(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(51) 。Int. Cl. [7]
H04L 9/08

(11) 공개번호 특2002-0026547
(43) 공개일자 2002년04월10일

(21) 출원번호 10-2002-7000913
(22) 출원일자 2002년01월22일
번역문 제출일자 2002년01월22일
(86) 국제출원번호 PCT/GB2000/02813
(86) 국제출원출원일자 2000년07월20일
(87) 국제공개번호 WO 2001/08348
(87) 국제공개일자 2001년02월01일

(81) 지정국 국내특허 : 알바니아, 아르메니아, 오스트리아, 오스트레일리아, 아제르바이잔, 보스니아-헤르체고비나, 바베이도스, 불가리아, 브라질, 벨라루스, 캐나다, 스위스, 중국, 쿠바, 체코, 독일, 덴마크, 에스토니아, 스페인, 핀랜드, 영국, 그루지야, 헝가리, 이스라엘, 아이슬란드, 일본, 케냐, 키르기즈, 북한, 대한민국, 카자흐스탄, 세인트루시아, 스리랑카, 라이베리아, 레소토, 리투아니아, 룩셈부르크, 라트비아, 몰도바, 마다가스카르, 마케도니아, 몽고, 말라위, 멕시코, 노르웨이, 뉴질랜드, 슬로베니아, 슬로바키아, 타지키스탄, 투르크메니스탄, 터어키, 트리니다드토바고, 우크라이나, 우간다, 미국, 우즈베키스탄, 베트남, 폴란드, 포르투칼, 루마니아, 러시아, 수단, 스웨덴, 싱가포르, 아랍에미리트, 코스타리카, 도미니카연방, 알제리, 모로코, 탄자니아, 남아프리카, 벨리즈, 모잠비크, 그레나다, 가나, 감비아, 크로아티아, 인도네시아, 인도, 시에라리온, 유고슬라비아, 짐바브웨,
AP ARIPO특허: 케냐, 레소토, 말라위, 수단, 스와질랜드, 우간다, 시에라리온, 가나, 감비아, 짐바브웨,
EA 유라시아특허: 아르메니아, 아제르바이잔, 벨라루스, 키르기즈, 카자흐스탄, 몰도바, 러시아, 타지키스탄, 투르크메니스탄,
EP 유럽특허: 오스트리아, 벨기에, 스위스, 리히텐슈타인, 독일, 덴마크, 스페인, 프랑스, 영국, 그리스, 아일랜드, 이탈리아, 룩셈부르크, 모나코, 네덜란드, 포르투칼, 스웨덴, 핀랜드, 사이프러스,
OA OAPI특허: 부르키나파소, 베넹, 중앙아프리카, 콩고, 코트디브와르, 카메룬, 가봉, 기네, 말리, 모리타니, 니제르, 세네갈, 차드, 토고, 기네비쏘, 적도기네,

(30) 우선권주장 99305870.0 1999년07월23일 EP(EP)

(71) 출원인 브리티쉬 텔리커뮤니케이션즈 파블릭 리미티드 캄퍼니
내쉬 로저 윌리엄
영국 런던(우편번호 이시1에이 7에이제이) 뉴게이트 스트리트 81

(72) 발명자 브리스코로버트존
영국서포크주우드브리지파햄홈팜

(74) 대리인 김명신
이동기
이기성
최민서

심사청구 : 없음

(54) 데이터 분산방법

요약

본 발명은 데이터 분산에 관한 것으로, 데이터 분산시스템에서 데이터는 복수의 응용 데이터 유닛으로 분할되고, 키의 시퀀스는 조직적으로 생성되며, 소스에서 각각의 데이터 유닛을 암호화하기 위해 다른 키가 사용되고, 수신기에서 대응하는 키가 생성되고 상기 데이터에 대한 액세스를 얻기 위해 데이터 유닛을 해독하기 위해 사용되며, 상기 키를 생성하기 위해 사용되는 구성은 키의 전체 시퀀스의 본질적으로 제한된 서브셋이 하나 이상의 시드값의 선택된 조합을 전달함으로써 사용자가 사용가능하도록 이루어지는 것을 특징으로 한다.

대표도

도 9

명세서

기술분야

본 발명은 데이터에 대한 제어된 액세스를 제공하는 데이터 분산시스템에 관한 것이다. 예를 들어, 상기 시스템은 사용자의 선불에 대한 답례로 제한된 시간주기동안 멀티캐스트(multicast) 오디오 또는 비디오 프로그램 자료에 대한 액세스를 제공하기 위해 사용될 수 있다. 그러나 본 발명은 멀티캐스트 패킷 네트워크에서 결코 사용이 제한되지 않고, 또한 예를 들어 DVD(digital versatile disk)와 같은 저장매체를 포함한 다른 벌크 데이터 분산채널과 함께 사용될 수 있다.

멀티캐스팅 기술은 데이터 소스로부터 복수의 수신기로의 효율적인 데이터 분산을 허용하는 인터넷에서 사용하기 위해 개발되어 왔다. 그러나, 현존하는 멀티캐스팅 프로토콜의 효율성 및 기준화가능성(scalability)은 데이터 소스가 데이터 수신기의 정보를 전혀 필요로 하지 않는다는 사실에 부분적으로 종속된다. 그러나, 이것은 예를 들어 텔레비전 프로그램과 같은 유출되는(streamed) 비디오 데이터가 상기 프로그램을 수신하기 위해 돈을 지불한 가입자에게만 전송되도록 하기 위해서 데이터 소스와 수신자 사이에 보안관계를 설정할 필요가 있을 때 문제점을 나타낸다.

일반적으로, 그러한 보안관계는 데이터 소스에서 데이터를 암호화한 후 데이터를 해독하기 위해 필요한 키에 대한 사용자의 액세스를 제어함으로써 설정될 수 있다. 한가지 간단한 방법은 데이터를 암호화하기 위해 싱글 세션 키를 이용하는 것이다. 상기 세션 키는 새로운 사용자가 데이터를 액세스하고자 할 때까지, 또는 현존 사용자중의 한 사용자가 제외될 때까지 변화되지 않고 유지된다. 그 시점에서, 새로운 세션 키가 요구되고, 그 키는 모든 사용자에게 분산되어야 한다. 키 분산구조의 효율성이 키의 계층구조를 이용한 범위(그 일부만이 특정 고객을 제외 또는 포함하기 위해 변경될 필요가 있을 수도 있다)까지 개선될 수 있더라도, 그러한 구조에서는 그룹에 새로운 고객 참여 또는 이적(leaving)과 관련된 상당한 전송 오버헤드가 여전히 불가피하게 존재하게 된다.

본 출원인의 공동 계류중인 국제특허출원 제PCT/GB98/03753호(BT case: A25728/WO)에 기재된 대안적인 방법에서, 데이서 소스에서의 데이터는 일련의 ADU(application data unit)로 분할되고, 각각의 ADU에 대해 다른 키가 사용된다. 상기 키는 최초 시드값(seed value)으로부터 조직적으로 시퀀스로서 생성된다. 상기 시드값은 또한 각각의 고객 단말기에서 보안모듈로 전달되고, 상기 보안모듈은 최종 사용자에 대한 키의 사용가능성을 제어한다.

본 발명의 한 측면에 따르면,

(a) 키의 시퀀스중의 한 키를 이용하여 복수의 데이터 유닛을 암호화하는 단계;

(b) 상기 암호화된 데이터 유닛을 복수의 사용자 단말기로 전달하는 단계;

(c) 적어도 하나의 시드값을 하나의 사용자 단말기로 전달하는 단계;

(d) 상기 사용자 단말기에 전달된 시드값의 수보다 많은 수의 키 시퀀스를 시드값(들)으로부터 생성하는 단계; 및

(e) 상기 키 시퀀스를 이용하여 상기 사용자 단말기에서 데이터 유닛을 해독하는 단계를 포함하고,

단계 (d)에서, 단계 (a)의 키 시퀀스의 임의로 이중 제한된 부분을 구성하는 키 시퀀스가 생성되며, 상기 부분의 하위 및 상위 경계의 시퀀스내 위치는 단계 (c)에서 전달된 적어도 하나의 시드값에 의해 결정되는 것을 특징으로 하는 데이터 분산방법이 제공된다.

본 발명은 상기한 공동 계류중인 출원서에 개시된 시스템에서와 같이 연속적인 데이터 유닛이 다른 키의 시퀀스를 이용하여 암호화되는 데이터 분산 방법을 제공한다. 그러나, 본 발명의 방법은 또한 각각의 사용자가 사용가능한 키의 시퀀스의 범위가 종래 시스템에서와 같이 잠재적으로 제한되지 않는 것은 아니지만, 이중으로 제한되는, 즉 사용자에게 사용가능한 키의 시퀀스의 시작과 끝이 미리 결정된다는 점에서 상당한 이점을 제공한다. 후술되는 바와 같이, 데이터 송신기 또는 데이터 송신기의 역할을 하는 키 발행부는 임의로 시작점 및 종료점을 결정하여 사용자에게 송신되는 시드값을 선택함으로써 사용자가 사용가능한 키 시퀀스의 길이를 결정할 수 있다. 키의 시퀀스의 임의의 원하는 부분에서, 상기 부분에 대한 액세스 및 상기 부분만에 대한 액세스를 제공할 한 세트의 시드(seed)가 존재한다. 잠재적으로 제한이 없는 키의 세트보다는 이중으로 제한된 키의 세트에 대한 액세스를 사용자에게 제공함으로써, 본 발명은 상기 키에 대한 사용자의 액세스를 제어하기 위해 각각의 고객 단말기에서 데이터 송신기의 제어하에서 보안모듈을 가질 필요가 없도록 한다.

대개, 상기 단계 (a)에서 사용된 키 시퀀스는:

(A) 하나 이상의 최초 시드값에 대해 연산하고, 더 많은 수의 중간 시드값을 생성하는 단계;

(B) 이전 단계에서 생성된 값에 대해 연산하여, 더 많은 수의 추가값을 생성하는 단계; 및

(C) 생성된 값의 수가 단계 (a)에서 필요한 키의 수 이상이 될 때까지 단계 (B)를 반복하는 단계에 의해 생성되며,

상기 중간 시드값은 상기 최초 시드값을 블라인딩(blinding)하고,

상기 추가값은 이전 단계에 의해 생성된 값을 블라인딩한다.

대개, 상기 방법은:

(Ⅰ) 한 세트의 서로 다른 블라인딩 함수의 각각의 함수를 이용하여 적어도 하나의 루트 시드값에 대해 연산하여 복수의 추가값을 생성하는 단계;

(Ⅱ) 상기 서로 다른 블라인딩 함수 세트의 각각의 함수를 이용하여 이전 단계에 의해 생성된 추가값 또는 그로부터 유도된 값에 대해 연산하는 단계;

(Ⅲ) 단계 (Ⅱ)를 반복하여, 상기 또는 각각의 반복에 의해 값의 트리에서 다음 연속층을 생성하는 단계;

(Ⅳ) 단계 (a)에서, 단계 (Ⅲ)에 의해 생성된 하나 이상의 층에서 시드 시퀀스로부터 유도된 값들을 키 시퀀스로서 이용하는 단계; 및

(Ⅴ) 단계 (c)에서, 상기 트리 바디내의 적어도 하나의 값을 사용자 단말기로 전달하여, 상기 사용자 단말기에 전달된 상기 또는 각각의 값의 트리에서의 위치로부터 데이터 유닛 해독시 사용하기 위해 사용자가 사용가능한 키 시퀀스 일부의 위치 및 범위를 결정하는 단계를 포함한다.

함수가 알려진 경우에도 입력값이 출력값으로부터 용이하게 유도될 수 없는 경우에 출력값을 생성하기 위해 입력값에 대해 연산되는 함수를 블라인딩 함수라고 한다. 상기 블라인딩 함수는 예를 들어 MD5(message digest number 5)와 같은 암호 해시 함수를 포함하기도 한다.

본 발명자는 사용자가 사용가능한 시퀀스 부분의 위치 및 범위에 대한 용이한 제어를 촉진하는 동안 일련의 키를 계통적으로 생성하는 특별히 효과적인 방법은 한 세트의 서로 다른 블라인딩 함수를 이용하여 반복된 조작에 의해 값의 트리를 생성하는 것이라는 것을 발견했다. 좀더 상세하게 후술된 예시에서, 한쌍의 대칭적인 블라인딩 함수로부터 형성된 2진 트리가 사용된다. 해시 함수에 따른 우측 회전 이동이 한 함수이고, 해시 함수에 따른 좌측 회전 이동이 다른 함수이다. 그러나 이러한 본 발명의 특징은 2진 트리를 이용하는 것이 제한되지 않지만, 3진 또는 더 높은 차수 트리를 이용하여 구현될 수도 있다.

대개, 단계 (c)에서, 통신네트워크를 통해 고객 단말기로 시드값이 전달되고, 이러한 경우 대개 상기 시드값은 복수의 키 관리노드로부터 고객 단말기로 전달되고, 여기서 노드들은 서로 다른 각각의 위치에서 네트워크와 연결된다.

적절한 실시예에서 본 발명의 다른 이점은 인코딩된 데이터에 대한 액세스를 제공하기 위해 시드값을 분산하는 프로세스가 일부 또는 전부가 데이터 소스로부터 원격일 수도 있는 서로 다른 위치에서 복수의 키 관리노드에 의존될 수 있다는 것이다. 이러한 방법에서, 데이터 제어시스템은 많은 수의 수신기와 함께 사용하기 위해 용이하게 기준화될 수 있다.

대개, 데이터 유닛 및 시드값은 동일한 통신네트워크상에서 분산될 것이다. 그러나, 이것은 반드시 그런 것은 아니다. 예를 들어, 데이터 유닛은 DVD와 같은 벌크 데이터 저장매체상에 분산될 수도 있고, 상기 시드값은 인터넷을 통해 고객 온라인으로 순차 전달된다. 이러한 예시는 단지 설명을 위해 제공된 것이고, 많은 다른 구성이 적용될 수 있다는 것이 명백할 것이다.

전체 키 시퀀스의 특정 하위 범위에 대한 키를 구성하기 위해 임의의 수신기에 의해 요구되는 시드는 각각의 시드를 묵시적으로 식별하는 순서대로 전달된다. 이러한 경우, 상기 시드의 인덱스는 요구되는 최소 및 최대 값 및 각각의 시드의 인덱스 번호를 밖으로 나열하지 않고 시드를 전달하는 선배치(pre-arranged) 순서의 정보로부터 추론된다. 대개, 각각의 암호화된 데이터 유닛은 시퀀스내 키가 상기 데이터 유닛을 해독하기 위해 사용되어야 하는 임의의 수신기로 식별될 암호화되지 않은 인덱스 번호를 전달한다.

본 발명의 다른 측면에 따르면,

(a) 하나 이상의 블라인딩 함수를 이용하여 적어도 하나의 루트 시드값에 대해 연산하여, 복수의 추가값을 생성하는 단계;

(b) 하나 이상의 블라인딩 함수를 이용하여 이전 단계에 의해 생성된 추가값 또는 그로부터 유도된 값에 대해 연산하는 단계;

(c) 단계 (b)를 반복하여 상기 또는 각각의 반복에 의해 값의 트리내 다음 연속하는 층을 생성하는 단계;

(d) 단계 (c)에 의해 생성된 하나 이상의 층으로부터 유도된 키 값의 시퀀스를 이용하여 복수의 데이터 유닛을 암호화하는 단계를 포함하는 것을 특징으로 하는 분산용 데이터 암호화 방법이 제공된다.

본 발명의 다른 측면에 따르면, 복수의 데이터 유닛을 키 시퀀스 중 한 키를 이용하여 각각 암호화하는 단계, 및 상기 암호화된 데이터 유닛을 복수의 사용자 단말기로 전달하는 단계를 포함하는 데이터 분산방법이 제공되고, 상기 키 시퀀스는 생성되어 키 구성 알고리즘에 따라 응용 데이터 유닛에 할당되고, 사용중에 수신기가 임의의 데이터 송신기(들)에 대한 참조없이 키 관리자로부터 상기 데이터의 임의의 부분에 대한 액세스를 위한 키를 획득할 수 있도록 상기 키 구성 알고리즘의 사본이 복수의 키 관리자에게 분산된다.

이러한 본 발명의 측면은 키 관리가 복수의 키 관리노드로 위임될 수 있는 데이터 분산에 대한 방법을 제공하여, 상기 시스템이 많은 수의 사용자를 포함하도록 스케일되도록 한다.

본 발명의 다른 측면에 따르면, 상기 측면에 따른 방법에서 사용하기 위해 암호화된 복수의 데이터 유닛을 포함하는 데이터 캐리어가 제공된다. 상기 데이터 캐리어는 예를 들어 DVD 디스크, 컴퓨터 메모리 영역, 또는 데이터 유닛으로 인코딩된 데이터 전송신호와 같은 데이터 저장매체가 될 수 있다.

본 발명은 또한 본 발명에서 사용하기 위한 고객 단말기, 데이터 서버, 및 키 관리자 및 상기 디바이스 및 상기 디바이스를 포함한 네트워크의 사용방법을 포함한다.

지금부터 첨부한 도면을 참조하여 본 발명을 실시하는 시스템을 좀더 상세하게 설명할 것이다.

도 1은 본 발명을 실시하는 네트워크의 개략도,

도 2는 도 1의 네트워크에서 이용하기 위한 고객 단말기의 구조를 나타내는 도면,

도 3은 도 1의 네트워크에서 이용하기 위한 키 관리노드의 구조를 나타내는 도면,

도 4는 도 1의 네트워크에서 이용하기 위한 데이터 송신기의 구조를 나타내는 도면,

도 5는 도 1의 네트워크상에 전송된 데이터 패킷의 포맷을 나타내는 도면,

도 6은 키 관리노드를 통한 키의 분산을 나타내는 도면,

도 7은 양방향 해시 체인을 나타내는 도면,

도 8은 2개 블라인딩 함수의 생성을 나타내는 도면,

도 9는 2진 해시 트리를 나타내는 도면,

도 10은 연속적인 2진 해시 트리를 나타내는 도면,

도 11a-11b는 하이브리드 2진 해시 체인트리의 요소를 나타내는 도면,

도 12는 하이브리드 2진 해시 체인트리를 나타내는 도면,

도 13은 하이브리드 2진 해시 체인트리의 구성을 나타내는 도면,

도 14는 2진 해시 체인트리의 성장 순서를 나타내는 도면,

도 15는 연속적인 2진 해시 체인트리 하이브리드를 나타내는 도면,

도 16a-16b는 제 2의 2진 해시트리의 요소를 나타내는 도면,

도 17은 BHC-T내 노출 및 블라인딩 시드쌍을 나타내는 도면,

도 18은 상기 제 2의 2진 해시트리내 시드의 노출 및 블라인딩을 나타내는 도면,

도 19는 다차원 키 시퀀스를 나타내는 도면, 및

도 20a-20e는 공통 모델을 나타내는 도면이다.

데이터 통신시스템은 데이터 통신네트워크(2)를 통해 복수의 고객 단말기(3)와 연결된 데이터 서버(1)를 포함한다. 쉽게 설명하기 위해 소수의 고객 단말기만을 도시하였지만, 실제로 데이터 서버(1)는 많은 단말기와 동시에 통신할 수도 있다. 본 예시에서, 데이터 통신네트워크(2)는 공용 인터넷이고, 노드(4)에 의해 상호연결된 복수의 서브네트워크(2a-2c)로부터 형성된다. 서브네트워크 및 관련 라우터(router)는 IP(Internet Protocol) 멀티캐스팅을 지원한다.

본 예시에서, 데이터 서버(1)는 비디오 서버이다. 상기 데이터 서버는 대량기억장치(1a)로부터 비디오 데이터스트림을 판독하고, MPEG2와 같은 적절한 압축 알고리즘을 이용하여 데이터를 압축한다. 그 후, 데이터 서버(1)내 암호화 모듈은 압축된 비디오 데이터스트림을 응용 데이터 유닛(ADU; application data unit)으로 분할한다. 예를 들어, 각각의 ADU는 비디오 신호의 1분에 해당하는 데이터를 포함할 수도 있다. 그 후, 암호화 알고리즘은 계획적으로 생성된 키의 시퀀스를 이용하여 ADU를 암호화하기 위해 사용되고, 다른 키는 각각의 ADU를 위해 사용된다. 적당한 암호화 알고리즘에는 DES(data encryption standard)(미국 표준 FIPSPUB46)가 포함된다. 이것은 종래의 전용 키 알고리즘이다. 키의 시퀀스 생성시 사용되는 시드값은 또한 데이터 서버(1)로부터 복수의 키 관리노드로 전달된다. 키 관리노드는 서로 다른 위치에서 데이터 통신네트워크를 통해 확산된다. 고객 단말기 중 하나로부터의 요청에 응하여, 키 관리노드는 복수의 시드값을 단말기로 전달한다. 시드값 발행전에 각각의 키 관리노드는 예를 들어 관련 고객 단말기가 요청된 데이터를 액세스할 권리를 갖도록 설정되어 있는가를 점검할 수도 있다. 예를 들어, 고객은 비디오 데이터 서버(1)로부터 멀티캐스트되는 특정 영화에 대한 액세스 권한을 요청했을 수도 있다. 이 영화가 페이퍼뷰(pay-per-view) 기반으로 사용가능하다면, 키 관리노드는 고객이 비디오 데이터 서버의 오퍼레이터와 거래가 있고 영화에 대해 적절하게 선불했는가를 점검할 책임이 있다. 만일 이러한 조건이 만족된다면, 키 관리노드는 영화를 만드는 ADU를 암호화하기 위해 데이터 서버에서 사용된 키 시퀀스의 일부에 해당하는 키를 고객이 생성하도록 하기 위해 선택된 고객 시드값으로 발행한다. 후술되는 바와 같이, 키 시퀀스의 생성을 위해 사용되는 알고리즘은 원본 키 시퀀스의 임의로 제한된 부분에 대한 액세스를 제공하기 위해 시드값의 적절한 선택이 사용될 수 있도록 되어 있다.

도 2는 고객 단말기(3) 중의 하나의 주요 기능적 성분을 나타낸다. 네트워크 인터페이스(22)는 데이터 통신네트워크(2)로/로부터 ADU를 전달한다. ADU는 인터페이스(22)로부터 액세스 모듈(23)로 전달된다. 액세스 모듈(23)이 본 발명을 실시하는 시스템에서 예를 들어 스마트카드에 대해서 분리된 보안모듈내에 위치했던 종래 시스템과 대조적으로, 상기 액세스 모듈은 단순하게 고객 단말기의 주 프로세서상에서 실행되는 소프트웨어 모듈이 될 수도 있다. 액세스 모듈(23)은 해독 모듈(D), 키 생성모듈(K), 및 시드 기억장치(SS)를 포함한다. 상기 시드 기억장치는 키 관리노드로부터 수신된 시드값을 저장하고, 일련의 키를 생성하기 위해 후술되는 바와 같은 키 구성 알고리즘을 이용하여 상기 시드값을 처리한다. 상기 일련의 키는 시드 기억장치(SS)에 보유된 시드값에 의해 결정된 개시점과 종료점을 갖는다. 이러한 시퀀스로부터의 키는 해독 모듈(D)로 순차적으로 전달된다. 해독 모듈(D)은 인터페이스(22)로부터 수신된 일련의 ADU를 해독하고, 이들을 응용층 모듈(24)로 전달한다. 이것은 예를 들어 MPEG2 해독 알고리즘을 이용하여 추가 처리를 수행하고, 출력장치로 결과 데이터를 전달하며, 이 예시에서는 VDU(video display unit)(25)이다. 적절한 구성에서, 인터페이스(22)는 ISDN 모뎀에 의해 하드웨어에서, 그리고 TCT-IP(Transport Control Protocol-Internet Protocol) 스택에 의해 소프트웨어에서 구현될 수 있다.

고객 단말기는 복수의 형태 중 임의의 한 형태로 구현될 수 있다. 예를 들어, 단말기는 텔레비전과 관련하여 인터넷 액세스를 제공하기 위해 설계된 셋탑(set-top)박스, 적절한 네트워크 인터페이스, 및 지능 이동전화와 함께 개인용 컴퓨터를 포함할 수도 있다.

도 3은 도 1의 네트워크에서 사용하기 위한 키 관리노드의 한 예의 구조를 나타낸다. 상기 노드는 TCP-IP 스택을 통해 데이터 송신기 및 고객 단말기 또는 "수신기" 모두와 패킷을 통신한다. 패킷은 종래 방법에서의 공용키 암호화 알고리즘을 이용하여 SSL(secure sockets layer)(32)상에서 전달된다. 키 관리 애플리케이션(33)은 데이터 송신기로부터 시드값을 수신하고, 후술되는 방법으로 고객 단말기로 시드값을 발행한다. 키 관리 애플리케이션(33)과 관련된 데이터 기억장치(330)는 그 또는 각각의 데이터 송신기로부터 수신된 시드값을 보유한다. 사용자는 예를 들어 HTML(hypertext mark-up language) 및 CGI를 이용할 수도 있는 사용자 인터페이스(34)를 통해 고객 단말기에 대한 서버 웹페이지로 키 관리 애플리케이션과 상호작용한다.

도 4는 도 1의 네트워크에서 사용하기 위한 데이터 송신기의 한 예의 구조를 나타낸다. 데이터 애플리케이션(41)은 본 예시에서 응용 데이터 유닛으로 분할된 MPEG2 비디오 스트림을 포함하는 데이터를 출력한다. 비디오 프로그램 재료는 기억장치(410)로부터 유도된다. ADU는 액세스 모듈(42)로 전달된다. 이것은 암호화 서브모듈, 키 생성 서브모듈(K), 및 시드 기억장치(SS)를 포함한다. 서브모듈(K)은 후술된 바와 같은 키 구성 알고리즘과 관련하여 무작위로 생성된 시드값을 이용하여 키의 시퀀스를 생성한다. 시드값은 SSL(43) 및 TCP-IP 스택(44)을 통해 액세스 모듈(42)로부터 출력될 수도 있다. 암호화된 ADU 또한 TCP-IP 스택(44)을 통해 출력될 수 있다.

데이터 서버 및 키 관리노드 모두 COMPAQ Proliant ™ 서버 또는 Sun Microsystems Enterprise 5000 ™ 서버와 같은 상업적으로 이용가능한 플랫폼을 이용하여 구현될 수 있다.

도 5는 데이터 송신기에 의해 출력된 데이터 프레임 중 하나의 포맷을 나타낸다. 이것은 암호화된 ADU, 키 인덱스(k_i) 및 세션 식별자(SE)를 전달하는 데이터 페이로드(payload)를 포함한다. 키 인덱스 및 세션 식별자는 명문(明文)으로 송신된다.

일반적으로, 키 값은 ADU의 분산을 위해 사용되는 것과 동일한 통신네트워크상에서 고객 단말기로 분산될 수도 있다.

용어 "응용 데이터 유닛" (ADU; application data unit)은 상업적인 관점 또는 보안에서 유용한 데이터의 최소 유닛을 설명하기 위해 본 명세서에서 사용된다. ADU의 크기는 필요한 보안 및 응용에 다라 가변될 수 있다. 비디오 시퀀스에서 초기화 프레임 및 "P-프레임"의 관련 세트가 될 수 있거나, 또는 네트워크 게임에 대한 10분간의 액세스가 될 수도 있다. 암호화를 위해 사용되는 ADU는 다른 응용층에서 사용된 것과 다를 수도 있다. 예를 들어, 본 예시에서 ADU는 MPEG 압축 알고리즘에 의해 처리된 비디오 프레임과 다른 지속기간, 및 고객이 구입한 각각의 프로그램 아이템과 다른 지속기간을 갖는다. 상기 시스템의 성능을 향상시키기 위해, ADU는 부분적으로만 암호화되고, 나머지는 명문으로 송신될 수도 있다. ADU 크기는 컨텐트에 종속적인 스트림의 지속기간을 통해 가변될 수 있다. 만일 백만개의 수신기가 15분내에 멀티캐스트 데이터 스트림과 만나지만 ADU 크기 또한 15분이어서, 단 하나의 리-키(re-key) 상황만을 요구하는 경우, 응용 데이터 유닛의 크기는 시스템 기준화가능성의 제 1 결정인자가 된다. 지금부터 다른 키 시퀀스 구성 알고리즘을 좀더 상세하게 설명할 것이다.

송신기 분리 구조송신기 분리 구조

본 발명은 도 1의 방법에서와 같이 간단한 시간 시퀀스로 사용하는 것에 제한되지 않는다. 예를 들어, 본 발명은 큰 규모의 네트워크 게임에 적용될 수 있다.

그러한 게임에서, ADU의 경제적 가치는 시간 또는 데이터 크기와 관련되지 않지만, 완전히 애플리케이션 지정 인수에만 관련되지 않는다. 이 예시에서, 참가자는 실시간과 직접 관련되지는 않지만 게임 타임키퍼(time-keeper)에 의해 한정되고 시그낼링되는 지속기간인 '게임기간'당 요금을 낸다. 상기 게임은 많은 가상구역(virtual zone)으로 구성되고, 각각은 서로 다른 구역 제어장치에 의해 조절된다. 상기 구역 제어장치는 구역을 살아있게 하는 배경 이벤트 및 데이터를 제공한다. 그들은 구역당 하나의 멀티캐스트 어드레스로 이 암호화된 데이터를 송신하지만, 동일한 ADU 인덱스 및 키가 모든 구역에서 한번에 사용된다. 따라서, 전체 게임은 많은 멀티캐스트 어드레스상에 확산되어 있음에도 불구하고 하나의 단일 안전 멀티캐스트 세션이다. 플레이어는 현재 키를 가지고 있는 한 임의의 구역에 대한 배경 데이터를 동조시킬 수 있다. 상기 구역내에 플레이어에 의해 형성된 전경(foreground) 이벤트는 암호화되지 않지만, 상기 배경 데이터를 참조하지 않고서는 아무 의미가 없다.

도 6은 상기 게임에서의 데이터 흐름을 나타낸다. 게임 보안과 관련된 흐름 및 셋업동안이 아니라 일단 게임이 진행되면 송신되는 흐름만 도시되어 있다. 모든 플레이어가 데이터를 송신하고 있지만, 도 6에서는 암호화하는 송신기(S) - 구역 제어장치만을 도시하고 있다. 유사하게, 해독하는 수신기(R) - 게임 플레이어만이 도시되어 있다. 게임 제어장치는 게임 보안을 셋업하고, 이는 도 6에 도시되어 있지 않지만 후술된다. 키 관리 동작은 보안 웹 서버 기술을 이용하는 복수의 복제된 키 관리자(KM)에게 위임된다.

상기 보안 멀티캐스트 세션에 대한 키는 매 게임-기간(매 ADU)마다 시퀀스내에서 변경된다. 모든 암호화된 데이터는 ADU 인덱스에 의해 명문으로 앞에 있게 되고, 이는 해독을 위해 필요한 키를 인용한다. 셋업단계 이후, 게임 제어장치, 구역 제어장치 및 키 관리자는 그들이 게임의 전체 지속기간동안 사용될 키의 시퀀스를 계산할 수 있도록 하는 최초 시드를 보유한다. 대안적으로, 단계화된 셋업이 사용될 수도 있다.

게임 셋업게임 셋업

1. 게임 제어장치(도시되지 않음)는 그 아이덴티티의 확실성 그 자체를 만족시킨 후 모든 KM 및 S에게 공유된 '제어 세션 키'를 유니캐스트한다. 모든 KM뿐만 아니라 모든 S는 상기 세션 키가 고객이 입증된 보안 소켓층(SSL; secure sockets layer) 통신을 이용하는 각각의 공용키로 암호화된 각각에게 전송될 수 있도록 하기 위해 웹 서버 보안을 실행한다. 게임 제어장치는 또한 제어 메시지로 사용할 멀티캐스트 어드레스를 모든 KM 및 S에게 알려주고, 그들은 즉시 참여한다.

2. 게임 제어장치는 그후 전체 키 시퀀스를 구성하기 위해 최초 시드를 생성하고, 그들을 모든 KM 및 모든 S로 멀티캐스트하여, 제어 세션 키로 메시지를 암호화하고 관련된 대개 소수인 타겟에 적당한 신뢰할만한 멀티캐스트 프로토콜을 이용한다.

3. 멀티캐스트에 있어서 규칙적으로 반복되는 Mark Handley(UCL)의 " On Scalable Internet Multimedia Conferencing Systems" (PhD 논제(1997.11.14))에 기재되어 잇는 바와 같이 인증된 세션 디렉토리 공고로 안내된다. 인증된 공고는 가짜 지불 서버를 셋업하는 공격자가 게임 소득을 수집하는 것을 방지한다. 공고 프로토콜은 키 관리자 어드레스의 세부사항 및 게임기간 당 가격을 포함하도록 향상된다. 키 관리자는 게임기간의 현재 가격을 얻기 위해 수신기뿐만 아니라 이러한 공고도 청취한다. 상기 공고는 또한 어떤 키 시퀀스 구성이 사용중인가를 지정해야 한다.

수신기 세션 셋업, 지속기간 및 종료수신기 세션 셋업, 지속기간 및 종료

1. 세션 디렉토리내에 광고된 것을 듣고서 게임에 참여하기 위해 돈을 지불하기를 원하는 수신기는 적절한 형태를 이용하여 특정 수의 게임기간을 요청하는 KM 웹 서버와 접촉한다. 이것은 도 6에서 '유니캐스트 셋업'으로 도시되어 있다. R은 요청된 게임기간의 비용을 KM에게 지불하고, 아마도 그 신용카드 세부사항을 제공하거나, 또는 이전 게임에서 승리한 토큰 또는 일부 e-캐시 형태로 지불한다. 돌아와서, KM은 가져온 키 시퀀스의 하위 범위(sub-range)를 R이 단지 계산하도록 할 중간 시드의 한 세트를 송신한다. 다음 섹션에서 설명된 키 시퀀스 구성은 이것을 효율적으로 가능하게 만든다. 이 전부는 R이 아닌 인증을 필요로 하는 KM과의 SSL 통신상에서만 발생한다.

2. R은 가져온 중간 시드를 이용하여 관련 키를 생성한다.

3. R은 게임 애플리케이션에 의해 결정된 관련 멀티캐스트와 만나고, 그중 하나는 항상 하나의 S로부터 암호화된 배경 구역 데이터가 된다. R은 이들 메시지를 해독하기 위해 이전 단계에서 계산된 키 시퀀스를 이용하여, 나머지 게임 데이터가 의미있게 만든다.

4. 타임 키퍼가 (제어 멀티캐스트상에서) 새로운 게임기간을 시그낼링할 때마다, 모든 구역 제어장치는 그 ADU 인덱스를 인크리먼트하고, 시퀀스내 다음 키를 이용한다. 그들은 모두 동일한 ADU 인덱스를 이용한다. 각각의 R은 S로부터의 메시지내 ADU 인덱스가 인크리먼트된 것을 알리고, 시퀀스내 적절한 다음 키를 이용한다.

5. 게임기간 인덱스가 R이 가져온 시퀀스의 종단(end)에 접근할 때, 애플리케이션은 플레이어가 액세스를 못하게 되기 전에 '동전 투입' 경고를 제공한다. 필요한 키가 R이 적당하게 계산할 수 있는 범위밖에 포인트가 도달할 때까지 게임기간이 계속 인크리먼트된다. 만일 R이 추가 게임기간을 호출하지 않았다면, 게임에서 빠져야 한다.

이러한 시나리오는 키 관리자가 각각의 ADU 인덱스의 재정값 또는 일부 선배치(pre-arrangement)를 통한 각각의 ADU에 대한 액세스 정책을 아는 한 송신기가 모든 수신기 참여 또는 이적(leaving) 행위로부터 완전히 분리될 수 있는 방법을 설명한다. 키 관리자와 송신기 사이에는 통신이 전혀 필요없다. 송신기는 어떠한 수신기 행위에 관해서도 전혀 들을 필요가 없다. 만일 키 관리자가 이미 전송된 ADU를 판매하지 않으려면, 단지 송신기로부터 ADU 시퀀스 갯수의 변경 스트림과 동기화할 필요가 있다. 예시에서, 키 관리자는 멀티캐스트 데이터 그 자체를 들어서 동기화한다. 다른 시나리오에서, 명시된 동기화 신호에 의해, 또는 날짜 시간(time-of-day) 동기화에 의해 묵시적으로 순수하게 시간 기반으로 동기화될 수도 있다. 다른 시나리오(예를 들어 상업적 소프트웨어의 멀티캐스트 분산)에서, 전송 시간은 무관계일 수도 있다. 예를 들어, 수신기가 이후 어느 때에 동조할 수 있는 시퀀스의 일부에 키를 판매하면서, 전송이 규칙적으로 반복될 수 있다.

이 예시에서, 시드를 구매하기 위해 선불이 사용된다. 이것은 키 관리자가 그 고객에 관한 상태를 전혀 보유하지 않는다는 것을 보장한다. 이것은 만일 시드가 계산되었고 고객의 계좌 상태가 점검될 필요가 잇는 경우에서와 같이 중심 상태 저장소가 필요하지 않기 때문에 그들이 무한하게 복제될 수 있다는 것을 의미한다.

지금부터 키 구성의 다른 방법을 설명한다.

키 시퀀스 구성키 시퀀스 구성

후술되는 모든 시퀀스 구성에서, 다음의 기수법이 사용된다:

· b(v)는 v값을 블라인딩하는 함수를 위해 사용되는 기수법이다. 즉, 계산이 제한된 상대는 b(v)로부터 v를 찾을 수 없다. 블라인딩 또는 일방적 함수의 예로는 MD5 해시 [IETF RFC1321] 또는 표준 보안 해시 1 [NIST Sha-1]와 같은 해시 함수가 있다. 우수한 해시 함수는 일반적으로 경량인 계산 자원만을 요구한다. 해시 함수는 임의의 크기의 입력을 고정된 크기의 출력으로 축소하기 위해 설계된다. 모든 경우에서, 크기 감소 특성이 아닌 단순히 해시의 블라인딩 특성을 이용하여 이미 출력과 동일한 크기인 입력을 이용할 것이다.

· $b^h(v)$는 전부 h번, 이전 결과에 반복적으로 적용된 함수 b()를 의미한다.

· r(v)은 한 세트의 입력값을 그 자체에 사상하는 임의의 계산이 빠른 1대 1 함수이다. 원형(회전) 비트 이동은 그러한 함수의 한 예이다.

· $c(v_1,v_2,...)$는 결과 및 오퍼랜드의 하나를 제외한 전부가 제공되면 남아있는 오퍼랜드가 간단하게 추론될 수 있도록 v_1, v_2 등의 값을 결합하는 함수이다. c()는 만일 오퍼랜드의 비트들이 독립적이고 바이어스되지 않는다면 상기 결과의 비트 또한 독립적이고 바이어스되지 않도록 선택되어야 한다. XOR 함수는 그러한 조합함수의 간단한 예이다. c()는 이상적으로 XOR의 경우에서와 같이 남아있는 오퍼랜드를 간단하게 추론하기 위해 사용될 수 있는 함수가 되어야 한다, 즉: $v_1 = c(c(v_1,v_2,...),v_2,...)$.

모든 구성에 대한 공통 모델이 섹션 4.5에 소개될 것이지만, 그 자체 항목에 대한 각각의 구조 먼저 소개하는 것이 더 명확하다.

양방향 해시 체인(BHC; Bi-directional chain)

양방향 해시 체인 구성은 제한된 형태로 보안된다는 것만 입증하지만, 본 발명자는 이후 구조의 기반을 제한된 버전 형태로 설명하기를 주장한다. 제한되지 않은 형태가 사용되는 시나리오가 있을 수 있다:

1. 발신기는 2개 초기 시드값 v(0,0)과 v(0,1)를 무작위로 생성한다. 구체적인 예에서와 같이, 이들 값을 128비트 폭으로 할 것이다.

2. 발신기는 필요한 최대 키 시퀀스 길이 H를 결정한다.

3. 발신기는 동일한 길이 H의 2개 시드 체인을 생성하기 위해 각각의 시드에 대해 동일한 블라인딩 함수를 반복적으로 적용한다. 따라서 상기 값들은 v(0,0) 내지 v(H-1,0) 및 v(0,1) 내지 v(H-1,1)가 된다. 항목 H-1이 종종 나타내는 바와 같이, 간결하게 하기 위해 다른 상수 G=H-1를 소개할 것이다.

따라서, 식으로,

수학식 4.1.1

$$v(h,0)=b^h(v(0,0)); \quad v(h,1)=b^h(v(0,1))$$

4. 키 k_0를 생성하기 위해, 송신기는 체인 제로 v(0,0)로부터의 처음 시드를 체인 1 v(G,1)로부터의 마지막 시드와 결합한다.

키 k_1를 생성하기 위해, 상기 송신기는 체인 제로 v(1,0)로부터의 두번째 시드를 체인 1 v(G-1,1)로부터의 끝에서 두번째 시드와 결합하는 방식이다.

식으로 하면,

수학식 4.1.2

$$k_h=c(v(h,0),v(G-h,1))$$

엄격하게, 사용중인 스트림 암호는 128b 키를 필요로 하지 않을 수도 있어서, 최상위(최하위) 비트의 끊기에 의해 이 조합의 결과로부터 더 짧은 키, 일반적으로 64b까지가 유도될 수도 있다. 스트림 암호의 선택은 고속이고 안전한 동안은 무관하다.

5. 상기 송신기는 스트림을 멀티캐스트하고, k_0으로 ADU_0(응용 데이터 유닛 0)을, k_1로 ADU_1을 암호화하기 시작하지만 적어도 ADU 시퀀스 갯수를 명문으로 남겨둔다.

6. 만일 송신기가 키 관리를 위임하는 경우, 2개의 초기 시드값을 키 관리자에게 비공식적으로 전달해야 한다. 새로운 초기 시드쌍이 생성되고, 미리 계산된 키로 암호화된 데이터가 흘러나오는 것과 동시에 키 관리자에게 전달된다.

수신기는 다음과 같이 시퀀스의 일부를 재구성한다:

1. 수신기가 ADU_m 내지 ADU_n 액세스 권한을 부여받는 경우, 송신기(또는 키 관리자)는 시드 v(m,0) 및 v(G-n,1)를 상기 수신기로 유니캐스트한다.

2. 상기 수신기는 수학식 4.1.1을 이용하여 전송된 시드에 블라인딩 함수를 반복적으로 적용함으로써 시드 체인 v(m,0) 내지 v(n,0) 및 v(G-n,1) 내지 v(G-m,1)을 생성한다.

3. 수신기는 송신기가 한 것과 같이 수학식 4.1.2를 이용하여 키(k_m 내지 k_n)를 생성한다.

그러나, (h < m)일 경우 시드 v(h,0) 또는 (h n)일 경우 시드 v(h,1)는 송신기가 노출한 시드에 '선행하는' 블라인딩된 시드의 철저한 탐색없이 이 수신기에 의해 적당하게 알려질 수 없다. 따라서, 범위(k_n 내지 k_m) 외부의 키는 이러한 수신기에 의해 적당하게 계산될 수 없다.

4. 임의의 다른 수신기는 그 범위의 경계에서 관련 시드를 송신하여 ADU의 완전히 다른 범위에 대한 액세스가 제공될 수 있다; 첫번째 체인으로부터 '개시' 시드 및 두번째 체인으로부터 '종료' 시드.

도 7에서, 짙은 회색 배경의 시드 범위는 처음 언급된 수신기로부터 블라인딩된 것을 나타낸다. 이것은 이 수신기로부터 블라인딩되는 짙은 회색 배경의 키들을 유도한다.

따라서, 각각의 수신기는 세션당 단지 2개 시드/수신기를 송신함으로써 임의의 인접한 범위의 키에 대한 액세스가 주어질 수 있다. 좋지 못하게, 이러한 구성은 각각의 수신기가 하나의 송신기 시퀀스내에 노출된 키의 한 범위를 갖는 것에 대해서만 제한되는 한 사용이 제한된다. 만일 수신기가 다른 이후 범위 앞의 범위(즉, k_0 내지 k_1 이면 k_{G-1} 내지 k_G)에 대한 액세스 권한을 부여받는다면, 2개 사이의 모든 값(k_0 내지 k_G)을 계산할 수 있다. 이것은 시드 v(0,0), v(G-1,1), v(G-1,0) 및 v(G,1)가 노출되었을 것이지만, v(0,0) 및 v(G,1)만 전체 시퀀스를 노출하기 때문이다.

이러한 제한을 둘러싼 한가지 방법은 새로운 시드값(즉, H를 낮게 유지하는)으로 두번째 체인을 규칙적으로 재시작하고, H인 ADU내 한 수신기에 대한 서로의 2개 액세스를 허용하지 않는 것이다. 그러나, 이것은 키 관리자에서 고객 상태에 대한 보유를 필요로 한다. 고객이 가입을 연장만 할 수 있고 철회할 수 없어서 재임명하는 상업적 모델과 같이, 이러한 구조가 적절한 적소의 애플리케이션이 있을 수 있다. 그러한 경우, 이것은 매우 효율적인 구성이 된다. 이러한 제한을 둘러싼 두번째 방법은 만일 2개의 최소한으로 짧은 체인 사이의 갭을 위한 방이 있는 경우에만 2개의 만나지 않는 체인이 가능하다는 점에 유의하는 것이다. 다시 말해서, H< 4인 체인은 항상 안전할 것이다. 그러한 짧은 체인은 그다지 사용되지 않을 것 같지만, 이후에 짧은 BHC 프래그먼트로부터 혼성 구성을 설계하기 위해 이러한 특징을 이용할 것이다.

2진 해시 트리(BHT; binary hash tree) 2진 해시 트리(BHT; binary hash tree)

2진 해시 트리는 잘 알려진 2개 블라인딩 함수 $b_0()$ 및 $b_1()$을 필요로 한다. 이들을 '좌측' 및 '우측' 블라인딩 함수라고 할 것이다. 일반적으로 그들은 블라인딩 함수전에 2개의 간단한 1대 1 함수 $r_0()$ 및 $r_1()$ 중의 하나를 적용함으로써 단일 블라인딩 함수 b()로부터 구성될 수 있다. 도 8에 도시된 바와 같다.

따라서:

$b_0(s)=b(r_0(s))$; $b_1(s)=b(r_1(s))$

예를 들어, 첫번째 공지된 블라인딩 함수가 MD5 해시가 뒤따른 1비트 좌측 원형 이동이 될 수 있는 반면, 두번째 블라인딩 함수는 MD5 해시가 뒤따른 1비트 우측 원형 이동이 될 수 있다. 1 또는 공지된 단어와 연결하여 XOR를 이용하여 하나의 블라인딩 함수를 선행하는 다른 대안이 있을 수 있다. 최소이지만 동일한 양의 프로세서 자원을 소비하는 2개 함수를 선택하는 것이 모든 경우에서 부하를 균형맞추고 그렇지 않으면 프로세서 부하(load)의 레벨이 실행되는 함수의 선택을 노출하면 나타낼 숨어있는 채널에 대한 민감성을 제한함에 따라 유리해 보인다. 대신, 효율성을 위해, 예를 들어 2개의 서로 다른 초기화 벡터를 갖는 MD5와 같은 해시 함수의 2개 변형이 이용될 수 있다. 그러나, 시도 및 테스트된 알고리즘으로 간섭하는 것이 잘못 알려진 것 같다.

이러한 키 시퀀스는 다음과 같이 구성된다:

1. 송신기는 무작위로 초기 시드값 s(0,0)을 생성한다. 다시, 구체적인 예로써, 그 값을 128비트 너비로 설정할 것이다.

2. 송신기는 필요한 최대 트리 깊이 D를 결정하고, 이것은 새로운 초기 시드가 요구되기전에 최대 키 시퀀스 길이 $N_0=2^D$를 유도할 것이다.

3. 송신기는 2개의 '좌측' 및 '우측' 제 1 레벨 중간 시드값을 생성하여, 각각 '좌측' 및 '우측' 블라인딩 함수를 초기 시드에 적용한다:

$s(1,0)=b_0(s(0,0))$; $s(1,1)=b_1(s(0,0))$

송신기는 4개의 제 2 레벨 중간 시드값을 생성한다:

$s(2,0)=b_0(s(1,0))$; $s(2,1)=b_1(s(1,0))$;

$s(2,2)=b_0(s(1,1))$; $s(2,3)=b_1(s(1,1))$

이런 식으로, D레벨의 깊이까지 중간 시드값의 2진트리를 생성한다.

식으로 하면, 만일 $s_{d,i}$ 가 초기 시드 $s_{0,0}$ 아래의 d 레벨인 중간 시드라면:

수학식 4.2.1

$$s_{d,i}=b_p(s_{(d-1),i/2})$$

여기서 i가 짝수이면 p=0이고, 홀수이면 p=1이다.

4. 그 후, 키 시퀀스는 이전과 같이 트리의 리프(leaf) 또는 그 끝이 잘린 미분상에서의 시드값으로부터 구성된다.

즉, 만일 D=5, $k_0=s(5,0)$; $k_1=s(5,1)$;... $k_{31}=s(5,31)$.

식으로 하면,

수학식 4.2.2

$$k_i = s_{D,i}$$

5. 송신기는 상기 스트림을 멀티캐스팅하기 시작하고, k_0으로 ADU_0을, k_1로 ADU_1을 암호화하지만, 적어도 ADU 시퀀스 갯수는 명문으로 남겨둔다.

6. 만일 송신기가 키 관리를 위임하는 경우, 비공식적으로 초기 시드를 키 관리자에게 전달해야 한다. 새로운 초기 시드가 생성되고, 이전에 계산된 키로 암호화된 데이터 흐름과 동시에 키 관리자에게 전달된다.

수신기는 다음과 같이 시퀀스의 일부를 재구성한다:

1. 수신기가 ADU_m 내지 ADU_n까지의 액세스 권한을 부여받은 경우, 송신기(또는 키 관리자)는 (예를 들어 SSL을 이용하여) 상기 수신기에게 한 세트의 시드를 유니캐스트한다. 상기 세트는 요구된 키의 범위밖의 임의의 키를 계산할 수는 없고 요구된 키의 범위를 계산할 수 있는 트리 루트에 가장 가까운 중간 시드로 구성된다.

이들은 짝수 인덱스가 항상 '좌측' 자식(child)인 동안 홀수 인덱스는 항상 '우측' 자식인 사실을 이용하여 최소 및 최대 시드의 인덱스(i)를 테스트함으로써 식별된다. 테스트는 트리의 각각의 층에서 수행되고, 리프로부터 시작하여 윗방향으로 진행된다. '우측' 최소값 또는 '좌측' 최대값은 항상 레벨을 이동하기전에 노출될 필요가 있다. 만일 시드가 노출되면, 상기 인덱스는 한 시드만큼 안으로 이동하여, 층을 옮기기전에 최소값 및 최대값이 항상 각각 짝수 및 홀수가 되도록 한다. 층을 이용하기 위해서, 최소 및 최대 인덱스는 반으로 나누어지고, 필요하다면 발올림된다. 이것은 그들 사이의 차이가 예측으로 2만큼 감소한다는 것을 보장한다. 홀수/짝수 테스트는 새로운 인덱스상에서 반복되어 이전과 같이 '우측' 최소값 또는 '좌측' 최대값을 노출한다. 상기 프로세스는 최소값과 최대값이 교차하거나 만날 때까지 계속된다. 그들은 한쪽 또는 모두가 안으로 이동한 후에 교차될 수 있다. 그들은 위로 이동한 후에 만날 수 있고, 이 경우 그들이 만나는 시드는 프로시저를 종료하기전에 노출될 필요가 있다. 이러한 프로시저는 첨부 A의 C-유사 코드로, 좀더 수식적으로 설명된다.

2. 명확하게, 각각의 수신기는 주어진 각각의 시드가 트리내에 존재하는 장소를 알 필요가 있다. 상기 시드 및 그 인덱스는 그들이 노출될 때 명확하게 쌍으로 이루질 수 있다. 대신, 필요한 대역폭을 감소시키기 위해, 프로토콜은 각각의 인덱스가 최소 및 최대 인덱스로부터 묵시적으로 계산될 수 있도록 전송되는 순서 및 상기 시드의 순서를 지정할 수도 있다. 이것은 임의의 키의 한 범위의 재생성을 허용하는 시드의 단 하나의 최소 세트가 있기 때문에 가능하다.

각각의 수신기는 그 후 상기 송신기가 키의 시퀀스(k_m 내지 k_n)를 재생성함에 따라서 이들 중간 시드에 대해 동일한 쌍의 블라인딩 함수를 반복할 수 있다.(수학식 4.2.1 및 수학식 4.2.2)

3. 임의의 다른 수신기는 중간 시드의 다른 세트를 송신함으로써 완전히 다른 범위의 ADU에 대해 액세스가 제공될 수 있다.

예를 들면, 본 발명자는 한 수신기가 k_3 내지 k_9의 키 시퀀스를 재생성하도록 하는 관련 중간 시드를 우회한다. 이러한 수신기로부터 블라인딩되어 유지되는 시드 및 키는 회색 배경으로 도시되어 있다. 물론, 4보다 큰 D의 값이 실제로 일반적이다.

각각의 층이 상기 층을 유일하게 식별하는 한 임의값 d로 지정될 수 있다는 점에 유의해야 한다. d 또는 D의 실제값에 의존하는 것은 없다. 따라서, 송신기는 트리가 위로 얼마나 멀리 연장하는가를 노출할 필요가 없어서, 보안을 개선할 수 있다.

종종 한 세션은 시작될 때 알려지지 않은 지속기간을 가질 것이다. 명확하게, D의 선택은 임의의 한 시작 포인트로부터 키 시퀀스의 최대 길이를 제한한다. 가장 간단한 작업은 단순히 새로운 초기 시드를 생성하고 만일 필요하다면 오래된 시드와 나란히 새로운 2진 해시 트리를 시작하는 것이다. 만일 d가 모든 송신기 및 수신기에 의해 공지된다면, 최대 키 인덱스 2^D를 넘는 키의 범위는 즉시 모든 부분에 명백해질 것이다. 그러한 경우, 각각의 새로운 트리에 대한 '트리 id'를 할당하고, 각각의 트리에서 시드를 따라 이것을 지정하는 것이 필요하다.

상기의 앞에서의 제약을 회피하는 또다른 방법은 D를 상수 대신에 예를 들면, $D=D_0+f(i)$와 같은 변수로 만드는 것이다. 도 10은 $D_0=4$이고, D는 매 1 M 키(key)마다 증가하는 연속적인 BHT를 도시한다. 이 예에서 M은 7이라는 고정된 값을 갖는다. 그러나, 상기 트리 개개의 노드에 대해 일반적인 시드만이 상기 트리, $S_{d,2}$ d의 멀리있는 오른쪽 줄기를 따라갈 때 이러한 복잡성을 더하는 것은 중요하지 않다. 이들중 어떤 것이라도 드러난다면, 전체 미래의 트리가 드러나게된다. 그러므로, 이 '개선'은 수신기에 대한 키의 임의의 영역과 그것이 저장하는 모두를 나타내는 것이 키관리자(key manager)에 대한 사소한 메시지에서 매우 임시적으로 새로운 초기 시드를 통과하는 송신기일 때는, 효율을 높이는 데에 사용할 수 없다. 반대로, 아무리 많은 철저한 조사가 할 가치가 있더라도 '무한한' 값의 시드의 세트를 만든다면 보안의 취약성을 드러낸다. 또 한편으로는 첫번째 작업주기(work-round)에서처럼, 정기적으로 새로운 초기 시드를 생성해야만 한다는 것은 공격에 대한 BHT의 취약성의 최고한계를 정하는 것이다.

2진 해시 체인-트리 하이브리드(Binary hash chain-tree hybrid)(BHC-T) 2진 해시 체인-트리 하이브리드(Binary hash chain-tree hybrid)(BHC-T)

상기 구조는 2진 해시 트리(BHT)가 오직 두개의 시드 길이의 이중방향 해시 체인(bi-directional hash chain)(BHCs)의 단편으로부터 만들어지기 때문에 하이브리드라고 불리워진다. 이해를 위해서는 도 11에 도시된 것처럼, BHC의 단편을 구축하기위하여 리프(leaf)방향으로 루트에서 트리를 만들면서 설명을 시작하면 된다. 이것이 이해를 용이하게 할것이다. 나중에 트리를 만드는 최상의 방법은 루트보다는 차라리 측면에서부터라는 것을 제시할 것이다.

1. 우리가 랜덤하게 생성된 s(0, 0)과 s(0, 1)이라는 두개의 초기 시드값을 가지고 있다고 가정하자. 다시 구체적인 예로서, 우리는 그 값들이 128비트의 폭을 갖는다고 가정한다.

2. 우리는 지금 동일한 블라인딩 함수를 각 시드에 적용하여 두개의 블라인딩된 시드 v(1, 0)과 v(1, 1)을 생성한다.

3. 자식 시드, s(1, 1)를 생성하기 위하여 우리는 첫번째 시드, s(0, 0)를 블라이딩된 두번째 시드 v(1, 1)과 결합한다.

자식 시드, s(1, 2)를 생성하기 위하여 우리는 두번째 시드, s(0, 1)를 블라인딩된 첫번째 시드, v(1, 0)과 결합한다.

4. 우리가 지금 세번째 초기 시드를 랜덤하게 생성하고, 그것을 블라인딩하여 v(1, 2)를 생성한다면, 우리는 그 두번째, 세번째 초기 시드와 그것들의 반대의 블라인딩된 값들을 결합하여 동일한 방법으로 두개 이상의 자식 시드, s(1, 3)과 s(1, 4)를 생성할 수 있다. 이것은 모든 부모 시드는 한쪽면에 그것의 형제(sibling)와 (근친으로) 결합할 때 두개, 다른 면에 그것의 절반의 형제와 결합할 때 두개, 네개의 자식을 생성한다는 것을 의미한다. 결과적으로, 이 구조는 그들의 부모가 각 세대에서 두배의 시드의 수를 갖기 때문에 새로운 자식 시드가 블라인딩되고 결합되면 2진 트리를 생성한다. 그러나, 상기 트리는 시드의 맨 꼭대기 열의 중앙 아래에서만이 노드를 친다(둘 이상의 시드가 이 열을 따라서 만들어진다고 가정)). 상기 트리의 경계는 맨위에서부터 만들어진다면 내부로 '시든다(wither)'(그러나 나중에 보인다).

수식으로 나타내면,

수학식 4.3.1

$$s(d,i) = c(s(d-1,i/2),v(2d-1,i/2+1)) \quad i\text{는홀수}$$
$$= c(s(d-1,i/2),v(2d-1,i/2-1)) \quad i\text{는짝수}$$

여기서

v(h, j)=b(s((h-1)/2, j))

도 11a는 두 쌍의 BHC-T 하이브리드의 부모 시드 < s(0, 0), s(0, 1)과 < s(0, 1), s(0, 2)를 도시한다. 외부 링에서 외부값이 다이어그램의 경계를 벗어나 있음에도 불구하고, 우리는 바로 중심의 부모시드, s(0, 1)의 자손에 초점을 맞추기 때문에 상기 링은 각 쌍에 대해 일반적인 부모 시드를 식별한다. 도 11b는 2진 트리가 어떻게 형성되는지 더 잘 도시하기 위해, 동일한 네개의 자식들을 낳고 절대 통신하지 않는 것처럼 보이는 블라인딩된 시드를 감추는 동일한 세개의 부모를 도시한다. 보다 낮으 다이어그램에서 고리모양의 부모 시드는 상위의 다이어그램에서 도시된 동일한 세개의 고리모양의 시드를 나타낸다. 오른쪽으로 계속되는 두개의 점선으로된 화살표는 부모 시드, s(0, 2)가 또 다른 부모가 오른쪽에 있을 때 어떻게 또 다른 두개의 자식을 만드는지를 도시한다. 각 쌍의 화살표와 결합하는 점선은 이 선 위의 두 부모가 그 아래의 두 자식들을 생성한다는 사실을 나타낸다. 우리는 나중의 다이어그램에서 간단화된 형태를 사용하여 이 구조를 나타낼 것이다.

도 2는 하이브리드 트리의 예의 한 부분을 도시한다. 2진 해시 트리와 함께, 연속하는 ADU를 암호화 하기위해 사용되는 상기 키는 상기 트리나 그것들의 단면의 파생부의 리프에서의 시드의 시퀀스이다. 상기 형태는 특정한 수신기에 대한 고리모양의 시드를 나타냄에 의해 키의 예시적인 영역 k_3 - k_9를 나타내는 방법을 도시한다.

우리는 지금 루트보다는 측면에서부터 트리를 형성하는 방법을 설명하기위해 상기 구조를 더 한층 뒤틀리도록(twist) 한다. 두 피연산함수(operand)의 XOR이 세번째 값을 생성한다면, 상기 세 값중 둘은 XOR되어 세번째 값을 생성하기 때문에 XOR함수가 선택된다는 것을 앞서 언급했었다. 이것은 도 13에서 도시되었고, 거기에서 모든 시드의 값은 도 11에서처럼 동일하다. s(0, 1)이 초기에 알려져 있지 않고, s(0, 0)과 s(1, 2)는 알려져있다면, 그때, s(0, 1)이 s(1, 1)라는 것은 이 '뒤틀림'의 특성때문에 유도된다;

s(0, 1)= c(s(1, 2), b(s(0, 0))) 그리고 그때

s(1, 1)=c(s(0, 0), b(s(0, 1)))

도 14는 상기 '측면'에서부터 어떻게 송신기가 BHC-T 하이브리드 구조를 만들수 있는지를 도시한다. 시드를 만드는 순서는 숫자를 매기는 것에 의한 순환에 의해 도시된다. 어떤 순서에 의해서건 만들어질 수 있는 시드는 모두 특별한 문자에 의해 뒤이어지는 동일한 숫자로 배치된다. 고리모양의 노드 다음의 보다 짙은 순환은 랜덤하게 생성되어야하는 시드를 나타낸다. 우리는 이것들을 일차시드라고 칭할 것이다. 이것은 다음의 고리모양의 노드까지 모든 일련의 중간의 시드의 값을 고정시킨다.

1. 상기 송신기는 랜덤하게 128비트 값의 시드 0을 생성시킨다.

2. 시드 1과 2는 그때 생성된다. 그것들은 네개의 시드의 상자의 대각선의 코너를 만들고 결과적으로 반대편 코너의 값을 설정한다. '뒤틀림' 알고리듬에 의해 3은 그때 4가 된다.:

식으로 나타내면

수학식 4.3.2

$$s(d-1,i/2) = c(s(d,i),v(2d-1,i/2+1)) \quad i\text{는홀수}$$
$$= c(s(d,i),v(2d-1,i/2-1)) \quad i\text{는짝수}$$

여기서

v(h, j) = b(s((h - 1)/2, j)).

루트 시드에 대해 d=0이면, d는 리프에서 루트 방향으로 음으로 점차 증가한다.

3. 시드(5)는 그때 생성되어야만 하며, 시드(2)와 대각선 코너의 또다른 쌍을 형성한다.

4. 이것은 반대편의 코너를 나타내고, 시드(6)는 방정식(4.3.2)에 의해 시드(7)가 된다.

5. 시드(7)과 시드(2)는 그때 4개의 또 다른 상자의 꼭대기 코너를 형성하며, 방정식(4.3.1)에 의해 시드(8a, 8b)를 설정한다.

6. 상기 패턴은 그럭저럭 시드(9)가 랜덤하게 생성된 후에 계속되었다. 이 구조의 장점은 상기 트리가 무한하게 증가할 수 있어서 미리 어떤 제한선을 결정할 필요가 없다는 것이다.

7. 상기 송신기는 상기 스트림을 멀티캐스팅하는 것을 시작하고, k_0로 ADU_0를 암호화하고 k_1로 ADU_1를 암호화하며, 적어도 ADU의 순서 번호를 명문으로 남긴다. 즉,

수학식 4.3.3

$$k_i = s(D,i)$$

여기서 D=0

8. 송신기가 키관리를 위임한다면, 그것은 비공식적으로 키관리자에 대해 1차 시드와 통신할 것이다. 새로운 1차 시드는 앞서 계산된 키로 암호화된 스트리밍 데이터에 병행하여 생성되고 키관리자와 통신할 수 있다.

수신기는 다음과 같이 시퀀스의 부분을 재구축할 수 있다.

1. 수신기가 ADU_m로부터 ADU_n까지의 엑세스를 받을 때, 송신기(또는 키관리자)는 그 수신기에 대한 시드의 세트를 유니캐스트한다. 상기 세트는 키의 필요한 영역의 연산을 할 수 있도록 하는 트리에서의 중간 시드의 최소의 세트로 구성된다.

이것은 인덱스를 BHT에 대해 유사하지만 미러드(mirrored) 방식으로 최소 최대 시드의 인덱스 i를 테스트함으로써 식별된다. '왼쪽'의 최소와 '오른쪽'의 최대는 항상 단계를 높이기전에 나타낼 필요가 있다. 시드가 드러나면, 상기 인덱스는 한 계층을 올라가기 전에 한 시드에 의해 내적으로 변화되어 최소 최대 인덱스는 각각 항상 홀수이고 짝수가 된다. 한 계층을 올라가기 위해, 최소 최대 인덱스는 필요하다면 이등분되고 끝수를 잘라버린다. 이것은 그것들 사이의 차이가 예상할 수 있게 하나씩 감소하도록 보장한다. 상기 홀/짝수 테스트는 새로운 인덱스에서 반복된다. 상기 프로세스는 최소 최대가 둘 또는 셋으로 분리될때까지 계속된다. 그것들이 둘로 분리되면, 그것들은 그들 사이의 시드와 함께 나타난다. 셋으로 분리되면, 그것들은 최소가 짝수일때 그들 사이의 양쪽 시드와 함께 나타난다. 홀수이면, 한층 더 올라갈 가치가 있으며 그때 아무것도 드러나지 않고 일련의 과정이 한번 더 허용될 것이다. 테스트를 시작하기전에, 예외적인 초기조건이 테스트되며, 거기에서 요구되는 영역은 두개의 폭보다는 작다.

상기 프로세스는 첨부 자료 B의 C 코드에서 보다 명시적으로 설명될 것이다.

2. 명백히 각 수신기는 주어진 각 시드가 트리의 어디에 속하는 지를 알 필요가 있다. 상기 시드와 그들의 인덱스는 그것들이 드러날때 명시적으로 짝을 이룰 수 있을 것이다. 대안적으로 필요로하는 대역폭을 줄이기 위해, 프로토콜은 시드가 전송되어 각 인덱스가 최소 최대 인덱스와 상기 시드의 순서(order)로부터 반드시 연산될 수 있도록 하는 순서를 지정한다. 예를 들면, 첨부 자료 B에서의 알고리듬은 항상 같은 영역의 키에 대한 같은 순서에서 동일한 시드를 나타낸다.

3. 각 수신기는 그때 송신기가 키의 시퀀스 k_m - k_n를 재생산할 때 이 중간의 시드에 대한 동일한 쌍의 블라인딩과 결합 함수를 반복할 수 있다(수학식 4.3.1, 4.3.2와 4.3.3).

4. 어떤 다른 수신기도 중간 시드의 다른 세트를 수신함에 의해 ADU의 완전히 다른 영역에 대한 접근이 주어질 수 있다.

BHC-T는 측면에서부터 만들어질 수 있기 때문에, 알려지지않은 기간의 세션에 대해서 이상적이다. 새로운 중간루트 시드의 연속적인 랜덤한 생성은 상기 시퀀스의 연속적인 연산을 공격하고 용인하는 취약성을 제한한다. 취약성을 훨씬 더 제한하기 위해, 송신기는 시퀀스에서의 특정한 미래지점(future point)을 초과하는 키를 연산하는 어떠한 수신기의 능력도 거부하기 위하여 미래시드의 생성을 지연할 수 있다. 이것은 시드-공간의 맹목적인 힘의 탐색(brute force search)에 대해 유효한 시간을 제한할 것이다. 그럼에도 불구하고 측면에서부터 트리를 만드는 것은 키의 수를 지수적으로 증가하는 개개의 새로운 루트 시드(그리고 결과적으로 그 시드에 대한 공격의 값)에 대해 의존적이 되도록 한다.

루트 시드의 값은 리프(leaf)의 레벨이 되도록 정의되는 레벨을 정기적으로 증가시키고, 그것을 M 키의 각 시퀀스 이후(첫번째는 제외)의 루트에 대해 보다 근접한 층으로 이동시킴으로써 한정될 수 있다(bound).

수식으로 나타내면 이것은 식(4.3.3)을 다음으로 대신하게 한다.

수학식 4.3.4

$$k_i = s(-i/M, i) \quad i < M,$$
$$k_i = s(1-i/M, i) \quad i = M$$

이것은 도 15에서 M=8로 도시된다. 물론 실제로 M은 모든 적당한 길이의 수신기의 세션이 트리의 왼쪽 꼭대기의 노드를 건드리지 않고 효과적으로 설명될 수 있도록 하기 위해서 약간 더 크다.

우리는 BHC가 H< 4일 때 내재적으로 안전하다는 것을 앞에서 설명하였다. 상기 BHC의 단편은 H=2를 노드는 체인-트리의 하이브리드에서 사용하였고, 그것은 하이브리드 스키마의 보안성을 보장하였다. 이것은 또한 2진 체인 트리 하이브리드가 보안을 구성하지 않고 길이 3(H=3)의 체인 단편으로부터 구축될 수 있다는 것을 제안한다. 이 경우에, 각 부모시드는 그것의 형제와 절반의 형제(half-sibling)와 쌍을 이룰때 여섯의 자식을 만들고, 각 레벨에서 트리의 폭에서의 세배의 성장을 한다(삼원의 트리 BHC3-T). 이 구조는 도 20e에 도시되지만 전체분석은 앞으로 할 것이다. 조금만 더 복잡해진다면, BHC-T보다 효율적이 되는 것이 가능하다.

2진 해시 트리Ⅱ(BHT2) 2진 해시 트리Ⅱ(BHT2)

우리는 지금 BHT와 BHC-T를 결합하는 구조에 기반한 그 이상의 2진 트리를 맹목적인 힘의 공격에 대하여 보안을 매우 튼튼하게 하는 방법에 접근시키고 있다. 우리는 상기 시드에 대해 동일한 노테이션을 사용하며, BHT에 관한 루트에서 d에 대한 원점과 함께, 그 리프에 접근할 때 그것의 값은 증가한다. 상기 트리의 하나의 요소가 도 16에 도시되어 있다. 우리는 이 구조에서 b_0 (),와 b_1 ()의 두개의 블라인딩 함수를 사용하고, 그것을 BHT의 경우처럼 우리는 각각 '왼쪽'과 '오른쪽'이라고 부른다.

1. 우리가 두개의 랜덤하게 생성된 초기 시드 값 s(0, 0)과 s(0, 1)을 가지고 있다고 가정한다. 다시 구체적인 예를 들면, 우리는 그 값을 128비트의 범위로 취한다.

2. 송신기는 필요로하는 최대 트리 깊이 D를 결정한다,

우리는 이 개개의 초기 시드로부터 두 개의 블라인딩된 값을 만드는데, 개개의 블라인딩 함수는 다음과 같다.

$v(1, 0)=b_0(s(0, 0)$; $v(1, 1)=b_1(s(0, 0))$;

$v(1, 2)=b_0(s(0, 1)$; $v(1, 3)=b_1(s(0, 1))$.

3. 자식 시드 s(1, 1)를 만들기 위해, 우리는 두개의 왼쪽의 블라인딩된 시드 v(1, 1)과 v(1, 2)를 결합한다.

자식 시드 s(1, 2)를 만들기 위해, 우리는 오른쪽의 블라인딩된 시드v(1, 1)과 v(1, 3)를 결합한다.

4. 우리가 지금 세번째 초기 시드 s(0, 2)를 랜덤하게 생성한다면, 우리는 두 개의 자식 시드 s(1, 3)와 s(1, 4)를 더 추가하여 생성하는 동일한 방법으로 두번째와 세번째 초기 시드를 결합할 수 있을 것이다. BHC-T 하이브리드처럼, 이것은 모든 부모 시드가 두개의 자식으로 하여금 우리가 2진 트리를 만들 수 있도록 하지만, 내부로는 '쇠퇴하는(withering)' 경계를 가진다. 사실, 층 d가 n_d시드를 포함한다면, $n_{(d+1)}=2n_d-2$이다. 둘 이상의 초기 시드가 사용되는 한, 상기 트리는 2진 트리가 되기 쉽다.

식으로 나타내면.

수학식 4.4.1

$$s(d,i)=c(v(2d-1,i/2),v(2d-1,i/2+1)) \quad i\text{는홀수}$$
$$=c(v(2d-1,i/2),v(2d-1,i/2-1)) \quad i\text{는짝수}$$

여기서 $v(h, j)=b(s((h-1)/2, j))$.

5. 키 시퀀스는 그때 트리의 리프를 가로지르는 시드 값으로부터 구축된다.

수식으로 나타내면,

수학식 4.4.2

$$k_i=s_{D,}$$

6. 송신기는 상기 스트림을 k_0로 ADU_0를 암호화하고, k_1로 ADU_1를 암호화하면서, 멀티캐스팅을 시작하고, 적어도 ADU 시퀀스의 번호를 명확하게 남긴다.

도 16은 BHT2의 두 부모 시드 쌍 < s(0, 0), s(1, 0)과 < s(0, 1), s(0, 2)을 도시한다. 상기 링은 상기 형상의 두부분 a)와 b)의 각 쌍에 대해 BHC-T 하이브리드를 도시하기 위해 사용되는 것과 정확하게 같은 방식으로 일반적인 부모시드를 식별한다. 앞에서처럼, 도 16b)는 BHT2로 만들어진 시드의 트리가 어떻게 나타는지 도시하며, 명확하게 하기위해 시야로부터 중간의 블라인딩된 값을 감춘다. 이 내부값이 감춰지면, 그 결과인 BHT2는 도 4.3.2에서 BHC-T 하이브리드에 대해 동일하게 보인다.

키의 범위를 나타내기 위해 나타나는 시드를 연산하기 위한 알고리듬은 또한 첨부자료 B에서 BHC-T 하이브리드에 대한 그것과 동일하며, 결과적으로 도 12에서의 고리모양의 시드는 특정한 수신기에 대해 k_3 - k_9로 나타낸다.

깊이 D에 대한 세개의 초기 시드(층 0에서)로부터 BHT2의 리프를 가로지르는 키의 최대수는 2^D+2이다. 연속적인 트리가 필요하다면, 상기 키는 도 10에 도시된 연속적인 BHT에 대해 유사한 그것들을 가로지르는 레벨에 머무르는 것보다는 차라리 중간의 시드의 층으로 내려가도록 정의될 수 있다.

우리는 (4.4.1)에서 네개의 블라인딩된 값의 두개의 결합을 사용하여서 만이 2진 트리를 만드는 법을 도시해왔다.

한 번에 두개의 값을 취하면서, 여섯개의 가능한 결합을 준다;

c1=c(v(1, 0), v(1, 1))

c2=c(v(1, 2), v(1, 3))

c3=c(v(1, 0), v(1, 2))

c4=c(v(1, 1), v(1, 3))

c5=c(v(1, 0), v(1, 3))

c6=c(v(1, 1), v(1, 2))

c1과 c2는 오직 개개의 부모 시드에 대해서만 의존적이다. 그러므로, 부모만을 나타내는 것은 자식을 나타내는 것이며 어느쪽 하나만을 사용하는 것은 불가능하다. 게다가 c6=c(c3, c4, c5), c5=c(c3, c4, c6) 등등 이다. 그러므로 이 결합의 어느 세개를 나타내는 것은 반드시 네번째를 나타내는 것이다. 그럼에도 불구하고, 이 결합중에서 어느 세개는 BHT2에서 사용되는 두 개와 마찬가지로 사용될 수 있다. 결과로서 나타나는 세번째 트리(BHT3)의 분석은 다음의 작업으로 남긴다.

공통 모델(Common model) 공통 모델(Common model)

네개의 키 시퀀스 구조를 제공하면서, 우리는 공통모델을 제공하였고, 그것은 모든 스키마와 그밖의 다른 것들이 동일한 방식으로 설명되도록 한다.

우리는 두개의 좌표평면을 정의한다.

· 좌표(h, j)에 놓여지고 불연속적인 값 v를 가진 '블라인딩' 평면은 일반적으로 h좌표에서 값이 h+1에서의 값과 스키마에 의존하는 특정한 매핑을 만들기 위해 블라인딩된다.

· 좌표 (d, j)에 놓인 불연속적인 값 s를 가진 '결합' 평면은 다시 스키마에 의존하는 방식으로 블라인딩 평면으로부터의 값을 결합한 결과이다.

개개의 구조는 블라인딩 평면에서 기초적인 수학적 '분자(molecule)'로부터 만들어진다. 도 20a-20e는 h=0의 축에서 시작하여 v의 한 값에서 다음 값으로 매핑하는 블라인딩 함수를 나타내는 두꺼운 검은 화살표의 집합으로서 이 분자들을 도시한다. j축 방향에서 상기구조가 어떻게 성장하는지 도시하기 위하여, 두껍지만 매우 밝은 회색의 화살표가 다음의 분자를 완결하는 인접한 값의 블라인딩을 나타낸다. 분자는 다음의 세 상수에 의해 정의된다.

· H, 블라인딩 평면의 h축을 따라가는 한 분자의 높이

· P, 한 분자내에서 사용되는 블라인딩 함수의 수

· Q, 결합평면에서 개개의 값을 만들어내는 블라인딩 평면에서의 각각의 분자로부터 결합되는 값의 수

블라인딩 평면에서 한분자의 초기값 v는 결합평면에서의 선행된 값 s로부터 직접적으로 매핑한다.(도 20a - 20e에서 줄그어진 선의 체인으로 도시됨);

수학식 4.5.1

$$h \mod H=0\text{이면};\quad v(h,j)=s(h/H,j)$$

블라인딩 평면 분자에서의 결과로써 발생하는 값은 선행한 값으로부터 블라인딩된다(두꺼운 화살표로 도시됨);

수학식 4.5.2

$$h \mod H \neq 0\text{이면};\quad v(h,j)=b_p(v((h-1),j/P))$$

여기서 p=j mod P

블라인딩 평면 분자에서의 결과적인 최종값은 그때 결합되어 결합평면에서 다음의 값을 만든다(얇은 선으로 도시됨).

수학식 4.5.3

$$s(d,i)=c(v(h_0,j_0),\ldots v(h_q,j_q),\ldots v(h_{(Q-1)},j_{(Q-1)}))$$

여기서 h_q와 j_q는 패러미터 q의 함수로써 각 구조에 대해 정의된다.

결과적으로, d는 블라인딩 평면에서 h축을 따라서 모든 H에 대한 결합평면에서 증가한다.

표 1은 H, P 그리고 Q값과 각 구조를 정의하는 h_q와 j_q에 대한 식을 준다. 또한 그것은 일반모델을 사용하는 개개의 구조를 도시하는 형상을 참조한다.

[표 1]각 키의 시퀀스 구조를 정의하는 일반 모델의 계수

	BHT2	BHT	BHC	BHC - T	BHC3 - T
도	20a	20b	20c	20d	20e
H	2	2	H	2	3
P	2	2	1	1	1
Q	2	1	2	2	2
h_q	H_d - 1		H(d - 1) +q(H - 1)+(1 - 2q)(i mod H)		
j_q	i - 1+2q	i	i/H + q		

모든 경우에, 연속하는 구조가 바람직하지 않다면, 상기 시퀀스에서부터 구축된 키는 다음과 같이 정의된다:

수학식 4.5.4

$$k_i=s(D,i)$$

여기서 $D=\log(N_0)$

여기서 N_0는 요구되는 키의 최대값.

부수적으로, 일방향 함수(one-way function)(OFT)트리[McGrew98]는 세팅으로부터 나온다.{...?}

프로세스에 대한 절충 스토리지(trading off storage against processing) 프로세스에 대한 절충 스토리지(trading off storage against processing)

모든 MARKS 구조에서, 작은 수의 시드는 암호화 전의 송신기에서와 암호 해독 전의 수신기 양쪽에서 보다 많은 수의 키를 생성하는데 사용된다. 어느 한 경우에, 키 시퀀스에 대한 저장 능력은 제한 될 수도 있으며, 그것은 지수적으로 시드보다는 스토리지를 더욱 필요로한다. 그런 경우에, 첫번째 소수의 키는 잔여부분이 연산되도록 하는 시드를 저장하는 동안 연산될 것이다. 일반적으로 스토리지나 프로세스 시간이 최단기의 공급에 있어서인가의 경우에 따라서 스토리지가 저장될 수 있거나 동일한 연산의 반복이 회피되거나 할 수 있다.

BHC의 경우, 전체 리버스 체인(reverse chain)은 첫번째 키가 연산될 수 있기 전에 횡단되어져야만(traverse) 한다. 그러나 모든 값이 저장될 필요는 없다. 절반의 지점, 4분의 3지점의 값 등은 저장될 수 있고, 나머지는 폐기된다. 시퀀스가 이 리버스 체인으로 다시 들어갈 때, 그 다음의 값은 미리 저장된 값으로부터의 해시 체인을 다시 가동함으로써 항상 다시 연산될 수 있으며, 상기 과정에서 요구되는 것 보다 더 많은 값을 저장한다.

모든 트리의 구조의 경우를 고려하면, 첫번째 키에 대한 트리의 노드 아래의 어떤 중간 시드도 상기 세션이 시작 할 수 있기 전에 연산될 필요가 있지만, 그것들 모두가 다시 저장될 필요는 없다. 그것들은 가장 빨리 다음의 소수의 키를 연산할 필요가 있을 때, 상기 리프에 가장 근접한것이 저장되어야만 한다(캐쉬된(cached)). 상기 루트에 좀더 가까운 중간 시드가 요구될 때, 그것들은 처음에 키 관리자에 의해 전송된 시드가 결코 폐기되지 않는다면 그것들은 다시 연산될 것이다.

효율(Efficiency) 효율(Efficiency)

이미 설명되었듯이, H3을 가진 BHC는 대단히 효율적이지만, 안전하지 못하다. 그러므로 우리는 우리가 완전히 분석한 2진 트리에 기반한 구조에 대한 논의를 한정시킬 것이다. 표 5.2.1은 안전한 멀티 캐스트 세션당의 BHT, BHC-T와 BHT2의 다양한 패러미터를 보여준다. 여기서;

R, S와 KM은 각각 수신기, 송신기 그리고 키 관리자이며, 섹션 3에 정의 된 것처럼, $N(=n-m+1)$은 수신기가 요구하는 키의 영역의 길이이며, 키 공간에 랜덤하게 위치한 w_s는 시드의 크기이며(일반적으로 128b) w_h는 프로토콜 헤더의 오버헤드(overhead)의 크기이며 t_s는 시드를 블라인딩하는 프로세서 시간이다(기능을 변화 또는 결합하는 상대적으로 무시해도 좋은 순환 하나를 더함).

[표 2]

			BHT	BHC-T	BHT2
per R	(unicast 메시지 크기)/w_s - w_h또는(min storage)/w_s	min	1	3	3
		max	2(log(N+2) - 1)	2logN	2logN
		mean	O(log(N) - 1)	O(log(N))	O(log(N))
per R	(processing latency)/t_s	min	0	0	0
		max	log(N)	2(log(N) - 1)	4(log(N) - 1)
		mean	O(log(N)/2)	O(log(N) - 1)	O(2(log(N) - 1))
per R 또는 S	(processing per key)/t_s	min	1	2	4
		max	log(N)	2(log(N) - 1)	4(log(N) - 1)
		mean	2	4	8
per S 또는 KM	(min storage)/w_s		1	3	3
per S	(min random bits)/w_s				

배치된 각 수신기의 세션에 대한 유니캐스트 메시지의 크기는 각 수신기가 요구하는 스토리지의 최소량을 나타낸다. 이것은 오직 수신기가 위에서 묘사한 것같은 프로세스에 대한 스토리지를 절충(trade off)하도록 선택하기만 한다면이다. 동일한 절충은 최소의 송신기 스토리지 열에대해 사용된다. 프로세스 대기시간은 어느 수신기가 그것의 세션에 대한 유니캐스트 셋업 메시지를 받은 후에 들어온 데이터를 해독하도록 준비하는데 필요한 시간이다. 계획하지 않은 퇴거(eviction)를 제공하는 스키마에서처럼, 다른 멤버가 합류하고 떠날때에 대기시간의 손실은 없음을 명심하라. 키 당 프로세스에 대한 형상은 키의 연속적인 접근이라고 가정한다. 이경우, 어떤 세션동안(트리의 구조를 인가하여 나타나는 최소한의 세트보다) 저장하는 가장 효율적인 값은 현재 사용하는 루트에서 키까지의 노드 위의 형상이다. 키당 평균 프로세스는 리프에서의 키의 수에 의해 나누어지는 전체 트리에서의 해시 작용의 수이다. 오직 송신기(또는 다중 송신기가 있다면 그룹 컨트롤러)만이 상기 초기 시드에 대한 랜덤한 비트를 생성하도록 요구받을 수 있다. 요구되는 비트의 수는 이 초기 시드의 최소 송신기 스토리지와 명백하게 동일하다.

상기 그룹 멤버십의 크기에 의존하는 패러미터만이 수신기당의 패러미터라고 이해될 수 있다. 이 두개의(스토리지와 프로세스 대기시간) 손실(cost)은 그룹의 멤버십에 걸쳐서 분배되고 그 결과 수신기당 일정하게 된다. 오직 유니캐스트 메시지 크기만이 그룹 멤버십 크기가 선형적으로 증가하는 키 관리자에서의 손실을 일으키지만, 상기 소비는 수신기 세션당 한번으로 한정된다. 확실히, 단위당 수신기의 손실중 어떤 것도 계획되지 않은 퇴거를 인가하는 스키마에서 처럼 상기 그룹의 크기에 의존하지 않는다. 결과적으로, 제공된 모든 구조는 매우 가치가 있다.

서로 다른 것들과 스키마를 비교해보면, 아마 놀랍게도, 하이브리드 BHC-T와 BHT2는 전달 기한에서의 BHT만큼 효율적이다. 그것들은 둘다 오직 메시지를 준비하는 수신기 세션당 평균적인 하나 이상의 시드를 요구한다. N이 크다면, 단위 수신기 세션당 요구되는 키의 수에 비해 이것은 중요하지 않다. 평균으로, BHC-T는 프로세싱보다 2배를 필요로 하고, BHT2는 BHT의 4배를 필요로한다. 그러나, 우리는 보안으 개선이 비용을 투자할 가치가 있음을 이해하게 될 것이다.

BHTBHT

BHT와 함께, 상기 트리에서의 각 시드는 용량에서 그것의 자식의 2배이다. 그러므로, 상기 트리에서 현재 가장 높게 알려진 시드값을 블라인딩하는 정확한 값에 대한 시드 공간을 철저하게 조사하는 것이 유발된다. MD5 해시에 대해, 이것은 평균적으로 2^{127} 의 MD5 작용을 포함할 것이다. 정확하지 않다는 것이 알려지고 알려진 값과 충돌을 일으키는 값을 블라인딩하는 값이 가능하다(일반적으로 MD5와 모든 2^{64} 이 작동한다고 알려져있다.). 이것은 키의 영역을 만드는 시드를 사용하고 그것과 아마 암호화되는 일련의 데이터에서의 그것을 테스트함으로써 명백하게 될 것이다. 한 레벨을 깨는데에 서공하면, 다음 레벨은 다시 두배의 값이 되지만, 크랙킹을 위해 동일한 무자비한-힘의 노력을 필요로한다. 128b 입력의 MD5 해시(가능한 소스)가 선 스팍 서버-1000에서 약 4us가 든다는 것을 생각해보라. 결과적으로 2^{128} 의 MD5는 4e25년이 걸린다.

BHC-TBHC-T

BHC-T 하이브리드의 경우, 공격에 대비하는 능력은 상기 공격이 취하는 방향에 달려있다. 우리가 BHC-T의 단일한 요소(element)를 취한다면, 표 5.3.1에 보이고 도 17에 도시된 것처럼, 두 부모와 두 자식의 네개의 시드값을 가진다. 네 값중에 하나만 주어진다면, 오류를 검증하는 정보가 불충분하기 때문에 어떤 다른 값도 연산될 수 없다. 네개중 세개의 값이 주어진다면, 네번째 값은 단지 하나의 블라인딩 작용으로 언제든지 연산될 수 있다. 상기 값중 두개가 주어진다면, 상기 테이블은 어떤 두값이 주어지느냐에 따라서 다른 두값을 연산하는 것이 얼마나 어려운지를 일람표로 제시한다. 문자 'i'는 입력값을 나타내고, 셀에서의 상기 값은 입력값이 주어진 출력값의 쌍을 알아내는 것을 보장하는데 필수적인 블라인딩 함수의 작용의 수를 나타낸다. 'w'는 숫자 공간에서의 비트의 수이다(MD5에 대해 128). 도 17은 회색빛의 배경에 도시된 고리모양으로 둘러싼 입력값과 블라인딩된 값을 가지고 그림으로 동일한 정보를 도시한다.

도 17 - BHC-T에서의 나타내고 블라인딩하는 시드쌍

[표 3]

<table>
<tr><td rowspan="2">부모</td><td>s(0, 0)</td><td>i</td><td rowspan="2">$1+2^{(w+1)}$</td><td>i</td><td>$1+2^{w}$</td><td>i</td><td>2</td></tr>
<tr><td>s(0, 1)</td><td>i</td><td>$1+2^{w}$</td><td>i</td><td rowspan="2">2</td><td>i</td></tr>
<tr><td rowspan="2">자식</td><td>s(1, 1)</td><td rowspan="2">2</td><td>i</td><td>i</td><td>$1+2^{w}$</td><td rowspan="2"></td><td>i</td></tr>
<tr><td>s(1, 2)</td><td>i</td><td>$1+2^{w}$</td><td>i</td><td>i</td><td>2</td></tr>
</table>

'사각형'의 한측면의 아래의 부모와 자식이 주어진다면, 반대편 부모는 그것을 블라인딩하고 두개의 주어진 값의 XOR과 비교한 각 값을 가지고 철저하게 조사될 수 있다. 결과적으로, '측면' 공격에 대한 2^{w} 의 블라인딩 작용 후에 성공이 보장된다.

오직 두개의 자식의 값만 주어진다면, 상기 부모중 하나에 대한 철저한 조사가 약간 더 포함될 것이다. 즉, 한 부모의 값, s(0, 1)은 풀어지고, 다음 것이 맞는다면 그것은 정답이 된다.

c(s(0, 1), b(c(s(1, 1), b(s(0, 1))))) = s(1, 2)

결과적으로, '위쪽으로'의 공격에 대한 $2^{(w+1)}$ 의 블라인딩 작용이후에 성공이 보장된다.

두 개의 주어진 값에 적합할 뿐만 아니라 값의 정확한 쌍이 아닌(이중 충돌(double collision)) 두 개의 알려지지 않은 값을 알아낼 가능성은 이 구조에서는 작다. 그러한 쌍이 나타난다면, 그것들은 그것과 함께 키를 만들고 암호화된 데이터에 대한 키를 테스트 함으로써만이 테스트될 수 있다. 그러므로 이중 충돌가능성을 좀더 낮추는 것이 공격자의 임무의 복잡도를 조금 낮추게 된다.

측면 공격은 가장 높은 시드가 이미 알려진 동일한 레벨에서 기껏해야 하나의 시드만을 얻을 수 있다. 오른쪽으로의 공격은 한 값이 오른쪽으로 다음의 '상자'에서 알려질 때만이 짝수의 인덱스된 자식에 닿을 수 있다. 유사하게 왼쪽으로의 공격은 홀수로 인덱스된 자식까지 끝에 닿는다. 위로의 공격이 그때 유일하게 남는 선택이다. 한번의 성공적인 위로의 공격은 그외의 키를 주지 않지만, 그때 측면 공격이 이어지면 마지막 측면 공격의 두배의 키가 드러난다.

BHT2BHT2

공격에 대한 BHT2의 능력은 위로의 공격에 대한 능력이 훨씬 더 크도록 고안되었다는 것을 제외하고는 BHC-T 하이브리드의 그것과 동일한 형태를 가진다. BHCT를 가지고, '사각형'의 네개중 알려진 하나의 값은 다른 어떤 것도 나타내지 못한다. 그러나, BHC-T와는 달리, 세개의 값이 필수적으로 즉시 네번째 것을 주지는 않는다. 한 부모만 알려지지 않는다면, 2^{w} 의 블라인딩 작용이 그것을 알아내도록 보장하도록 요구받는다. 상기 값의 두개가 주어지면, 표 5.3.2는 어떤 두개가 주어질 것인가에 따라 다른 두개를 연산하는 것이 얼마나 어려운가를 리스트로 나타낸다. 전에서처럼, 셀에서의 값은 입력값이 주어진 출력값의 쌍을 알아내도록 보장하기 위해 필요한 블라인딩 함수 작용의 개수를 나타낸다. 도 18은 회색 배경위에 도시된 고리모양의 이력값과 블라인딩된 값으로 동일한 정보를 그림으로 도시한다.

[표 4]

부모	s(0,0)	i	2^{2w}	i	$3+2^{w}$	i	$3+2^{w}$
	s(0,0)	i		$3+2^{w}$	i	$3+2^{w}$	l
드	s(1,1)	4	i	i	$3+2^{w}$		l
	s(1,2)		i	$3+2^{w}$	i	i	$3+2^{w}$

도 18 - BHT2에서의 시드 서브세트를 나타내고 블라인딩함

부모와 '사각형'의 한측면 아래의 자식중 하나가 주어질때, 반대편 부모는 그것을 블라인딩하고 두개의 알려진 값의 XOR과 그것을 비교함으로써 테스트된 개개의 값으로 철저하게 조사될 수 있다. 결과적으로, 성공은 '측면'공격에 대하여 2^{w} 의 블라인딩 작용 이후에 보장된다. 부모와 반대편의 자식이 주어진다면 동일한 것이 적용된다.

오직 두개의 자식 값만 주어진다면, 부모에 대한 철저한 조사는 BHT2에 보다 훨씬 더 포함되도록 고안된다. 오른쪽 부모 값에서의 각각의 추측(guess)에 대해, 블라인딩되도록 남겨져야하며, 그때 첫번째 왼쪽의 블라인딩된 추측과 결합할 때 주어진 왼쪽의 자식의 값을 주는 왼쪽의 블라인딩된 값을 찾도록 철저히 조사되어야만 한다. 그러나, 이 두 부모 추측이 오른쪽으로 블라인딩되었을 때, 정확한 오른쪽 자식을 주도록 결합하지는 않을 것이다. 결과적으로, 오른쪽 부모에서의 다음의 추측은 왼쪽 부모의 블라인딩된 값 등의 철저한 조사를 하면서 결합되어야만 한다. 이것은 s(1,1)과 s(1, 2)만이 주어진 이어지는 동시에 존재하는 방정식을 푸는 것에 상당한다.

$c(b_0(s(0,0)), b_0(s(0,1))) = s(1,1)$

$c(b_1(s(0,0)), b_1(s(0,1))) = s(1,2)$

성공을 보장하기 위해 그러므로 상기 두 부모의 결합의 정방행렬의 철저한 조사를 필요로하며, 그것은 2^{2w} 블라인딩 작용이다. 자식에서 부모로의 무자비한 힘의 공격에 반한 더욱 큰 힘은 보다 진한 회색의 배경의 형상에 도시된다. 대안은 그것들을 다시 연산하도록 저장하기 위해 한 부모의 왼쪽과 오른쪽 모두 블라인딩된 값을 저장하는 것이다. 그러나 한 부모의 모든 가능한 값중 왼쪽으로 블라인딩된 인덱스되지 않은 값은 5e27TB의 저장능력보다 더 소비하며, 그것의 소비는 다른 공격수단을 보다 경제적으로 가치있게 한다!

이중 충돌에 관한 동일한 코멘트가 네개의 해시 충돌이 발견된다면, 잘못된 쌍의 값이 동시에 거의 영인 작은 가능성을 가진 이벤트를 나타내는 것을 제외한 BHC-T에 대해 하는 것 처럼 BHT2에 적용된다.

BHT2에서의 측면공격은 BHC-T에 있는 것처럼 기껏해야 하나의 '상자'의 어느 한 방법으로 한정된다. 그러므로, 의미있는 키의 수를 얻기위해, 위로의 공격이 즉시 직면해야만한다. 측면 공격에 대한 2^{w} 의 블라인딩 작용은 공격당하는 키를 합법적으로 얻는 것보다 훨씬 더 비쌀 것이다. 한번 위로의 공격을 직면한다면, 2^{w} 의 블라인딩 작용이 다른 방법을 찾기 위해 명확하게 유발된다.

일반적인 보안일반적인 보안

일반적으로, 트리를 만들기 위해 필요한 랜덤한 값이 많을 수록, 새로운 랜덤한 시드로부터 만들어진 서브트리의 한계내에서 일련의 공격을 더 많이 견제할 수 있다. 그러나, 오래동안 가동하는 세션에 대해,연속적인 트리에 관한 위에서 논의한 것처럼 보안과 연속적인 키공간의 편리함 사이에는 절충안이 있다. 랜덤하게 생성된 시드의 무작위성은 올바르게 고안되어야할 또다른 잠재적인 취약한 영역이다.

모든 MARKS구조는 유효한 그룹 멤버들 사이에서의 공모에 대해 약점이 있다. 멤버의 하위그룹이 그들 각각에게 각기 다른 키공간의 영역을 받아들이도록 한다면, 그것들은 모두 전송된 시드를 공유할 수있어서 그들 모두 그들의 독립된 키공간의 결합에 접근할 수 있을 것이다. 중재(arbitrage)는 이미 논의 되었던 다양한 멤버의 공모(collusion)이다. 이것은 한 그룹의 멤버가 전체 키 시퀀스를 사고 판매가격보다 더 낮게 그것의 부분을 판매하는 곳이고, 대부분의 키가 한 명이상의 손님에게 팔린다면 이익을 보게된다. 그룹이 아닌 멤버에 대한 방비는 워터마킹(watermarking)에 대한 6.3장에서 논의 된다.

마침내, 어떤 특정한 응용프로그램에 대한 전체적인 시스템보안은 상기 세션을 준비할 때 사용되는 보안상의 능력에 명확하게 달려있다. 위에서의 예시한 시나리오는 어드레스를 할당(address)는데 필요하고 그것들이 부합하기 위한 표준적인 암호기술을 제안하는 이슈를 묘사한다. 항상 그렇듯이, MARKS구조를 사용하는 응용프로그램의 전체적인 보안은 가장 취약한 부분 만큼 강하다.

현재 작업에서 묘사된 상기의 키관리 스키마는 그들자신을 아래에 묘사된 첨가적인 상업적 요구에 부합하는 다른 메카니즘과의 모듈의 결합을 이끄도록한다.

다중 송신기 멀티캐스트다중 송신기 멀티캐스트

다중-송신기 멀티캐스트 세션은 모든 송신기가 동일한 키 시퀀스를 사용하도록 조정하는 한 MARKS 구조를 사용하여 안전할 수 있다. 그것들은 모두 그들이 사용하는 키가 동일한 시퀀스의 부분을 사용하는한 동시에 같은 키를 사용할 필요는 없다. 수신기는 암호화된 ADU에 대한 헤더로서 ADU의 인덱스가 명백하게 전송되는 한 각 송신기가 다른 것들과 시퀀스에서 벗어날지라도 어떤 키가 사용되는지 알 수 있다.3장에서 예시된 시나리오는 이것이 상업적인 응용프로그램의 모델에 중요하다면 다중 송신기가 그들이 모두 사용하는 ADU 인덱스를 어떻게 동기화하는지를 묘사하였다.

다중 송신기 멀티캐스트에서 각 송신기가 각기 다른 키나 키 시퀀스를 사용한다면, 그것들 모두 동일한 멀티캐스트 주소를 사용함에도 불구하고 각 송신기는 각각 다른 보안 멀티캐스트 세션을 만들고 있는 것이다. 이것은2.1장에서 정의되었던 멀티캐스트 세션과 보안 멀티캐스트 세션사이의 차별로부터 이어지는 것이다. 그런 경우에 각 보안 멀티캐스트 세션은 다른 것으로부터 개별적으로 만들어지고 유지되어야만 한다. 그러나, 상환된 초기화(amortised initialisation) [Balen99]이라고 칭하여지는 것에 대한 일종의 유효범위가 있다. 즉, 뚜렷한 보안 멀티캐스트 세션은 모두 메시지를 저장하는 동일한 셋업 데이터를 사용한다. 예를 들면, 상업적인 모델은 소비자는 항상 그들이 각각 모두로부터 어떤것이든 살때, 관계된 송신기의 세트중 매 하나로부터 동일한 ADU를 사야만한다는 것이다. 그런 시나리오에서, 각 송신기는 모든 송신기에 일반적인 키의 MARKS 시퀀스를 그 송신기에 대해 특정한 긴 문장(long-term) 키와 결합해야만 한다. 상기 소비자는 키의 일반적인 영역에 대해 적절한 시드를 살 수 있고, 그때 암호를 해독하기 바라는 각 송신기에 대해 첨가적인 긴문장의 키를 얻을 수 있다.

비연속적이고 다중연속적인 키 접근(Non-sequential and multi-sequential key access) 비연속적이고 다중연속적인 키 접근(Non-sequential and multi-sequential key access)

MARKS구조는 주어진 각 수신기가 폭이 더 넓은 시퀀스의 임의의 하위영역인 키 시퀀스에 접속할 때 효율적이 되도록 고안되었지만 거기에서 상기 데이터는 순차적(SEQUENTIAL)이지 않거나 시퀀스의 임의의 어긋난 부분이 필요로한다. 결과적으로 MARKS는 실시간 멀티미디어 스트림처럼 1차원에서 자연스럽게 연속적인 데이터 스트림에 목표를 맞춘다.

그러나, 수신기는 키의 영역에 대한 접근(access)을 가지면, 명백히 연속적인 순서에서 그것들에 접근하는 것이 강제적이지는 않다. 예를들면, 수신기는 MARKS 키 시퀀스 중의 하나를 사용하는 암호화된 인터넷에 대해 멀티캐스팅하는 음악의 스트림의 하위영역을 저장할 수도 있다. 다운로드받은 트랙의 인덱스를 사용하면, 수신기는 MARKS 시퀀스로부터 순서를 끄집어낸 관련된 키를 사용하여 랜덤한 순서로 듣고자하는 트랙을 나중에 끄집어낼 수 있다.

MARKS는 또한 다중차원에서만 연속적인 데이터에 대한 접근을 금지하는 데 사용될 수있다. 그러한 응용의 일련의 예가 런던의 HIPPARCH 워크샵의 회보(1998년 6월)에 " ALF 디자인에 대한 이름 접근(A Naming Approach for ALF Design)" , M.Fuchs, C.Diot, T.Turletti, M.Hoffman에 설명되어있다. 이차원 키 시퀀스 공간이 도 19에 도시되어있다.

예를들면, 멀티캐스트 저장 인용문(quotes)에 대한 접근은 예약기간(duration of subscription)과 예약된 미래 시장의 영역 양쪽에 의해 팔린다. 각 인용은 그때 함께 XOR된 두개의 중간 키로 암호화될 필요가 있을 것이다. 결과적으로 실제로 암호화에 사용된 '마지막 키'는

$k_{i,j} = c(k^1_{0,j}, k^1_{1,j})$가 될 것이다.

한 중간 키는 i가 매분 증가하는 시퀀스 $k^1_{0,j}$ 에서 나온다. 다른 중간 키는 j가 앞으로의 인용문의 달의 수를 나타내는 시퀀스 $k^1_{1,j}$ 로부터 나온다. 일이년 안에 전문적인 트레이더(trader)는 그녀가 얼마나 오랜동안 예약하고자 하는지에 따라 $k^1_{0,i}$ 의 적절한 하위영역을 살뿐만 아니라 중간 키 $k^1_{1,12} - k^1_{1,24}$ 의 영역을 산다.

Haifa(1997년 1월)에서 " 카멜레온 - 새로운 종류의 기호" 부호매김, Ross nderson & Charalampos Manifavas(캠브리지 대학)과 같은 접근은 데이터의 스트림을 해독하는데 사용되는 키를 워터마크하도록 사용될 수 있다. 결과적으로, 임의의 MARKS 구조중 하나로 생성된 키는 중간 키로서 다루어진다. 송신기는2.2장에서 설명한 것처럼 각 중간 키를 긴-문장의 키 블럭(구체적인 예에서 512kb)과 결합하는 마지막 키의 시퀀스를 만든다. 각 수신기는 중간 키의 시퀀스에서부터 워터마크된 마지막 키의 시퀀스를 만들기 위한 동일한 블럭의 긴 문장으로 워터마크된 버전이 주어지며, 결과적으로 워터마크된 해독이 되도록 강요한다.

그러나, 이 접근은 카멜레온의 일반적인 결점으로 괴로움을 당한다. 그것은 완전히 인가되지 않은 수신기에 대해서 불충분한 인가된 수신기를 무시하는 키나 데이터에 대한 감사추적(audit trail)을 만들고, 그것은 긴 문장의 키블럭이 없는 수신기이다. 그런 경우, 상기 키나 데이터를 나타내는 배반자(traitor)는 상기 키나 데이터가 추적되면 추적될 수 있다. 그러나, 마지막 것보다는 중간 키가 어떤 때에는 여전히 유효한 긴 문장의 키블록이 주어진 수신기를 통과할 수 있다. 결과적으로 일정한 중간 키(워터마크되지 않은)에 대해 자격이 주어지지 않은 수신기는 그 자신으로 워터마크된 마지막 키를 만들 수 있고 그에 따라 기호화된 스트림(cipherstream)을 해독할 수 있다. 생성된 키와 데이터가 그 자신의 워터마크로 특징지어짐에도 불구하고, 이것은 오직 누설경로의 목표에 대한 감사추적을 할 뿐이지 소스에 대한것은 아니다. 그러므로, 이런종류의 '내부' 배반자에 대하여는 거의 대책이 없다.

MARKS구조의 특정한 경우로 돌아와서, 카멜레온이 가진 이 일반적인 결점은 중간 시드나 중간 키가 감사추적의 두려움 없이 내부를 차례로 돌수 있다는 것을 의미한다. 예를들면, 위의 네트워크 게임의 예에서, 한그룹의 플레이어는 각각 서로 다른 게임시간을 사도록 공모하고 그들 사이에서 구입할 수 있는 중간 시드를 공유할 수 있다. 실질적인 키를 만들기 위해, 그때 각 플레이어는 게임을 하는데 필요한 그 자신의 워터마크된 긴 문장의 키 블럭을 사용할 수 있다. 감사추적은 워터마크되지 않은 중간 시드를 이용하는 사람을 추적하도록 만들어지지 않는다. 그러나, 어떤 플레이어라도 최근에 게임을 하지 않아서, 그러므로 자신의 유효한 긴 문장의 키블록이 없는 사람에 대한 워터마크된 키나 데이터를 사용한다면 감사추적을 한다. 유사하게, 대신에 배반하는 수신기에 대해 추적할 수 있는 워터마크를 가지고 있으면서 플레이어중 한사람이 그들의 긴 문장의 키블록을 사용한다면 감사추적을 한다.

계획되지 않은 퇴거(Unplanned eviction) 계획되지 않은 퇴거(Unplanned eviction)

이미 지적되었듯이, MARKS 구조는 임의의 시간에서 상기 그룹의 퇴거를 허용하는데 오직 그 시간에 계획된 각 수신기 세션이 준비가 되었을 때이다. 미리 준비된 퇴거가 일반적인 경우라면, 가끔씩 계획되지 않은 퇴거가 필요한데, MARKS 스키마 중 어떤 것도 임시적인 계획되지 않은 퇴거를 허용하는 LKH++[Chang99]와 같은 다른 스키마와 결합될 수 있다. 위에서 워터마킹을 가지고 한 것처럼 이것을 달성하기 위하여, MARKS구조에 의해 생성된 키 시퀀스는 중간 키로서 다루어진다. 이것은 데이터 스트림을 해독하기위해 사용되는 마지막 키를 생성하기 위해 예를 들면 LKH++를 사용하여 분포된 그룹의 키(예를 들면 XOR된)와 결합한다. 결과적으로 MARKS 중간 키와 LKH++ 중간 키 모두 어느 한순간에 마지막 키를 생성할 필요가 있다.

사실, 어떤 수의 중간 키도(예를 들면 XOR을 사용하는) 동시에 다중 요구에 부합하도록 결합될 수 있다. 예를 들면, MARKS, LKH++ 그리고 카멜레온 중간키는 낮은 손실의 계획된 퇴거와 임시적인 계획되지 않은 퇴거와 긴 문장 그룹의 바깥으로의 누설에 대한 워터마크된 감사추적을 동시에 달성하기 위해 결합될 수 있다.

식으로 나타내면, 마지막 키는

$k_{ij...} = c(k^1_{0,i}, k^1_{1,j}, ...)$

여기서 중간키 k^1은 MARKS 구조를 사용하는 시퀀스나 워터마킹하고 다중 차원의 키 시퀀스에 대한 선행한 두 섹션에서 묘사된 것처럼 다른 수단에서 생성될 수 있다.

일반적으로, 이런 방법에서의 결합은 각각의 컴포넌트 스키마의 집합인 스토리지 손실(cost)을 가지고 총계의 스키마를 만든다. 그러나, 대부분의 퇴거가 계획된 MARKS와 LKH++를 결합하는 것은 계획되지 않은 퇴거가 실제로 요구되지 않는다면 LKH++의 다시 입력된(re-keying) 모든 메시지를 제거한다.

본발명은 결코 멀티 캐스트 데이터 네트워크의 사용에 제한되지 않는다. 두개의 다른 사용의 영역이 예로써 아래 설명된다.

가상사설망(VPN)가상사설망(VPN)

큰 회사에서는 종업원과 계약인들이 VPN을 구축함으로써 인터넷상의 어디에서건 회사의 다른 부분과 통신할 수 있도록한다. 이것을 달성하는 한가지 방법은 모든 종업원들에게 전체 회사에서 사용되는 그룹키를 주는 것이다. 결과적으로 매시간 종업원들은 회사에 들어오고 떠나고하며, 그룹 키는 바꿔야한다. 그렇지 않으면 상기 키는 종업원들이 회사에 들어오건 떠나건 간에 MARKS 구조중 하나에 의해 결정되는 시퀀스에 있어서 정기적으로 바꿔져야한다. 각각의 새로운 채용계약이 체결될때, 그 계약이 갱신될 때 까지 그들로 하여금 시퀀스에서 다음의 키를 연산하도록 시드가 각 종업원들에게 주어진다. 일찍 떠나는 종업원들은 누구나 계획되지 않은 퇴거로써 다루어진다.

디지털 다기능 디스크(DVD)디지털 다기능 디스크(DVD)

DVD는 원래 디지털 비디오 디스크를 나타내었는데, 그 능력이 상기 매체에 대해 적합하기 때문이었다. 그러나, 그것은 소프트웨어나 오디오 같은 보다 저장공간을 덜 요구하는 컨텐츠(content)를 저장하는데에 사용될 수있다. 오디오 트랙이나 소프트웨어 타이틀의 각 선택에 대해서 여러가지 부족하게 채워진 DVD를 찍어내는 대신에, 본 발명을 사용한다면, 각각의 DVD는 수백의 많은 관련된 트랙이나 타이틀로 용량을 채워서 만들어진다. 각 트랙이나 타이틀은 ADU를 조성한다. 각 ADU는 MARKS 구조중 하나를 사용하여 만들어진 시퀀스로부터 다양한 키를 가지고 암호화 될 수 있다. 이 DVD는 대량 생산되어 무료로 배포될 수도 있다(예를 들면 잡지의 커버 디스크로). 이 DVD 중 하나를 가지고 있는 사람은 누구건 간에 DVD상의 락이 걸려있지 않은(unlock) 일련의 ADU에 대한 영역의 키에 접근하는 인터넷에 대한 시드를 얻을 수 있다. MARKS는 상기 암호키가 DVD가 찍히자 마자 변화될 수는 없기 때문에 그러한 시나리오에 이상적으로 맞고, 그러한 물리적 매체를 사용하는 상업적 모델은 계획되지 않은 퇴거에 의존하는 경향은 없다. 이 스키마는 상기 키와 데이터를 워터마크하는 카멜레온과 유용하게 결합될 수 있다.

우리는 위에서 매우 큰 그룹의 키를 다루기 이한 해법을 설명하여왔다. 송신기를 완전히 흡수함으로써 인터네 멀티캐스트를 일으킨 수신기의 측정성(scalability)을 합류하고 떠나는 활동을 하는 모든 수신기로부터 보존한다. 송신기는 또한 이 수신기의 활동을 흡수하는 키 관리자에게서 완전히 흡수된다. 우리는 많은 상업 응용프로그램들이 정해진 상태가 없는 키관리자를 필요로하는 모델을 가지고 있는 것을 보아왔으며, 그런 경우에서 제한 없는 키관리자의 복사가 실현가능하다. 정형화되지 않은 키관리자의 복사된 세트중 하나가 결핍할 때, 시스터 서버에서의 작업에서의 처리에는 영향이 없고, 결과적으로 문제로부터 전반적인 시스템을 고립시키며 복원력을 개선한다. 우리는 이점을 도시하기 위해 분당 요금을 매기는 커다란 규모의 네트워크 게임의 작업한 예를 제시하였다.

이것의 이점은 수신기가 리킹(re-keying)을 구동하는 활동에 참여하거나 떠나는 것보다는 체계적인 그룹 키의 변화를 사용하여 달성되어왔다. 흡수(decoupling)는 송신기와 키관리자가 멜티캐스트 데이터 스트림에서 금전적인 가치의 유닛을 미리 배열함으로써 달성된다(금전 지금에 관한 '응용 데이터 유닛'). 체계적인 키의 변화는 그때 데이터에서 선언된 ADU 인덱스를 증가함으로서 신호로 보내질 수 있다. 이 모델을 사용하면, 수신기가 합류하거나 떠날때 송신기에게 뿐만이 아니라 다른 수신기에게도 부작용이 일어나지 않는다. 우리는 또한 멀티캐스트가 키 관리를 위해서가 아니라 데이터 전송만을 위해서 사용되도록한다. 전형적인 키 관리 솔루션은 그룹 멤버의 계획되지 않은 퇴거를 허용하는 기술의 가능성을 성공적으로 개선시켜왔지만, 최상의 기술은 메시징분야에서는 비싸다. 반대로, 우리는 계획된 퇴거의 문제에 초점을 맞추어왔다. 즉, 일련의 임의의 미래 ADU이후의 퇴거로 그때 계획되어서 수신기는 세션을 요청한다. 우리는 선불 예약에 기반한 많은 상업적 시나리오들이 계획되지 않은 퇴거를 필요로하지 않고 임의의 계획된 퇴거를 필요로 한다는 것을 가정하여왔다. 그 예들은 유료예약TV, TV나 네트워크게임에서의 시청당 요금이다.

계획된 하지만 임의의 퇴거를 달성하기 위해, 전체적으로 그룹키를 변화시키는 송신기에 의해 사용되는 키 시퀀스를 고안해왔다. 그것들은 시퀀스의 하위영역이 적은 수의 시드(각각 16B)를 나타냄으로써 재구축될 수 있도록 고안되었다. 결과로써 수신기는 상기 데이터 시퀀스의 임의의 하위영역에 대한 접근이 주어질 수 있었다. 모든 실질 적인 스키마는 O(log(N)) 시드를 시용한 개개의 수신기에 대한 N키를 나타낼 수 있다. 상기 스키마는 개개의 키를 연산하는 프로세스 로드에서 차이가 있으며, 그것은 보안에 대해 절충을 이룬다. 가장 중요한 스키마는 시작되기 위해 평균적으로 그저 O(2(log(N)-1)의 빠른 해시 공정을 요구하며, 그때 평균적으로 시퀀스에서 개개의 새로운 키를 연산하기 위해 해시를 16이상 더 필요로하고, 그것은 미리 행해질 수 있다. 가장 가벼운 스키마는 이것보다 네배 더 적은 프로세스를 요구한다.

문맥안에 이작업을 넣기 위해, 15분의 시간동안 채널을 돌리고 들르는 천만의 시청자의 10%에게 시간당 TV 요금을 매기기 위해, 최상의 대안적인 스키마(Chang 그 밖의 문서에서)는 모든 그룹 멤버에 대해 RB방식의 10개의 순서의 수천 바이트의 시간당 멀티캐스트 리키(re-key)메시지를 생성할 수도 있다. 본 작업은 약 4시간의 시청의 처음에 각 수신기에 대해 오직 한번의 몇백 바이트의 유니캐스트의 메시지만을 필요로 한다.

첨부 A - BHT를 위한 중간 시드의 최소 세트를 식별하기 위한 알고리즘

다음의 C-유사 코드 프래그먼트에서

· 함수 odd(x)는 x가 홀수인가를 테스트하고

· 함수 reveal(d,i)는 시드 $s_{d,i}$ 를 수신기에 대해 노출시킨다

```
min=m; max=n;

if (min max) error(); // min max 거절

for(d=D; d=0; d--) { // 트리의 바닥부터 작업...

// 각각의 루프에서 트리를 한 레벨씩 이동

if (min == max) { // min과 max가 수렴...

reveal(d,min); //... 그래서 서브트리의 루트를 노출...
```

```
break; //... 그리고 빠져나간다
}
if odd(min) { // 홀수 min 값이 좌측에 없다
chldren...
reveal(d,min); // ... 그래서 홀수 min 시드를 노출
min++; // 그리고 min을 우측으로 한 시드 안으로 옮김
}
if !odd(max) { // 짝수 max 값이 우측에 없다
children...
reveal(d,max); // ... 그래서 짝수 max 시드를 노출
max--; // 그리고 max를 좌측으로 한 시드 안으로 옮김
}
if (min max) break; // min과 max가 등가가 되어 빠져나감
min/=2; // min을 이등분하고...
max/=2; // max를 이등분한다...
} // 다음 라운드 루프의 레벨 상승을 위해
```

첨부 A - BHC-T를 위한 중간 시드의 최소 세트를 식별하기 위한 알고리즘

다음의 C-유사 코드 프래그먼트에서

· 함수 odd(x)는 x가 홀수인가를 테스트하고

· 함수 reveal(d,i)는 시드 $s_{d,i}$ 를 수신기에 대해 노출시킨다

```
min=m; max=n;
if (min max) error(); // min max 거절
d=0; // 트리의 바닥부터 작업
if (max <= min+1); { // min과 max가 인접/동일하기를 요청...
reveal(d,min); // ... 그래서 좌측 노출...
```

```
if (max < min) // min 과 max가 동일하지 않기를 요청...
reveal(d,max); //...역시 우측 노출...
break; //... 그리고 빠져나감
}
for(d=0; ;d++) { // 각각의 루프에서 트리를 한 레벨씩 이동
if (max < = min+3) { // min과 max가 2개 또는 3개로 분리...
if (max < min+3) { // min과 max가 2개로 분리되었다...
reveal(d,min); //... 그래서 좌측 노출...
reveal(d,max); //... 그리고 우측
reveal(d,min+1); //... 그리고 중심...
break; //... 그리고 빠져나감
} else { // min과 max가 3개로 분리되었다, 그래서..
if (!odd(min)) { // min이 짝수인 경우에만...
reveal(d,min+1) //... 좌측 중심 노출...
reveal(d,max - 1) //... 그리고 우측 중심...
break; //... 그리고 빠져나감
}
}
}
if !odd(min) { // 짝수 min값은 우측에 없다
children...
reveal(d,min); // ... 그래서 짝수 min 시드를 노출
min++; // 그리고 min을 우측으로 한 시드 안으로 옮김
}
if odd(max) { // 홀수 max값은 좌측에 없다
```

children...

reveal(d,max); //... 그래서 홀수 max 시드 노출

max- -; //max를 좌측으로 한 시드 안으로 옮김

}

min/=2; // min을 이등분하고...

max/=2; // max를 이등분한다...

} // 다음 라운드 루프의 레벨 상승을 위해

(57) 청구의 범위

청구항 1.

(a) 키의 시퀀스중의 한 키를 이용하여 복수의 데이터 유닛을 암호화하는 단계;

(b) 상기 암호화된 데이터 유닛을 복수의 사용자 단말기로 전달하는 단계;

(c) 적어도 하나의 시드값을 하나의 사용자 단말기로 전달하는 단계;

(d) 상기 사용자 단말기에 전달된 시드값의 수보다 많은 수의 키 시퀀스를 시드값(들)으로부터 생성하는 단계; 및

(e) 상기 키 시퀀스를 이용하여 상기 사용자 단말기에서 데이터 유닛을 해독하는 단계를 포함하고,

단계 (d)에서, 단계 (a)의 키 시퀀스의 임의로 이중 제한된 부분을 구성하는 키 시퀀스가 생성되며, 상기 부분의 하위 및 상위 경계의 시퀀스내 위치는 단계 (c)에서 전달된 적어도 하나의 시드값에 의해 결정되는 것을 특징으로 하는 데이터 분산방법.

청구항 2.

제 1 항에 있어서,

상기 단계 (a)에서 사용된 키 시퀀스는:

(A) 하나 이상의 최초 시드값에 대해 연산하고, 더 많은 수의 중간 시드값을 생성하는 단계;

(B) 이전 단계에서 생성된 값에 대해 연산하여, 더 많은 수의 추가값을 생성하는 단계; 및

(C) 생성된 값의 수가 단계 (a)에서 필요한 키의 수 이상이 될 때까지 단계 (B)를 반복하는 단계에 의해 생성되며,

상기 중간 시드값은 상기 최초 시드값을 블라인딩(blinding)하고,

상기 추가값은 이전 단계에 의해 생성된 값을 블라인딩하는 것을 특징으로 하는 데이터 분산방법.

청구항 3.

제 1 항 또는 제 2 항에 있어서,

단계 (d)는 복수의 서로 다른 시드값으로부터 유도된 값들을 조합하는 단계를 포함하는 것을 특징으로 하는 데이터 분산방법.

청구항 4.

제 1 항 내지 제 3 항 중 어느 한 항에 있어서,

단계 (d)는 복수의 서로 다른 블라인딩 함수 각각을 이용하여 복수의 시드값에 대해 연산하는 단계를 포함하는 것을 특징으로 하는 데이터 분산방법.

청구항 5.

제 4 항에 있어서,

(Ⅰ) 한 세트의 서로 다른 블라인딩 함수의 각각의 함수를 이용하여 적어도 하나의 루트 시드값에 대해 연산하여 복수의 추가값을 생성하는 단계;

(Ⅱ) 상기 서로 다른 블라인딩 함수 세트의 각각의 함수를 이용하여 이전 단계에 의해 생성된 추가값 또는 그로부터 유도된 값에 대해 연산하는 단계;

(Ⅲ) 단계 (Ⅱ)를 반복하여, 상기 또는 각각의 반복에 의해 값의 트리에서 다음 연속층을 생성하는 단계;

(Ⅳ) 단계 (a)에서, 단계 (Ⅲ)에 의해 생성된 하나 이상의 층에서 시드 시퀀스로부터 유도된 값들을 키 시퀀스로서 이용하는 단계; 및

(Ⅴ) 단계 (c)에서, 상기 트리 바디내의 적어도 하나의 값을 사용자 단말기로 전달하여, 상기 사용자 단말기에 전달된 상기 또는 각각의 값의 트리에서의 위치로부터 데이터 유닛 해독시 사용하기 위해 사용자가 사용가능한 키 시퀀스 일부의 위치 및 범위를 결정하는 단계를 포함하는 것을 특징으로 하는 데이터 분산방법.

청구항 6.

제 5 항에 있어서,

단계 (Ⅰ)에서,

(ⅰ) 상기 서로 다른 블라인딩 함수 세트를 이용하여 복수의 서로 다른 시드값에 대해 연산하는 단계;

(ⅱ) 상기 서로 다른 블라인딩 함수 각각에 대해서, 한 블라인딩 함수를 이용하여 상기 시드값 중 하나에 대해 연산한 결과를 동일하거나 다른 블라인딩 함수를 이용하여 각각의 시드값 중 다른 시드값에 대해 연산한 결과와 조합하여 복수의 추가값을 생성하는 단계를 포함하는 것을 특징으로 하는 데이터 분산방법.

청구항 7.

제 3 항에 있어서,

(Ⅰ) 각각의 제 1 및 제 2 블라인딩 함수 체인으로부터 유도된 제 1과 제 2 값을 조합하여, 다음 제 1 시드 또는 키를 생성하는 단계;

(Ⅱ) 상기 제 1 값의 위치 뒤에 있는 상기 제 1 체인내 위치로부터 유도된 값을 상기 제 2 값의 위치 앞에 있는 상기 제 2 체인내 위치로부터 유도된 값과 조합하여, 다음 추가 시드 또는 키 값을 생성하는 단계를 포함하고,

상기 제 1 및 제 2 블라인딩 함수 체인은 서로 다른 각각의 시드를 갖는 것을 특징으로 하는 데이터 분산방법.

청구항 8.

제 7 항에 있어서,

단계 (Ⅱ)를 반복하여 상기 제 1 체인내 이전 위치 뒤에 있는 위치 및 조합되는 상기 제 2 체인내 이전 위치 앞에 있는 위치로부터의 각각의 반복값에서 추가 키 값을 생성하는 단계를 포함하는 것을 특징으로 하는 데이터 분산방법.

청구항 9.

제 1 항 내지 제 8 항 중 어느 한 항에 있어서,

상기 시드값은 통신네트워크를 통해 사용자 단말기로 전달되는 것을 특징으로 하는 데이터 분산방법.

청구항 10.

제 9 항에 있어서,

상기 시드값은 복수의 키 관리노드로부터 고객 단말기로 전달되는 것을 특징으로 하는 데이터 분산방법.

청구항 11.

분산용 데이터를 암호화하는 방법에 있어서,

(a) 하나 이상의 블라인딩 함수를 이용하여 적어도 하나의 루트 시드값에 대해 연산하여, 복수의 추가값을 생성하는 단계;

(b) 하나 이상의 블라인딩 함수를 이용하여 이전 단계에 의해 생성된 추가값 또는 그로부터 유도된 값에 대해 연산하는 단계;

(c) 단계 (b)를 반복하여 상기 또는 각각의 반복에 의해 값의 트리내 다음 연속하는 층을 생성하는 단계;

(d) 단계 (c)에 의해 생성된 하나 이상의 층으로부터 유도된 키 값의 시퀀스를 이용하여 복수의 데이터 유닛을 암호화하는 단계를 포함하는 것을 특징으로 하는 분산용 데이터 암호화 방법.

청구항 12.

한 그룹의 사용자에게 데이터를 전달하는 방법에 있어서,

(a) 분산용 데이터를 암호화하는 단계;

(b) 그룹 멤버쉽 변경에 계통적이고 독립적으로 상기 분산용 데이터 암호화시 사용된 키를 변경하는 단계;

(c) 상기 데이터를 상기 사용자에게 전달하는 단계; 및

(d) 상기 사용자 단말기에서 상기 데이터를 해독하는 단계를 포함하고,

복수의 최초 시드값으로부터 더 많은 중간 시드값을 생성하고, 분산용 데이터 암호화시 사용된 상기 복수의 키를 상기 중간 시드값으로부터 유도하는 것을 특징으로 하는 한 그룹의 사용자에게 데이터를 전달하는 방법.

청구항 13.

제 12 항에 있어서,

상기 키의 시퀀스의 모든 가능한 서브세트(sub-set)는 시드값들 각각의 조합으로부터 유도가능한 것을 특징으로 하는 한 그룹의 사용자에게 데이터를 전달하는 방법.

청구항 14.

제 1 항 내지 제 13 항 중 어느 한 항에 있어서,

각각의 암호화된 데이터 유닛은 상기 시퀀스내의 어떤 키가 상기 데이터 유닛을 해독하기 위해 사용되어야 하는지를 식별하도록 암호화되지 않은 인덱스 번호를 임의의 수신기로 운반하는 것을 특징으로 하는 방법.

청구항 15.

제 1 항 내지 제 14 항 중 어느 한 항에 있어서,

전체 키 시퀀스 중 특정 하위 범위(sub-range)에 대한 키를 구성하기 위해 임의의 수신기에 의해 요구되는 시드는 각각의 시드를 묵시적으로 식별하는 순서대로 전달되는 것을 특징으로 하는 방법.

청구항 16.

제 1 항 내지 제 15 항 중 어느 한 항에 있어서,

복수의 데이터 송신기는 동일하거나 다른 데이터 유닛을 암호화하기 위해 서로 동일한 키 시퀀스를 이용하는 것을 특징으로 하는 방법.

청구항 17.

제 1 항 내지 제 16 항 중 어느 한 항에 있어서,

상기 시드로부터 생성된 시퀀스내 각각의 키는 상기 데이터 유닛을 암호화 또는 해독하기 위한 키 시퀀스를 생성하기 위해 다른 중간 키 또는 키 시퀀스와 조합될 중간키로서 사용되는 것을 특징으로 하는 방법.

청구항 18.

복수의 데이터 유닛을 키 시퀀스 중 한 키를 이용하여 각각 암호화하는 단계, 및 상기 암호화된 데이터 유닛을 복수의 사용자 단말기로 전달하는 단계를 포함하는 데이터 분산방법에 있어서,

상기 키 시퀀스는 키 구성 알고리즘에 따라 생성되어 응용 데이터 유닛에 할당되고, 사용중에 수신기가 임의의 데이터 송신기(들)에 대한 참조없이 키 관리자로부터 상기 데이터의 임의의 부분에 대한 액세스를 위한 키를 획득할 수 있도록 상기 키 구성 알고리즘의 사본이 복수의 키 관리자에게 분산되는 것을 특징으로 하는 데이터 분산방법.

청구항 19.

사용자 단말기를 동작하는 방법에 있어서,

a) 하나의 키 시퀀스를 이용하여 암호화된 복수의 데이터 유닛을 수신하는 단계;

b) 하나 이상의 시드값을 수신하는 단계;

c) 단계 b)에서 수신된 시드의 수보다 많은 수의 임의적으로 이중 제한된 키 시퀀스를 상기 하나 이상의 시드값으로부터 생성하는 단계; 및

d) 단계 c)에서 생성된 값 또는 그로부터 유도된 값을 이용하여 상기 응용 데이터 유닛을 해독하는 단계를 포함하는 것을 특징으로 하는 사용자 단말기 동작방법.

청구항 20.

제 18 항에 따른 방법에 의해 동작하도록 배치된 키 관리자.

청구항 21.

제 19 항에 따른 방법에 의해 동작하도록 배치된 고객 단말기.

청구항 22.

제 1 항 내지 제 19 항 중 어느 한 항에 따른 방법에 의해 동작하도록 배치된 통신네트워크.

청구항 23.

제 22 항에 있어서,

상기 데이터는 멀티캐스트 또는 방송 전송모드를 이용하여 분산되는 것을 특징으로 하는 통신네트워크.

청구항 24.

제 22 항 또는 제 23 항에 있어서,

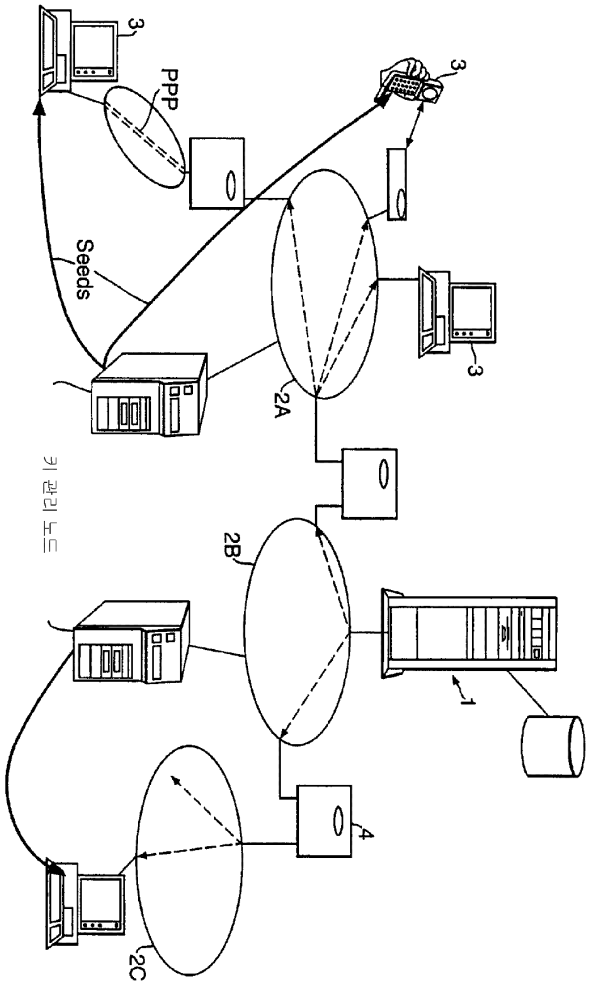
상기 네트워크는 VPN(virtual private network)를 포함하고, 데이터 해독을 위해 키의 서로 다른 하부 범위를 구성하기 위한 서로 다른 시드의 조합은 VPN에 대한 서로 다른 액세스 주기를 VPN의 멤버에게 제공하는 것을 특징으로 하는 통신네트워크.

청구항 25.

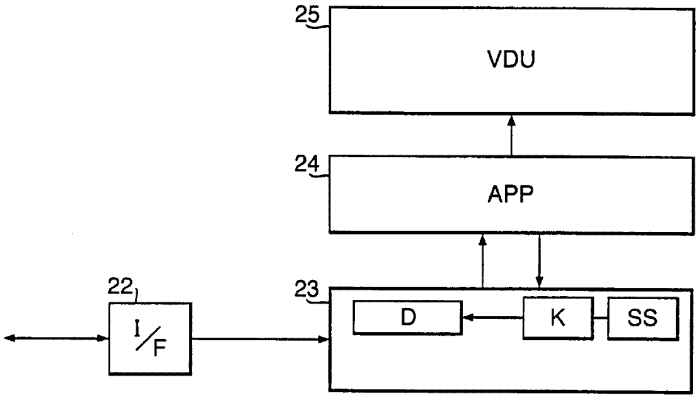
제 1 항 내지 제 19 항 중 어느 한 항에 따른 방법에서 사용하기 위해 암호화된 복수의 데이터 유닛을 포함하는 데이터 캐리어.

도면

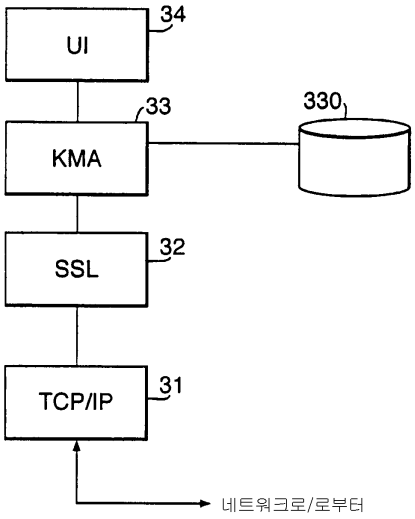
도면 1



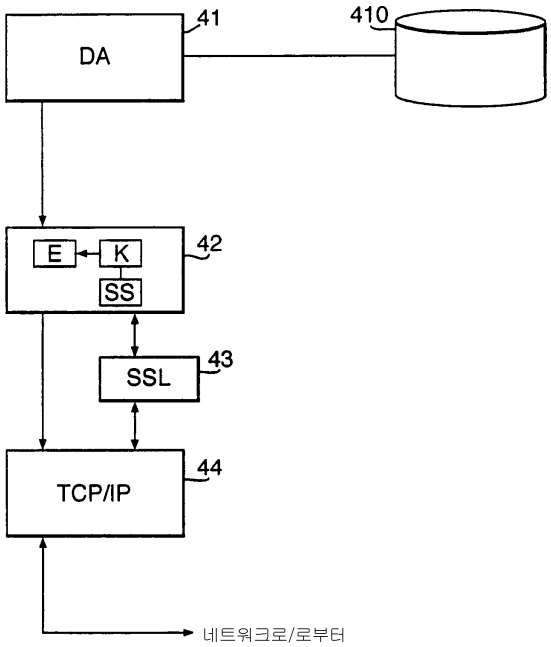
도면 2


25
VDU
24
APP
22
I/F
23
D
K
SS

도면 3

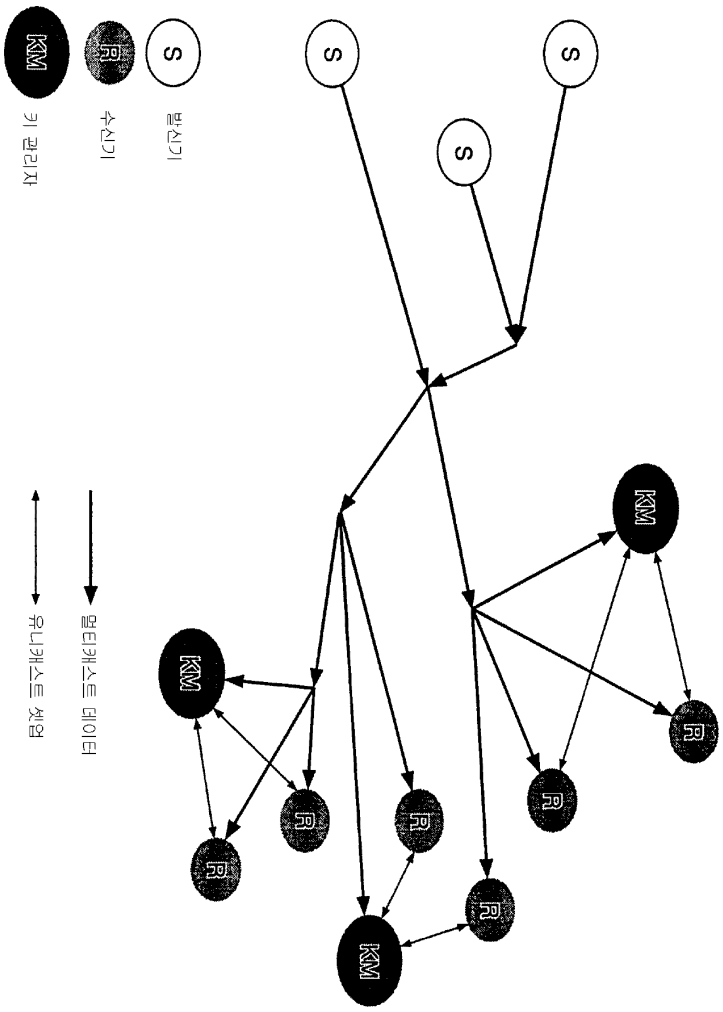

34
UI
33
330
KMA
32
SSL
31
TCP/IP
네트워크로/로부터

도면 4

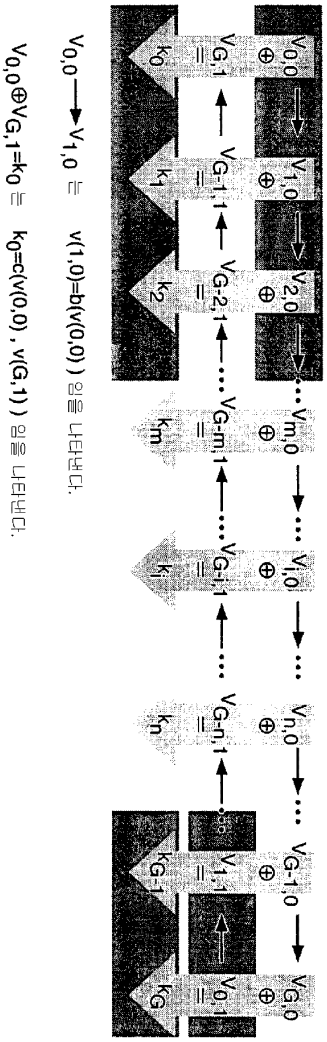


도면 5

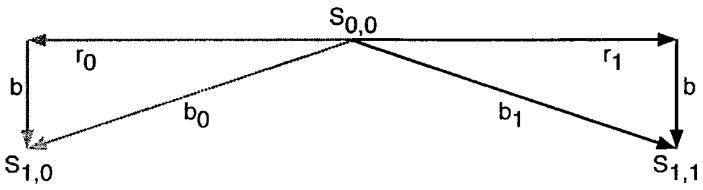
도면 6



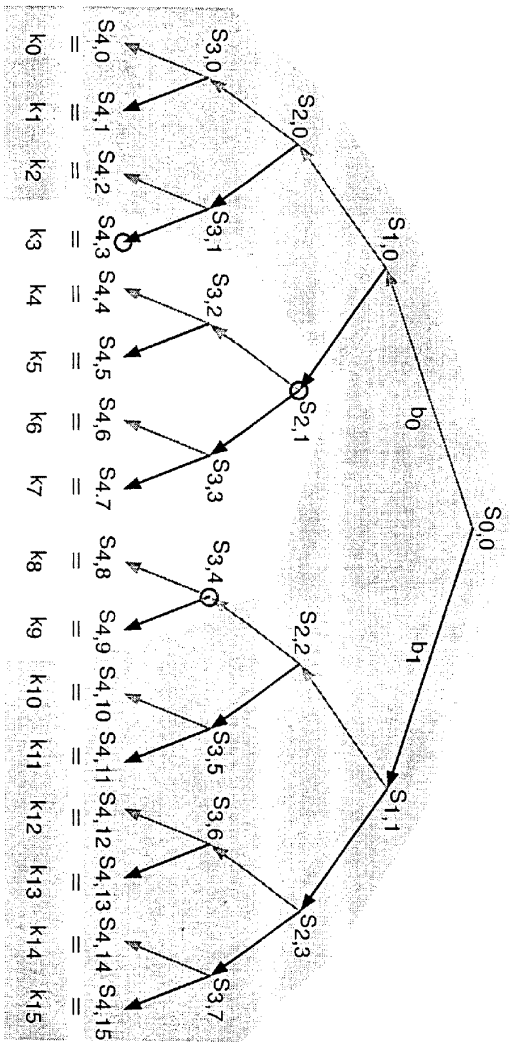
도면 7



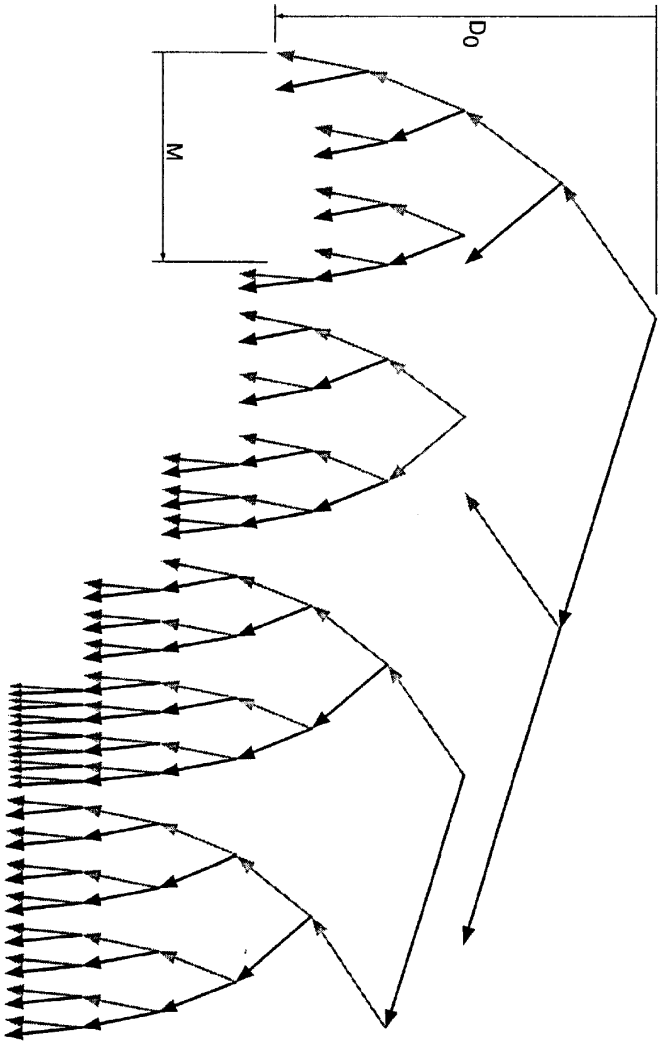
도면 8



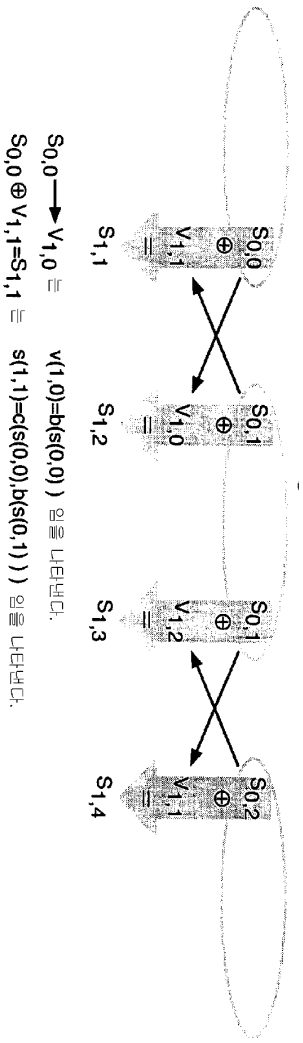
도면 9


S0,0
S1,0
S1,1
S2,0
S2,1
S2,2
S2,3
S3,0
S3,1
S3,2
S3,3
S3,4
S3,5
S3,6
S3,7
S4,0
S4,1
S4,2
S4,3
S4,4
S4,5
S4,6
S4,7
S4,8
S4,9
S4,10
S4,11
S4,12
S4,13
S4,14
S4,15
b0
b1
k0
k1
k2
k3
k4
k5
k6
k7
k8
k9
k10
k11
k12
k13
k14
k15

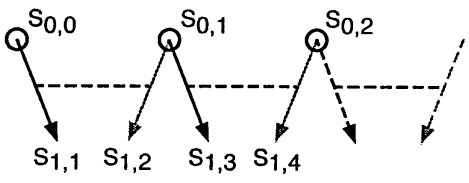
도면 10



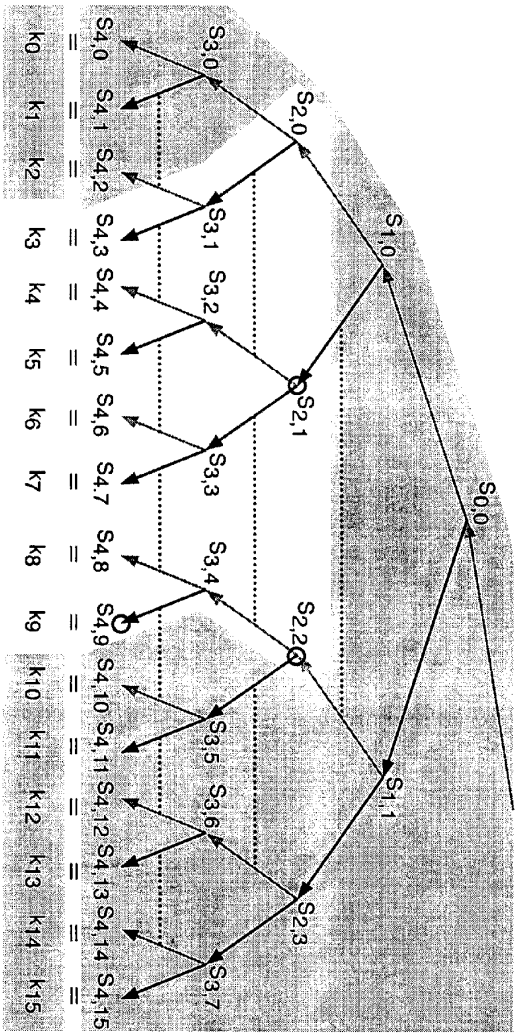
도면 11a


$S_{0,0} \rightarrow V_{1,0}$ 라
$S_{0,0} \oplus V_{1,1} = S_{1,1}$ 라
v(1,0)=b(s(0,0)) 임을 나타낸다.
s(1,1)=c(s(0,0),b(s(0,1))) 임을 나타낸다.
$S_{0,0}$
$V_{1,1}$
$S_{1,1}$
$S_{0,1}$
$V_{1,0}$
$S_{1,2}$
$S_{0,1}$
$V_{1,2}$
$S_{1,3}$
$S_{0,2}$
$V_{1,1}$
$S_{1,4}$

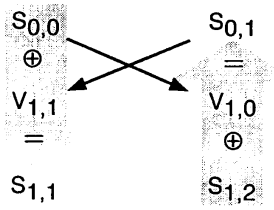
도면 11b


$S_{0,0}$
$S_{0,1}$
$S_{0,2}$
$S_{1,1}$
$S_{1,2}$
$S_{1,3}$
$S_{1,4}$

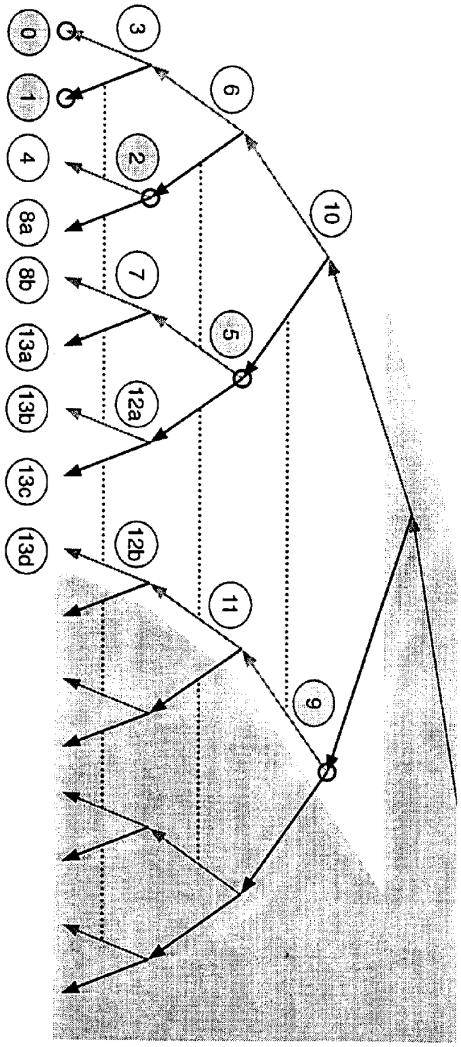
도면 12


S0,0
S1,0
S1,1
S2,0
S2,1
S2,2
S2,3
S3,0
S3,1
S3,2
S3,3
S3,4
S3,5
S3,6
S3,7
S4,0
S4,1
S4,2
S4,3
S4,4
S4,5
S4,6
S4,7
S4,8
S4,9
S4,10
S4,11
S4,12
S4,13
S4,14
S4,15
k0
k1
k2
k3
k4
k5
k6
k7
k8
k9
k10
k11
k12
k13
k14
k15

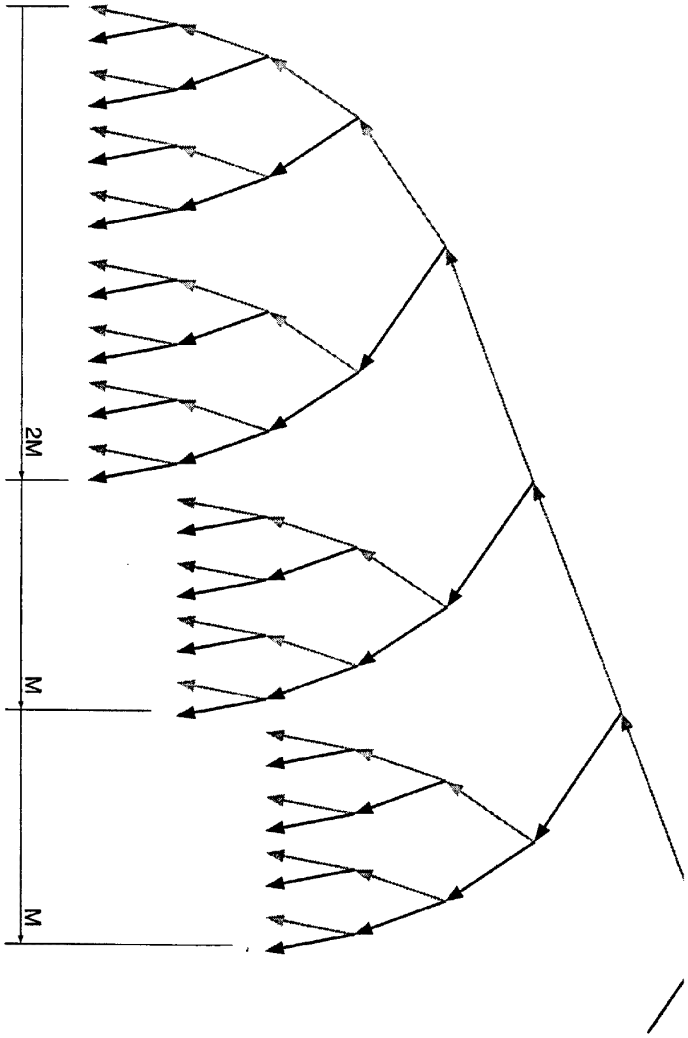
도면 13


S0,0
⊕
V1,1
=
S1,1
S0,1
=
V1,0
⊕
S1,2

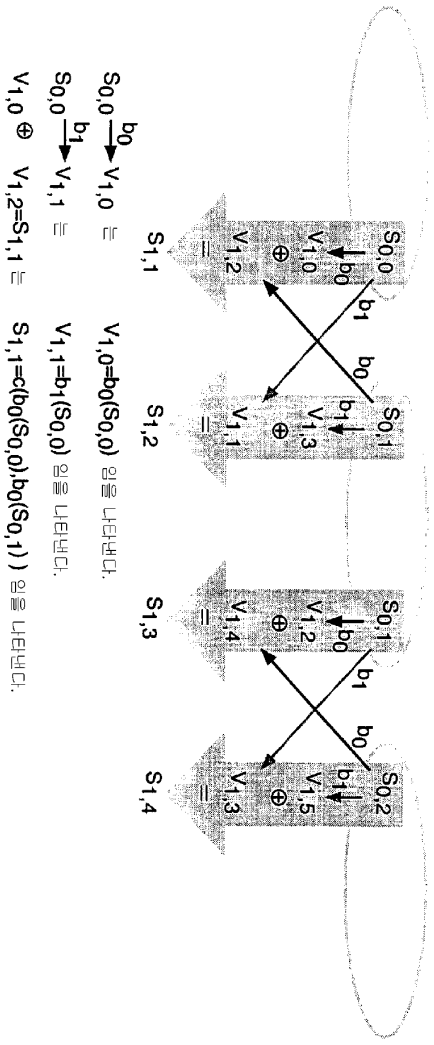
도면 14



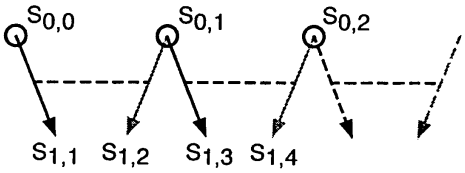
도면 15



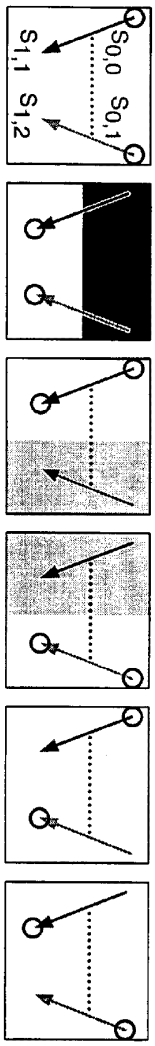
도면 16a


$S_{0,0}$
$S_{0,1}$
$S_{0,2}$
$S_{1,1}$
$S_{1,2}$
$S_{1,3}$
$S_{1,4}$
b_0
b_1
$V_{1,0}$
$V_{1,1}$
$V_{1,2}$
$V_{1,3}$
$V_{1,4}$
$V_{1,5}$
$S_{0,0}$ b_0 $V_{1,0}$ 로
$V_{1,0}=b_0(S_{0,0})$ 임을 나타낸다.
$S_{0,0}$ b_1 $V_{1,1}$ 로
$V_{1,1}=b_1(S_{0,0})$ 임을 나타낸다.
$V_{1,0}$ ⊕ $V_{1,2}=S_{1,1}$ 로
$S_{1,1}=c(b_0(S_{0,0}),b_0(S_{0,1}))$ 임을 나타낸다.

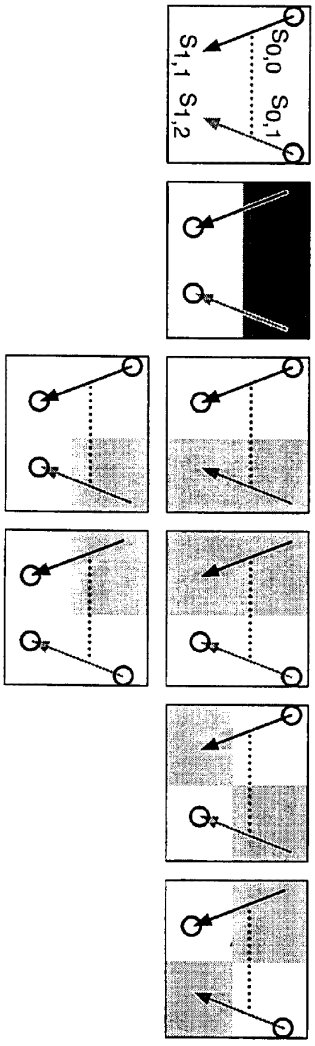
도면 16b


$S_{0,0}$
$S_{0,1}$
$S_{0,2}$
$S_{1,1}$
$S_{1,2}$
$S_{1,3}$
$S_{1,4}$

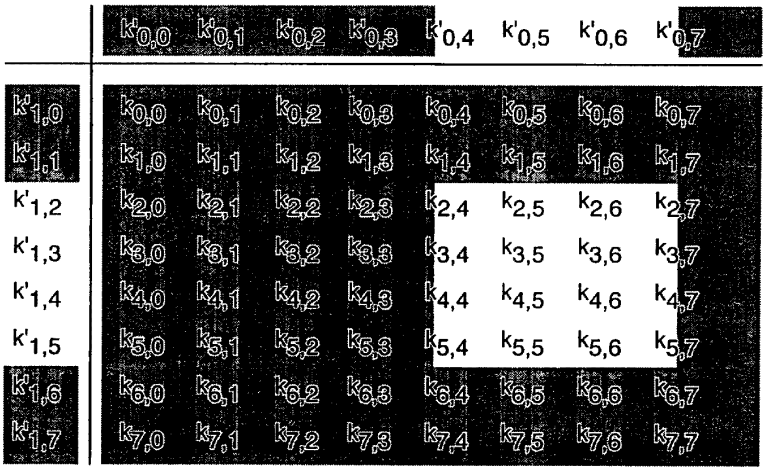
도면 17



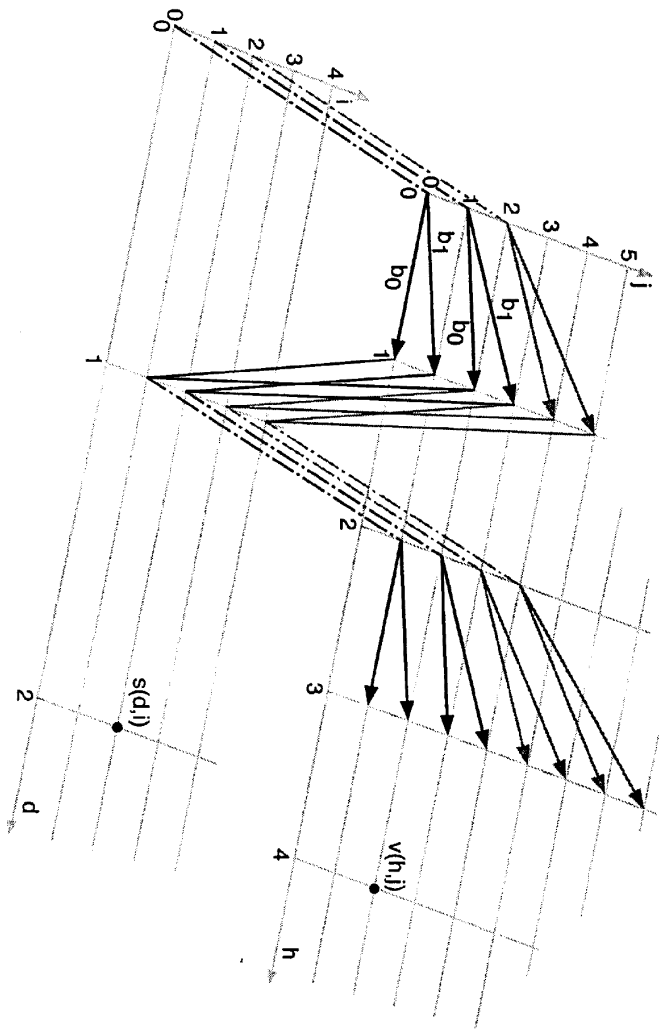
도면 18



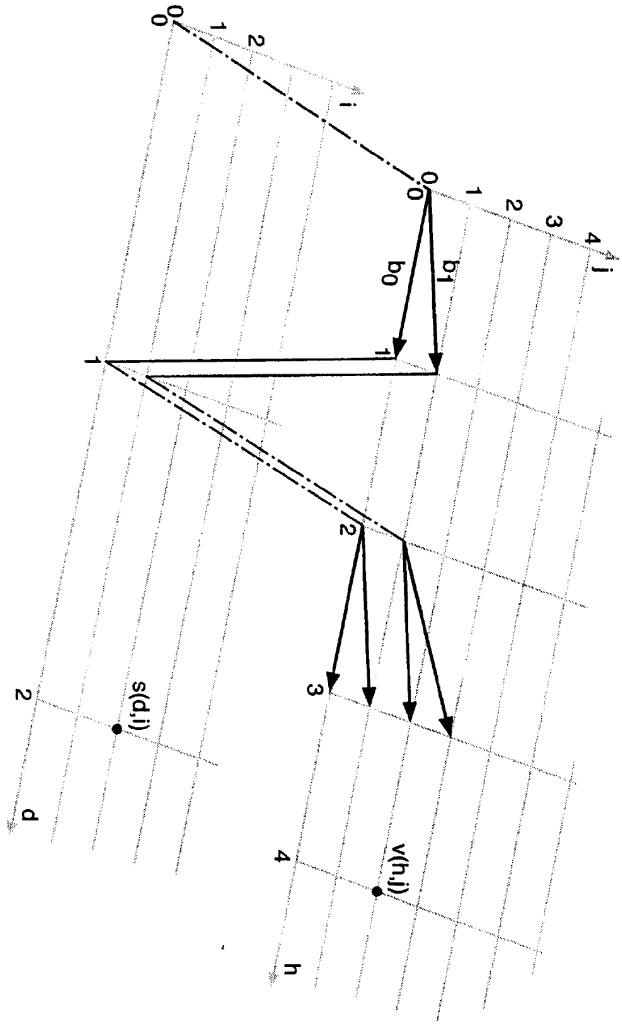
도면 19


k'0,0 k'0,1 k'0,2 k'0,3 k'0,4 k'0,5 k'0,6 k'0,7
k'1,0 k0,0 k0,1 k0,2 k0,3 k0,4 k0,5 k0,6 k0,7
k'1,1 k1,0 k1,1 k1,2 k1,3 k1,4 k1,5 k1,6 k1,7
k'1,2 k2,0 k2,1 k2,2 k2,3 k2,4 k2,5 k2,6 k2,7
k'1,3 k3,0 k3,1 k3,2 k3,3 k3,4 k3,5 k3,6 k3,7
k'1,4 k4,0 k4,1 k4,2 k4,3 k4,4 k4,5 k4,6 k4,7
k'1,5 k5,0 k5,1 k5,2 k5,3 k5,4 k5,5 k5,6 k5,7
k'1,6 k6,0 k6,1 k6,2 k6,3 k6,4 k6,5 k6,6 k6,7
k'1,7 k7,0 k7,1 k7,2 k7,3 k7,4 k7,5 k7,6 k7,7

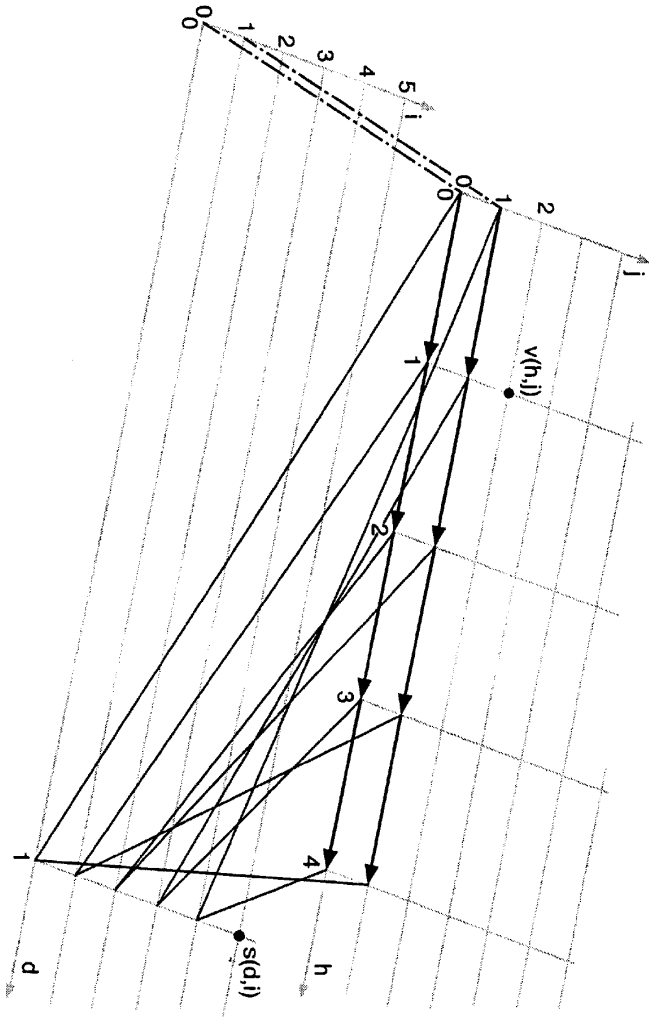
도면 20a



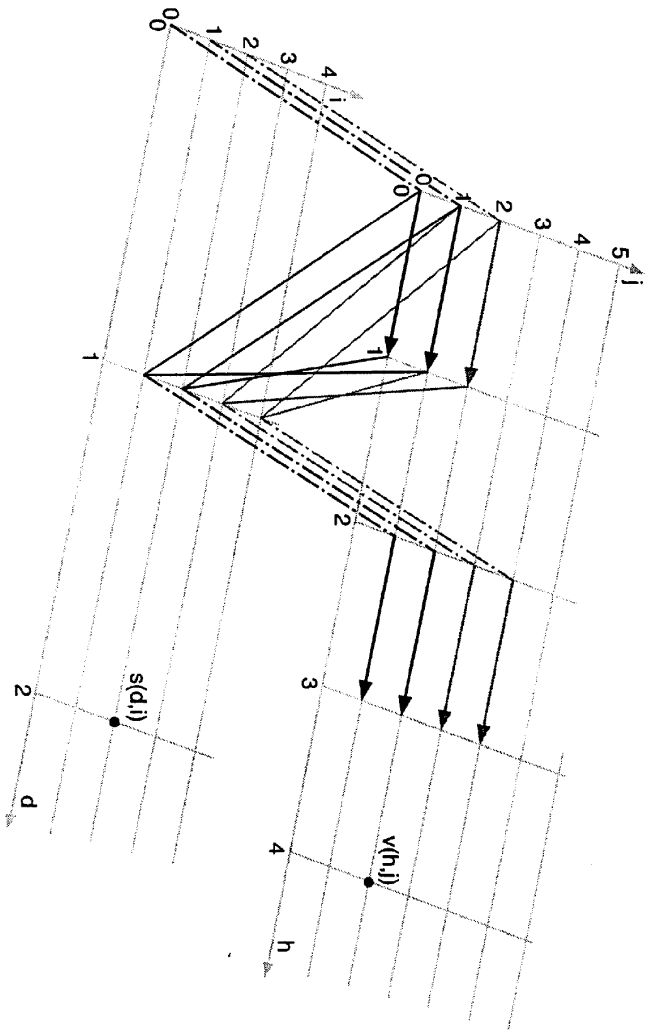
도면 20b



도면 20c



도면 20d



도면 20e