비트의 값을 제1 비트에 배정하는 제8 단계를 수행함으로써 구현될 수 있다.

[0071] 한편, 도 9b를 참조하면, 상기 제3 치환 연산은, 상기 S4에 입력되는 값의 최하위 비트인 제1 비트 내지 상기 입력되는 값의 최상위 비트인 제4 비트에 대하여, 제4 비트와 제1 비트를 AND 연산한 결과값과 제3 비트를 XOR 연산한 결과값을 다시 제3 비트에 배정(assign)하는 제1 단계; 제3 비트와 제4 비트를 XOR 연산한 결과값을 다시 제4 비트에 배정하는 제2 단계; 제2 비트와 제1 비트를 XOR 연산한 결과값을 다시 제1 비트에 배정하는 제3 단계; 제2 비트와 제1 비트를 OR 연산한 결과값과 제3 비트를 XOR 연산한 결과값을 다시 제3 비트에 배정하는 제4 단계; 제4 비트와 제2 비트를 XOR 연산한 결과값을 다시 제2 비트에 배정하는 제5' 단계; 제1 비트와 제3 비트를 OR 연산한 결과값과 제4 비트를 XOR 연산한 결과값을 다시 제4 비트에 배정하는 제6 단계; 및 제4 비트와 제3 비트를 AND 연산한 결과값과 제1 비트를 XOR 연산한 결과값을 다시 제1 비트에 배정하는 제7' 단계를 수행한 후에, 제1 비트와 제2 비트의 값을 서로 뒤바꾸는 연산(swap)의 제8' 단계를 더 수행하거나 하드웨어 배선(wiring)을 통하여 바로 제1 비트와 제2 비트의 값이 서로 뒤바뀌어 산출되게 함으로써 구현될 수 있다.

[0072] 다시 도 4를 참조하면, 본 개시서의 방법에 이용된 상기 비트 간 위치변환 연산은, 행 전환(row conversion) 및 행별 로테이션(rotation)으로써 구현될 수 있는데, 이는 도 10에 예시되어 있다.

[0073] 도 10은 본 개시서의 데이터 암호화 방법 및 데이터 복호화 방법에서 이용되는 비트 간 위치변환 연산 P_{128}을 128비트 평문에 대하여 적용하는 방식을 개념적으로 나타낸 도면이다.

[0074] 도 10에 예시된 비트 간 위치변환 연산에 속한 제1 연산인 행 전환은 그 모든 궤도(orbit)들이 (0, 5, 3, 4, 2)(1, 6)(7)로 표현 가능하고, 동 비트 간 위치변환 연산에 속한 제2 연산인 행별 로테이션은 제1 열부터 순서대로 1, 0, 9, 10, 2, 8, 15, 7비트만큼 왼쪽 로테이션을 하는 연산으로 표현 가능하다.

[0075] 엄밀하게는 데이터 복호화 방법에서 치환 연산 S_{128}이 그대로 이용되는 것과 달리 데이터 복호화 방법에서 비트 간 위치변환 연산 P_{128}의 역연산이 이용되는데, 이 또한 행별 로테이션 및 행 전환으로 수행될 수 있음은 자명하

다(합성 함수의 역함수).

[0076] 이제 위에서 설명된 데이터 암호화 방법에 대응하는 데이터 복호화 방법을 설명하기로 한다.

[0077] 본 개시서에 따른 데이터 복호화 방법은, 소정 라운드 횟수인 자연수 M에 대하여, 제1 내지 제M+1 라운드 키를 생성하는 단계; 및 생성된 상기 제1 내지 제M+1 라운드 키를 이용한 복호화로써 암호문으로부터 평문을 산출하는 단계를 포함하는데, 라운드 키를 생성하는 단계는 데이터 암호화 방법에서와 같다.

[0078] 전술한 단계(S200)의 내용을 이해한 통상의 기술자에게는 자명할 것이지만 설명의 명확성을 위하여 다소 부언하면, 상기 암호문으로부터 상기 평문을 산출하는 단계(전술한 S220 내지 S240에 대응)는, (a) n=M, M-1, ..., 1에 대하여, 복호화부(220)가, 순차적으로 (i) 제n+1 라운드 키를 이용하여 제n+1 중간값으로부터 제n 결과값을 산출하는 단계 및 (ii) 제n 결과값에 치환 연산(substitution)을 적용한 결과에 비트 간 위치변환 연산(bit permutation)의 역연산을 적용하거나 상기 제n 결과값에 상기 비트 간 위치변환 연산의 상기 역연산을 적용한 결과에 상기 치환 연산을 적용하여 제n 중간값을 산출하는 단계를 반복 수행하는 단계로서, 상기 암호문은 제M+1 중간값인, 단계; 및 (b) 상기 제1 라운드 키를 이용하여 제1 중간값으로부터 평문을 산출하는 단계를 포함할 수 있다.

[0079] 지금까지 설명된 본 개시서의 블록 암호화 및 복호화 방법에서는 라운드 함수의 구성요소인 치환 연산(S_{128})과 비트 간 위치변환 연산(P_{128})이 비트슬라이스 구현에 적합하도록 면밀히 선택되었다. 128비트의 블록(평문, 암호문 등)을 8×16의 비트열로 나타낸 도 5를 다시 참고하면, 각각의 행은 16비트 레지스터에 대응하는 것으로 볼 수 있다. 따라서 S_8의 8비트 비트슬라이스 구현 논리를 8개의 레지스터로 시행하면, S_{128}의 8개 S-박스를 병렬 연산으로 구현할 수 있는 장점이 있다. 또한 비트 간 위치변환 연산(P_{128})은 레지스터 간 위치 변경과 16비트 로테이션(rotation) 연산만으로 구현이 가능하므로 연산의 종류가 단순화되는 장점도 있는데, 그 역연산도 마찬가지이다.

[0080] 본 개시서의 방법들은 설명의 편의상 하나의 프로세서에서 실현되는 것으로 예시되었으나, 암호화·복호화부는 하나의 프로세서 내에서 하나의 코어 또는 복수의 코어를 가진 것으로 구성될 수 있을 뿐만 아니라 복수개의 프로세서들이 서로 연동되도록 구성되어 병렬 연산이 가능해질 수도 있다는 점이 이해될 것이다. 또한 전술한 본 발명 방법의 각 단계는, 하나의 연산 장치가 직접 수행하거나 상기 하나의 연산 장치가 상기 하나의 연산 장치에 연동되는 타 연산 장치로 하여금 수행하도록 지원함으로써 수행될 수 있다.

[0081] 이와 같이 본 발명은 그 모든 실시 예 및 변형례에 걸쳐, 비트슬라이스 구현이 가능한 8비트 치환 테이블과 왼쪽 로테이션 연산만을 이용하여 설계되었으므로 IoT 환경에서의 경량 구현에 적합할 뿐만 아니라 최소한의 비선형 연산만으로 구현이 가능하기 때문에 비교적 적은 추가 리소스로도 부채널 분석 대응 마스킹 기법의 적용이 가능한 효과가 있다. 뿐만 아니라, 이러한 효과를 낼 수 있도록 차분도가 16 이하이며, 비선형도가 96 이상인 동시에 DBN이 3인 8비트 S-박스가 본 개시서에 예시적으로 개시된 S_8에 한정되지 않을 수 있음을 통상의 기술자는 이해할 수 있을 것이다.

[0082] 위 실시 예의 설명에 기초하여 해당 기술분야의 통상의 기술자는, 본 발명의 방법 및/또는 프로세스들, 그리고 그 단계들이 하드웨어, 소프트웨어 또는 특정 용례에 적합한 하드웨어 및 소프트웨어의 임의의 조합으로 실현될 수 있다는 점을 명확하게 이해할 수 있다. 더욱이 본 발명의 기술적 해법의 대상물 또는 선행 기술들에 기여하는 부분들은 다양한 컴퓨터 구성요소를 통하여 수행될 수 있는 프로그램 명령어의 형태로 구현되어 기계 판독 가능한 기록 매체에 기록될 수 있다. 상기 기계 판독 가능한 기록 매체는 프로그램 명령어, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 기계 판독 가능한 기록 매체에 기록되는 프로그램 명령어는 본 발명을 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 분야의 통상의 기술자에게 공지되어 사용 가능한 것일 수도 있다. 기계 판독 가능한 기록 매체의 예에는, 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체, CD-ROM, DVD, Blu-ray와 같은 광기록 매체, 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 ROM, RAM, 플래시 메모리 등과 같은 프로그램 명령어를 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 프로그램 명령어의 예에는, 전술한 장치들 중 어느 하나뿐만 아니라 프로세서, 프로세서 아키텍처 또는 상이한 하드웨어 및 소프트웨어의 조합들의 이종 조합, 또는 다른 어떤 프로그램 명령어들을 실행할 수 있는 기계 상에서 실행되기 위하여 저장 및 컴파일 또는 인터프리트될 수 있는, C와 같은 구조적 프로그래밍 언어, C++ 같은 객체지향적 프로그래밍 언어 또는 고급 또는 저급 프로그래밍 언어(어셈블리어, 하드웨어 기술 언어들 및 데이터베이스 프로그래밍 언어 및 기술들)를 사용하여 만들어질 수 있는바, 기계어 코드, 바이트코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될

수 있는 고급 언어 코드도 이에 포함된다.

[0083] 따라서 본 발명에 따른 일 태양에서는, 앞서 설명된 방법 및 그 조합들이 하나 이상의 연산 장치들에 의하여 수행될 때, 그 방법 및 방법의 조합들이 각 단계들을 수행하는 실행 가능한 코드로서 실시될 수 있다. 다른 일 태양에서는, 상기 방법은 상기 단계들을 수행하는 시스템들로서 실시될 수 있고, 방법들은 장치들에 걸쳐 여러 가지 방법으로 분산되거나 모든 기능들이 하나의 전용, 독립형 장치 또는 다른 하드웨어에 통합될 수 있다. 또 다른 일 태양에서는, 위에서 설명한 프로세스들과 연관된 단계들을 수행하는 수단들은 앞서 설명한 임의의 하드웨어 및/또는 소프트웨어를 포함할 수 있다. 그러한 모든 순차 결합 및 조합들은 본 개시서의 범위 내에 속하도록 의도된 것이다.

[0084] 예를 들어, 상기 하드웨어 장치는 본 발명에 따른 처리를 수행하기 위해 하나 이상의 소프트웨어 모듈로서 작동하도록 구성될 수 있으며, 그 역도 마찬가지이다. 상기 하드웨어 장치는, 프로그램 명령어를 저장하기 위한 ROM/RAM 등과 같은 메모리와 결합되고 상기 메모리에 저장된 명령어들을 실행하도록 구성되는 MPU, CPU, GPU, TPU와 같은 프로세서를 포함할 수 있으며, 외부 장치와 신호를 주고받을 수 있는 입출력부를 포함할 수 있다. 덧붙여, 상기 하드웨어 장치는 개발자들에 의하여 작성된 명령어들을 전달받기 위한 키보드, 마우스, 기타 외부 입력장치를 포함할 수 있다.

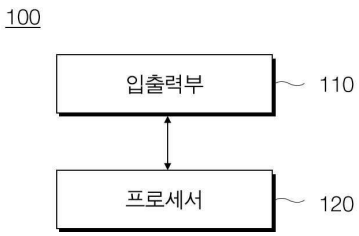
[0085] 이상에서 본 발명이 구체적인 구성요소 등과 같은 특정 사항들과 한정된 실시 예 및 도면에 의해 설명되었으나, 이는 본 발명의 보다 전반적인 이해를 돕기 위해서 제공된 것일 뿐, 본 발명이 상기 실시 예들에 한정되는 것은 아니며, 본 발명이 속하는 기술분야에서 통상적인 지식을 가진 사람이라면 이러한 기재로부터 다양한 수정 및 변형을 꾀할 수 있다.

[0086] 따라서, 본 발명의 사상은 상기 설명된 실시 예에 국한되어 정해져서는 아니되며, 후술하는 특허청구범위뿐만 아니라 이 특허청구범위와 균등하게 또는 등가적으로 변형된 모든 것들은 본 발명의 사상의 범주에 속한다고 할 것이다.

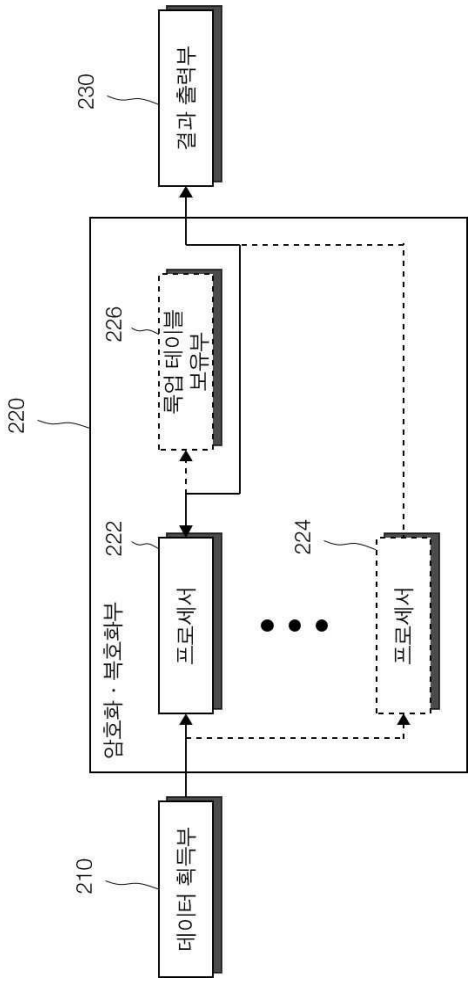
[0087] 그와 같이 균등하게 또는 등가적으로 변형된 것에는, 예컨대 본 발명에 따른 방법을 실시한 것과 동일한 결과를 낼 수 있는, 논리적으로 동치(logically equivalent)인 방법이 포함될 것인바, 본 발명의 진의 및 범위는 전술한 예시들에 의하여 제한되어서는 아니되며, 법률에 의하여 허용 가능한 가장 넓은 의미로 이해되어야 한다.

도면

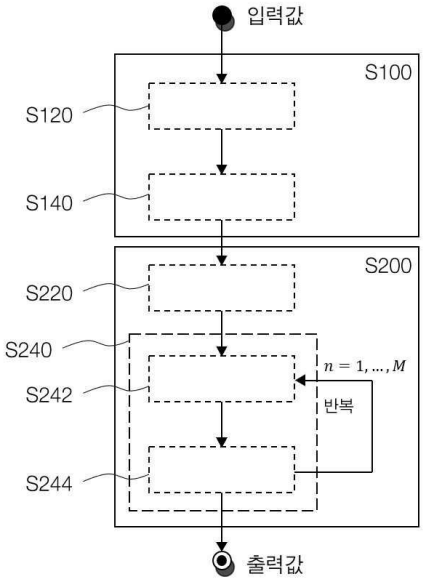
도면1



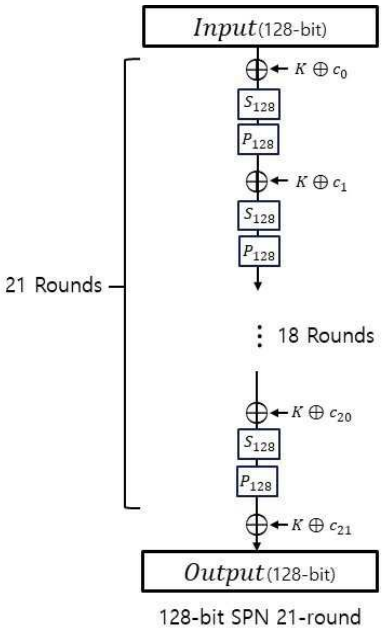
도면2


230
결과 출력부
220
226
룩업 테이블 보유부
222
프로세서
224
프로세서
암호화 · 복호화부
210
데이터 획득부

도면3

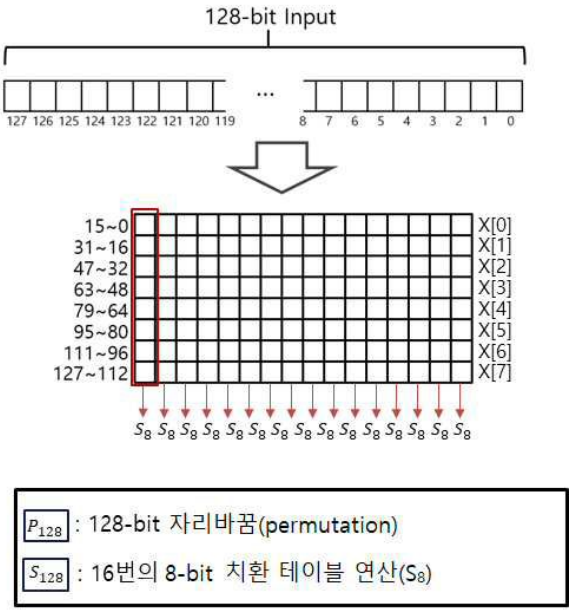

입력값
S100
S120
S140
S200
S220
S240
S242
S244
$n = 1, \ldots, M$
반복
출력값

도면4



128-bit SPN 21-round

도면5



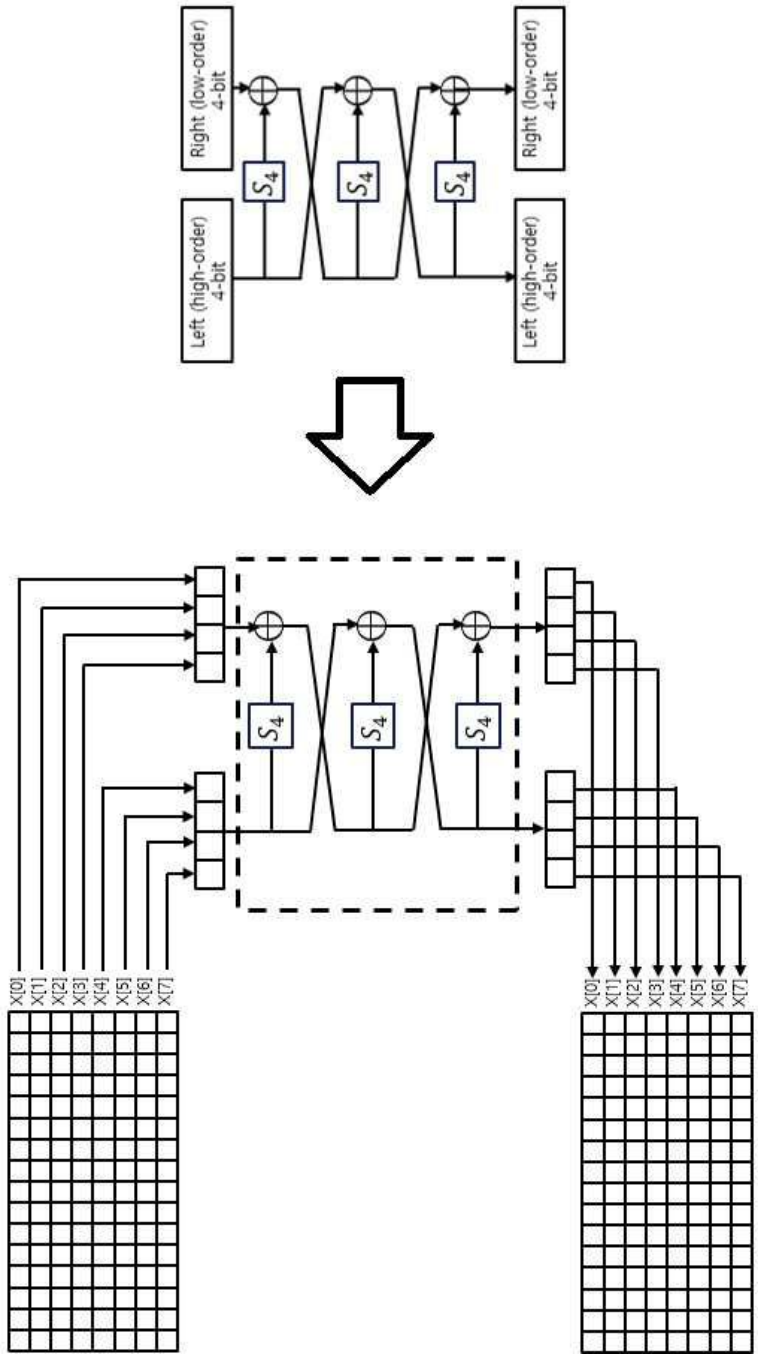
P_{128} : 128-bit 자리바꿈(permutation)

S_{128} : 16번의 8-bit 치환 테이블 연산(S_8)

도면6

Left (high-order) 4-bit \ Right (low-order) 4-bit	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0x00	0xCF	0xD5	0xF7	0x57	0x3A	0x2B	0x8E	0x92	0xAF	0x68	0x17	0xE7	0x75	0xBF	0x4A
1	0xF8	0x6F	0xA8	0x5C	0x81	0xB8	0x72	0x0B	0x41	0x28	0x39	0x9D	0x1C	0xD6	0xCC	0xE8
2	0x5E	0xC2	0x6D	0x94	0x80	0xB9	0x34	0x4F	0x19	0x7C	0xA1	0x06	0xEA	0x2D	0xD4	0xF6
3	0x77	0x87	0x48	0xDB	0x26	0x59	0x93	0xAE	0xB6	0x1A	0x05	0x66	0xCD	0xE9	0xF5	0x3F
4	0x7D	0x18	0xC9	0x64	0x88	0x45	0xB2	0x98	0x32	0xAA	0x0F	0xFA	0xE2	0xDF	0x58	0x27
5	0xA5	0x8B	0x9B	0x53	0xD0	0x7E	0x67	0x04	0x4E	0x35	0xFD	0xC6	0x13	0xE5	0x20	0xBD
6	0xB3	0x99	0x62	0xA7	0x43	0xEC	0x3B	0x56	0x0A	0x73	0xFC	0xC7	0xD9	0x22	0x85	0x11
7	0xE3	0xDE	0x16	0x69	0x96	0x0D	0xC0	0x30	0x78	0xB0	0xA4	0x8A	0x29	0x40	0x55	0xF3
8	0x24	0x14	0x91	0xE1	0xF9	0x6E	0xC1	0x31	0x44	0x89	0x7B	0x51	0xB4	0xD3	0x07	0xA0
9	0xFE	0x82	0x08	0x36	0x23	0xD8	0x74	0xE6	0x47	0x61	0x9A	0x52	0xB7	0x1B	0xCA	0xA3
A	0x8F	0x2A	0xF0	0x9F	0x7A	0x50	0xA6	0x63	0x12	0xE4	0x49	0xDA	0xC4	0xBA	0x37	0x09
B	0x79	0xB1	0x46	0x60	0x8C	0xEF	0x38	0x9C	0x15	0x25	0xAD	0xDD	0xC3	0x5F	0xFB	0x0E
C	0x76	0x86	0x21	0xBC	0xAC	0xDC	0x5B	0x6B	0xED	0x42	0x9E	0xF1	0x1E	0x3C	0xCE	0x01
D	0x54	0xF2	0xEE	0x8D	0x2E	0x02	0x1D	0xD7	0x95	0x6C	0xAB	0x33	0xC5	0xBB	0x71	0x4D
E	0xFF	0x83	0x4C	0x70	0xA9	0x5D	0x97	0x0C	0x1F	0x3D	0x2C	0xEB	0x65	0xC8	0xD2	0xB5
F	0xA2	0xCB	0xD1	0x7F	0xF4	0x3E	0x2F	0x03	0x10	0x84	0x4B	0xBE	0x6A	0x5A	0x90	0xE0

도면7



도면8

input	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
output	0	C	D	F	5	3	2	8	9	A	6	1	E	7	B	4

도면9a

```
# Input: (MSB) X[3], X[2], X[1], X[0] (LSB)
X[2] = XOR (X[2], AND (X[0], X[3]))
X[3] = XOR (X[3], X[2])
X[0] = XOR (X[0], X[1])
X[2] = XOR (X[2], OR (X[1], X[0]))
T = XOR (X[1], X[3])
X[3] = XOR (X[3], OR (X[0], X[2]))
X[1] = XOR (X[0], AND (X[3], X[2]))
X[0] = T
# Output: (MSB) X[3], X[2], X[1], X[0] (LSB)
```

도면9b

```
# Input: (MSB) X[3], X[2], X[1], X[0] (LSB)
X[2] = XOR (X[2], AND (X[0], X[3]))
X[3] = XOR (X[3], X[2])
X[0] = XOR (X[0], X[1])
X[2] = XOR (X[2], OR (X[1], X[0]))
X[1] = XOR (X[1], X[3])
X[3] = XOR (X[3], OR (X[0], X[2]))
X[0] = XOR (X[0], AND (X[3], X[2]))
# Output: (MSB) X[3], X[2], X[0], X[1] (LSB)
```

도면10