(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2014년03월05일
(11) 등록번호 10-1370223
(24) 등록일자 2014년02월26일

(51) 국제특허분류(Int. Cl.)
H04L 9/06 (2006.01) *H04L 9/14* (2006.01)
(21) 출원번호 10-2011-7012214
(22) 출원일자(국제) 2009년10월28일
심사청구일자 2011년05월27일
(85) 번역문제출일자 2011년05월27일
(65) 공개번호 10-2011-0089319
(43) 공개일자 2011년08월05일
(86) 국제출원번호 PCT/US2009/062391
(87) 국제공개번호 WO 2010/056531
국제공개일자 2010년05월20일
(30) 우선권주장
12/262,092 2008년10월30일 미국(US)
(56) 선행기술조사문헌
US20060265563 A1*
Cryptography and network security : principles and practice / William Stallings / Stallings, William / Prentice Hall (2006)
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
퀄컴 인코포레이티드
미국 92121-1714 캘리포니아주 샌 디에고 모어하우스 드라이브 5775
(72) 발명자
호크스 필립 마이클
미국 92121 캘리포니아주 샌디에고 모어하우스 드라이브 5775
샤오 루
미국 92121 캘리포니아주 샌디에고 모어하우스 드라이브 5775
(뒷면에 계속)
(74) 대리인
특허법인코리아나

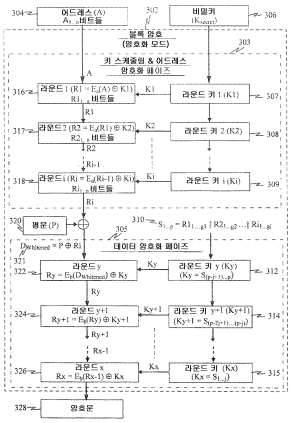
전체 청구항 수 : 총 43 항 심사관 : 이형일

(54) 발명의 명칭 **로우 레이턴시 블록 암호**

(57) 요 약

데이터를 그 데이터가 저장될 메모리 어드레스에 기초하여 암호화함으로써 데이터를 보안하는 블록 암호가 제공된다. 메모리 어드레스에의 저장을 위한 데이터를 암호화할 때, 메모리 어드레스는 제 1 복수의 블록 암호 라운드들에서 암호화된다. 데이터 라운드 키들이 제 1 복수의 블록 암호 라운드들로부터의 정보를 이용하여 생성된다. 저장될 데이터는 암호화된 메모리 어드레스와 결합되고, 데이터 라운드 키들을 이용하여 제 2 복수의 블록 암호 라운드들에서 암호화된다. 암호화된 데이터는 그 후 메모리 위치에 저장된다. 데이터를 복호화할 때, 메모리 어드레스는 다시 앞서와 같이 암호화되는 한편, 암호화된 저장 데이터는 부분적으로 복호화된 데이터를 획득하기 위해 데이터 라운드 키들을 이용하여 제 2 복수의 블록 암호 라운드들에서 복호화된다.
부분적으로 복호화된 데이터는 완전히 복호화된 데이터를 획득하기 위해 암호화된 메모리 어드레스와 결합된다.

대 표 도 - 도3



(72) 발명자

로즈 그레고리 고든

미국 92121 캘리포니아주 샌디에고 모어하우스 드라이브 5775

밀렌도르프 스티브

미국 92121 캘리포니아주 샌디에고 모어하우스 드라이브 5775

특허청구의 범위

청구항 1

저장 디바이스의 메모리 어드레스에의 저장을 위한 데이터를 암호화하는 방법으로서,

상기 메모리 어드레스를, 메모리 어드레스 암호화 라운드로부터의 출력이 다음의 메모리 어드레스 암호화 라운드에 대한 입력으로 사용되는, 제 1 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하는 단계;

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 하나 이상의 중간 (intermediate) 라운드들로부터의 정보를 이용하여 데이터 라운드 키들을 생성하는 단계;

결합된 데이터를 획득하기 위해 상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 후에 상기 데이터를 상기 암호화된 메모리 어드레스와 결합하는 단계;

상기 데이터 라운드 키들을 이용하여 상기 결합된 데이터를, 데이터 암호화 라운드로부터의 출력이 다음의 데이터 암호화 라운드에 대한 입력으로 사용되는, 제 2 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하는 단계; 및

상기 암호화된, 결합된 데이터를 상기 저장 디바이스의 상기 메모리 어드레스에 저장하는 단계를 포함하고,

상기 메모리 어드레스를 암호화하는 단계는, 상기 데이터가 이용가능하기 전에 시작되는, 데이터 암호화 방법.

청구항 2

삭제

청구항 3

제 1 항에 있어서,

상기 메모리 어드레스를 암호화하는 단계는,

상기 메모리 어드레스를 제 1 변환 함수에 따라 변환하는 단계; 및

상기 변환된 메모리 어드레스를 라운드 키와 혼합하는 단계를 포함하는, 데이터 암호화 방법.

청구항 4

삭제

청구항 5

제 1 항에 있어서,

상기 데이터 라운드 키들을 생성하는 단계는,

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 적어도 일부에 대해 상기 암호화된 메모리 어드레스의 복수의 중간 라운드들로부터 복수의 비트들을 추출하는 단계; 및

상기 추출된 복수의 비트들의 세그먼트들로부터 상기 데이터 라운드 키들을 선택하는 단계를 포함하는, 데이터 암호화 방법.

청구항 6

제 5 항에 있어서,

상기 데이터 라운드 키들을 생성하는 단계는,

상기 추출된 복수의 비트들을 상기 데이터 라운드 키들이 선택되는 스트링에 연쇄시키는 단계를 포함하는, 데이터 암호화 방법.

청구항 7

제 5 항에 있어서,

상기 제 2 복수의 순차적 반복적 블록 암호 라운드들 중 초기의 (earlier) 라운드들에 대해 이용되는 상기 데이터 라운드 키들은, 상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 최근의 (later) 라운드들로부터의 상기 암호화된 메모리 어드레스로부터의 비트들을 이용하여 생성되는, 데이터 암호화 방법.

청구항 8

제 1 항에 있어서,

상기 데이터는 가역 연산 (invertible operation) 에 의해 상기 암호화된 메모리 어드레스와 결합되는, 데이터 암호화 방법.

청구항 9

제 1 항에 있어서,

상기 결합된 데이터를 암호화하는 단계는,

상기 결합된 데이터를 제 2 변환 함수에 따라 변환하는 단계; 및

상기 변환된 결합된 데이터를 상기 데이터 라운드 키들 중 하나 이상과 혼합하는 단계를 포함하는, 데이터 암호화 방법.

청구항 10

제 1 항에 있어서,

상기 결합된 데이터를 암호화하는 단계는,

상기 결합된 데이터를 복수의 데이터 세그먼트들로 세그먼트화하는 단계; 및

상이한 데이터 세그먼트들에 대해 비트 치환을 수행하는 단계를 더 포함하는, 데이터 암호화 방법.

청구항 11

제 1 항에 있어서,

상기 제 2 복수의 순차적 반복적 블록 암호 라운드들에서의 블록 암호 라운드들의 수는 상기 제 1 복수의 순차적 반복적 블록 암호 라운드들에서의 블록 암호 라운드들의 수보다 큰, 데이터 암호화 방법.

청구항 12

제 1 항에 있어서,

상기 결합된 데이터는 상기 제 2 복수의 순차적 반복적 블록 암호 라운드들에 걸쳐 순차적으로 암호화되는, 데이터 암호화 방법.

청구항 13

삭제

청구항 14

메모리 어드레스를, 메모리 어드레스 암호화 라운드로부터의 출력이 다음의 메모리 어드레스 암호화 라운드의 입력으로 사용되는, 제 1 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하기 위한 어드레스 암호화 모듈;

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 하나 이상의 중간 (intermediate) 라운드들로부터의 정보를 이용하여 데이터 라운드 키들을 생성하기 위한 키 스케줄링 모듈;

결합된 데이터를 획득하도록 상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 후에 상기 암호화된 메모리

어드레스와 데이터를 결합하기 위한 결합기; 및

상기 데이터 라운드 키들을 이용하여 상기 결합된 데이터를, 데이터 암호화 라운드로부터의 출력이 다음의 데이터 암호화 라운드로부터의 출력으로 사용되는, 제 2 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하기 위한 데이터 암호화 모듈을 포함하고,

상기 메모리 어드레스를 암호화하는 것은, 상기 데이터가 이용가능하기 전에 시작되는, 블록 암호 디바이스.

청구항 15

삭제

청구항 16

제 14 항에 있어서,

상기 어드레스 암호화 모듈은 또한,

상기 메모리 어드레스를 제 1 변환 함수에 따라 변환하고;

상기 변환된 메모리 어드레스를 라운드 키와 혼합하도록 구성되는, 블록 암호 디바이스.

청구항 17

삭제

청구항 18

제 14 항에 있어서,

상기 키 스케줄링 모듈은 또한,

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 적어도 일부에 대한 상기 암호화된 메모리 어드레스로부터 복수의 비트들을 추출하고;

상기 추출된 복수의 비트들의 세그먼트들로부터 상기 데이터 라운드 키들을 선택하도록 구성되는, 블록 암호 디바이스.

청구항 19

제 18 항에 있어서,

상기 키 스케줄링 모듈은 또한,

상기 추출된 복수의 비트들을 상기 데이터 라운드 키들이 선택되는 스트링에 연쇄시키도록 구성되는, 블록 암호 디바이스.

청구항 20

제 18 항에 있어서,

상기 제 2 복수의 순차적 반복적 블록 암호 라운드들 중 초기의 (earlier) 라운드들에 대해 이용되는 상기 데이터 라운드 키들은, 상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 최근의 (later) 라운드들로부터의 상기 암호화된 메모리 어드레스로부터의 비트들을 이용하여 생성되는, 블록 암호 디바이스.

청구항 21

삭제

청구항 22

제 14 항에 있어서,

상기 데이터 암호화 모듈은 또한,

상기 결합된 데이터를 제 2 변환 함수에 따라 변환하고;

상기 변환된 결합된 데이터를 상기 데이터 라운드 키들 중 하나 이상과 혼합하도록 구성되는, 블록 암호 디바이스.

청구항 23

제 14 항에 있어서,

상기 데이터 암호화 모듈은 또한,

상기 결합된 데이터를 복수의 데이터 세그먼트들로 세그먼트화하고;

상이한 데이터 세그먼트들에 대해 비트 치환을 수행하도록 구성되는, 블록 암호 디바이스.

청구항 24

제 14 항에 있어서,

상기 제 2 복수의 순차적 반복적 블록 암호 라운드들에서의 블록 암호 라운드들의 수는 상기 제 1 복수의 순차적 반복적 블록 암호 라운드들에서의 블록 암호 라운드들의 수보다 큰, 블록 암호 디바이스.

청구항 25

제 14 항에 있어서,

상기 결합된 데이터는 상기 제 2 복수의 순차적 반복적 블록 암호 라운드들에 걸쳐 순차적으로 암호화되는, 블록 암호 디바이스.

청구항 26

메모리 어드레스를, 메모리 어드레스 암호화 라운드로부터의 출력이 다음의 메모리 어드레스 암호화 라운드에 대한 입력으로 사용되는, 제 1 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하는 수단;

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 하나 이상의 중간 (intermediate) 라운드들로부터의 정보를 이용하여 데이터 라운드 키들을 생성하는 수단;

결합된 데이터를 획득하기 위하여 상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 후에 상기 암호화된 메모리 어드레스와 데이터를 결합하는 수단; 및

상기 데이터 라운드 키들을 이용하여 상기 결합된 데이터를, 데이터 암호화 라운드로부터의 출력이 다음의 데이터 암호화 라운드에 대한 입력으로 사용되는, 제 2 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하는 수단을 포함하고,

상기 메모리 어드레스를 암호화하는 것은, 상기 데이터가 이용가능하기 전에 시작되는, 블록 암호 디바이스.

청구항 27

프로세싱 회로를 포함하는 프로세서로서,

상기 프로세싱 회로는,

메모리 어드레스를, 메모리 어드레스 암호화 라운드로부터의 출력이 다음의 메모리 어드레스 암호화 라운드에 대한 입력으로 사용되는, 제 1 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하고;

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 하나 이상의 중간 (intermediate) 라운드들로부터의 정보를 이용하여 데이터 라운드 키들을 생성하고;

결합된 데이터를 획득하기 위해 상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 후에 상기 암호화된 메모리 어드레스와 데이터를 결합하며;

상기 데이터 라운드 키들을 이용하여 상기 결합된 데이터를, 데이터 암호화 라운드로부터의 출력이 다음의 데이터 암호화 라운드로부터의 입력으로 사용되는, 제 2 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하도

록 구성되고,

상기 메모리 어드레스를 암호화하는 것은, 상기 데이터가 이용가능하기 전에 시작되는, 프로세싱 회로를 포함하는 프로세서.

청구항 28

블록 암호 암호화를 위한 명령들을 포함하는 컴퓨터-판독가능 매체로서,

상기 명령들은, 하나 이상의 프로세서들에 의한 실행 시에, 상기 프로세서들로 하여금,

메모리 어드레스를, 메모리 어드레스 암호화 라운드로부터의 출력이 다음의 메모리 어드레스 암호화 라운드에 대한 입력으로 사용되는, 제 1 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하게 하고;

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 하나 이상의 중간 (intermediate) 라운드들로부터의 정보를 이용하여 데이터 라운드 키들을 생성하게 하고;

결합된 데이터를 획득하기 위해 상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 후에 상기 암호화된 메모리 어드레스와 데이터를 결합하게 하며;

상기 데이터 라운드 키들을 이용하여 상기 결합된 데이터를, 데이터 암호화 라운드로부터의 출력이 다음의 데이터 암호화 라운드에 대한 입력으로 사용되는, 제 2 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하게 하고,

상기 메모리 어드레스를 암호화하는 것은, 상기 데이터가 이용가능하기 전에 시작되는, 컴퓨터-판독가능 매체.

청구항 29

저장 디바이스의 메모리 어드레스로부터 검색된 데이터를 복호화하는 방법으로서,

암호화된 메모리 어드레스를 획득하기 위해 상기 메모리 어드레스를, 메모리 어드레스 암호화 라운드로부터의 출력이 다음의 메모리 어드레스 암호화 라운드에 대한 입력으로 사용되는, 제 1 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하는 단계;

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 하나 이상의 중간 (intermediate) 라운드들로부터의 정보를 이용하여 데이터 라운드 키들을 생성하는 단계;

상기 저장 디바이스의 상기 메모리 어드레스로부터 암호화된 데이터를 검색하는 단계;

부분적으로 복호화된 데이터를 획득하기 위해 상기 데이터 라운드 키들을 이용하여 상기 암호화된 데이터를, 데이터 복호화 라운드로부터의 출력이 다음의 데이터 복호화 라운드에 대한 입력으로 사용되는, 제 2 복수의 순차적 반복적 블록 암호 라운드들에서 복호화하는 단계; 및

완전히 복호화된 데이터를 획득하기 위해 상기 부분적으로 복호화된 데이터를 상기 암호화된 메모리 어드레스와 결합하는 단계를 포함하고,

상기 메모리 어드레스를 암호화하는 단계는, 상기 데이터가 이용가능하기 전에 시작되는, 데이터 복호화 방법.

청구항 30

삭제

청구항 31

삭제

청구항 32

제 29 항에 있어서,

상기 메모리 어드레스를 암호화하는 단계는,

상기 메모리 어드레스를 제 1 변환 함수에 따라 변환하는 단계; 및

상기 변환된 메모리 어드레스를 라운드 키와 혼합하는 단계를 포함하는, 데이터 복호화 방법.

청구항 33

삭제

청구항 34

제 29 항에 있어서,

상기 데이터 라운드 키들을 생성하는 단계는,

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 적어도 일부에 대해 상기 암호화된 메모리 어드레스의 복수의 중간 라운드들로부터 복수의 비트들을 추출하는 단계; 및

상기 추출된 복수의 비트들의 세그먼트들로부터 상기 데이터 라운드 키들을 선택하는 단계를 포함하는, 데이터 복호화 방법.

청구항 35

제 34 항에 있어서,

상기 데이터 라운드 키들을 생성하는 단계는,

상기 추출된 복수의 비트들을 상기 데이터 라운드 키들이 선택되는 스트링에 연쇄시키는 단계를 포함하는, 데이터 복호화 방법.

청구항 36

제 34 항에 있어서,

상기 제 2 복수의 순차적 반복적 블록 암호 라운드들 중 초기의 (earlier) 라운드들에 대해 이용되는 상기 데이터 라운드 키들은, 상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 초기의 라운드들로부터의 상기 암호화된 메모리 어드레스로부터의 비트들을 이용하여 생성되는, 데이터 복호화 방법.

청구항 37

제 29 항에 있어서,

상기 부분적으로 복호화된 데이터는 가역 연산 (invertible operation) 에 의해 상기 암호화된 메모리 어드레스와 결합되는, 데이터 복호화 방법.

청구항 38

제 29 항에 있어서,

상기 암호화된 데이터를 복호화하는 단계는,

상기 암호화된 데이터를 제 2 역변환 함수에 따라 변환하는 단계; 및

상기 변환된 암호화된 데이터를 상기 데이터 라운드 키들 중 하나 이상과 혼합하는 단계를 포함하는, 데이터 복호화 방법.

청구항 39

삭제

청구항 40

제 29 항에 있어서,

상기 제 2 복수의 순차적 반복적 블록 암호 라운드들에서의 블록 암호 라운드들의 수는 상기 제 1 복수의 순차적 반복적 블록 암호 라운드들에서의 블록 암호 라운드들의 수보다 큰, 데이터 복호화 방법.

청구항 41

제 29 항에 있어서,

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들은 상기 제 2 복수의 순차적 반복적 블록 암호 라운드들과 동시에 실행되는, 데이터 복호화 방법.

청구항 42

암호화된 메모리 어드레스를 획득하기 위해 메모리 어드레스를, 메모리 어드레스 암호화 라운드로부터의 출력이 다음의 메모리 어드레스 암호화 라운드에 대한 입력으로 사용되는, 제 1 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하기 위한 어드레스 암호화 모듈;

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 하나 이상의 중간 (intermediate) 라운드들로부터의 정보를 이용하여 데이터 라운드 키들을 생성하기 위한 키 스케줄링 모듈;

부분적으로 복호화된 데이터를 획득하기 위해 상기 데이터 라운드 키들을 이용하여 암호화된 데이터를, 데이터 복호화 라운드로부터의 출력이 다음의 데이터 복호화 라운드에 대한 입력으로 사용되는, 제 2 복수의 순차적 반복적 블록 암호 라운드들에서 복호화하기 위한 데이터 복호화 모듈; 및

완전히 복호화된 데이터를 획득하기 위해 상기 부분적으로 복호화된 데이터를 상기 암호화된 메모리 어드레스와 결합하기 위한 결합기를 포함하고,

상기 메모리 어드레스를 암호화하는 것은, 상기 데이터가 이용가능하기 전에 시작되는, 블록 암호 디바이스.

청구항 43

삭제

청구항 44

제 42 항에 있어서,

상기 키 스케줄링 모듈은 또한,

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 적어도 일부에 대한 상기 암호화된 메모리 어드레스로부터 복수의 비트들을 추출하고;

상기 추출된 복수의 비트들의 세그먼트들로부터 상기 데이터 라운드 키들을 선택하도록 구성되는, 블록 암호 디바이스.

청구항 45

제 42 항에 있어서,

상기 제 2 복수의 순차적 반복적 블록 암호 라운드들 중 초기의 (earlier) 라운드들에 대해 이용되는 상기 데이터 라운드 키들은, 상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 초기의 라운드들로부터의 상기 암호화된 메모리 어드레스로부터의 비트들을 이용하여 생성되는, 블록 암호 디바이스.

청구항 46

삭제

청구항 47

제 42 항에 있어서,

상기 제 2 복수의 순차적 반복적 블록 암호 라운드들에서의 블록 암호 라운드들의 수는 상기 제 1 복수의 순차적 반복적 블록 암호 라운드들에서의 블록 암호 라운드들의 수보다 큰, 블록 암호 디바이스.

청구항 48

암호화된 메모리 어드레스를 획득하기 위해 메모리 어드레스를, 메모리 어드레스 암호화 라운드로부터의 출력이

다음의 메모리 어드레스 암호화 라운드에 대한 입력으로 사용되는, 제 1 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하는 수단;

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 하나 이상의 중간 (intermediate) 라운드들로부터의 정보를 이용하여 데이터 라운드 키들을 생성하는 수단;

부분적으로 복호화된 데이터를 획득하기 위해 상기 데이터 라운드 키들을 이용하여 암호화된 데이터를, 데이터 복호화 라운드로부터의 출력이 다음의 데이터 복호화 라운드에 대한 입력으로 사용되는, 제 2 복수의 순차적 반복적 블록 암호 라운드들에서 복호화하는 수단; 및

완전히 복호화된 데이터를 획득하기 위해 상기 부분적으로 복호화된 데이터를 상기 암호화된 메모리 어드레스와 결합하는 수단을 포함하고,

상기 메모리 어드레스를 암호화하는 것은, 상기 데이터가 이용가능하기 전에 시작되는, 블록 암호 디바이스.

청구항 49

제 48 항에 있어서,

상기 메모리 어드레스로부터 상기 암호화된 데이터를 검색하는 수단을 더 포함하는, 블록 암호 디바이스.

청구항 50

프로세싱 회로를 포함하는 프로세서로서,

상기 프로세싱 회로는,

암호화된 메모리 어드레스를 획득하기 위해 메모리 어드레스를, 메모리 어드레스 암호화 라운드로부터의 출력이 다음의 메모리 어드레스 암호화 라운드에 대한 입력으로 사용되는, 제 1 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하고;

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 하나 이상의 중간 (intermediate) 라운드들로부터의 정보를 이용하여 데이터 라운드 키들을 생성하고;

부분적으로 복호화된 데이터를 획득하기 위해 상기 데이터 라운드 키들을 이용하여 암호화된 데이터를, 데이터 복호화 라운드로부터의 출력이 다음의 데이터 복호화 라운드에 대한 입력으로 사용되는, 제 2 복수의 순차적 반복적 블록 암호 라운드들에서 복호화하며;

완전히 복호화된 데이터를 획득하기 위해 상기 부분적으로 복호화된 데이터를 상기 암호화된 메모리 어드레스와 결합하도록 구성되고,

상기 메모리 어드레스를 암호화하는 것은, 상기 데이터가 이용가능하기 전에 시작되는, 프로세싱 회로를 포함하는 프로세서.

청구항 51

블록 암호 복호화를 위한 명령들을 포함하는 컴퓨터-판독가능 매체로서,

상기 명령들은, 하나 이상의 프로세서들에 의한 실행 시에, 상기 프로세서들로 하여금,

암호화된 메모리 어드레스를 획득하기 위해 메모리 어드레스를, 메모리 어드레스 암호화 라운드로부터의 출력이 다음의 메모리 어드레스 암호화 라운드에 대한 입력으로 사용되는, 제 1 복수의 순차적 반복적 블록 암호 라운드들에서 암호화하게 하고;

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 하나 이상의 중간 (intermediate) 라운드들로부터의 정보를 이용하여 데이터 라운드 키들을 생성하게 하고;

부분적으로 복호화된 데이터를 획득하기 위해 상기 데이터 라운드 키들을 이용하여 암호화된 데이터를, 데이터 복호화 라운드로부터의 출력이 다음의 데이터 복호화 라운드에 대한 입력으로 사용되는, 제 2 복수의 순차적 반복적 블록 암호 라운드들에서 복호화하게 하며;

완전히 복호화된 데이터를 획득하기 위해 상기 부분적으로 복호화된 데이터를 상기 암호화된 메모리 어드레스와

결합하게 하고,

상기 메모리 어드레스를 암호화하는 것은, 상기 데이터가 이용가능하기 전에 시작되는, 컴퓨터-판독가능 매체.

청구항 52

제 1 항에 있어서,

상기 데이터 라운드 키들의 생성은 상기 제 1 복수의 순차적 반복적 블록 암호 라운드들의 완료 이전에 개시되는, 데이터 암호화 방법.

청구항 53

제 1 항에 있어서,

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들의 최근의 (later) 중간 라운드들로부터의 정보를 이용하여 초기의 (early) 데이터 라운드 키들이 계산되는, 데이터 암호화 방법.

청구항 54

제 1 항에 있어서,

상기 데이터 라운드 키들의 생성은 상기 메모리 어드레스의 암호화와 오버랩하는, 데이터 암호화 방법.

청구항 55

제 1 항에 있어서,

상기 제 1 복수의 순차적 반복적 블록 암호 라운드들 중 하나 이상의 중간 라운드들로부터의 정보는 2 개 이상의 블록 암호 라운드들 사이의 중간 결과들인, 데이터 암호화 방법.

명 세 서

기 술 분 야

[0001] 일 특징은 메모리 컨텐트의 보호에 관한 것으로, 특히, 블록 암호들을 이용하여 메모리 디바이스들에 저장된 컨텐트를 보안하는 방법에 관한 것이다.

배 경 기 술

[0002] 디지털 프로세서들은 셀룰러 폰, 컴퓨터, 개인 휴대 정보 단말기 (PDA), 무선 네트워크 액세스 포인트 등과 같은 다수의 디바이스들 내에 존재한다. 점차, 메모리에 저장된 프로그램 및 데이터가 상당히 복잡한 툴들을 가진 공격자들에 대하여 안전하게 될 필요가 있다. 디지털 저작권 관리 애플리케이션들이 또한 디지털 데이터 또는 하드웨어에 대한 액세스 또는 디지털 데이터 또는 하드웨어의 사용을 제어하기 위한 제한들을 부과한다. 예를 들면, 데이터를 액세스하기 위한 시도가 디바이스를 파괴, 추측컨대 데이터가 액세스될 수 있기 전에 데이터를 스크램블 또는 파괴하도록, 데이터 액세스 라인들이 CPU (central processing unit) 또는 기판 (board) 내에 매립되는 것을 보장함으로써 보안될 수도 있는, 일부 온-칩 메모리 저장장치를 CPU 가 갖는 경우가 대개 있다.

발명의 내용

해결하려는 과제

[0003] 규모 및 경제적 이유로, 메모리를 별개의 칩에 패키징할 수 있는 것이 바람직하다. 그러나, 메모리 디바이스의 별개의 칩에의 패키징은, 데이터가 CPU 와 메모리 칩들 사이에서 이동할 때 노출되기 때문에 공격자들이 단순한 툴들, 이를 테면 프로브 (probe) 들의 이용에 의해 액세스하기가 비교적 쉬워진다.

[0004] 별개의 메모리 칩에 데이터를 저장할 때의 보안의 결여를 다루기 위한 일 방법은 메모리 칩에 기록된 데이터가 공격자에게 무용지물이 되도록 CPU 상에 암호화 프로세스를 갖는 것이다. 반대로, 데이터가 메모리로부터

인출되는 경우, 그 데이터는 CPU 에 의해 복호화된다. 메모리의 특정 블록에 대한 어드레스 정보, 및 CPU 에만 알려져 있는 암호 키 (cryptographic key) 는 암호화 알고리즘에 대한 다른 입력들이다.

[0005] 메모리 위치들이 종종 패터닝된 데이터로 반복적으로 기록될 수 있기 때문에, 스트림 암호들 및 카운터 모드 (CTR) 와 같은 블록 암호들에 대한 대응하는 동작 모드들은 적절하지 않다. 암호 블록 체이닝 (Cipher Block Chaining; CBC) 과 같은 모드에 대한 초기화 벡터로서 어드레스를 이용한 블록 암호들은 여기서 적절한 메커니즘이다 (FIPS special publication 800-38A - 블록 암호들에 대한 동작 모드들을 참조). 그러나, 대개 일 동작에서 암호화될 메모리의 블록들은 암호의 고유한 블록 크기와 비교하면 작다 (예를 들어, 대개는 단지 단일 블록). 따라서, CBC 모드의 "체이닝" 으로의 생각은 단일 블록들에 적용될 때는 반직관적 (couter-intuitive) 이다.

[0006] 현대 블록 암호들은 반복된 블록 암호로 종종 지칭되는 구조를 갖는다. 각각의 반복은 *라운드*라 불리고, 반복된 함수는 *라운드 함수*라 불린다 (예를 들어, 대략 4 내지 32 사이가 통상적이다). 각각의 라운드에서, 라운드 함수는 입력 블록에 적용될 때 소정량의 혼돈 (confusion) 및 확산 (diffusion) 을 달성한다. 입력 블록을 암호화하기 위해, 암호는 입력 블록의 순열 (permutation) 을 생성한다. 복호화는 프로세스를 역으로 실행함으로써 달성된다. 블랙 박스로서 보면, 암호는 입력으로서 고정된 크기의 데이터의 단일 블록을 받아들이고, 비밀 키가 반복적으로 라운드 함수를 입력 블록에 적용하여, 암호 출력의 단일 블록을 출력한다.

일부 암호들은 가변 크기의 키들을 허용하며, 키 크기는 블록 크기보다 작거나, 같거나 또는 클 수도 있다.

예를 들어, 고급 암호화 표준 (Advanced Encryption Standard; AES) 알고리즘은 128-비트 블록 크기를 가지며, 128, 192 또는 256 비트의 키들을 받아들일 수 있다.

[0007] 암호 안에는, 다수의 라운드들 (예를 들어, 128-비트 키를 가진 AES 의 경우에는 10 개의 라운드들) 이 있다.

각각의 라운드는 그의 입력의 일부로서 라운드 키를 갖는다. 라운드 키들은 *키 스케줄링*이라 불리는 프로세스에서 비밀 키로부터 유도된다. 각각의 라운드는, 라운드 키와 블록의 부분들에 대해 일부 비선형 치환 (nonlinear substitution) 을 수행한 후, 각각의 치환 효과를 전체 블록에 퍼뜨리기 위해 일부 (종종 선형) 확산 연산을 수행하는 것으로 의도된다. 이들 액션들은 선형 및 미분 암호해독과 같은 널리 알려져 있는 형태의 암호해독을 좌절시키는 것으로 의도된다.

[0008] 메모리로 전송된 데이터를 암호화하기 위해, 메모리 어드레스가 초기화 벡터로서 이용될 수도 있다. 이는 동일 데이터를 가진 상이한 메모리 위치들이 그럼에도 불구하고 상이하게 암호화되는 것을 보증한다. 암호화는 :

[0009] $$C = E_K(P \oplus A)$$

[0010] 로서 기록될 수 있으며, 여기서 P 는 입력 *평문* (원래의 데이터 블록) 이고, A 는 메모리 어드레스이고, C 는 출력 *암호문* (어드레스 A 에서의 메모리 칩에 보이는 출력 데이터 블록) 이고, $\oplus$ 는 비트단위 배타적 OR (XOR) 연산이며, E_K 는 데이터의 블록을 비밀 키 K 로 암호화하기 위해 블록 암호를 이용하는 것을 의미한다.

대응하여, 데이터가 메모리에서 다시 판독되는 경우, 역연산 :

[0011] $$P = D_K(C) \oplus A$$

[0012] 이 이용되며, 여기서 D_K 는 그의 복호화 모드에서 블록 암호를 이용하는 것을 의미한다. 그러나, 통상의 블록 암호 애플리케이션들은 메모리 액세스 속도와 비교하여 상당히 하이 레이턴시를 갖는다. 파이프라이닝이 대량의 암호화의 경우 이런 문제를 다루지만, 단일 메모리 위치들을 암호화할 경우에는 도움이 되지 않는다.

[0013] 따라서, 레이턴시를 저감시키면서 적은 수의 메모리 위치들에 대한 블록 암호 암호화를 구현하기 위한 방법이 필요하다.

과제의 해결 수단

[0014] 데이터가 저장될 메모리 어드레스에 기초하여 데이터를 암호화함으로써 데이터를 보안하는 블록 암호가 제공된다. 메모리 어드레스에의 저장을 위한 데이터를 암호화할 때, 메모리 어드레스는 제 1 복수의 블록 암호 라운드들에서 암호화된다. 제 1 복수의 블록 암호 라운드들로부터의 정보를 이용하여 데이터 라운드 키들이 생성된다. 저장될 데이터가 암호화된 메모리 어드레스와 결합되고, 데이터 라운드 키들을 이용하여 제 2 복수의 블록 암호 라운드들에서 암호화된다. 암호화된 데이터는 그 후 메모리 위치에 저장된다. 데이터를

복호화할 때, 메모리 어드레스는 다시 앞서와 같이 암호화되는 한편, 암호화된 저장 데이터는 부분적으로 복호화된 데이터를 획득하기 위해 데이터 라운드 키들을 이용하여 제 2 복수의 블록 암호 라운드들에서 복호화된다.

완전히 복호화된 데이터를 획득하기 위해, 부분적으로 복호화된 데이터는 암호화된 메모리 어드레스와 결합된다.

[0015] 메모리 어드레스 내의 데이터의 암호화의 일 예에서, 메모리 어드레스는 제 1 복수의 블록 암호 라운드들에서 암호화된다. 메모리 어드레스를 암호화하는 것은 : (a) 제 1 변환 함수에 따라 메모리 어드레스를 변환하는 것, (b) 변환된 메모리 어드레스를 라운드 키와 혼합하는 것, (c) 메모리 어드레스를 세그먼트화하는 것, 및/또는 (d) 상이한 메모리 어드레스 세그먼트들에 대해 비트 치환을 수행하는 것을 포함할 수도 있다. 메모리 어드레스는 저장될 데이터보다 먼저 이용가능할 수도 있다. 그 결과, 메모리 어드레스를 암호화하는 것은 데이터가 이용가능하기 전에 시작될 수도 있다.

[0016] 제 1 복수의 블록 암호 라운드들 중 하나 이상으로부터의 정보를 이용하여 데이터 라운드 키들이 생성될 수도 있다. 데이터 라운드 키들을 생성하는 것은 : (a) 제 1 복수의 블록 암호 라운들 중 적어도 일부에 대한 암호화된 메모리 어드레스로부터 복수의 비트들을 추출하는 것, (b) 추출된 복수의 비트들의 세그먼트들로부터 데이터 라운드 키들을 선택하는 것, 및/또는 (c) 추출된 복수의 비트들을 데이터 라운드 키들이 선택되는 스트링에 연쇄 (concatenate) 시키는 것을 포함할 수도 있다.

[0017] 데이터는 제 1 복수의 블록 암호 라운드들 후에 암호화된 메모리 어드레스와 결합될 수도 있다. 예를 들어, 데이터는 가역 연산 (invertible operation) (예를 들어, 모듈러 덧셈/뺄셈, 비트단위 XOR 연산 등) 에 의해 암호화된 메모리 어드레스와 결합될 수도 있다. 데이터는 그 후 데이터 라운드 키들을 이용하여 제 2 복수의 블록 암호 라운드들에서 암호화될 수도 있다. 제 2 복수의 블록 암호 라운드들은 제 1 복수의 블록 암호 라운드들보다 크다. 데이터를 암호화하는 것은 : (a) 제 2 변환 함수에 따라 데이터를 변환하는 것, (b) 변환된 데이터를 데이터 라운드 키들 중 하나 이상과 혼합하는 것, (c) 데이터를 복수의 데이터 세그먼트들로 세그먼트화하는 것, 및/또는 (d) 상이한 데이터 세그먼트들에 대해 비트 치환을 수행하는 것을 포함할 수도 있다.

[0018] 메모리 어드레스는 제 1 복수의 블록 암호 라운드들에 걸쳐 반복적으로 암호화될 수도 있으며, 데이터는 제 2 복수의 블록 암호 라운드들에 걸쳐 반복적으로 암호화될 수도 있다. 일 예에서, 제 2 복수의 블록 암호 라운드들 중 초기의 (earlier) 라운드들에 대해 이용되는 데이터 라운드 키들은 제 1 복수의 블록 암호 라운드들 중 최근의 (later) 라운드들로부터의 암호화된 메모리 어드레스로부터의 비트들을 이용하여 생성될 수도 있다.

암호화된 데이터는 후속하여 메모리 어드레스에 저장될 수도 있다.

[0019] 메모리 어드레스 내의 데이터를 복호화하는 다른 예에서, 메모리 어드레스는 암호화된 메모리 어드레스를 획득하기 위해 제 1 복수의 블록 암호 라운드들에서 암호화된다. 메모리 어드레스를 암호화하는 것은 데이터가 이용가능하기 전에 시작될 수도 있다.

[0020] 메모리 어드레스를 암호화하는 것은 : (a) 제 1 변환 함수에 따라 메모리 어드레스를 변환하는 것, (b) 변환된 메모리 어드레스를 라운드 키와 혼합하는 것, (c) 메모리 어드레스를 세그먼트화하는 것, 및/또는 (d) 상이한 메모리 어드레스 세그먼트들에 대해 비트 치환을 수행하는 것을 포함할 수도 있다.

[0021] 데이터 라운드 키들이 제 1 복수의 블록 암호 라운드들 중 하나 이상으로부터의 정보를 이용하여 생성될 수도 있다. 데이터 라운드 키들을 생성하는 것은 : (a) 제 1 복수의 블록 암호 라운드들 중 적어도 일부에 대한 암호화된 메모리 어드레스로부터 복수의 비트들을 추출하는 것, (b) 추출된 복수의 비트들의 세그먼트들로부터 데이터 라운드 키들을 선택하는 것, 및/또는 추출된 복수의 비트들을 데이터 라운드 키들이 선택되는 스트링에 연쇄시키는 것을 포함할 수도 있다.

[0022] 암호화된 데이터는 메모리 어드레스로부터 검색될 수도 있다. 암호화된 데이터는 부분적으로 복호화된 데이터를 획득하기 위해 데이터 라운드 키들을 이용하여 제 2 복수의 블록 암호 라운드들에서 복호화될 수도 있다.

암호화된 데이터를 복호화하는 것은 : (a) 제 2 역변환 함수에 따라 암호화된 데이터를 변환하는 것, (b) 변환된 암호화된 데이터를 데이터 라운드 키들 중 하나 이상과 혼합하는 것, (c) 암호화된 데이터를 복수의 암호화된 데이터 세그먼트들로 세그먼트화하는 것, 및/또는 (d) 상이한 암호화된 데이터 세그먼트들에 대해 비트 치환을 수행하는 것을 포함할 수도 있다.

[0023] 부분적으로 복호화된 데이터는 완전히 복호화된 데이터를 획득하기 위해 암호화된 메모리 어드레스와 결합될 수도 있다. 일 예에서, 부분적으로 복호화된 데이터는 가역 연산 (예를 들어, 모듈러 덧셈/뺄셈, 비트단위 XOR 연산 등) 에 의해 암호화된 메모리 어드레스와 결합될 수도 있다. 제 2 복수의 블록 암호 라운드들 중

초기의 라운드들에 대해 이용되는 데이터 라운드 키들은 제 1 복수의 블록 암호 라운드들 중 초기의 라운드들로부터의 암호화된 메모리 어드레스로부터의 비트들을 이용하여 생성된다. 제 2 복수의 블록 암호 라운드들은 제 1 복수의 블록 암호 라운드들보다 크다. 제 1 복수의 블록 암호 라운드들은 제 2 복수의 블록 암호 라운드들과 동시에 실행될 수도 있다.

[0024] 이들 방법들은 하드웨어, 소프트웨어 및/또는 이들의 조합에서 구현될 수도 있다.

도면의 간단한 설명

[0025] 본 양태들의 특징, 본질 및 이점은 동일한 참조 부호들이 대응하여 식별하는 도면들과 함께 얻어진 이하 기술되는 상세한 설명으로부터 보다 명백해질 수도 있다.

도 1 은 평문 입력 블록이 이용가능하기 전에 블록 암호를 부분적으로 프로세싱하여 블록 암호의 레이턴시를 저감시킴으로써 블록 암호 암호화 프로세싱을 개선하는 제 1 특징을 예시한 블록도이다.

도 2 는 블록 암호의 제 1 부분을 블록 암호의 제 2 부분과 병렬로 프로세싱하여 블록 암호의 레이턴시를 저감시킴으로써 블록 암호 복호화를 개선하는 제 2 특징을 예시한 블록도이다.

도 3 은 메모리에 저장될 데이터를 암호화하도록 구성된 블록 암호의 일 예를 예시한 블록도이다.

도 4 는 평문 데이터를 암호화하도록 구성된 블록 암호 디바이스의 기능적 컴포넌트들을 예시한 블록도이다.

도 5 는 메모리 어드레스 암호화 또는 변환 모듈의 일 예를 예시한 블록도이다.

도 6 은 평문 데이터 암호화 또는 변환 모듈의 일 예를 예시한 블록도이다.

도 7 은 데이터가 저장될 메모리 어드레스를 이용하여 데이터를 암호화하는 블록 암호 데이터 암호화 방법을 예시한 도면이다.

도 8 은 메모리 어드레스로부터 판독된 데이터를 복호화하기 위한 블록 암호의 프로세싱을 예시한 블록도이다.

도 9 는 암호문 데이터를 복호화하도록 구성된 블록 암호 디바이스의 기능적 컴포넌트들을 예시한 블록도이다.

도 10 은 암호문 데이터 복호화 또는 역변환 모듈의 일 예를 예시한 블록도이다.

도 11 은 블록 암호의 레이턴시를 저감시키기 위해 암호화된 데이터를 복호화하면서 메모리 어드레스를 동시에 암호화하는 블록 암호를 이용함으로써 암호화된 데이터를 복호화하는 방법을 예시한 도면이다.

도 12 는 블록 암호의 어드레스 암호화 라운드들로부터의 결과들에 기초하여 데이터 암호화 및 복호화 라운드들에 대한 라운드 키들이 생성될 수도 있는 방법의 일 예를 예시한 블록도이다.

도 13 은 효율적인 블록 암호 암호화 및 복호화를 수행하도록 구성될 수도 있는 디바이스를 예시한 블록도이다.

발명을 실시하기 위한 구체적인 내용

[0026] 다음의 설명에서, 본 실시형태들의 완전한 이해를 제공하기 위해 특정 상세가 주어진다. 그러나, 이들 특정 상세 없이 본 실시형태들이 실시될 수도 있다는 것이 당업자에 의해 이해될 것이다. 예를 들어, 본 실시형태들을 불필요하게 상세화하여 불명료하게 하지 않기 위하여 회로가 블록도로 도시될 수도 있고, 또는 전혀 도시되지 않을 수도 있다. 다른 경우에, 본 실시형태들을 불명료하게 하지 않기 위하여 널리 알려져 있는 회로, 구조 및 기술은 상세하게 도시되지 않을 수도 있다.

[0027] **개관**

[0028] 여러 신규한 특징들은 메모리로/로부터 보안된 데이터를 기록 및 판독하기 위한 블록 암호의 이용에 의해 야기되는 레이턴시를 다룬다. 일반적으로, 기록 동작들보다 판독 동작들이 더욱 많이 있다. 종래 기술에서는, 데이터가 내부 버스 상에서 이용가능하거나 (기록) 또는 메모리로부터 인출된 (판독) 후에 암호화 및 복호화 연산들이 시작된다. 그러나, 통상의 하드웨어 설계에서, 어드레스 정보는 특히 판독 메모리의 경우에는 데이터 전에도 이용가능하다.

[0029] 도 1 은 평문 입력 블록이 이용가능하기 전에 블록 암호를 부분적으로 프로세싱하여, 블록 암호의 레이턴시를 저감시킴으로써 블록 암호 암호화 프로세싱을 개선하는 제 1 특징을 예시한 블록도이다. 이 암호화 프로세

스는 메모리 기록 동작의 일부로서 수행될 수도 있고, 메모리 어드레스 (A) (106) 가 암호화되는 어드레스 페이즈 (102) 및 데이터가 암호화되는 데이터 페이즈 (104) 를 포함한다. 반복된 블록 암호의 다수의 암호 라운드들은 블록 암호 (100) 에 대한 비밀 키 (108) 및 메모리 어드레스 (A) (106) 를 이용하여 미리 프로세싱된다.

평문 데이터 (112) 는 블록 암호 암호화의 어느 정도의 라운드들 후에 삽입된다 (110) (예를 들어, 어드레스 페이즈 (102) 의 결과와 비트단위 XOR 된다). 특히, 블록 암호 (100) 의 일부 라운드들 (즉, 어드레스 페이즈 라운드들 (102)) 은 평문 데이터 (112) 가 도입되기 전에 수행되고, 일부 라운드들 (즉, 데이터 페이즈 라운드들 (104)) 은 평문 데이터 (112) 가 도입된 후에 수행되어 암호문 (114) 을 생성한다. 어드레스 페이즈 (102) 는 메모리 어드레스 (A) (106) 를 암호화할 수도 있을 뿐만 아니라, 그 라운드들의 결과들을 이용하여 후속 데이터 페이즈 라운드들 (104) 에 대한 암호화 키들을 생성할 수도 있다 (즉, 키 스케줄링). 평문 데이터 (112) 가 도입되기 전과 후의 암호 라운드들의 수는 동일할 수도 있고 또는 상이할 수도 있다. 이 암호화 프로세스는 메모리에 저장될 데이터를 암호화할 때 레이턴시를 저감시키기 위해 평문 데이터 (112) 전에 메모리 어드레스 (A) (106) 가 이용가능하다는 사실을 이용한다.

[0030] 부가적으로, 보다 효율적인 키 스케줄링이 블록 암호에 대해 수행될 수도 있다. 블록 암호의 각각의 라운드에 대한 라운드 키들은 실제 평문 데이터가 이용가능하기 전에, 어드레스 데이터 및 비밀 키에 기초하여 생성될 수도 있다. 라운드 키들은 메모리 어드레스에 기초하여 생성되기 때문에, 이는 블록 암호 변환이 각각의 메모리 어드레스 마다 상이하여, 암호해독에 이용가능한 리소스들을 심하게 제약하고, 블록 암호의 보안성을 증가시킨다는 것을 의미한다.

[0031] 도 2 는 블록 암호의 제 1 부분을 블록 암호의 제 2 부분과 병렬로 프로세싱하여 블록 암호의 레이턴시를 저감시킴으로써 블록 암호 복호화를 개선하는 제 2 특징을 예시한 블록도이다. 이 복호화 프로세스는 메모리 판독 동작의 일부로서 수행될 수도 있으며, 메모리 어드레스 (A) (206) 가 암호화되는 어드레스 페이즈 (202) 및 데이터가 복호화되는 데이터 페이즈 (204) 를 포함한다. 전체 블록 암호를 종래의 복호화 방법에서 행해진 것과 같이 역방향으로 실행하는 대신에, 블록 암호 (200) 의 데이터 페이즈 (204) 는 암호문 (214) 에서 시작하여 역방향으로 프로세싱된다. 동시에, 블록 암호 (200) 의 어드레스 페이즈 (202) 가 입력으로서의 메모리 어드레스 (A) (206) 및 비밀 키 (208) 를 이용하여 순방향으로 실행된다. 이들 프로세스들이 중간 (210) 에서 조우할 때, 부분적으로 복호화된 암호문과 부분적으로 암호화된 메모리 어드레스를 XOR 함으로써 평문 데이터 (212) 가 유도된다.

[0032] **블록 암호를 이용한 효율적인 암호화**

[0033] 도 3 은 메모리에 저장될 데이터를 암호화하도록 구성된 블록 암호의 일 예를 예시한 블록도이다. 이 프로세스에서, 메모리 어드레스 (A) (304) 는 저장될 데이터 블록 (즉, 평문 (320)) 이 이용가능하기 전에 이용가능할 수도 있다. 반복된 블록 암호 (302) 는 데이터 (평문 (320)) 가 메모리에 저장될 때 그 데이터를 암호화하는데 이용될 수도 있다. 예를 들어, CPU 는 데이터를 저장을 위한 메모리 디바이스로 전송하기 전에 암호화할 수도 있다. 이 예에서, 블록 암호 (302) 는 키 스케줄링 및 어드레스 암호화 페이즈 (303) 및 데이터 암호화 페이즈 (305) 를 포함할 수도 있다.

[0034] 키 스케줄링 및 어드레스 암호화 페이즈 (303) 에서, 반복된 블록 암호 (302) 의 다수의 라운드들은 블록 암호에 대한 비밀 키 (K_{secret}) (306) 및 메모리 어드레스 (A) (304) 를 이용하여 미리 프로세싱된다. 예를 들어, 블록 암호 (302) 의 대응하는 어드레스 암호화 라운드들 (316, 317 및 318) 에 대한 복수의 라운드 키들 (K1 (307), K2 (308) 및 Ki (309)) 은 실제 평문 데이터 블록 (P) (320) 이 이용가능하기 전에 비밀 키 (K_{secret}) (306) 에 기초하여 생성된다. 각각의 라운드 키 (K1 (307), K2 (308) 및 Ki (309)) 는 이전의 라운드 키에 기초하여 유도될 수도 있다 (예를 들어, K1 은 K_{secret} 에 기초하고, K2 는 K1 에 기초하며, 등등이다). 일 예에 따르면, 비밀 키 (306) 는 w-비트 길이일 수도 있고, 각각의 라운드 키 (K1, K2 및 Ki) 는 n-비트 길이이며, 여기서 $n < w$ 이다. 각각의 라운드 키 (K1, K2 및 Ki) 는 비밀 키로부터 연속적인 n 개의 비트들을 얻음으로써 생성되며, 여기서 비밀 키 (306) 는 마지막엔 랩어라운드하는 것으로 간주된다. 각각의 라운드 키 (K1, K2 및 Ki) 는 비밀 키 (306) 의 상이한 그룹의 연속적인 비트 시퀀스들을 이용할 수도 있다.

[0035] 블록 암호 (302) 의 복수의 어드레스 암호화 라운드들 (316, 317 및 318) 은 메모리 어드레스 (304) 및 대응하는 라운드 키들 (K1 (307), K2 (308) 및 Ki (309)) 에 기초하여 생성될 수도 있다. 예를 들어, 라운드 1 (316) 은 제 1 선형 및/또는 비선형 함수 (E_a) 를 이용하여 메모리 어드레스 (A) (304) 의 전부 또는 일부를 변

환하고, 키 (K1) 와의 가역 연산 (예를 들어, 모듈러 덧셈/뺄셈, 비트단위 XOR 연산 등) 에 기초하여 더욱 변환된다 (예를 들어, $R1 = E_a(A) \oplus K1$). 유사하게, 라운드 2 (317) 는 제 1 선형 및/또는 비선형 함수 (E_a) 를 이용하여 라운드 1 (316) 의 결과 (R1) 를 변환하고, 그 결과를 대응하는 키 (K2) 와의 가역 연산 (예를 들어, 비트단위 XOR 연산) 에 기초하여 더욱 변환한다 (예를 들어, $R2 = E_a(R1) \oplus K2$). 이 프로세스는 각각의 변환 연산의 효과를 전체 블록에 확산시키기 위해 다수 회 반복될 수도 있다. 예를 들어, 라운드 i (318) 는 제 1 선형 및/또는 비선형 함수 (E_a) 를 이용하여 이전의 라운드의 결과 (Ri-1) 를 변환하고, 그 결과를 대응하는 키 (Ki) 와의 가역 연산 (예를 들어, 비트단위 XOR 연산 등) 에 기초하여 더욱 변환한다 (예를 들어, $Ri = E_a(Ri\text{-}1) \oplus Ki$). 제 1 블록 암호 라운드들 (303) (메모리 어드레스 암호화 페이즈) 은 데이터가 데이터 암호화 페이즈 (305) 에서의 암호화에 이용가능하기 전이라도 (적어도 부분적으로) 수행될 수도 있다. 평문 데이터 블록 (P) (320) 이 이용가능하기 전에 블록 암호를 부분적으로 프로세싱 (또는 미리 프로세싱) 함으로써, 블록 암호에서의 레이턴시 (즉, 지연) 가 저감될 수도 있다.

[0036] 부가적으로, 키 스케줄링 페이즈 (303) 동안, 데이터 암호화 페이즈 (305) 에 대한 키들 (312, 314 및 315) 이 시간을 절약하기 위해 생성될 수도 있다. 데이터 암호화 페이즈 (305) 키들 (Ky, Ky+1 및 Kx) 은 어드레스 암호화 페이즈 (303) 의 각각의 암호 라운드 결과 (R1, R2, Ri) 의 결과에 기초하여 생성될 수도 있다. 일 예에서, 라운드 결과들 (R1, R2 및 Ri) 은 n 비트 길이 (여기서 n 은 포지티브 정수이다) 일 수도 있고, 적어도 복수의 이들 라운드들로부터의 비트들의 수 g 는 데이터 암호화 페이즈 키들 (Ky, Ky+1 및 Kx) 을 생성하는데 이용되며, 여기서 g 는 n 보다 작은 정수이다. 예를 들어, 비트들의 세트 S (310) 는, $S_{1\ldots p} = R1_{1\ldots g1} \,||\, R2_{1\ldots g2} \ldots || \, Ri_{1\ldots gi}$ 가 되도록 다양한 라운드 결과들 (R1, R2, Ri) 로부터의 추출된 비트들을 연쇄 (심볼 $||$) 시킴으로써 획득될 수도 있으며, 여기서 p 는 비트들의 세트 S (310) 내의 총 비트들의 수를 나타내는 정수 값이다. 일부 구현에서, 각각의 라운드에 대한 비트들의 수 (g1, g2, ..., gi) 는 동일할 수도 있는 한편, 다른 구현에서는, 비트들의 수 (g1, g2, ..., gi) 가 상이할 수도 있다. 키 스케줄링 페이즈 (303) 동안, 데이터 암호화 페이즈 키들 (Ky, Ky+1 및 Kx) 은 각각의 키에 대한 비트 세트 S (310) 로부터 비트들의 세그먼트를 추출함으로써 생성될 수도 있다. 일 예에서, 키 스케줄링 및 어드레스 암호화 페이즈 (303) 의 최근의 암호 라운드들에 대응하는 비트들은 데이터 암호화 페이즈 (305) 에서의 초기의 키들에 대해 이용될 수도 있다. 예를 들어, 키 (Ky) (312) 는 이 예에서는 $Ri_{1\ldots g}$ 로부터의 비트들의 서브세트에 대응하는, 비트 세트 S (310) 의 비트들 $S_{(p-j+1),,,p}$ 로부터 얻어질 수도 있으며, 여기서 $j < g$ (g = g1, g2,..., gi) 이다. 유사하게, 키 (Ky+1) (314) 는 비트 세트 S (310) 의 비트들 $S_{(p-2j+1)\ldots(p-j)}$ 과 같을 수도 있으며, 키 (Kx) 는 비트 세트 S (310) 의 비트들 $S_{1\ldots j}$ 과 같을 수도 있다. 일부 구현에서, $j < g$ 인 경우, 키 스케줄링 페이즈 (303) 에서의 라운드들의 수는 데이터 페이즈 (305) 에서의 라운드들의 수보다 적을 수도 있다.

예를 들어, 라운드 결과들 (R1, R2 및 Ri) 이 63 비트 길이 (즉, n = 63) 인 경우, 각각의 라운드로부터의 45 개의 비트들 (즉, g = 45) 이 비트들의 세트 S (310) 를 위해 이용되도록 추출될 수도 있으며, 각각의 데이터 페이즈 키 (Ky (312), Ky+1 (314) 및 Kx (315)) 는 32 비트 길이 (즉, j = 32) 일 수도 있다.

[0037] 일반적인 의미로, 하나 이상의 라운드 키 함수들 (KSx) 은 라운드 키들 (K1, K2, Ki, Ky, Ky+1..., Kx) 각각을 생성하는데 이용될 수도 있다. 일 예에서, 제 1 키 스케줄링 함수 (KS1) 는 (어드레스 암호화 페이즈에 대한) 키들 (K1, K2, Ki) 을 생성하는데 이용될 수도 있고, 제 2 키 스케줄링 함수 (KS2) 는 (데이터 암호화 페이즈에 대한) 키들 (Ky, Ky+1, Kx) 을 생성하는데 이용될 수도 있다. 예를 들어, 제 1 키 스케줄링 함수 (KS1) 는 $Ki = KS1(K_{secret}, i)$ 가 되도록 키 (Ki) 를 생성하는데 이용될 수도 있는 한편 (여기서, "i" 는 어드레스 암호화 페이즈 (303) 에 대한 라운드 수이다), 제 2 키 스케줄링 함수 (KS2) 는 $Ky+i = KS2(S_{1\ldots p}, i)$ 가 되도록 키 (Ky+i) 를 생성하는데 이용될 수도 있다 (여기서 "y+i" 는 데이터 암호화 페이즈 (305) 에 대한 라운드 수이다).

[0038] 평문 데이터 블록 (P) (320) 이 이용가능해질 때, 그 평문 데이터 블록 (P) (320) 은 블록 암호 (302) 의 하나 이상의 라운드들 (316, 317 및 318) 이 수행된 후 (예를 들어, 키 스케줄링 페이즈 (303) 후) 에 블록 암호 (302) 에 삽입될 수도 있다. 평문 데이터 블록 (P) (320) 은 그 평문 데이터 블록 (P) (320) 을 종종 *화이트닝* (whitening) 이라 불리는 프로세스에서 마지막으로 미리 프로세싱된 라운드 (예를 들어, 라운드 i (318))

의 결과 (Ri) 와 (비트단위로) XOR 함으로써 블록 암호 (302) 에 삽입될 수도 있다. 평문 데이터 블록 (P) (320) 이 도입된 후에, 데이터 암호화 페이즈 (305) 의 하나 이상의 라운드들 (322, 324 및 326) 이 대응하는 라운드 키들 (Ky (312), Ky+1 (314) 및 Kx (315)) 을 이용하여 수행된다.

[0039] 라운드 y (322) 에서의 데이터 암호화 페이즈 (305) 동안, 화이트닝된 데이터 블록 ($D_{whitened}$) (321) 은 제 2 선형 및/또는 비선형 함수 (E_b) 에 의해 변환되고, 대응하는 라운드 키 (Ky) 와의 가역 연산 (예를 들어, 비트단위 XOR 연산) 에 기초하여 더욱 변환된다 (예를 들어, $Ry = E_b(D_{Whitened}) \oplus Ky$). 유사하게, 라운드 y+1 (324) 변환에서는, 제 2 선형 및/또는 비선형 함수 (E_b) 를 이용하여 라운드 y (322) 의 결과 (Ry) 를 변환하고, 그 결과를 대응하는 키 (Ky+1) 와의 가역 연산 (예를 들어, 모듈러 덧셈/뺄셈, 비트단위 XOR 연산 등) 에 기초하여 더욱 변환한다 (예를 들어, $Ry+1 = E_b(Ry) \oplus Ky+1$). 이 프로세스는 각각의 변환 연산의 효과를 전체 블록에 확산시키기 위해 다수 회 반복될 수도 있다. 예를 들어, 라운드 x (326) 는 제 2 선형 및/또는 비선형 함수 (E_b) 를 이용하여 이전의 라운드의 결과 (Rx-1) 를 변환하고, 그 결과를 대응하는 키 (Kx) 와의 가역 연산 (예를 들어, 비트단위 XOR 연산 등) 에 기초하여 더욱 변환하여 암호문 (328) 을 획득한다.

[0040] 다양한 구현에서, 키 스케줄링 및 어드레스 암호화 페이즈 (303) 및 데이터 암호화 페이즈 (305) 의 라운드들의 수는 동일할 수도 있고 또는 상이할 수도 있다. 데이터 암호화 페이즈 (305) 동안의 라운드들의 수는 블록 암호 (302) 의 레이턴시를 저감시키기 위해 도입되는 평문 데이터 블록 (P) (320) 에 충분한 확산을 제공하면서 블록 암호 (302) 의 레이턴시를 저감시키도록 선택될 수도 있다.

[0041] 도 4 는 평문 데이터를 암호화하도록 구성된 블록 암호 디바이스의 기능적 컴포넌트들을 예시한 블록도이다. 블록 암호 디바이스 (402) 는 저장될 메모리 어드레스 (406) 및 비밀 키 (408) 에 기초하여 평문 데이터 (404) 를 암호화할 수도 있다. 블록 암호 디바이스 (402) 는 변환 또는 암호화 함수 및 라운드 키 생성기 (416) 에 의해 제공되는 키에 따라 메모리 어드레스 (406) 를 변환 및/또는 암호화하는 어드레스 암호화 모듈 (412) 을 포함할 수도 있다. 라운드 키 생성기 (416) 는 비밀 키 (408) 에 기초하여 하나 이상의 라운드 키들을 생성하도록 구성될 수도 있다. 키 스케줄링 모듈 (414) 이 또한 어드레스 암호화 모듈 (412) 의 결과들에 기초하여 하나 이상의 데이터 키들을 생성할 수도 있다. 하나 이상의 데이터 키들은 데이터 키 저장장치 (422) 모듈에 저장될 수도 있다. 어드레스 암호화 및 데이터 스케줄링 함수들은 각각의 라운드에서의 라운드 키 생성기 (416) 로부터의 상이한 라운드 키를 이용하여 다수의 라운드들에서 반복적으로 수행될 수도 있다. 복수의 라운드들 후에, 결합기 (418) 는 평문 데이터 (404) 를 가역 연산 (예를 들어, 모듈러 덧셈/뺄셈, 비트단위 XOR 등) 을 이용하여 어드레스 암호화 모듈 (412) 의 마지막 결과들과 결합할 수도 있다. 결과의 화이트닝된 평문 데이트는 그 후 암호문 (424) 을 생성하기 위해 데이터 키 저장장치 (422) 로부터의 저장된 데이터 키들 및 변환 또는 암호화 함수를 이용하여 하나 이상의 라운드들에서 데이터 암호화 모듈 (420) 에 의해 반복적으로 변환 또는 암호화될 수도 있다. 암호문 (424) 은 메모리 어드레스 (406) 에서 메모리 디바이스 (426) 에 저장될 수도 있다.

[0042] 일 예에 따르면, 블록 암호 디바이스는 시스템에서 바이트 어드레스가능한 메모리로 구현될 수도 있다. 예를 들어, 블록 암호를 구현하는 CPU 의 워드 크기는 32 비트일 수도 있으며, 메모리 어드레스도 또한 32 비트일 수도 있다. 전술한 바와 같이, 블록 암호 디바이스는 어드레스 암호화 페이즈 및 데이터 암호화 페이즈를 수행하도록 구성될 수도 있다.

[0043] 도 5 는 메모리 어드레스 암호화 또는 변환 모듈의 일 예를 예시한 블록도이다. 어드레스 암호화 페이즈 (502) 동안, 입력 메모리 어드레스 (64 비트에 패딩) 는 복수의 치환-순열 (substitution-permutation) 암호 라운드들을 실행함으로써 변환될 수도 있다. 선택적으로는, 입력 메모리 어드레스 (504) 는 먼저 라운드 키와 XOR 함으로써 화이트닝될 수도 있다. 어드레스 세그먼트화 모듈 (506) 이 64-비트 메모리 어드레스 (504) 를 8 개의 8-비트 세그먼트들로 분할할 수도 있다. 각각의 8-비트 세그먼트는 그 후 8×8 치환 박스 (508) (예를 들어, AES (Advance Encryption Standard) 8×8 치환 박스) 를 통과한다. 각각의 치환 박스 (508) 로부터의 결과들은 그 후 전체 세트의 세그먼트들에 대해 선형 변환을 수행하는 변환 모듈 (510) 로 전달된다. 선형 변환은 예를 들어, 행렬 곱셈 (matrix multiplication) $Y=CX$ 로 구현될 수도 있으며, 여기서 X 는 메모리 어드레스 벡터이고, C 는 변환 행렬이며, Y 는 출력 벡터이다. 일 예에서, 변환 행렬 C 는 브랜치 수 (9) 와 맵핑하는 MDS (Maximum Distance Separable) 로서 GF(2^8) 위의 8×8 행렬일 수도 있다. 행렬

C 는 모든 그 서브-행렬들이 정칙 (nonsingular) 인 경우 및 그런 경우에만 MDS 일 수도 있다. 블록 암호들 (예를 들어, SHARK 및 Khazad) 에서의 많은 확산 계층들이 이 요건을 충족할 수도 있다. 키 혼합 모듈 (512) 이 그 후 변환된 메모리 어드레스를 64-비트 라운드 키와 (예를 들어, 비트단위 XOR 을 이용하여) 혼합한다. 각각의 암호 라운드 마다, 데이터 라운드 키 추출 모듈 (514) 이 그 후 중간 암호화된 메모리 어드레스로부터 복수의 비트들을 추출하여 후속 데이터 암호화 프로세스에서 이용될 수도 있는 하나 이상의 데이터 라운드 키들 (518) 을 획득할 수도 있다. 복수의 이들 암호 라운드들 (예를 들어, 세그먼트화 (506), S-박스 계층 (508), 변환 (510) 및 키 혼합 (512)) 은 각각의 암호 라운드의 마지막에 수행되는 데이터 라운드 키 추출 (514) 과 함께 수행될 수도 있다.

[0044] 도 6 은 평문 데이터 암호화 또는 변환 모듈의 일 예를 예시한 블록도이다. 데이터 암호화 페이즈 (602) 동안, 평문 데이터 (604) 는 먼저 어드레스 암호화 페이즈로부터의 암호화된 메모리 어드레스 (603) 와 비트단위 XOR 모듈 (605) 에 의해 화이트닝될 수도 있다. 암호화된 메모리 어드레스 (603) 는 암호화된 평문 데이터가 저장될 메모리 어드레스에 대응할 수도 있다. 예를 들어, 평문 데이터 (604) 가 32-비트 블록 단위로 프로세싱되는 경우에는, 메모리 어드레스 페이즈로부터의 출력의 32 비트들과 XOR 될 수도 있다. 데이터 암호화 페이즈에 대한 라운드 키들은 메모리 어드레스 암호화 페이즈로부터 유도될 수도 있다. 데이터 세그먼트화 모듈 (606) 이 평문 데이터 (604) 를 4 개의 8-비트 세그먼트들로 분할 또는 스플리팅한다. 각각의 8-비트 데이터 세그먼트는 치환 박스 (608) (예를 들어, AES 8×8 치환 박스) 를 통과하게 된다. 치환 박스 (608) 로부터의 결과들은 그 후 선형 변환 모듈 (610) 에 의해 변환된다 (예를 들어, AES MDS 맵핑). 키 혼합 모듈 (612) 이 그 후 결과의 변환된 평문 데이터를 대응하는 라운드 키와 비트단위 XOR 할 수도 있다. 이 프로세스는 각각의 라운드 마다 상이한 라운드 키를 이용하여 다수 회 반복될 수도 있다. 데이터 암호화 페이즈 (602) 의 마지막 암호 라운드의 결과는 대응하는 어드레스 암호화 페이즈 동안 이용되는 메모리 어드레스에 저장될 수 있는 출력 암호문 (614) 이다.

[0045] 도 7 은 데이터를 암호화하기 위해 그 데이터가 저장될 메모리 어드레스를 이용하는 블록 암호 데이터 암호화 방법을 예시한다. 이 방법에서, 제 1 세트의 암호 라운드들은, 데이터가 실제로 저장할 준비가 되거나 저장에 이용가능하기 전에 메모리 어드레스를 암호화하고 데이터 라운드 키들을 생성하기 위해 실행된다. 그 후 제 2 세트의 암호 라운드들이 데이터를 암호화하기 위해 실행된다.

[0046] 프로세스는 데이터가 실제로 수신되기 전에 저장될 데이터에 대한 메모리 어드레스를 획득한다 (702). 메모리 어드레스는 제 1 복수의 블록 암호 라운드들에서 암호화될 수도 있다 (704). 이러한 메모리 어드레스 암호화는 : (a) 메모리 어드레스를 복수의 메모리 어드레스 세그먼트들로 세그먼트화하는 것, (b) 상이한 메모리 어드레스 세그먼트들에 대해 비트 치환을 수행하는 것, (c) 제 1 변환 함수에 따라 메모리 어드레스를 변환하는 것, 및/또는 (d) 변환된 메모리 어드레스를 라운드 키와 혼합하는 것을 포함할 수도 있다. 메모리 어드레스는 제 1 복수의 블록 암호 라운드들에 걸쳐 반복적으로 암호화될 수도 있다.

[0047] 데이터 라운드 키들이 제 1 복수의 블록 암호 라운드들 중 하나 이상으로부터의 정보를 이용하여 생성될 수도 있다 (706). 데이터 라운드 키들은 : (a) 제 1 복수의 블록 암호 라운드들 중 적어도 일부에 대한 암호화된 메모리 어드레스로부터 복수의 비트들을 추출하고, (b) 추출된 복수의 비트들의 세그먼트들로부터 데이터 라운드 키들을 선택하며/하거나 (c) 추출된 복수의 비트들을 데이터 라운드 키들이 선택되는 스트링에 연쇄시킴으로써 생성될 수도 있다.

[0048] 저장될 데이터는 그 후 제 1 복수의 블록 암호 라운드들 후에 암호화된 메모리 어드레스와 결합될 수도 있다 (708). 일 예에서, 데이터는 가역 연산 (예를 들어, 비트단위 XOR 연산) 에 의해 암호화된 메모리 어드레스와 결합될 수도 있다. 데이터는 그 후 데이터 라운드 키들을 이용하여 제 2 복수의 블록 암호 라운드들에서 암호화될 수도 있다 (710). 이러한 데이터 암호화는 : (a) 데이터를 복수의 데이트 세그먼트들로 세그먼트화하는 것, (b) 상이한 데이터 세그먼트들에 대해 비트 치환을 수행하는 것, (c) 제 2 변환 함수에 따라 데이터를 변환하는 것, 및/또는 (d) 변환된 데이터를 데이터 라운드 키들 중 하나 이상과 혼합하는 것을 포함할 수도 있다. 데이터는 제 2 복수의 블록 암호 라운드들에 걸쳐 반복적으로 암호화될 수도 있다. 일 예에서, 제 2 복수의 블록 암호 라운드들 중 초기의 라운드들에 대해 이용되는 데이터 라운드 키들은 제 1 복수의 블록 암호 라운드들 중 최근의 라운드들로부터의 암호화된 메모리 어드레스로부터의 비트들을 이용하여 생성된다. 제 2 복수의 블록 암호 라운드들은 제 1 복수의 블록 암호 라운드들보다 클 수도 있다. 암호화된 데이터는 그 후 메모리 어드레스에 저장될 수도 있다 (712).

[0049] **블록 암호를 이용한 효율적인 복호화**

[0050] 도 8 은 메모리 어드레스로부터 판독된 데이터를 복호화하기 위한 블록 암호의 프로세싱을 예시한 블록도이다. 데이터는 예를 들어 도 1 및 도 3 내지 도 7 에 예시된 방법을 이용하여 미리 암호화되었을 수도 있다. 복호화 모드에서, 전체 블록 암호를 종래의 복호화 방법에서 행한 것처럼 역방향으로 실행하는 대신에, 블록 암호 (802) 의 제 2 페이즈 (805) 가 암호문 (828) 에서 시작하여 역방향으로 프로세싱되는 한편, 제 1 페이즈 (803) 가 동시에 순방향으로 프로세싱된다. 일반적으로, 제 1 세트의 블록 암호 라운드들 (803) 은 제 2 세트의 블록 암호 라운드들 (805) 과 병렬로 프로세싱되어, 블록 암호 (802) 의 레이턴시를 저감시킨다. 즉, 제 1 블록 암호 라운드들 (803) (메모리 어드레스 암호화 페이즈) 은 암호화된 데이터가 제 2 블록 암호 라운드들 (805) 에 의한 프로세싱을 위해 이용가능하거나 검색되기 전이라도 (적어도 부분적으로) 수행될 수도 있다.

키 스케줄링 및 어드레스 암호화 페이즈 (803) 에서, (암호문 데이터 (828) 가 검색되고 있는) 메모리 어드레스 (804) 가 복수의 암호 라운드들에서 암호화된다. 한편, 데이터 복호화 페이즈 (805) 에서, 암호문 데이터 (828) 는 키 스케줄링 페이즈 (803) 에서 생성된 데이터 라운드 키들을 이용하여 복수의 암호 라운드들에서 복호화된다. 이러한 페이즈들 (803 및 805) 로부터의 결과들이 그 후 결합 (예를 들어, XOR) 되어 원래의 평문 (820) 이 생성된다.

[0051] 키 스케줄링 및 어드레스 암호화 페이즈 (803) 에서, 암호문 데이터 (828) 가 검색되고 있는 메모리 어드레스 (804) 가 암호화된다. 반복된 블록 암호 (802) 의 다수의 라운드들이 블록 암호 (802) 에 대한 비밀 키 (K_{secret}) (806) 및 메모리 어드레스 (A) (804) 를 이용하여 프로세싱된다. 예를 들어, 블록 암호 (802) 의 대응하는 어드레스 암호화 라운드들 (816, 817 및 818) 에 대한 복수의 라운드 키들 (K1 (807), K2 (808) 및 Ki (809)) 이 비밀 키 (K_{secret}) (806) 에 기초하여 생성된다. 각각의 라운드 키 (K1 (807), K2 (808) 및 Ki (809)) 는 이전의 라운드 키에 기초하여 유도될 수도 있다 (예를 들어, K1 은 K_{secret} 에 기초하고, K2 는 K1 에 기초하며, 등등이다). 일 예에 따르면, 비밀 키 (806) 는 w-비트 길이일 수도 있고, 각각의 라운드 키 (K1, K2 및 Ki) 는 n-비트 길이이며, 여기서 n < w 이다. 각각의 라운드 키 (K1, K2 및 Ki) 는 비밀 키 (806) 로부터 연속적인 n 개의 비트들을 얻음으로써 생성되며, 여기서 비밀 키 (806) 는 마지막엔 랩어라운드하는 것으로 간주된다. 각각의 라운드 키 (K1, K2 및 Ki) 는 비밀 키 (806) 의 상이한 그룹의 연속적인 비트 시퀀스들을 이용할 수도 있다.

[0052] 블록 암호 (802) 의 복수의 어드레스 암호화 라운드들 (816, 817 및 818) 은 메모리 어드레스 (804) 및 대응하는 라운드 키들 (K1 (807), K2 (808) 및 Ki (809)) 에 기초하여 생성된다. 예를 들어, 라운드 1 (816) 은 제 1 선형 및/또는 비선형 함수 (E_a) 를 이용하여 메모리 어드레스 A (804) 의 전부 또는 일부를 변환하고, 키 (K1) 와의 가역 연산 (예를 들어, 모듈러 덧셈/뺄셈, 비트단위 XOR 등) 에 기초하여 더욱 변환된다 (예를 들어, $R1 = E_a(A) \oplus K1$). 유사하게, 라운드 2 (817) 는 제 1 선형 및/또는 비선형 함수 (E_a) 를 이용하여 라운드 1 (816) 의 결과 (R1) 를 변환하고, 그 결과를 대응하는 키 (K2) 와의 비트단위 XOR 에 기초하여 더욱 변환한다 (예를 들어, $R2 = E_a(R1) \oplus K2$). 이 프로세스는 각각의 변환 연산의 효과를 전체 블록에 확산시키기 위해 다수 회 반복될 수도 있다. 예를 들어, 라운드 i (818) 는 제 1 선형 및/또는 비선형 함수 (E_a) 를 이용하여 이전의 라운드의 결과 (Ri-1) 를 변환하고, 그 결과를 대응하는 키 (Ki) 와의 비트단위 XOR 에 기초하여 더욱 변환한다 (예를 들어, $Ri = E_a(Ri\text{-}1) \oplus Ki$).

[0053] 부가적으로, 키 스케줄링 페이즈 (803) 동안, 데이터 복호화 페이즈 (805) 에 대한 키들 (812, 814 및 815) 이 시간을 절약하기 위해 생성될 수도 있다. 데이터 복호화 페이즈 (805) 키들 (Ky, Ky+1 및 Kx) 은 키 스케줄링 페이즈 키들 (K1, K2 및 Ki) 에 기초하여 생성될 수도 있다. 일 예에서, 암호 라운드 결과들 (R1, R2 및 Ri) 은 n 비트 길이 (여기서 n 은 포지티브 정수이다) 일 수도 있고, 이들 키들 각각으로부터의 비트들의 수 g 는 데이터 페이즈 키들 (Ky, Ky+1 및 Kx) 을 생성하는데 이용되며, 여기서 g 는 n 보다 작은 정수이다. 예를 들어, 비트들의 세트 S (810) 는 $S_{1\ldots p} = R1_{1\ldots g1} \| R2_{1\ldots g2} \ldots \| Ri_{1\ldots gi}$ 가 되도록 다양한 라운드 결과들 (R1, R2, Ri) 로부터의 추출된 비트들을 연쇄 (심볼 $\|$) 시킴으로써 획득될 수도 있으며, 여기서 p 는 비트들의 세트 S (810) 내의 비트들의 총 수를 나타내는 정수 값이다. 일부 구현에서, 각각의 라운드에 대한 비트들의 수 (g1, g2, ..., gi) 는 동일할 수도 있는 한편, 다른 구현에서는, 비트들의 수 (g1, g2, ..., gi)

가 상이할 수도 있다. 키 스케줄링 페이즈 (803) 동안, 데이터 암호화 페이즈 키들 (Ky, Ky+1 및 Kx) 은 각각의 키에 대한 비트 세트 S (810) 로부터 비트들의 세그먼트를 추출함으로써 생성될 수도 있다.

[0054] 일 예에서, 키 스케줄링 페이즈 (803) 의 초기 라운드들에 대응하는 비트들은 데이터 복호화 페이즈 (805) 에서 초기의 암호 라운드 키들에 대해 이용될 수도 있다. 이는 데이터 복호화 페이즈 (805) 의 어드레스 암호화 페이즈 (803) 와의 동시 또는 병렬 실행을 허용한다. 예를 들어, 키 (Kx) (815) 는 제 1 암호 라운드 ($R1_{1...gl}$) (816) 로부터 추출된 비트들의 일부에 대응하는 비트 세트 S (810) 의 비트들 ($S_{1...j}$) 과 동일할 수도 있다. 그 결과, R1 결과가 생성되자마자, 복호화 키 (Kx) (815) 가 획득될 수 있다. 유사하게, 키 (Ky+1) (814) 는 비트 세트 S (810) 의 비트들 ($S_{(p-2j+1)...(p-j)}$) 과 동일할 수도 있다. 마찬가지로, 키 (Ky) (812) 는 이 예에서는 $Ri_{1...gi}$ 로부터의 비트들의 서브세트에 대응하는 비트 세트 S (810) 의 비트들 ($S_{(p-j+1)...p}$) 로부터 얻어질 수도 있으며, 여기서 $j < g$ 이다. 일부 구현에서, $j < g$ 인 경우, 키 스케줄링 페이즈 (803) 에서의 암호 라운드들의 수는 데이터 복호화 페이즈 (805) 에서의 라운드들의 수보다 적을 수도 있다. 예를 들어, 라운드 결과들 (R1, R2 및 Ri) 이 63 비트 길이 (즉, $n = 63$) 인 경우, 각각의 라운드로부터의 45 개의 비트들 (즉, $g = 45$) 이 비트들의 세트 S (810) 에 이용되도록 추출될 수도 있으며, 각각의 데이터 복호화 페이즈 키 (Kx (815), Ky+1 (814) 및 Ky (812)) 는 32 비트 길이 (즉, $j = 32$) 일 수도 있다.

[0055] 일반적인 의미로, 하나 이상의 라운드 키 함수들 (KSx) 은 라운드 키들 (K1, K2, Ki, Ky, Ky+1, ... Kx) 각각을 생성하는데 이용될 수도 있다. 일 예에서, 제 1 키 스케줄링 함수 (KS1) 는 (어드레스 암호화 페이즈에 대한) 키들 (K1, K2, Ki) 을 생성하는데 이용될 수도 있고, 제 2 키 스케줄링 함수 (KS2) 는 (데이터 복호화 페이즈에 대한) 키들 (Ky, Ky+1, Kx) 을 생성하는데 이용될 수도 있다. 예를 들어, 제 1 키 스케줄링 함수 (KS1) 는 $Ki = KS1(K_{secret}, i)$ 가 되도록 키 (Ki) 를 생성하는데 이용될 수도 있는 한편 (여기서 "i" 는 어드레스 암호화 페이즈 (803) 에 대한 라운드 수이다), 제 2 키 스케줄링 함수 (KS2) 는 $Ky+i = KS2 (S_{1...p}, i)$ 가 되도록 키 (Ky+i) 를 생성하는데 이용될 수도 있다 (여기서 "y+i" 는 데이터 복호화 페이즈 (805) 에 대한 라운드 수이다).

[0056] 데이터 복호화 페이즈 동안, 암호문 데이터 (ct) (828) 는 다수의 라운드들에 걸쳐 키들 (Kx, Ky+1 및 Ky) 을 이용하여 복호화된다. 예를 들어, 라운드 x (826) 는 선형 및/또는 비선형 복호화 함수 (D_b) 를 이용하여 결과 암호문 (ct) (828) 을 변환하고, 그 결과를 대응하는 키 (Kx) 와의 연산 (가역 모듈러 덧셈/뺄셈, 비트단위 XOR 등) 에 기초하여 더욱 변환하여 (예를 들어, $Rx = D_b(ct) \oplus Kx$), 결과 (Rx) 를 획득한다. 이 복호화 프로세스는 저장된 데이터의 암호화를 원상태로 돌리기 위해 다수 회 반복될 수도 있다. 예를 들어, 라운드 (y+1) (824) 에서는, 선형 및/또는 비선형 복호화 함수 (D_b) 를 이용하여 이전의 라운드로부터 결과 (Ry+1) 를 변환하고, 그 결과를 대응하는 키 (Ky+1) 와의 비트단위 XOR 에 기초하여 더욱 변환하여 (예를 들어, $Ry = D_b(Ry+1) \oplus Ky+1$), 출력 (Ry) 을 획득한다. 라운드 (y) (822) 에서, 결과 (Ry) 는 선형 및/또는 비성현 복호화 함수 (D_b) 에 의해 변환되고, 대응하는 라운드 키 (Ky) 와의 비트단위 XOR 에 기초하여 더욱 변환되어 화이트닝된 데이터 블록 ($D_{Whitened}$) (821) 이 획득된다. 화이트닝된 데이터 블록 ($D_{Whitened}$) 은 그 후 가역 연산 (예를 들어, 모듈러 덧셈/뺄셈, 비트단위 XOR 등) 을 이용하여 어드레스 암호화 페이즈 (803) 로부터의 결과 (Ri) (예를 들어, 암호화된 어드레스) 와 결합되어 평문 데이터 블록 (P) (820) 이 획득된다.

[0057] 다양한 구현에서, 키 스케줄링 및 어드레스 암호화 페이즈 (803) 및 데이터 복호화 페이즈 (805) 의 라운드들의 수는 동일할 수도 있고 또는 상이할 수도 있다. 데이터 암호화 페이즈 (305) (도 3) 에서 이용되는 암호화 함수 (E_b) 에 의한 암호화를 원상태로 돌리기 위해 데이터 복호화 페이즈 (805) 에서 이용되는 복호화 함수 (D_b) 가 선택될 수도 있다. 예를 들어, 복호화 함수 (D_b) 는 암호화 함수 (E_b) 변환의 역변환일 수도 있다.

[0058] 도 9 는 암호문 데이터를 복호화하도록 구성된 블록 암호 디바이스의 기능적 컴포넌트들을 예시한 블록도이다. 블록 암호 디바이스 (902) 는 변환 또는 암호화 함수 및 라운드 키 생성기 (916) 에 의해 제공되는 키에 따라 메모리 어드레스 (906) 를 변환 및/또는 암호화하는 어드레스 암호화 모듈 (912) 을 포함할 수도 있다. 메모리 어드레스 (906) 는 암호문 데이터 (924) 가 메모리 디바이스 (926) 로부터 검색되고 있는 위치일 수도 있다. 라운드 키 생성기 (916) 는 비밀 키 (908) 에 기초하여 하나 이상의 라운드 키들을 생성하도록 구성될 수도 있다. 키 스케줄링 모듈 (914) 이 또한 어드레스 변환 모듈 (912) 의 결과들에 기초하여 하나 이상

의 데이터 키들을 생성할 수도 있다. 하나 이상의 데이터 키들은 데이터 키 저장장치 (922) 모듈에 저장될 수도 있다. 어드레스 암호화 및 데이터 스케줄링 함수들은 각각의 라운드에서의 라운드 키 생성기 (916) 로부터의 상이한 라운드 키를 이용하여 다수의 라운드들에서 반복적으로 수행될 수도 있다. 동시에 또는 병렬로, 암호문 데이터 (924) 는 화이트닝된 평문 데이터를 생성하기 위해 데이터 키 저장장치 (922) 로부터의 저장된 데이터 키들 및/또는 변환 또는 복호화 함수를 이용하여 하나 이상의 라운드들에서 데이터 복호화 모듈 (920) 에 의해 반복적으로 변환 또는 복호화될 수도 있다. 복수의 복호화 라운드들 후에, 결합기 (918) 는 데이터 복호화 모듈 (920) 의 마지막 결과 (화이트닝된 평문 데이터) 를 가역 연산 (예를 들어, 모듈러 덧셈/뺄셈, 비트단위 XOR 등) 을 이용하여 어드레스 암호화 모듈 (912) 의 마지막 결과들과 결합하여 평문 데이터 (904) 를 획득할 수도 있다.

[0059] 어드레스 암호화 모듈 (912) 에서, 메모리 어드레스는 암호화 모드에서 블록 암호 디바이스에 의해 행한 것처럼 암호화될 수도 있다. 예를 들어, 어드레스 암호화 모듈 (912) 은 도 5 에 예시한 바와 같이 복수의 치환-순열 암호 라운드들을 포함할 수도 있다.

[0060] 도 10 은 암호문 데이터 복호화 또는 역변환 모듈의 일 예를 예시한 블록도이다. 예를 들어, 이 암호문 데이터 복호화 또는 역변환 모듈 (1002) 은 데이터 복호화 모듈 (920) (도 9) 의 일부로서 포함될 수도 있다. 키 혼합 모듈 (1012) 은 입력 암호문 (1014) 과 대응하는 암호 라운드 키 간에 비트단위 XOR 연산을 수행할 수도 있다. 데이터 복호화 페이즈에 대한 암호 라운드 키들은 메모리 어드레스 암호화 페이즈로부터 유도될 수도 있다. 키 혼합 모듈 (1012) 로부터의 결과는 그 후 역선형 변환 모듈 (1010) 에 의해 변환된다 (예를 들어, AES MDS 맵핑). 역선형 변환 모듈 (1010) 로부터의 결과는 그 후 데이터 세그먼트화 모듈 (1009) 에 의해 복수의 8-비트 데이터 세그먼트들로 세그먼트화된다. 복수의 8-비트 데이터 세그먼트들은 그 후 치환 박스들 (1008) (예를 들어, AES 8×8 치환 박스) 을 통과하게 된다. 치환 박스들 (1008) 은 데이터 암호화 치환 박스들 (608) (도 6) 의 치환 박스들의 역일 수도 있다.

[0061] 데이터 결합기 모듈 (1006) 이 치환 박스들 (1008) 로부터의 결과의 출력을 결합하여 출력 화이트닝된 평문 데이터를 생성할 수도 있다. 이 프로세스는 각각의 라운드 마다 상이한 라운드 키를 이용하여 다수 회 반복될 수도 있다. 데이터 복호화 페이즈 (1002) 의 마지막 암호 라운드의 결과는 화이트닝된 평문 데이터이다. 화이트닝된 평문 데이터는 그 후 비트단위 XOR 모듈 (1005) 에 의해 암호화된 메모리 어드레스 (1003) 와 결합되어, 출력 평문 데이터 (1004) 를 생성한다. 암호화된 메모리 어드레스 (1003) 는 입력 암호문 데이터 (1014) 가 검색되었던 메모리 어드레스에 대응할 수도 있다.

[0062] 도 11 은 블록 암호의 레이턴시를 저감시키기 위해 암호화된 데이터를 복호화하면서 메모리 어드레스를 동시에 암호화하는 블록 암호를 이용함으로써 암호화된 데이터를 복호화하는 방법을 예시한다. 검색될 암호화된 데이터에 대한 메모리 어드레스가 획득된다 (1102). 메모리 어드레스는 암호화된 메모리 어드레스를 획득하기 위해 제 1 복수의 블록 암호 라운드들에서 암호화된다 (1104). 이러한 어드레스 암호화는 비밀 키에 기초하여 생성된 복수의 라운드 키들을 이용할 수도 있다. 부가적으로, 메모리 어드레스를 암호화하는 것은, (a) 변환된 메모리 어드레스를 라운드 키와 혼합하는 것, (b) 제 1 변환 함수에 따라 메모리 어드레스를 변환하는 것, (c) 메모리 어드레스를 세그먼트화하는 것, 및/또는 (d) 상이한 메모리 어드레스 세그먼트들에 대해 비트 치환을 수행하는 것을 포함할 수도 있다.

[0063] 데이터 라운드 키들은 또한 제 1 복수의 블록 암호 라운드들 중 하나 이상으로부터의 정보를 이용하여 생성될 수도 있다 (1106). 즉, 제 1 복수의 블록 암호 라운드들 중 적어도 일부로부터의 부분적으로 암호화된 메모리 어드레스는 데이터 라운드 키들을 생성하는데 이용될 수도 있다. 예를 들어, 데이터 라운드 키들을 생성하는 것은, (a) 제 1 복수의 블록 암호 라운드들 중 적어도 일부에 대한 암호화된 메모리 어드레스로부터 복수의 비트들을 추출하는 것, (b) 추출된 복수의 비트들의 세그먼트들로부터 데이터 라운드 키들을 선택하는 것, 및/또는 (c) 추출된 복수의 비트들을 데이터 라운드 키들이 선택되는 스트링에 연쇄시키는 것을 포함할 수도 있다.

[0064] 암호화된 데이터는 메모리 어드레스로부터 검색될 수도 있고 (1108), 부분적으로 복호화된 데이터를 획득하기 위해 데이터 라운드 키들을 이용하여 제 2 복수의 블록 암호 라운드들에서 복호화될 수도 있다 (1110). 제 2 복수의 블록 암호 라운드들 중 초기의 라운드들에 대해 이용되는 데이터 라운드 키들은 제 1 복수의 블록 암호 라운드들 중 초기의 라운드들로부터의 암호화된 메모리 어드레스로부터의 비트들을 이용하여 생성될 수도 있다. 일 예에서, 암호화된 데이터를 복호화하는 것은, (a) 변환된 암호화된 데이터를 데이터 라운드 키들 중 하나 이상과 혼합하는 것, (b) 제 2 역변환 함수에 따라 암호화된 데이터를 변환하는 것, (c) 암호화된 데이터

를 복수의 암호화된 데이터 세그먼트들로 세그먼트화하는 것, 및/또는 (d) 상이한 암호화된 데이터 세그먼트들에 대해 비트 치환을 수행하는 것을 포함할 수도 있다. 부분적으로 복호화된 데이터는 완전히 복호화된 데이터를 획득하기 위해 암호화된 메모리 어드레스와 결합될 수도 있다 (1112). 일 예에서, 부분적으로 복호화된 데이터는 가역 연산 (예를 들어, 비트단위 XOR 연산) 에 의해 암호화된 메모리 어드레스와 결합된다.

[0065] 제 1 복수의 블록 암호 라운드들은 제 2 복수의 블록 암호 라운드들과 동시에 실행될 수도 있어 복호화 프로세스를 더 신속히 처리할 수도 있다. 또한, 제 2 복수의 블록 암호 라운드들은 제 1 복수의 블록 암호 라운드들보다 클 수도 있다.

[0066] **블록 암호에 대한 효율적인 키 스케줄링**

[0067] 일 특징에 따르면, 키 스케줄링이 데이터를 효율적으로 암호화 및 복호화하기 위해 수행될 수도 있다. 어드레스 암호화 페이즈 동안, 복수의 암호 라운드들이 메모리 어드레스를 암호화하기 위해 반복적으로 실행될 수도 있으며, 여기서 메모리 어드레스는 데이터가 저장될 위치 또는 데이터가 검색될 위치이다. 각각의 암호 라운드는 암호화된 메모리 어드레스를 생성한다. 이들 암호 라운드들 중 하나 이상에 의해 생성되는 암호화된 메모리 어드레스는, 데이터 암호화/복호화 페이즈 라운드 키들을 생성하는데 (완전히 또는 부분적으로) 이용될 수도 있다.

[0068] 도 12 는 블록 암호의 어드레스 암호화 라운드들로부터의 결과들에 기초하여 데이터 암호화 및 복호화 라운드들에 대한 라운드 키들이 생성될 수도 있는 방법의 일 예를 예시한 블록도이다. 블록 암호가 데이터를 암호화중인 경우, 데이터 라운드 키들은 어드레스 암호화 페이즈 (1202) 의 결과들에 기초하여 생성된다. 어드레스 암호화 페이즈 (1202) 의 초기 라운드들의 결과들 (예를 들어, R1 (1206), R2 (1208)...) 은 데이터 암호화 페이즈 (1204) 에서 이용될 최근의 데이터 암호화 라운드 키들 (키-E6 (1212), 키-E5 (1214) ...) 을 생성하는데 이용된다. 유사하게, 어드레스 암호화 페이즈 (1202) 의 최근의 라운드들의 결과들 (예를 들어, R3 (1210) ...) 은 초기 데이터 암호화 라운드 키들 (키-E1 (1222), 키-E2 (1220) ...) 을 생성하는데 이용된다.

[0069] 유사하게, 블록 암호가 데이터를 복호화중인 경우, 데이터 라운드 키들은 어드레스 암호화 페이즈 (1202) 의 결과들에 기초하여 생성된다. 어드레스 암호화 페이즈 (1202) 의 초기 라운드들의 결과들 (예를 들어, R1 (1206), R2 (1208) ...) 은 데이터 복호화 페이즈 (1224) 에서 이용될 초기 데이터 암호화 라운드 키들 (키-D1 (1226), 키-D2 (1228) ...) 을 생성하는데 이용된다. 유사하게, 어드레스 암호화 페이즈 (1202) 의 최근의 라운드들의 결과들 (예를 들어, R3 (1210)...) 은 최근의 데이터 복호화 라운드 키들 (키-D6 (1236), 키-D5 (1234) ...) 을 생성하는데 이용된다. 그 결과, 이는 데이터 복호화 페이즈 (1224) 가 어드레스 암호화 페이즈 (1202) 와 동시에 (중복된 시간 주기 또는 병렬로) 실행되는 것을 허용하여, 데이터를 보다 효율적으로 복호화한다.

[0070] 다양한 구현에서, 어드레스 암호화 페이즈 (1202), 데이터 암호화 페이즈 (1204) 및/또는 데이터 복호화 페이즈 (1224) 의 암호 라운드들의 수는 이 예에서 도시된 것보다 많거나 적을 수도 있다. 부가적으로, 일 선택적 특징에 따르면, 어드레스 암호화 페이즈 (1202) 의 마지막 라운드의 결과 (예를 들어, R4 (1211)) 의 적어도 일부분은 평문 데이터의 화이트닝 연산용으로 예비될 수도 있다. 그 결과, 어드레스 암호화 페이즈 (1202) 의 마지막 라운드의 이 결과 (예를 들어, R4 (1211)) 는 데이터 라운드 키 생성을 위해 이용되지 않을 수도 있다.

[0071] 일부 구현에서, 데이터 암호화 라운드 키 (또는 데이터 복호화 라운드 키) 가 어드레스 암호화 페이즈 (1202) 의 하나 이상의 결과들 (예를 들어, R1 (1206), R2 (1208), ...) 로부터의 비트들의 서브세트에 기초할 수도 있다. 예를 들어, 키-E1 (1222) 은 R3 (1210) 으로부터의 비트들의 서브세트에 기초할 수도 있고, 키-E2 는 R2 (1208) 와 R3 (1210) 양자로부터의 비트들의 서브세트에 기초할 수도 있다.

[0072] 메모리 어드레스가 데이터 암호화/복호화 페이즈들 (1204/1224) 에 대한 암호화/복호화 키들을 생성하기 위해 블록 암호에 의해 이용되기 때문에, 이는 평문/암호문의 블록 암호 변환이 각각의 메모리 어드레스 마다 상이하여 암호해독에 이용가능한 리소스들을 심하게 제약하고, 블록 암호의 보안성을 증가시킨다는 것을 의미한다. 반드시 초기 라운드들이 최근의 라운드들과 동일한 블록 크기를 가질 필요가 있는 것은 아니라는 것을 알아야 한다. 예를 들어, 메모리는 32-비트 블록 단위로 암호화될 가능성이 상당한 한편, 어드레스는 그보다 클지도 모른다. 제 1 라운드들에서의 병렬화 (parallelization) 에 의해 효율이 얻어지게 될 것이다.

[0073] 블록 암호의 일 예에 따르면, 데이터 암호화/복호화는 바이트 어드레스가능한 메모리일 수도 있다. 구체적으로, 블록 암호를 실행하는 프로세스의 워드 (데이터 블록) 크기는 32 비트이고 어드레스 또한 32 비트이다.

어드레스 암호화 페이즈로부터의 결과의 마지막 32 비트는 화이트닝 키로서 이용될 수도 있다. 어드레스 암호화 결과들 (예를 들어, 암호화된 메모리 어드레스) 로부터의 나머지 비트들은 데이터 암호화 라운드 키들용으로 이용되는 세트 S 에 연쇄될 수도 있다. 32-비트 길이 데이터 암호화 라운드 키는 키-En = 세트 S 의 비트들 32 * (5-n) 내지 32 * (5-n) + 31 이 되도록 각각의 데이터 암호화 라운드 n (예를 들어, n = 0 ... 5) 에 대해 선택될 수도 있다. 반대로, 32-비트 길이 데이터 복호화 라운드 키는 키-Dn = 세트 S 의 비트들 32 * n 내지 32 * n + 31 이 되도록 각각의 데이터 복호화 라운드 n (예를 들어, n = 0 ... 5) 에 대해 선택될 수도 있다.

[0074] 도 13 은 효율적인 블록 암호 암호화 및 복호화를 수행하도록 구성될 수도 있는 디바이스를 예시한 블록도이다.

프로세싱 회로 (1302) 는 메모리 디바이스 (1306) 에 커플링될 수도 있다. 프로세싱 회로 (1302) 는 메모리 디바이스 (1306) 에 데이터를 기록할 수도 있고, 메모리 디바이스 (1306) 로부터 데이터를 판독할 수도 있다. 프로세싱 회로 (1302) 는 메모리 디바이스 (1306) 에 저장될 데이터를 암호화하고, 메모리 디바이스 (1306) 로부터 검색될 데이터를 복호화하는 블록 암호 (1304) 를 실행하도록 구성될 수도 있다. 이러한 암호화 및 복호화는 데이터가 기록되고 데이터가 판독되는 실제 메모리 어드레스에 기초할 수도 있다. 예를 들어, 블록 암호 (1304) 는 도 1 내지 도 12 에 설명된 동작들 중 하나 이상을 수행할 수도 있다.

[0075] 일반적으로, 본 개시물에 설명된 프로세싱의 대부분은 유사한 방식으로 구현될 수도 있다는 것이 인지되어야 한다. 회로(들) 또는 회로 섹션들 중 임의의 것은 하나 이상의 프로세서들을 가진 집적 회로의 일부로서 단독으로 또는 조합하여 구현될 수도 있다. 회로들 중 하나 이상은 집적 회로, ARM (Advance RISC Machine) 프로세서, 디지털 신호 프로세서 (DSP), 범용 프로세서 등 상에 구현될 수도 있다.

[0076] 또한, 본 실시형태들이 플로우차트, 플로우도, 구조도 또는 블록도로서 도시되는 프로세스로서 설명될 수도 있다는 것을 알게 된다. 플로우차트가 동작들을 순차적인 프로세스로서 설명할 수도 있지만, 동작들 대부분이 병렬로 또는 동시에 수행될 수 있다. 또한, 동작들의 순서가 재배열될 수도 있다. 프로세스는 그의 동작들이 완료될 때 종료된다. 프로세스는 방법, 함수, 절차, 서브루틴, 서브프로그램 등에 대응할 수도 있다. 프로세스가 함수에 대응하는 경우, 그의 종료는 함수의 호출 함수 또는 메인 함수로의 복귀에 대응한다.

[0077] 본 출원에서 이용한 바와 같이, "컴포넌트", "모듈", "시스템" 등의 용어들은 컴퓨터 관련 엔티티, 하드웨어, 펌웨어, 하드웨어와 소프트웨어의 조합, 소프트웨어, 또는 실행중인 소프트웨어 (software in execution) 중 어느 하나를 지칭하는 것으로 의도된다. 예를 들어, 컴포넌트는 프로세서 상에서 실행하는 프로세스, 프로세서, 오브젝트, 실행가능한 것, 실행의 스레드, 프로그램 및/또는 컴퓨터일 수도 있지만 이들로 제한되지는 않는다. 예시로, 컴퓨팅 디바이스 상에서 실행하는 애플리케이션과 컴퓨팅 디바이스 양자는 컴포넌트일 수 있다. 하나 이상의 컴포넌트들은 프로세스 및/또는 실행의 스레드 내에 상주할 수 있고, 컴포넌트는 하나의 컴퓨터 상에 로컬화될 수도 있고/있거나 2 개 이상의 컴퓨터들 사이에 분산될 수도 있다. 또한, 이들 컴포넌트들은 다양한 데이터 구조가 저장되어 있는 다양한 컴퓨터 판독가능 매체로부터 실행할 수 있다. 컴포넌트들은 이를 테면 하나 이상의 데이터 패킷들을 갖는 신호 (예를 들어, 로컬 시스템, 분산형 시스템 내의 다른 컴포넌트와 상호작용하고/하거나 신호에 의해 다른 시스템들과 인터넷과 같은 네트워크를 통해 상호작용하는 일 컴포넌트로부터의 데이터) 에 따라 로컬 및/또는 원격 프로세스들에 의해 통신할 수도 있다.

[0078] 또한, 저장 매체는 판독 전용 메모리 (ROM), 랜덤 액세스 메모리 (RAM), 자기 디스크 저장 매체, 광학 저장 매체, 플래시 메모리 디바이스 및/또는 정보를 저장하기 위한 다른 머신 판독가능 매체를 포함하는, 데이터를 저장하기 위한 하나 이상의 디바이스들을 나타낼 수도 있다. "머신 판독가능 매체" 란 용어는 휴대용 또는 고정 저장 디바이스들, 광학 저장 디바이스들, 무선 채널들 및 명령(들) 및/또는 데이터를 저장, 포함 또는 운반할 수 있는 다양한 다른 매체들을 포함하지만 이들로 제한되지는 않는다.

[0079] 또한, 실시형태들은 하드웨어, 소프트웨어, 펌웨어, 미들웨어, 마이크로코드, 또는 이들의 임의의 조합에 의해 구현될 수도 있다. 소프트웨어, 펌웨어, 미들웨어 또는 마이크로코드에서 구현한 경우, 필요한 태스크들을 수행하기 위한 프로그램 코드 또는 코드 세그먼트들은 저장 매체 또는 다른 저장장치(들)와 같은 머신 판독가능 매체에 저장될 수도 있다. 프로세서가 필요한 태스크들을 수행할 수도 있다. 코드 세그먼트는 절차, 함수, 서브프로그램, 프로그램, 루틴, 서브루틴, 모듈, 소프트웨어 패키지, 클래스 또는 명령들, 데이터 구조들 또는 프로그램 스테이트먼트들의 임의의 조합을 나타낼 수도 있다. 코드 세그먼트는 정보, 데이터, 인수, 파라미터 또는 메모리 컨텐츠를 전달 및/또는 수신함으로써 다른 코드 세그먼트 또는 하드웨어 회로에 커플링될 수도 있다. 정보, 인수, 파라미터, 데이터 등은 메모리 공유, 메시지 전달, 토큰 전달, 네트워크 송신 등을

포함하는 임의의 적절한 수단을 통해 전달, 포워딩 또는 송신될 수도 있다.

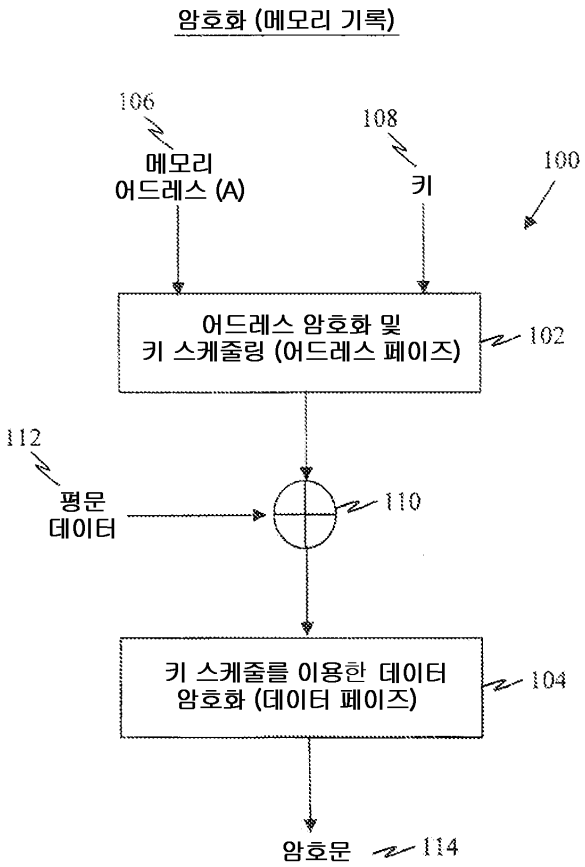
[0080] 도 1, 도 2, 도 3, 도 4, 도 5, 도 6, 도 7, 도 8, 도 9, 도 10, 도 11, 도 12 및/또는 도 13 에 예시된 컴포넌트들, 단계들 및/또는 함수들 중 하나 이상은 단일의 컴포넌트, 단계 또는 함수로 재배열 및/또는 결합될 수도 있고 또는 여러 컴포넌트들, 단계들 또는 함수들에 포함될 수도 있다. 부가적인 엘리먼트들, 컴포넌트들, 단계들 및/또는 함수들이 또한 추가될 수도 있다. 도 3, 도 4, 도 8, 도 9 및/또는 도 13 에 예시된 장치, 디바이스 및/또는 컴포넌트들은 도 1, 도 2, 도 5, 도 6, 도 7, 도 10, 도 11 및/또는 도 12 에 기술된 방법, 특징 또는 단계 중 하나 이상을 수행하도록 구성될 수도 있다. 본원에 설명된 신규한 알고리즘은 소프트웨어 및/또는 임베디드 하드웨어에서 효율적으로 구현될 수도 있다.

[0081] 당업자는 또한 본원에 개시된 실시형태들과 함께 설명되는 다양한 예시적인 로직 블록, 모듈, 회로 및 알고리즘 단계가 전자 하드웨어, 컴퓨터 소프트웨어 또는 양자의 조합으로서 구현될 수도 있다는 것을 알 것이다. 하드웨어와 소프트웨어의 이런 상호교환가능성을 명확히 예시하기 위해, 다양한 예시적인 컴포넌트, 블록, 모듈, 회로 및 단계가 일반적으로는 그들의 기능성의 관점에서 상술되었다. 이러한 기능성이 하드웨어로서 구현되는지 소프트웨어로서 구현되는지는 전체 시스템에 부과된 특정 애플리케이션 및 설계 제약에 의존한다.

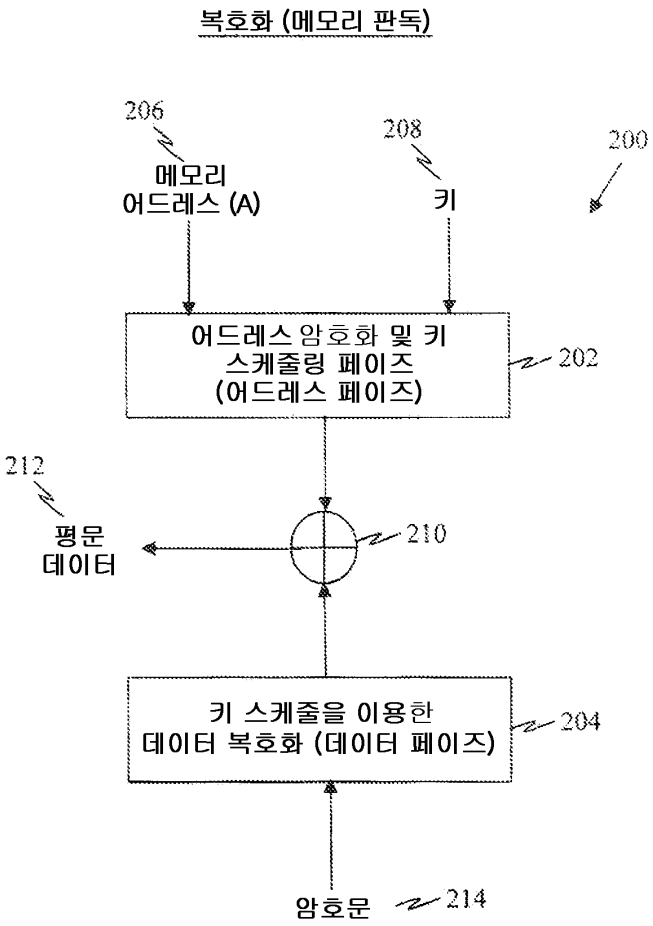
[0082] 본 실시형태들의 설명은 예시인 것으로 의도되며, 특허청구의 범위를 제한하지 않는다. 이로써, 본 교시는 다른 형태의 장치에 쉽게 적용될 수 있으며, 다수의 변경, 변형 및 변화가 당업자에게 명백할 것이다.

도면

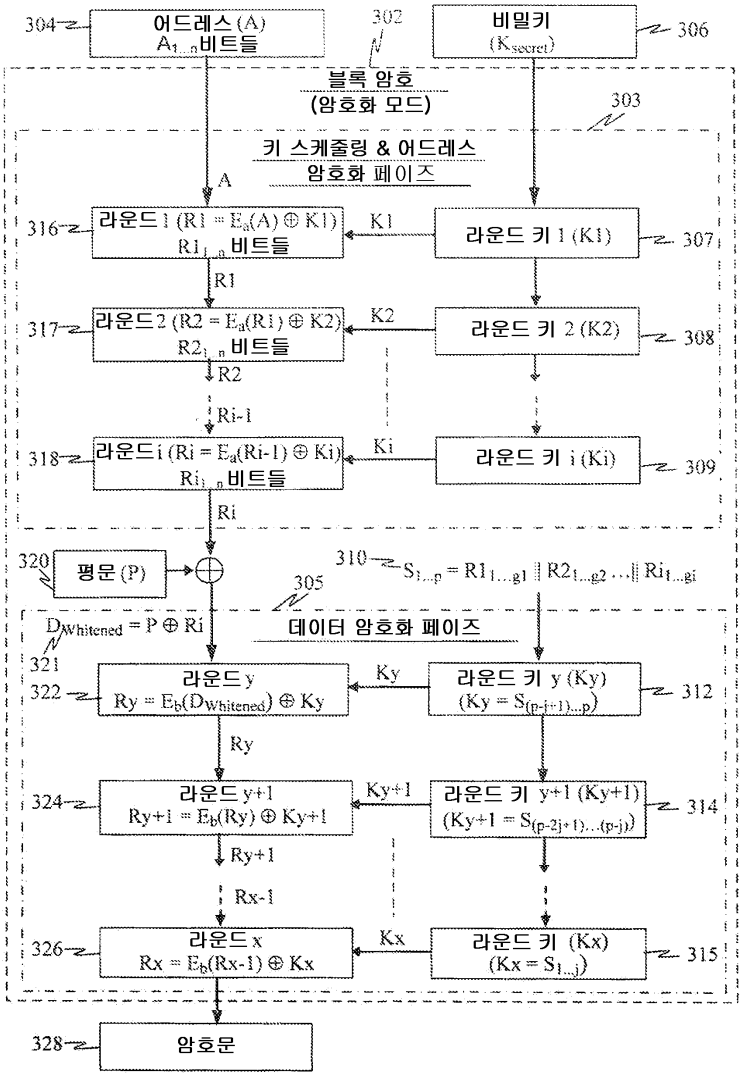
도면1



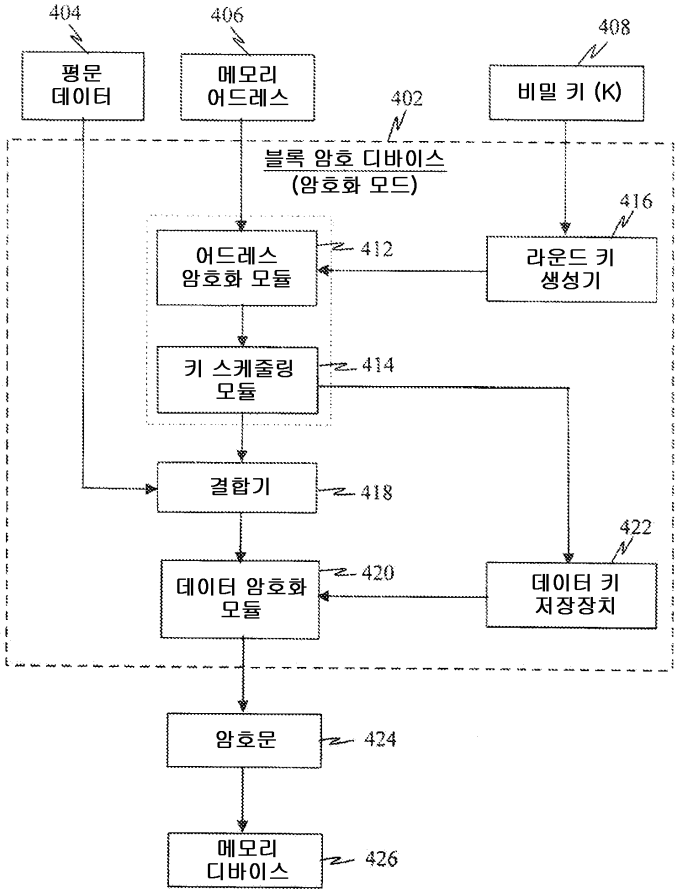
도면2


복호화 (메모리 판독)
206
메모리
어드레스 (A)
208
키
200
어드레스 암호화 및 키
스케줄링 페이즈
(어드레스 페이즈)
202
212
평문
데이터
210
키 스케줄을 이용한
데이터 복호화 (데이터 페이즈)
204
암호문
214

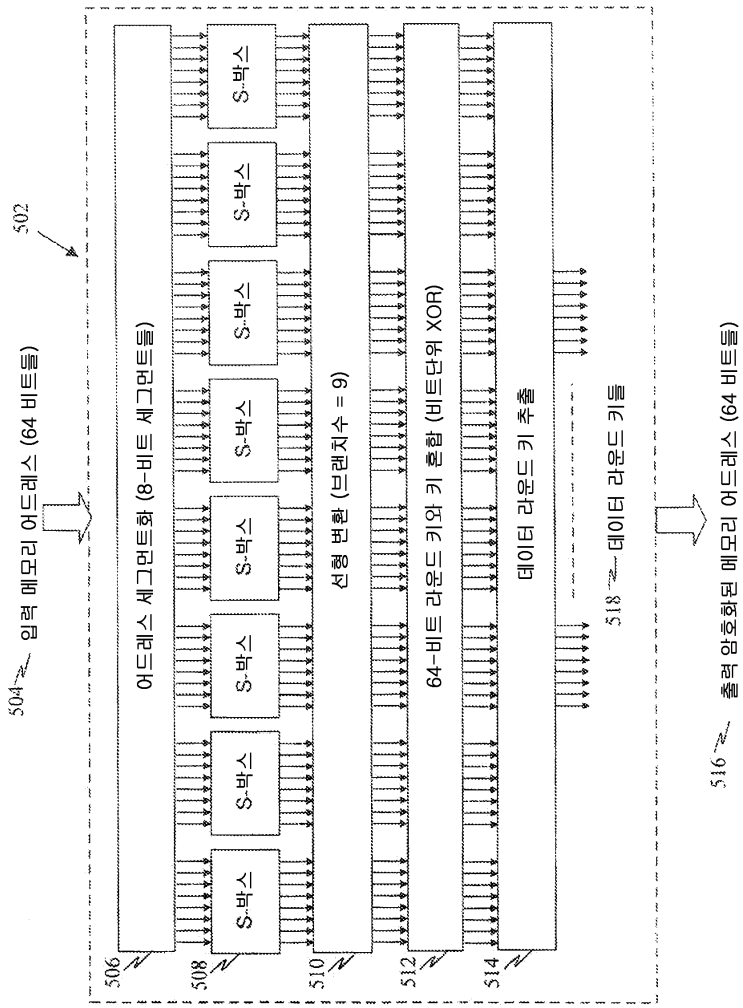
도면3


304
어드레스 (A)
$A_{1...n}$ 비트들
302
비밀키
(K_{secret})
306
블록 암호
(암호화 모드)
303
키 스케줄링 & 어드레스
암호화 페이즈
A
316
라운드 1 ($R1 = E_a(A) \oplus K1$)
$R1_{1...n}$ 비트들
K1
라운드 키 1 (K1)
307
R1
317
라운드 2 ($R2 = E_a(R1) \oplus K2$)
$R2_{1...n}$ 비트들
K2
라운드 키 2 (K2)
308
R2
Ri-1
318
라운드 i ($Ri = E_a(Ri-1) \oplus Ki$)
$Ri_{1...n}$ 비트들
Ki
라운드 키 i (Ki)
309
Ri
320
평문 (P)
310
$S_{1...p} = R1_{1...g1} \| R2_{1...g2} \ldots \| Ri_{1...gi}$
305
$D_{Whitened} = P \oplus Ri$
데이터 암호화 페이즈
321
322
라운드 y
$Ry = E_b(D_{Whitened}) \oplus Ky$
Ky
라운드 키 y (Ky)
($Ky = S_{(p-j+1)...p}$)
312
Ry
324
라운드 y+1
$Ry+1 = E_b(Ry) \oplus Ky+1$
Ky+1
라운드 키 y+1 (Ky+1)
($Ky+1 = S_{(p-2j+1)...(p-j)}$)
314
Ry+1
Rx-1
326
라운드 x
$Rx = E_b(Rx-1) \oplus Kx$
Kx
라운드 키 (Kx)
($Kx = S_{1...j}$)
315
328
암호문

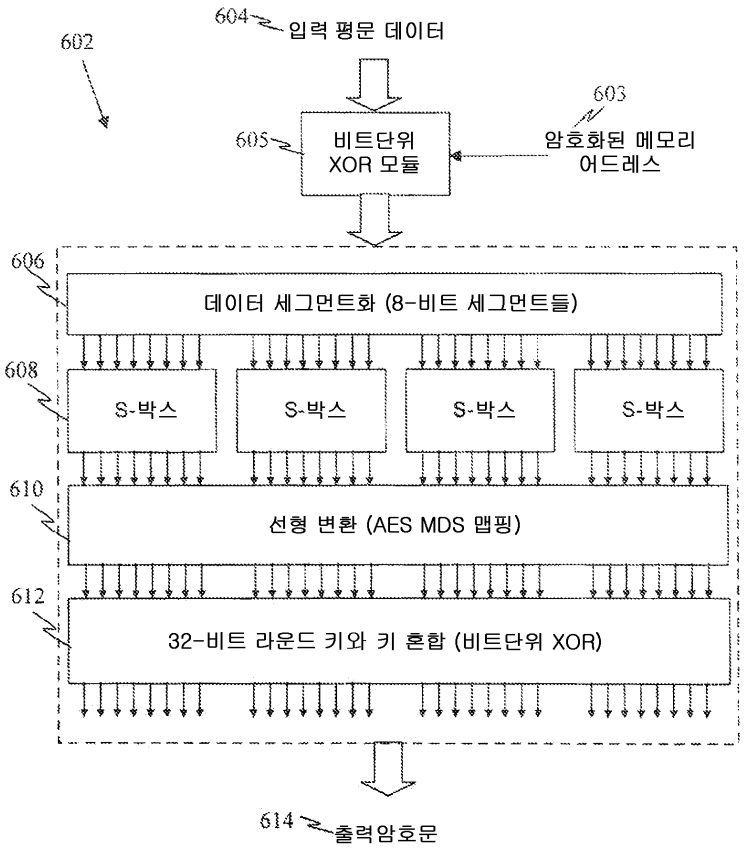
도면4


404
406
408
평문 데이터
메모리 어드레스
비밀 키 (K)
402
블록 암호 디바이스
(암호화 모드)
416
어드레스 암호화 모듈
412
라운드 키 생성기
키 스케줄링 모듈
414
결합기
418
422
데이터 암호화 모듈
420
데이터 키 저장장치
암호문
424
메모리 디바이스
426

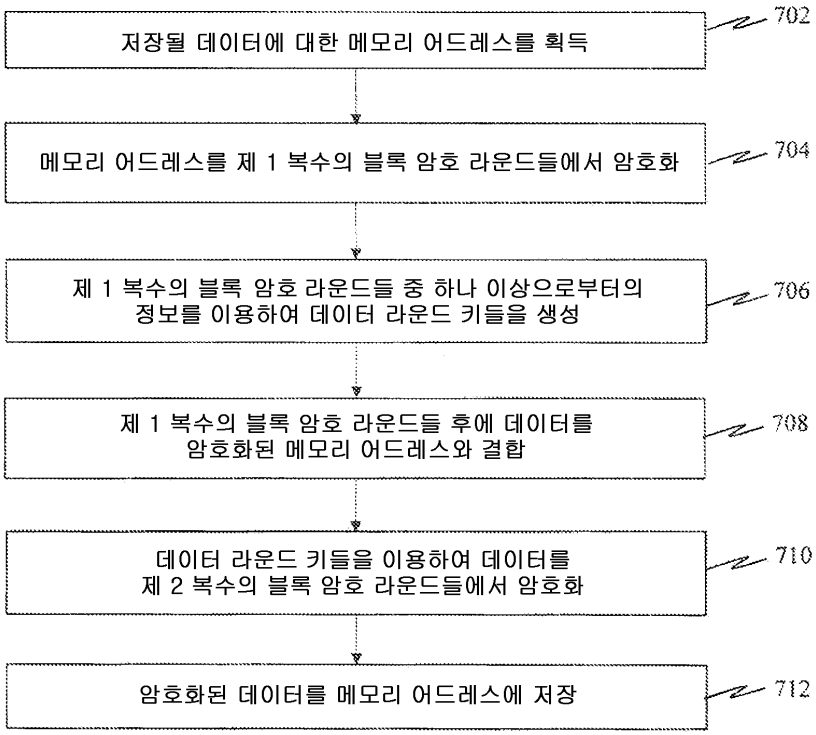
도면5


502
504
입력 메모리 어드레스 (64 비트들)
506
어드레스 세그먼트화 (8-비트 세그먼트들)
508
S-박스
S-박스
S-박스
S-박스
S-박스
S-박스
S-박스
S-박스
510
선형 변환 (브랜치수 = 9)
512
64-비트 라운드 키와 키 혼합 (비트단위 XOR)
514
데이터 라운드 키 추출
518
데이터 라운드 키들
516
출력 암호화된 메모리 어드레스 (64 비트들)

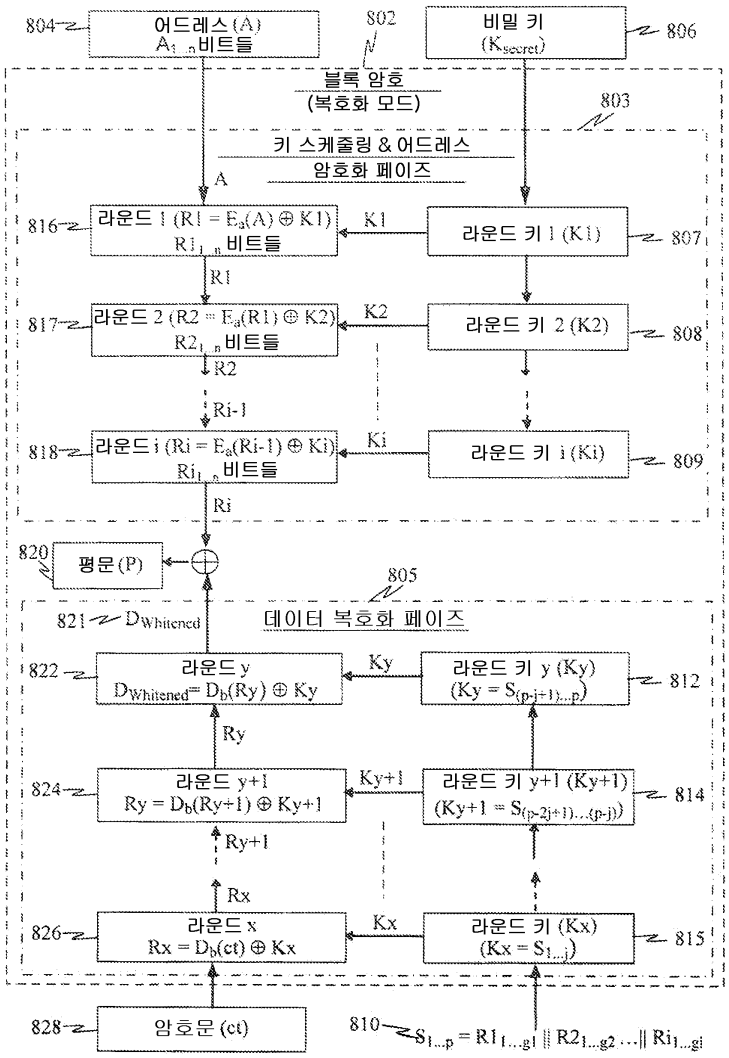
도면6


602
604
입력 평문 데이터
603
605
비트단위
XOR 모듈
암호화된 메모리
어드레스
606
데이터 세그먼트화 (8-비트 세그먼트들)
608
S-박스
S-박스
S-박스
S-박스
610
선형 변환 (AES MDS 맵핑)
612
32-비트 라운드 키와 키 혼합 (비트단위 XOR)
614
출력암호문

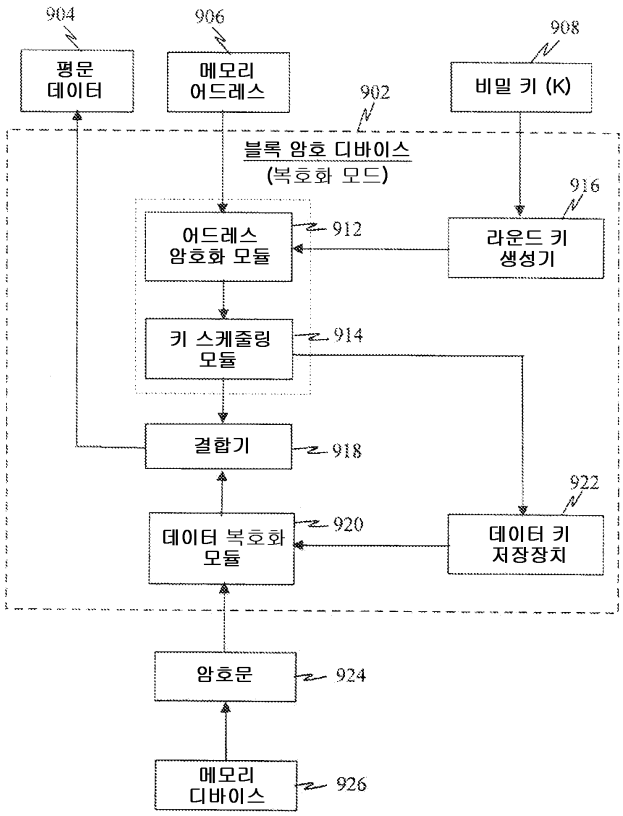
도면7


저장될 데이터에 대한 메모리 어드레스를 획득
702
메모리 어드레스를 제 1 복수의 블록 암호 라운드들에서 암호화
704
제 1 복수의 블록 암호 라운드들 중 하나 이상으로부터의
정보를 이용하여 데이터 라운드 키들을 생성
706
제 1 복수의 블록 암호 라운드들 후에 데이터를
암호화된 메모리 어드레스와 결합
708
데이터 라운드 키들을 이용하여 데이터를
제 2 복수의 블록 암호 라운드들에서 암호화
710
암호화된 데이터를 메모리 어드레스에 저장
712

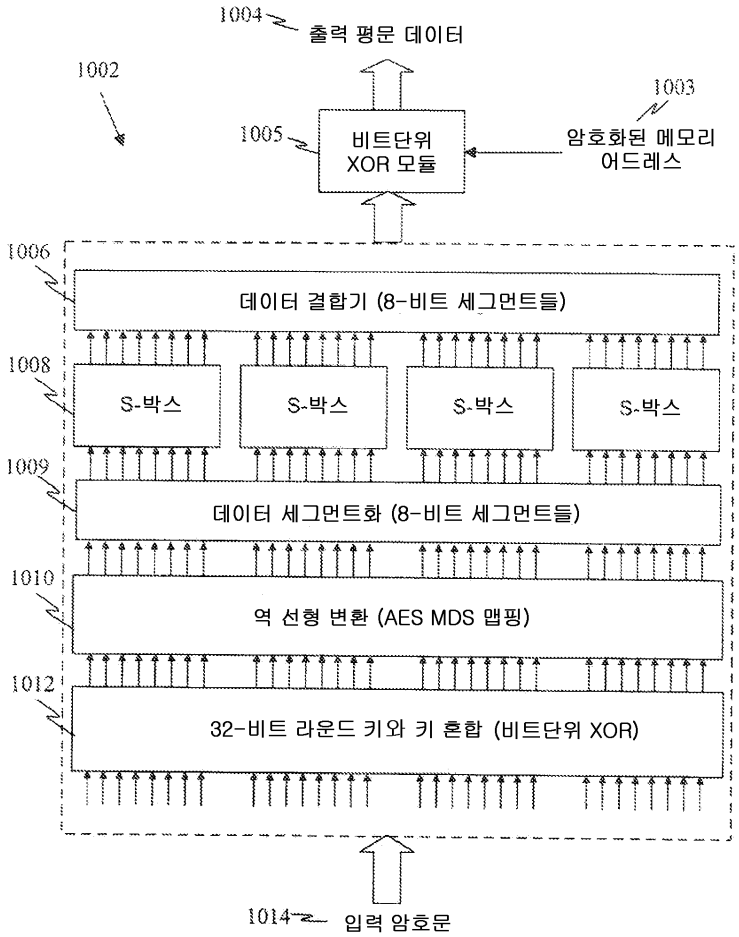
도면8


804
어드레스 (A)
$A_{1...n}$ 비트들
802
비밀 키
(K_{secret})
806
블록 암호
(복호화 모드)
803
키 스케줄링 & 어드레스
암호화 페이즈
A
816
라운드 1 ($R1 = E_a(A) \oplus K1$)
$R1_{1...n}$ 비트들
K1
라운드 키 1 (K1)
807
R1
817
라운드 2 ($R2 = E_a(R1) \oplus K2$)
$R2_{1...n}$ 비트들
K2
라운드 키 2 (K2)
808
R2
Ri-1
818
라운드 i ($Ri = E_a(Ri-1) \oplus Ki$)
$Ri_{1...n}$ 비트들
Ki
라운드 키 i (Ki)
809
Ri
820
평문 (P)
805
821
$D_{Whitened}$
데이터 복호화 페이즈
822
라운드 y
$D_{Whitened} = D_b(Ry) \oplus Ky$
Ky
라운드 키 y (Ky)
($Ky = S_{(p-j+1)...p}$)
812
Ry
824
라운드 y+1
$Ry = D_b(Ry+1) \oplus Ky+1$
Ky+1
라운드 키 y+1 (Ky+1)
($Ky+1 = S_{(p-2j+1)...(p-j)}$)
814
Ry+1
Rx
826
라운드 x
$Rx = D_b(ct) \oplus Kx$
Kx
라운드 키 (Kx)
($Kx = S_{1...j}$)
815
828
암호문 (ct)
810
$S_{1...p} = R1_{1...g1} \| R2_{1...g2} \ldots \| Ri_{1...gi}$

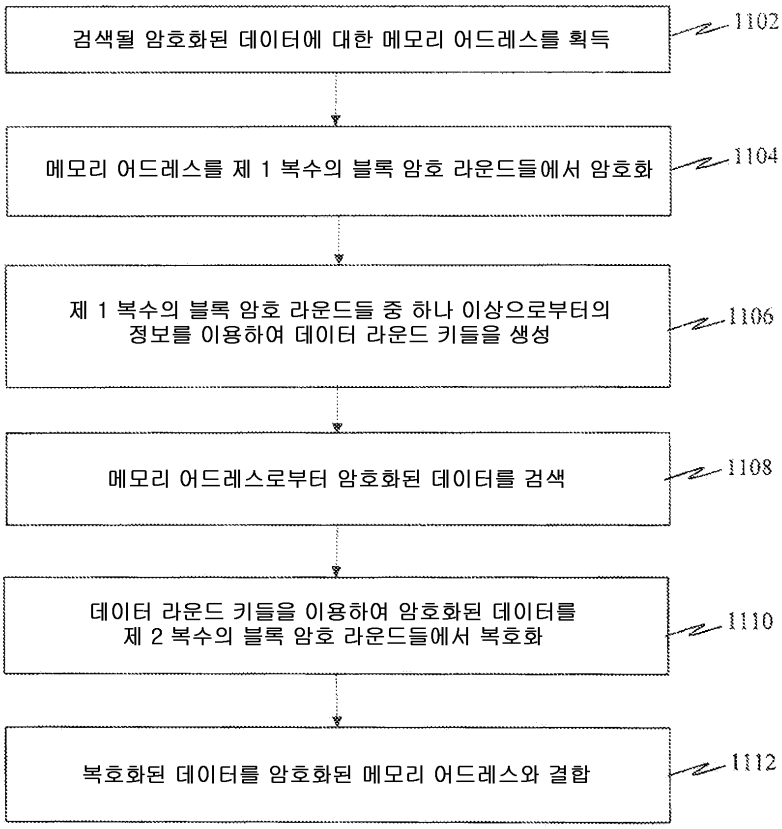
도면9


904
906
908
평문 데이터
메모리 어드레스
비밀 키 (K)
902
블록 암호 디바이스
(복호화 모드)
916
어드레스 암호화 모듈
912
라운드 키 생성기
키 스케줄링 모듈
914
결합기
918
922
데이터 복호화 모듈
920
데이터 키 저장장치
암호문
924
메모리 디바이스
926

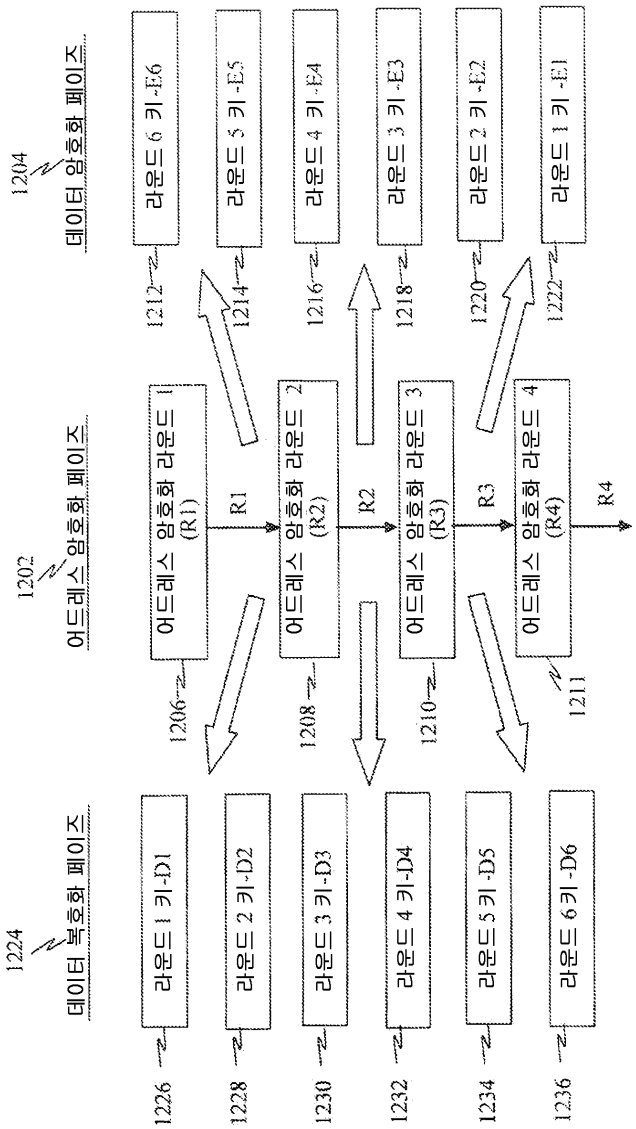
도면10


1004
출력 평문 데이터
1002
1003
1005
비트단위
XOR 모듈
암호화된 메모리
어드레스
1006
데이터 결합기 (8-비트 세그먼트들)
1008
S-박스
S-박스
S-박스
S-박스
1009
데이터 세그먼트화 (8-비트 세그먼트들)
1010
역 선형 변환 (AES MDS 맵핑)
1012
32-비트 라운드 키와 키 혼합 (비트단위 XOR)
1014
입력 암호문

도면11


검색될 암호화된 데이터에 대한 메모리 어드레스를 획득
1102
메모리 어드레스를 제 1 복수의 블록 암호 라운드들에서 암호화
1104
제 1 복수의 블록 암호 라운드들 중 하나 이상으로부터의 정보를 이용하여 데이터 라운드 키들을 생성
1106
메모리 어드레스로부터 암호화된 데이터를 검색
1108
데이터 라운드 키들을 이용하여 암호화된 데이터를 제 2 복수의 블록 암호 라운드들에서 복호화
1110
복호화된 데이터를 암호화된 메모리 어드레스와 결합
1112

도면12


1204
데이터 암호화 페이즈
1202
어드레스 암호화 페이즈
1224
데이터 복호화 페이즈
어드레스 암호화 라운드 1 (R1)
어드레스 암호화 라운드 2 (R2)
어드레스 암호화 라운드 3 (R3)
어드레스 암호화 라운드 4 (R4)
R1
R2
R3
R4
라운드 6 키 -E6
라운드 5 키 -E5
라운드 4 키 -E4
라운드 3 키 -E3
라운드 2 키 -E2
라운드 1 키 -E1
라운드 1 키-D1
라운드 2 키-D2
라운드 3 키-D3
라운드 4 키-D4
라운드 5 키 -D5
라운드 6 키 -D6
1206
1208
1210
1211
1212
1214
1216
1218
1220
1222
1226
1228
1230
1232
1234
1236

도면13