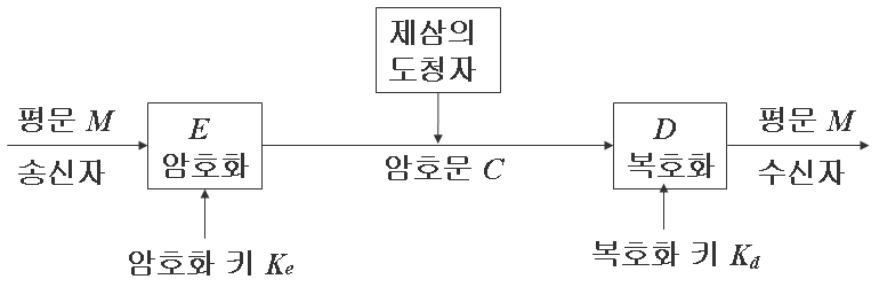
02 암호, 쉽게 이해하기

2-1 암호란? 정보보호를 가능하게 하는 수학적 기반기술

암호cryptography란 용어의 어원은 그리스어의 비밀이란 뜻을 가진 Cryptos로 알려져 있는데 평문을 해독 불가능한 형태로 변형하거나 또는 암호화된 통신문을 원래의 해독 가능한 상태로 변환하기 위한 모든 수학적인 원리, 수단, 방법 등을 취급하는 기술 또는 과학을 말한다. 즉 중요한 정보를 다른 사람들이 보지 못하도록 하는 방법이다. 그러나 현대의 정보화 사회에서는 정보를 감추는 기밀성뿐만 아니라 정보에 대한 적법한 권한을 가지고 있는지 확인하는 인증 및 접근통제, 정보의 변조 여부를 확인하는 무결성, 정보에 대한 사용자의 서명 등 좀 더 다양한 기능들을 요구하며 현대의 암호는 이런 기능들을 구현하기 위한 모든 수학적 기반기술이라고 말할 수 있다.

암호는 고도의 수학적인 내용을 포함하고 있어서 일반인들이 접근하기 어려운 순수 학문처럼 보이지만 실제로는 정보보호에의 적용을 목표로 하는 매우 실용적인 학문이라고 말할 수 있다. 안전한 암호를 사용하는 것만으로 모든 정보를 보호할 수 있다고 생각할 수는 없겠지만 암호를 사용하지 않고 궁극적인 정보보호를 성취하는 것은 불가능하다.

<그림 >은 암호 시스템이 어떻게 구성되는지 보여준다.



<암호 시스템의 구성>

송신자sender와 수신자receiver는 비밀리에 메시지를 주고받고자 한다. 송신자와 수신자는 자신들이 직접 소유하고 관리하는 컴퓨터를 이용하며 인터넷과 같은 공개된 공용 통신망을 통해 통신을 한다. 송신자는 보내고자 하는 평문을 암호화 알고리즘을 이용해 암호문으로 변환하고 이것을 공용 통신망을 통해 수신자에게 보낸다. 수신자는 복호화 알고리즘을 이용해 평문을 복구한다. 이때 암호화키는 송신자에 의해 암호화 알고리즘에 사용되고 복호화키는 수신자에 의해 복호화 알고리즘에 사용된다.

도청자eavesdropper는 정당한 참여자가 아닌 제삼자로서 통신망 상에서 관찰되는 암호문으로부터 암호해독cryptanalysis 기술을 이용해 평문을 해독하고 통신되는 메시지에 대한 정보를 획득하고자 하는 사람을 말한다. 제3자는 메시지의 도청과 같은 수동적인 공격뿐만 아니라 메시지를 위조하거나 전달을 방해하는 등 좀 더 능동적인 공격을 가할 수 있는데 이들을 통칭하여 공격자attacker라고 한다.

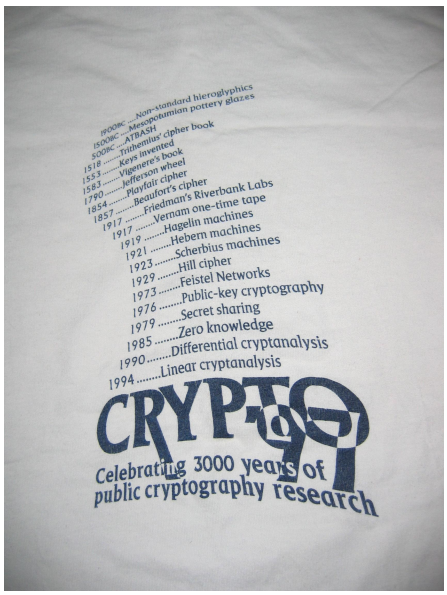
[박스원고입니다]

암호에는 어떤 방식들이 있나?

- o 대칭키 암호(비밀키 암호): 암호화와 복호화 알고리즘에 동일한 키가 사용되는 방식의 암호 알고리즘을 말한다. 복호화에 사용되는 키는 제3자에게 알려지면 안되므로 송신자와 수신자는 사용되는 키를 비밀히 공유하고 안전하게 보관해야 한다.
- o 블록 암호: 암호화와 복호화시 특정 크기의 블록 단위로 암호화/복호화 연산을 하는 방식의 암호를 블록 암호라고 한다. 각 블록의 암호화에는 동일한 키가 적용된다. 예를 들어 DES 암호는 64비트의 블록 단위로 암호화된다.
- o 스트림 암호: 스트림 암호는 비밀키 암호의 하나로 블럭의 크기를 1로 하여 블록마다 각각 다른 키를 사용하여 암호문을 생성하는 것으로 볼 수 있다. 암호화와 복호화시 키스트림 생성기를 이용하여 키스트림을 생성하며 이것을 평문과 연산하여 암호화하고 거꾸로 이것을 암호문과 연산하여 평문을 얻어낸다.
- o 비대칭키 암호(공개키 암호): 하나의 쌍이 되는 두 개의 키를 생성하여 하나는 암호화에 사용하고 다른 하나는 복호화에 사용한다. 암호화에 사용하는 키는 공개할 수 있어서 공개키라고 부르고 복호화에 사용하는 키는 사용자만이 비밀히 보관해야 하므로 개인키(비밀키)라고 부른다. 두 개의 키가 서로 다르므로 비대칭키 암호라고 부르며 하나의 키를 공개하므로 공개키 암호라고도 부른다.
- o 해쉬함수: 해쉬함수는 임의의 길이의 정보(비트스트링)를 입력으로 하여 고정된 길이의 출력값인 해쉬코드로 압축시키는 함수이다. 이것은 입력정보에 대한 변조할 수 없는 특징값을 나타내며 통신 중에 정보의 변조가 있었는지 여부를 확인하는 용도에 사용된다. 이런 용도로 사용될 수 있기 위해서 해쉬함수는 같은 해쉬값을 가지는 두 개의 입력 메시지를 찾는 것이 계산적으로 불가능해야 한다.
- o MAC 알고리즘: 메시지 인증 코드(MAC : Message Authentication Code)는 데이터가 변조(수정, 삭제, 삽입 등)되었는지를 검증할 수 있도록 데이터에 덧붙이는 코드이다. 종이 문서의 경우 원래의 문서를 고치거나 삭제하거나 하는 경우 그 흔적이 남아서 변조되었는지를 확인하지만 디지털 데이터인 경우 일부의 비트가 변경되거나 혹은 임의의 데이터가 삽입되거나 일부가 삭제되어도 흔적이 남지 않는다. 이런 문제를 해결하기 위해 원래의 데이터로만 생성할 수 있는 값을 데이터에 덧붙여서 확인하도록 하는 것이 MAC이다.
- o 전자서명: 전자문서에 대한 전자적인 방식에 의한 서명으로 서명자만이 생성할 수 있고 누구나 서명의 유효성을 검증할 수 있다. 전자서명은 공개키 암호 알고리즘을 사용하며 개인키로 서명하고 공개키로 검증한다. 개인키는 해당 서명자만이 가지고 있으므로 전자서명은 서명자의 정당한 서명으로 인정된다.
- o 키합의: 공개키 암호는 속도가 느리기 때문에 실제의 데이터를 암호화, 복호화하는 용도에 사용하지 않는다. 송신자와 수신자가 비밀키 암호를 사용하기 위해서는 미리 비밀키를 공유하거나 안전한 통신 채널을 사용하여 세션키를 전송하는 것이 필요하다. 송신자와 수신자가 직접 만나지 않고도 공개된 통신채널을 통해서 특정한 방법으로 세션키를 안전하게 공유하는 방식을 키합의라고 한다.

2-2 역사 속의 암호

<그림 >은 1997년 크립토Crypto라는 국제암호학회에서 참가자들에게 나누어준 T 셔츠에 도안된 암호의 역사에 관한 기록이다. 암호연구에 대한 역사가 벌써 3000년이 넘었다는 것을 말해주고 있다.



<그림 > 암호연구의 역사

암호기술의 발전 역사를 구분할 때 흔히 두 번의 큰 전환점을 기준으로 하여 고대암호, 근대암호, 현대암호의 세단계로 나눈다.

첫 번째 전환점은 1920년대 1차, 2차 세계대전에서 무선통신 기술의 발전을 기반으로 여러 가지 기계적, 전자적 암호장치를 개발하고 사용한 것이었고, 두 번째는 1970년대 들어 컴퓨터 사용이 활발해지면서 컴퓨터를 이용한 암호기술이 발전한 것이다.

이러한 전환점을 기준으로 고대로부터 1차, 2차 세계대전 이전까지의 시기에 사용된 초보적인 암호기술들을 고대암호라고 하며, 두 차례의 세계대전부터 1970년대까지의 복잡한 기계장치와 전자장치들을 이용한 암호기술을 근대암호, 컴퓨터가 개발된 이후 컴퓨터를 이용하는 발전된 암호기술을 현대암호라고 부른다.

고대암호 및 근대암호 시기에는 암호기술이 주로 특정 계층의 전문가들에 의해 군사용, 첩보용으로 쓰였고 일반인들은 암호기술에 대해 인식하거나 접할 기회가 없었다. 반면 현대암호 시기에 들어서는 일반인들도 암호기술에 대해 공부할 수 있게 되었고 또한 널리 사용하게 되었다.

이러한 암호기술의 역사를 살펴보면 인류는 고대로부터 정보를 보호해야 할 필요성을 느꼈고 이것을 성취하기 위해 그 당시의 최선의 지식과 기술을 이용하여 암호를 고안하고 이용하는데 많은 노력을 기울여 왔음을 알 수 있다.

2-2-1 고대암호

고대 봉건 사회에서는 황제나 군주가 지방관리에게 보내는 비밀문서, 전쟁 중의 작

전지시와 보고, 첩자들과의 통신 등 전쟁시나 첩보시에 정보를 비밀히 전달해야 하는 경우에 다양한 비밀통신 기법들이 사용되었는데 영화나 드라마에서 여러 가지 재미있는 사례들이 나타나곤 한다.

멀리 기밀정보를 전달해야 하는 경우 사자의 머리를 깎고 메시지를 쓴 후 머리를 길러서 보내면 받는 측에서는 사자의 머리를 깎고 메시지를 읽는 방법, 종이에 쓴 메시지가 그냥 보면 보이지 않지만 불빛에 비추거나 약품처리를 하면 메시지가 나타나도록 하는 방법, 비밀노출을 방지하기 위해 사자에게 말로 전달하도록 하는 방법 등 다양한 통신 방식이 이용됐다. 이러한 비밀통신 방법을 스테가노그래피(steganography)라고 하는데 적들도 이 통신방식을 알고 있으면 비밀을 유지하기가 어려운 취약한 방법이다.

반면 암호에서는 통신방식뿐만 아니라 키(key)라고 하는 부가적인 비밀정보를 가지고 정보를 처리하게 된다. 암호 알고리즘은 알려지더라도 키를 안전하게 보호함으로써 정보를 안전하게 감출 수 있다.

현대암호가 연구되기 전 고대암호와 근대암호에서 사용된 암호기법들을 분류해보면 문자의 위치를 서로 바꾸는 전치암호(transposition cipher), 문자를 다른 문자로 치환하는 환자암호(substitution cipher), 그리고 전치암호와 환자암호를 복합적으로 사용하는 적암호(product cipher)로 분류할 수 있다.

2-2-1-1 스키테일 암호

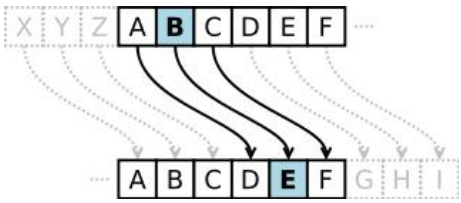
기원전 400년경 고대 그리스의 군사들은 스키테일scytale 암호라고 불리는 전치암호를 사용한 기록이 있다.

특정 지름을 갖는 막대에 종이를 감고 평문을 횡으로 쓴 다음 종이를 풀면 평문의 각 문자는 재배치되어 정보를 인식할 수 없게 되는데, 암호문 수신자가 송신자가 사용한 막대와 지름이 같은 막대에 종이를 감고 횡으로 읽으면 평문을 읽을 수 있다. 여기서 막대의 지름은 송신자와 수신자 사이에 공유된 비밀키가 된다.

<그림 > 스키테일 암호

2-2-1-2 시이저 암호

로마의 황제였던 줄리어스 시이저(Julius Caesar)는 시이저 암호라고 불리우는 환자암호를 사용하였다. 시저는 가족과 비밀통신을 할 때 각 알파벳 순으로 세자씩 뒤로 물려 읽는 방법으로 글을 작성했다. 즉 A는 D로, B는 E로 바꿔 읽는 방식이었다. 수신자가 암호문을 복호화하려면 암호문 문자를 좌측으로 3문자씩 당겨서 읽으면 원래의 평문을 얻을 수 있다. 송신자와 수신자는 몇 문자씩 이동할지를 비밀키로 하여 바꿔가면서 사용할 수 있다.



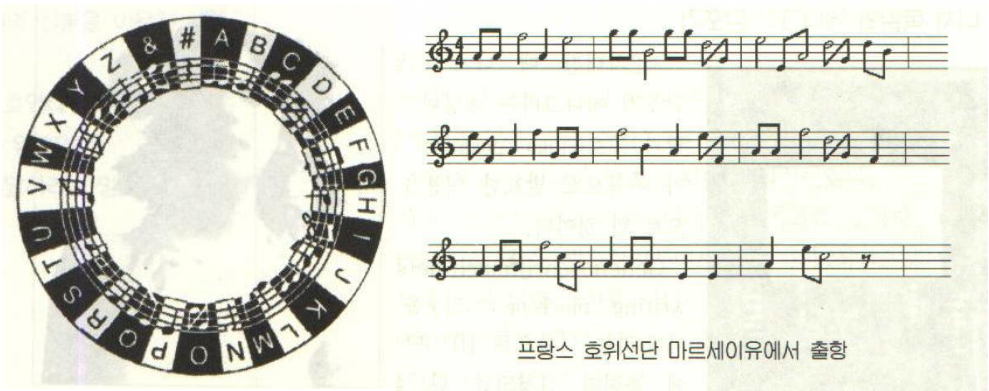
<그림 > 시이저 암호

시이저는 브루투스에게 암살당하기 전 가족들로부터 다음과 같은 긴급통신문을 받았다. 시이저가 받은 편지에는 'EH FDUHIXO IRU DVVDVVLQDWRU'라 돼 있었으나 3글자씩 당겨서 읽어보면 뜻은 'BE CAREFUL FOR ASSASSINATOR', 즉 '암살자를 주의하라'는 것이었다. 당시 시이저의 권세를 시기했던 일당은 시이저를 살해할 암살 음모를 꾸미고 있었으며 시이저 자신도 이를 어느 정도 눈치 채고 있었다. 하지만 시이저는 구체적으로 암살자가 누구인지 알 수 없었다. 결국 암호문을 전달받은 당일 시이저는 원로원에서 전혀 생각지도 못했던 브루투스에게 암살당하면서 "브루투스, 너마저…"라는 말을 남겼다.

2-2-1-3 악보 암호

악보암호는 전설적인 스파이 마타하리가 사용했던 방식이다. 마타하리는 일명 '첩보원 H21'이란 이름으로 프랑스 장교에 접근해 군사기밀 정보를 독일에 빼돌렸는데, 이때 비밀통신에 사용된 암호가 악보였다. 일정한 형태의 음표에 알파벳 하나씩을 대응시킨 형태로 얼핏 보기에 평범한 악보처럼 보이지만 실제로 연주하면 전혀 '음악'이 되지 않는다.

마타하리의 첩보활동은 20여만명에 달하는 프랑스군을 죽음으로 몰고 갔다. 그녀는 1차대전이 끝나기 1년 전 프랑스 정보부에 체포돼 사형 당했다.



<그림 > 마타하리의 악보 암호

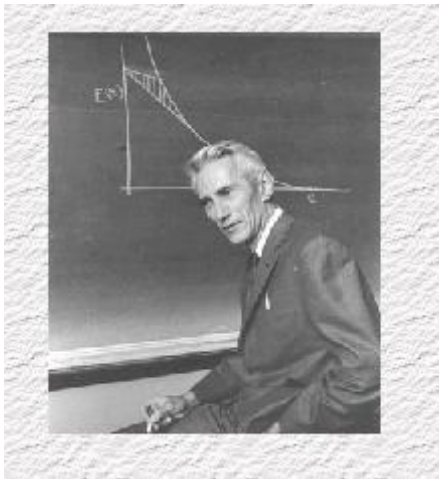
2-2-2 근대암호

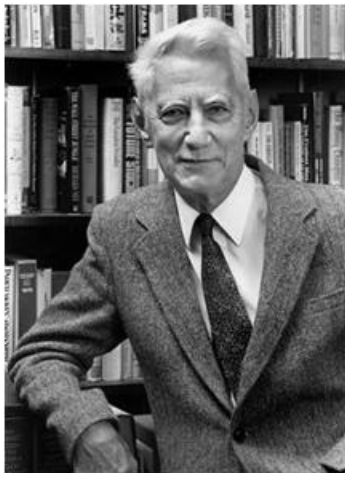
고대의 단순한 암호방식으로부터 17세기 근대 수학의 발전과 더불어 암호기술도 발전하기 시작하였는데 프랑스의 외교관이었던 비제네르Vigenere가 고안한 키워드를 이용

한 복수 시이저 암호형 방식, 플레이페어Playfair가 만든 2문자 조합 암호 등, 다양한 암호방식이 발전했다.

20세기 들어서는 통신기술의 발전과 기계식 계산기에 대한 연구를 바탕으로 두 차례의 세계대전을 통해 암호 설계와 해독에 대한 필요성이 높아지면서 암호에 대한 연구가 활발히 진행됐다.

근대 암호의 이론적 기초가 된 논문은 1920년 프리드만Freidman이 발표한 "일치 반복률과 암호응용"과 1949년 샤논Shannon이 발표한 "비밀시스템의 통신이론"이다. 샤논은 이 논문에서 일회성 암호체계가 안전함을 증명하였고 암호체계 설계의 두 가지 기본 원칙인 혼돈(confusion)과 확산(diffusion) 이론을 제시했다. 암호체계를 설계함에 있어서 혼돈은 평문과 암호문사이의 상관관계를 숨기는 반면 확산은 평문의 통계적 성격을 암호문 전반에 확산시켜 숨기는 역할을 한다. 혼돈과 확산이라는 두 가지 개념은 오늘날의 암호체계 설계에도 여전히 적용되고 있다.





<그림 > 클로드 샤논

프리드만은 2차 세계대전 중 독일군이 사용하던 이니그마Enigma 암호와 일본군이 사용하던 무라사끼 암호를 해독한 사람으로 유명하다. 이니그마 암호는 각기 다른 몇 개의 암호판을 전기적으로 연결하여 원문을 입력하면 전기적 연결에 의해 새로운 암호문을 출력하는 방식으로 이 기계가 존재하지 않으면 암호를 풀 수 없다.

< 독일군이 사용했던 Enigma 암호 >

[박스원고입니다]

미드웨이 해전에서의 암호전쟁

태평양 전쟁 당시 일본의 진주만 공습으로 큰 피해를 입고 전력이 약화됐던 미국은 일본의 그 다음 공격목표가 어디인지를 알아내야 했다.

1942년 4월, 하와이 주둔 미국 해군 정보부의 암호 해독반 블랙 챔버는 일본군의 무전이 증가하고 있음을 발견했다. 이미 일본 해군의 암호 체계인 JN-25를 해독하고 있던 해독반은 AF라는 문자가 자주 나타난다는 사실에 주목했다. AH는 진주만을 뜻하는 것이었다.

암호 해독반의 지휘관이었던 죠셉 로슈포르 중령은 AF를 미드웨이 섬이라고 생각했다. 일본의 정찰기가 "AF 근처를 지나고 있다"는 내용의 무선 보고를 해독한 적이 있었던 그는 정찰기의 비행경로를 추정해본 결과 AF가 미드웨이 섬일 것이라는 심증을 갖게 된 것이다.

로슈포르 중령은 체스터 니미츠 제독에게 일본군의 침공이 임박했다는 것과 AF가 자주 언급된다는 점, 그리고 AF가 미드웨이 섬일 것이라는 보고를 한 후, 미드웨이 섬의 담수 시설이 고장 났다는 내용의 가짜 전문을 하와이로 평문 송신하게 하자고 건의했다. 3월에 미드웨이 섬 근처에 일본 해군의 비행정이 정찰왔던 것을 알고 있던 니미츠 제독은 이 건의를 받아들였다. 사실 미드웨이 섬의 정수 시설은 아무런 문제가 없었다. 이틀 후, 도청된 일본군 암호 중 "AF에 물 부족"이라는 내용이 해독되었다. 이로써 일본군의 다음 공격 목표가 미드웨이 섬이라는 것이 분명해진 것이다.

미군은 암호해독을 통해 일본의 공격목표가 미드웨이라는 사실을 알아낸 후 전투에 대비하고 반격을 준비하여 일본의 태평양함대를 격파하고 전쟁을 승리로 이끌 수 있었다.

2-2-3 현대암호

현대암호는 1970년대 후반 스탠포드 대학과 MIT 대학에서 시작되었다. 1976년 스탠

포드 대학의 디피Diffie와 헬만Hellman은 “New Directions in Cryptography(암호의 새로운 방향)"라는 논문에서 처음으로 공개키 암호의 개념을 발표했다.

종래의 관용암호 방식 또는 대칭키암호 방식에서는 암호화키와 복호화키가 동일한 비밀키를 사용하기 때문에 송신자와 수신자는 비밀통신을 하기 전에 비밀키를 공유하고 있어야 한다. 반면 공개키 암호 방식에서는 하나의 쌍이 되는 공개키와 비밀키를 생성하여 암호화에 사용되는 공개키는 공개하고 복호화에 사용되는 비밀키는 사용자가 안전하게 보관하도록 한다. 공개키 암호 방식에서는 송신자와 수신자가 사전에 키를 공유할 필요가 없기 때문에 불특정 다수 사용자 간에 사전 준비가 없이도 암호통신망을 구축하는데 유용하게 사용할 수 있다.

1978년 MIT 대학의 론 리베스트Ron Rivest, 아디 샤미르Adi Shamir, 레오나드 에이들맨Leonard Adleman은 소인수분해 문제에 기반한 RSA 공개키 암호를 개발했는데 이것은 오늘날까지도 가장 널리 사용되는 공개키 암호 방식이다. 공개키 암호의 도입은 현대암호의 발전에 중요한 계기가 되었다.

한편 1977년 미국 상무성의 표준국은 상업용 암호 표준화의 필요성이 제기됨에 따라 암호 알고리즘을 공개 모집하여 당시 IBM사가 제안한 DES(Data Encryption Standard)를 표준 암호 알고리즘으로 채택했다. DES의 표준화를 계기로 하여 금융시스템을 중심으로 상업용 암호화의 이용이 증가하게 되었고 컴퓨터 통신망을 이용한 문서 전송, 전자자금이체 등이 활성화되었으며 암호 방식이 일반인들에게 알려지고 널리 사용되는 계기가 되었다.

이전의 암호방식에서는 사용하는 키뿐만 아니라 암호 알고리즘도 비밀로 하여 암호문의 비밀을 지키려고 하는 경우도 있었으나 현대 암호에서는 암호 알고리즘을 공개하도록 하고 있다. 1883년 커크호프(Auguste Kerckhoff)는 암호시스템의 안전성에 대해 “키 이외에 암호시스템의 모든 것이 공개되어도 안전해야 한다”고 했는데 이것을 커크호프의 원칙이라고 한다. 이렇게 함으로써 암호방식의 안전성을 공개적으로 검토하게 하여 안전성을 확인하는 것이다.

표준화된 암호와 표준화된 컴퓨팅 기기들을 사용하는 현대 암호에서는 암호 알고리즘을 감추기가 매우 어렵다. 또한 암호 알고리즘을 감춘다고 해서 암호의 보안성이 높아지는 것도 아니다. 비밀로 다루어진 암호 알고리즘이 일단 공개되고 나면 그 안전성에 문제가 발견되는 사례가 많다. 그러므로 암호 분야에서는 어떤 암호 알고리즘이 많은 암호학자들에 의해 장기간 세부적으로 수행된 분석에서도 잘 견디어 낼 때까지는 그 알고리즘을 안전하다고 인정하지 않는다. 즉 암호체계는 “무죄가 증명될 때까지는 유죄”이다.

2-2-4 암호 알고리즘의 표준화

암호기술은 송신자와 수신자 모두 합의된 같은 알고리즘을 사용해야 통신이 가능하게 된다. 그러므로 암호기술의 표준화는 정보보호의 확산에 매우 중요한 역할을 한다. 폐쇄된 소규모 그룹에서만 비밀스럽게 사용하는 암호기술이라면 표준화된 알고리즘이 필요 없을지도 모르지만, 인터넷과 같은 공용의 통신망과 누구나 사용하는 표준화된 컴퓨터 및 운영체제를 이용해 생면부지의 상대편과도 안전한 통신을 하기를 원한다면 표준화된 암호 알고리즘을 사용하지 않을 수 없다. 즉 모든 컴퓨터에 같은 암호 알고리즘

이 구현되어 있어야 서로 호환이 되고 안전한 통신이 가능한 것이다. 이러한 측면에서 산업계, 학계, 정부기관, 국제표준기구 등을 통해 암호 알고리즘을 표준화하고 이를 산업계에 적용하기 위한 노력이 기울여져 왔다.

표준 암호알고리즘을 사용한 최초의 사례는 1977년 미국에서 표준화된 블록암호 알고리즘인 DES(Data Encryption Standard)를 들 수 있다. 표준화된 DES 알고리즘이 널리 사용되면서 암호의 이용이 기존의 군사용 등 특수한 용도에서부터 일반적인 상업용으로 확산되었고 암호 방식이 일반인들에게 알려지고 널리 사용되는 계기가 되었다. 그러나 DES 알고리즘은 암호시스템 자체는 비교적 안전한 것으로 평가되지만 비밀키가 56비트로 너무 작아서 현재 우리가 사용하는 컴퓨팅 능력에 비해 취약하다는 단점이 있다. 1998년 기록으로 키 공간에 대한 전수조사를 통해 56시간 만에 DES의 키를 찾아냈으니 현재의 컴퓨터로는 이보다 훨씬 짧은 시간에 찾아낼 수 있어서 전혀 안전하지 않다고 볼 수 있다.

이에 따라 DES를 대체할 새로운 블록암호 표준의 필요성이 제기됐다. 미국표준기술연구소(NIST)는 블록암호 알고리즘을 전세계적으로 공모했고, 5년의 표준화 과정을 거쳐 2001년 11월 26일 AES를 새로운 표준 알고리즘으로 발표했다. 이 알고리즘은 벨기에의 암호학자인 존 대먼과 빈센트 라이먼에 의해서 만들어졌으며, 처음에는 두 사람의 이름을 합해서 레인달(Rijndael)이라는 이름을 썼다. 키의 크기(블록의 크기)로 128, 160, 190, 224, 256비트를 사용할 수 있으며, 미국 표준으로 인정받은 것은 128비트이다.

이밖에도 세계 각국은 암호 알고리즘의 표준화에 노력하고 있다. 우리나라에서는 한국정보보호진흥원과 ETRI 주도하에 개발된 대칭키 방식의 128비트 블록암호 알고리즘인 SEED가 한국의 표준 블록암호 알고리즘으로 제정되어 사용되고 있다.

[박스원고입니다]

SEED 암호 알고리즘이란?

1999년 2월 한국정보보호진흥원과 ETRI 주도하에 국내 암호 전문가들이 함께 개발한 암호 알고리즘으로 인터넷, 전자상거래, 무선통신 등에서 공개되는 경우 민감한 영향을 미칠 수 있는 중요 정보 및 개인 정보를 보호하기 위해 개발된 국내 블록암호 알고리즘이다. SEED는 1999년 9월 정보통신단체표준(TTA)으로 제정되었으며 2005년에는 국제 표준화 기구인 ISO/IEC 국제 블록암호 알고리즘 표준으로 제정되었다.

현재 유럽지역의 연구자들을 중심으로 스트림 암호의 표준화를 위해 이크립트 ECRYPT라는 프로젝트를 2004년부터 진행하고 있다. 이 이외에 해쉬함수, 전자서명 알고리즘 등도 표준화된 알고리즘들이 사용된다.

표준화된 암호 알고리즘들은 우리가 사용하고 있는 컴퓨터, 통신기기, 운영체제, 응용 소프트웨어, 하드웨어, 보안 프로토콜 등에 실제 구현되어 사용되고 있다. 윈도우, 리눅스, 맥 OS 등 운영체제에는 이들 표준 암호 알고리즘들이 내장되어 있다. 인터넷 보안에 사용되는 IPSec 프로토콜, 사용자간 세션 암호화에 사용되는 SSL/TLS 프로토콜 등은 암호 알고리즘들이 복합적으로 사용되어 통신 시스템의 정보보호 기능을 제공하는 좋은 사례이다. 그 밖에도 암호기술은 휴대폰을 이용한 이동통신의 보안, 저작권 보호, 이메일 보안, 공인인증, 전자상거래, 전자정부 서비스 등 이루 헤아리기 어려울 만큼 다방면에 응용되고 있다.

2-3 빠르고 효율적인 암호 시스템 : 대칭키 암호

대칭키 암호 방식에서는 암호화에 사용되는 암호화키와 복호화에 사용되는 복호화키가 동일하다는 특징이 있으며 이 키를 송신자와 수신자 이외에는 노출되지 않도록 비밀히 관리해야 한다. 우리가 일반적으로 사용하는 암호라는 의미로 관용암호라고도 하며 키를 비밀히 보관해야 한다는 의미로 비밀키 암호라고도 한다. 이 방식은 고대 암호로부터 연결된 오랜 역사를 가지고 있다.

사용되는 키가 짧아도 되고 속도가 빨라서 효율적인 암호 시스템을 구축할 수 있다. 이 암호방식은 알고리즘의 내부구조가 간단한 치환(대치)과 전치(뒤섞기)의 조합으로 되어 있어서 알고리즘을 쉽게 개발할 수 있고 컴퓨터 시스템에서 빠르게 동작한다. 그러나 송수신자 간에 동일한 키를 공유해야 하므로 많은 사람들과의 정보 교환시 많은 키를 생성, 유지, 관리해야 하는 어려움이 있다. 이러한 대칭키 암호 방식은 데이터를 변환하는 방법에 따라 블록암호와 스트림 암호로 구분된다.

2-3-1 블록 암호

블록 암호는 고정된 크기의 블록 단위로 암호화, 복호화 연산을 수행하며 각 블록의 연산에는 동일한 키가 이용된다. 고대 암호 알고리즘에는 평문의 문자를 특정한 다른 문자로 대치하는 환자암호 방식, 평문 문자의 순서를 특정 방식으로 섞어서 재배치하는 전치암호 방식, 그리고 이들을 혼합하여 사용하는 적 암호 방식으로 구분할 수 있다.

샤논의 암호이론에 의하면 전치와 환자를 반복시켜 암호화하면 평문의 통계적 성질이나 암호키와의 관계가 나타나지 않아 안전한 암호를 구성할 수 있다. 이러한 성질을 이용하여 파이스텔Feistel은 전치와 환자를 반복 적용한 파이스텔 네트워크Feistel Network 방식의 암호 알고리즘을 설계하였는데 DES 알고리즘이 대표적인 예이다.

이 방식은 암호화 과정과 복호화 과정이 적용되는 키만 다르고 알고리즘은 동일하기 때문에 하드웨어/소프트웨어 구현시 하나의 알고리즘만 프로그래밍 하면 되기 때문에 편리하며 안전성도 높다는 것이 밝혀졌기 때문에 현대 관용 암호 설계에 많이 이용된다. 대표적인 블록암호 알고리즘은 DES, Triple-DES, IDEA, FEAL, MISTY 등이 있으며 국내 암호표준인 SEED가 있다. AES는 DES알고리즘을 대치하여 사용되는 새로운 표준 블록암호 알고리즘이다.

<그림 > Horst Feistel

2-3-2 스트림 암호

스트림 암호는 블록 단위로 암호화, 복호화되는 블록암호와는 달리 이진화된 평문 스트림과 이진 키스트림의 배타적 논리합(XOR) 연산으로 암호문을 생성하는 방식이다. 이러한 스트림 암호는 키스트림이 평문과 관계없이 생성되어 동기식으로 사용해야만 하는 동기식 스트림 암호와 키스트림이 평문 혹은 암호문으로부터의 함수관계에 의해 생성되기 때문에 복호화시 동기가 흐트러졌더라도 스스로 동기화가 이루어져서 복호화가 가능한 자기 동기식 스트림 암호가 있다.

1970년대부터 유럽을 중심으로 발달한 스트림 암호는 주기, 선형복잡도 등 안전성과 관련된 수학적 분석이 가능하고 알고리즘 구현이 쉬운 특징이 있으며 군사 및 외교용으로 많이 사용되고 있다. 스트림 암호는 구현 여건이 제약되는 이동통신 환경에서도 구현이 용이하여 이동통신 등의 무선데이터 보호에 많이 사용된다.

2-4 나의 키를 공개한다 : 공개키 암호

2-4-1 많은 키들을 어떻게 관리하나? : 공개키 암호의 도입

1976년 디피Diffie와 헬만Hellman에 의해 공개키 암호의 개념이 도입되고 1978년 RSA라는 실용적인 공개키 암호 알고리즘이 개발된 이후 공개키 암호는 현대 암호를 이루는 중요한 요소가 되었다.

공개키 암호는 비대칭키 암호라고도 하는데 이것은 관용암호와는 달리 하나의 쌍이 되는 두 개의 키가 존재하여 하나의 키는 누구든지 사용할 수 있도록 공개하고 다른 하나는 자신만이 사용할 수 있도록 비밀리에 보관하는 방식이다. 이때 공개하는 키는 암호화에 사용되며 공개키라고 하고 비밀로 보관하는 키는 복호화에 사용하며 개인키라고 한다.

공개키로 암호화한 암호문은 이것의 쌍이 되는 개인키로만 복호화 할 수 있다. 이때 공개키가 공개되더라도 이것으로부터 개인키를 계산해 내는 것은 계산적으로 불가능한 특징이 있다. 공개키 암호방식은 관용암호에서와 같은 비밀키의 사전분배가 필요 없는 새로운 방식이다. 비밀통신을 하기 위해서는 송신자는 수신자의 공개키를 이용하여 메시지를 암호화해서 보내면 된다. 이 암호문은 개인키를 가진 정당한 수신자 이외에는

어느 누구도 복호화 할 수 없는 비밀 메시지가 된다.

관용암호를 원활하게 사용하기 위한 비밀키의 사전분배는 생각보다 매우 어려운 문제이다. 예를 들어 100명의 사용자들이 관용암호를 이용하여 서로 비밀통신을 하고자 하는 경우를 생각해 보자. 각 사용자는 99명의 사용자들과 비밀키를 사전에 공유하고 있어야 하며 이것이 노출되지 않도록 안전하게 보관해야 한다. 비밀키를 남들에게 노출되지 않도록 사전에 공유하기 위해서는 안전한 통신채널이 있어서 이것을 이용하여 전달하거나 직접 만나서 전달하는 수밖에 없는데, 안전한 통신채널이 이미 있다면 복잡한 암호를 쓸 필요도 없는 것이요, 직접 만나서 비밀키를 전달하는 것은 요즘 같은 정보화 시대에는 어울리지 않는 방식이다. 99개의 공유된 비밀키를 안전하게 보관하는 것도 쉽지 않은 문제이다. 컴퓨터 내의 데이터베이스에 보관하는 것도 해킹의 위협 등을 고려하면 쉽지 않고 많은 비밀키들을 기억하여 사용하는 것은 불가능하다도 보아야 할 것이다. 반면 공개키 암호 방식을 이용하면 각 개인은 자신의 개인키 하나만을 안전하게 보관하면 되기 때문에 키 관리가 매우 간단하고 안전하게 보관할 수 있게 된다.

공개키 암호는 키분배 문제를 해결했을 뿐만 아니라 통신망 상에서 사용자를 인증하는 중요한 도구로 사용된다. 사용자들이 비대면 상황에서 정보를 주고받으며 상호작용을 하게 되는 인터넷 통신망에서 지금 통신하고 있는 상대방이 누구인지 확인하고 상대방에게 나의 신분을 확인시키는 일은 매우 중요하다. 상대방의 신분이 서로 확인되어야 상거래와 같은 중요한 활동을 할 수 있기 때문이다.

예를 들면 사용자가 서버에 로그인 하는 간단한 예에서도 서버는 로그인 과정에서 사용자가 사용 권한이 있는 정당한 사용자인지 확인 후에 서비스를 제공해야 한다. 매우 중요한 서비스를 제공하고 있는 서버가 사용자의 신분 인증을 허술하게 하여 나쁜 의도를 가진 공격자에게 로그인을 허용했다고 가정하면 서비스의 안전성에 심각한 영향을 미칠 수 있는 것이다. 공개키 암호를 이용하면 사용자 인증 문제가 매우 간단하게 해결된다. 로그인 과정에서 사용자만이 비밀히 보관하고 있는 개인키를 이용해야 하도록 요구하면 서버는 사용자의 공개키를 이용하여 사용자의 신분확인이 가능하기 때문이다.

인증 문제를 메시지에 확대하면 전자서명도 가능하다. 사용자가 특정 문서에 자신의 개인키를 이용하여 서명이라고 하는 연산을 하면 어느 누구나 사용자의 공개키를 이용하여 서명의 검증이 가능하다. 개인키를 보관하고 있는 것은 해당 사용자 이외에는 아무도 없기 때문에 사용자가 특정 문서에 대해 개인키로 연산을 한 것은 정당한 서명으로 인정할 수 있다.

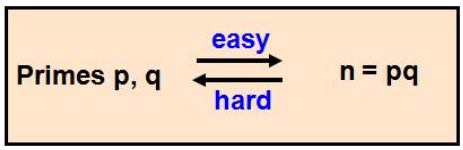
2-4-2 어려운 수학문제를 풀어보자 : 소인수분해와 이산대수 문제

공개키 암호에서는 하나의 쌍이 되는 두 개의 키가 존재하여 하나를 공개키로 사용하고 다른 하나는 개인키로 사용한다고 했는데 임의의 키가 쌍으로 사용될 수 있는 것이 아니고 특정한 성질을 만족시켜야 공개키/개인키 쌍으로 사용할 수 있는 것이다.

공개키 암호는 정수론에 기초한 일방향 함수one-way function를 사용하고 있다. 좀 더 자세히 말하면 공개키 암호를 구성하기 위해서는 비밀문trapdoor을 갖는 일방향 함수를 사용하고 있다. 쌍이 되는 두 개의 키 중에서 하나의 키를 공개키로 공개하더라도 다른 하나의 개인키는 노출되지 않아야 하기 때문이다. 공개키 암호에서 사용되는 대표적인 일방향 함수로는 소인수분해 문제와 이산대수 문제를 들 수 있다.

2-4-2-1 소인수분해 문제

소인수분해 문제(**Integer Factorization Problem**)는 큰 두 소수의 곱을 계산하는 것은 쉽지만 역으로 큰 두 소수의 곱인 합성수가 주어졌을 때 이것을 소인수분해 하는 것은 어렵다는 것이다. 소인수분해 문제는 NP(비결정론적 다항식 시간)의 의미에서 어려운 문제라고 알려져 있다.



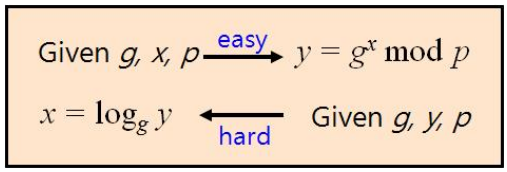
예를 들어 작은 소수인 43과 47을 곱하여 2021이라는 합성수를 구하는 것은 곱셈을 배운 초등학생이라도 쉽게 계산할 수 있는 쉬운 문제이다. 정해진 곱셈 규칙대로 계산하면 예상되는 시간 내에 분명히 답을 얻을 수 있다. 반면 2021이라는 숫자가 먼저 주어지고 이것이 어떤 두 정수의 곱이냐를 물어보면 아마 쉽게 답을 구할 수 없을 것이다. 이것은 비교적 작은 숫자이므로 두 숫자를 구하려고 노력한다면 여러 번의 시도 끝에 이 숫자가 이 숫자보다 작은 소수로 나누어떨어지는지 시도해 보고 답을 구할 수 있을 것이다.

운이 좋아서 43이라는 숫자를 처음에 시도해 보았다면 한 번에 답을 얻을 수도 있겠지만 운이 없다면 모든 소수를 다 시도해 보고 마지막에 답을 얻게 될지도 모른다. 이 문제를 푸는 데에는 여러 가지 가능성을 시도해 보는 이외에 정해진 다항식 시간의 알고리즘이 없기 때문에 NP 문제라고 하는 것이다.

만일 RSA 암호시스템에서 사용하는 300자리 정도의 큰 정수를 주고 곱이 되는 두 정수를 구하라고 하면 이것은 정말 까마득한 문제가 될 것이다. RSA 알고리즘은 소인수분해 문제의 어려움에 기반을 두어 고안된 공개키 암호 알고리즘이다.

2-4-2-2 이산대수 문제

이산대수 문제(**Discrete Logarithm Problem**)는 말로만 설명하기가 좀 복잡하지만 모듈러 연산으로 나머지만을 취하는 잉여계residue 시스템에서 위수를 구하는 것이 어렵다는 문제이다.



간단한 예를 들어 생성원 g=10을 밑으로 하고 임의의 정수로 승산을 한 후 소수 p=47로 나머지 연산을 하는 시스템을 생각해보자. 이 경우 10을 제곱하면 6이 되고 네 제곱을 계산하면 36이 된다. 이것을 이산대수 방식으로 표시하면 10을 밑으로 할 때 2는 6의 이산대수이고 4는 36의 이산대수라고 말한다.

$$10^{2} \bmod 47 = 100 \bmod 47 = 6$$
$$10^{4} \bmod 47 = 6^{2} \bmod 47 = 36$$

이것을 계산하는 것은 승산과 모듈러 연산을 할 줄 아는 사람이라면 누구나 계산할 수 있는 쉬운 문제이다. 그러나 만일 36이 mod 47로 연산할 때 10의 몇 제곱이냐를 물

으면 이 답을 구하는 것은 쉽지 않은 문제이다. 32는 mod 47에서 10의 몇 제곱일까? 이것을 구하기 위해서는 아마 10의 모든 승산을 계산해서 32가 되는 것을 찾아야 할 것이다. 답은 12제곱인데 독자들은 답을 쉽게 구했는지 모르겠다.

만일 실제 암호시스템에서 사용하는 300자리 정도의 큰 정수를 주고 이산대수를 구하라고 하면 이것도 까마득한 문제가 될 것이다. ElGamal 공개키 암호, DSA 전자서명 등은 이산대수문제의 어려움에 기반을 두어 고안된 공개키 암호 알고리즘이다.

2-4-2-3 공개키 암호의 해독

물론 이런 어려운 문제들을 푸는 데 있어서 위의 예에서 보인 것처럼 모든 공간을 검색해서 찾는 전수검사로만 할 수 있는 것은 아니며 이보다 빠르게 답을 찾는 방법이 연구되고 있다. 암호학자 또는 암호해독학자들은 소인수분해 문제, 이산대수 문제를 얼마나 빨리 계산할 수 있는지, 쉬운 계산 방법이 있는지에 대해 연구를 계속하고 있다. 이러한 암호해독 문제가 중요한 이유는 우리가 사용하는 암호시스템이 실제로 안전하지에 대한 확신을 가질 수 있어야 하기 때문이다. 만일 현재 사용하고 있는 암호 알고리즘이 공격에 대해 안전하지 않다면 안전한 수준의 암호 알고리즘으로 바꾸어 사용해야 하는 것이다. 이런 암호해독 분야의 사례로서 비교적 최근에 발표된 RSA-200의 해독사례를 소개한다.

RSA-200은 RSA 사에서 내건 소인수분해 공모 문제로서 다음과 같은 663비트의 수, 십진수로 200자리의 큰 숫자의 소인수분해를 구하라는 문제이다.

RSA-200 = 27,997,833,911,221,327,870,829,467,638,722,601,621,070,446,786,
955,428,537,560,009,929,326,128,400,107,609,345,671,052,955,360,
856,061,822,351,910,951,365,788,637,105,954,482,006,576,775,098,
580,557,613,579,098,734,950,144,178,863,178,946,295,187,237,869,
221,823,983

이 문제는 오랫동안 풀리지 않고 있었는데 소인수분해 기술의 발전에 따라 2005년 5월 9일 F. Bahr, M. Boehm, J. Franke, T. Kleinjung가 마침내 소인수를 찾아냈다. (http://www.loria.fr/~zimmerma/records/rsa200 **참조)** 그들은 GNFS(General Number Field Sieve)라는 방법을 이용하였으며 네트워크로 연결된 수많은 컴퓨터들의 계산량을 동원하여 2.2 GHz Opteron CPU 컴퓨터의 55년치 계산량에 해당하는 계산 후에 다음과 같은 두 개의 소인수를 찾아냈다.

p = 3,532,461,934,402,770,121,272,604,978,198,464,368,671,197,400,
197,625,023,649,303,468,776,121,253,679,423,200,058,547,956,528,
088,349

q = 7,925,869,954,478,333,033,347,085,841,480,059,687,737,975,857,
364,219,960,734,330,341,455,767,872,818,152,135,381,409,304,740,
185,467

시간이 충분히 있는 독자라면 이 두 숫자를 곱했을때 실제 위의 200자리 숫자가 나오는지 계산해 보는 것도 흥미있을 것이다.

이 사례는 663비트, 200자리의 숫자를 소인수분해 하는 문제는 더 이상 불가능한 문제가 아니며 많은 계산 량을 동원하면 풀릴 수도 있기 때문에 안전한 암호 시스템으로 사용할 수 없다는 것을 말해준다. 거꾸로 말하면 소인수분해 문제를 실제로 풀기 위해

서는 이만큼의 많은 계산량이 소요된다는 것을 보여주고 있다.

2-4-3 공개키 암호 알고리즘

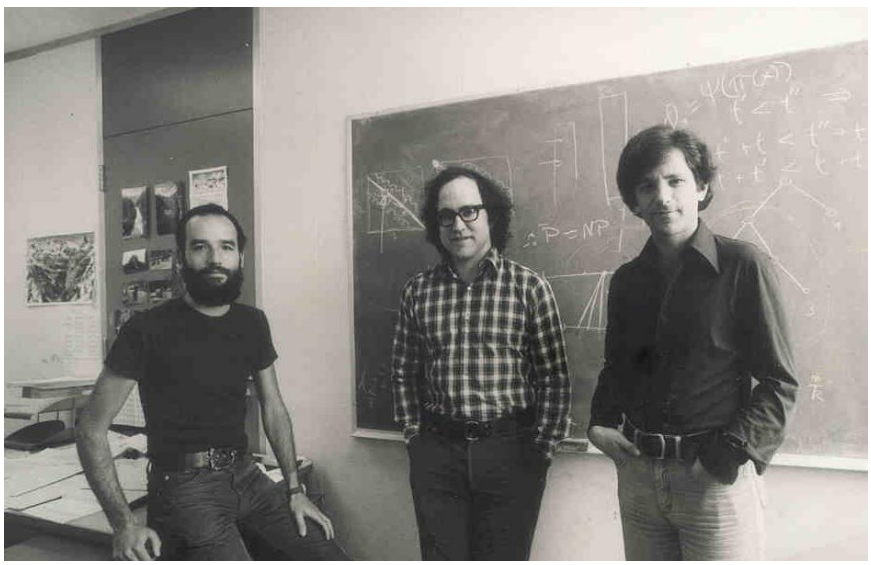
2-4-3-1 RSA 암호

RSA 암호는 1978년 MIT의 리베스트Rivest, 샤미르Shamir, 에이들맨Adleman에 의해서 제안된 공개키 암호 알고리즘으로 앞서 소개한 소인수분해 문제의 어려움에 이론적 기초를 둔 알고리즘이다. 설명하거나 이해하기에 가장 간단한 알고리즘이면서도 안전성이 높아서 현재까지도 제일 많이 사용되는 공개키 암호 알고리즘이다. 이것의 키생성, 암호화, 복호화 알고리즘은 다음과 같다.

■ 키생성
- 두 개의 큰 소수 p와 q를 선택하고 이들의 곱 n=pq를 계산한다.
- $\phi(n)=(p-1)(q-1)$을 계산한다.
- n과 서로소인 e를 선택하고 $de=1 \bmod \phi(n)$를 만족하는 수 d를 계산한다.
  (n, e)는 공개키로 공개하고 d는 개인키로 비밀히 보관한다.

■ 암호화
- 메시지 m을 암호화하기 위하여 송신자는 수신자의 공개키 (n, e)를 이용하여 다음과 같이 암호화한다. $c = m^e \bmod n$
- c를 메시지 m에 대한 암호문으로 수신자에게 전송한다.

■ 복호화
- 수신자는 자신의 비밀키 d를 이용하여 암호문 c를 다음과 같이 복호화 한다.
  $c^d \bmod n \rightarrow m$



<그림 > RSA 알고리즘을 개발한 리베스트(가운데), 샤미르(좌), 에이들맨(우)

[박스원고입니다]

RSA 암호는 어떻게 만들어졌나?

RSA란 리베스트Rivest, 샤미르Shamir, 에이들맨Adleman이라는 세 명의 이름에서 머리글자를 따서 만든 명칭이다.

리베스트는 오늘날 널리 사용되고 있는 '공개키'와 '개인키'를 이용하는 'PKC(Public Key Cryptography, 공개키 암호)'라는 혁명적인 개념을 도입한 스탠포드 대학의 디피와 헬만, 버클리 대학의 머클의 작업에서 영감을 얻고, MIT의 동료 교수인 에이들맨과 인도에서 온 샤미르를 설득해서 '공개키'와 '비밀키'라는 환상적인 개념을 실제 세상에 존재하는 알고리즘으로 구현했다.

디피, 헬만, 머클이 인류가 수천 년 동안 유일한 방법이라고 여겨왔던 암호화의 방식을 근본적으로 뒤바꿔 놓은 혁명적인 개념을 탄생시켰다면 세 명의 젊은 학자가 개발한 RSA 알고리즘은 혁명적인 개념을 현실화한 것이라 할 수 있다. (*자료출처 : 임백준, 누워서 읽는 알고리즘/한빛미디어, 2003)

RSA 암호에서 메시지와 암호문은 n보다 작은 정수로 표시되며 암호화와 복호화는 단순한 모듈러 승산으로 계산되는데 이 계산은 보기보다 많은 시간이 소요되는 연산이다. 하드웨어, 소프트웨어의 구현조건에 따라 다르겠지만 암호화 속도가 빠른 블록암호에 비해 공개키 암호는 100배~1000배 정도 더 많은 시간이 소요된다. 그러므로 공개키 암호 알고리즘은 많은 양의 데이터를 암호화하는 용도에는 적합지 않으며 블록암호에 사용되는 세션키를 안전하게 전달하는 용도에 주로 사용된다.

RSA 암호 알고리즘의 안전성은 사용되는 키의 길이에 따라 달라진다. 보편적으로 사용하는 키의 길이는 512비트, 768비트, 1024비트, 2048비트 등인데 사용자가 필요에 따라 키의 길이를 선택하여 사용할 수 있다. 키의 길이가 짧으면 암호화 속도가 빠르지만 안전성이 문제가 되며 키 길이가 길면 안전성은 높지만 암호화, 복호화 속도가 크게 느려지게 된다. 앞에서 살펴본 바와 같이 663비트의 길이를 가지는 숫자가 소인수분해 되었으므로 512비트, 768비트의 키 길이는 현재의 컴퓨팅 환경에서 안전하다고 볼 수 없으며 1024비트 이상의 키를 사용해야 한다.

2-4-3-2 엘가말 암호

이산대수 문제의 어려움에 기반한 최초의 공개키 암호 알고리즘은 1984년 스탠퍼드 대학의 암호학자 테하르 엘가말Tehar ElGamal이 제안한 엘가말ElGamal 암호 방식이다. 이것의 키생성, 암호화, 복호화 알고리즘은 다음과 같다.

■ 키생성
- 큰 소수 p를 선택하고 생성자 g를 선택한다.
- 비밀키 x를 선택하고 공개키 $y = g^e \bmod p$를 계산한다.
- (y, g, p)를 공개키로 공개하고 x는 비밀키로 안전하게 보관한다.

■ 암호화
- 메시지 m을 암호화하기 위해 난수 k를 선택하고 수신자의 공개키 (y, g, p)를 이용하여 다음을 계산한다.

$$r = g^k \bmod p,\quad s = my^k \bmod p$$

- 암호문 (r, s)를 수신자에게 전송한다.

■ 복호화

- 수신자는 자신의 비밀키 x를 이용하여 다음과 같이 복호화 한다.

$$s/r^{x} \bmod p \rightarrow m$$

엘가말 암호 알고리즘으로 암호화하면 메시지의 길이가 두 배로 늘어나는 특징이 있다. 그런데 암호화할 때 난수 k를 이용하므로 같은 메시지에 대해 암호화하여도 암호화 때마다 서로 다른 암호문을 얻게 되는데 이것은 정보보호 측면에서 큰 장점이 된다. RSA 암호 알고리즘에서는 난수를 사용하지 않기 때문에 같은 메시지에 대한 암호문은 항상 같다는 특징이 있는데 이것은 공격자가 암호문을 복호화하지 않고도 평문을 추측할 수 있는 단점이 있다. 그러므로 실제 적용시 RSA 알고리즘은 난수를 사용하는 OAEP(optimal asymmetric encryption padding)이라는 난수화 패딩 알고리즘과 함께 사용된다.

2-4-3-3 타원곡선 암호

최근 많은 연구가 이루어지고 있는 타원곡선 암호는 안전성이 높고 속도가 빨라서 새로운 공개키 암호 방식으로 각광받고 있다. 타원곡선 암호의 해독에는 그 특성상 소인수분해 문제, 이산대수 문제를 해독하기 위한 기존의 효율적인 해독방식이 적용되지 않는 것으로 밝혀져서 더 짧은 키를 안전하게 사용할 수 있고 이에 따라 같은 안전도를 갖는 공개키 암호 시스템을 구현한다고 할 때 암호화, 복호화 속도도 크게 빨라질 수 있다.

예를 들면 RSA의 1024비트의 키와 타원곡선 암호의 160비트의 키는 같은 수준의 안전도를 갖는 것으로 알려져 있다.

2-4-4 전자서명

지금까지 소개한 암호 알고리즘을 이용하면 인터넷과 같은 공개된 통신망에서도 정보를 비밀스럽게 주고받을 수 있다. 그런데 이러한 기밀성만 가지고 상거래가 이루어질 수 있을까? 비대면의 온라인상에서 상거래를 가능하게 하기 위해서는 상대방의 신분에 대한 확인, 문서에 대한 서명(계약서, 영수증) 등과 같은 것이 가능해야 한다. 종이문서에 사용하는 수기서명이나 도장을 이용한 서명은 인터넷 환경에서는 쓰기가 곤란하다. 그렇다면 수기서명이 포함된 전자문서를 법적으로 인정할 수 있을까? 전자문서는 쉽게 복사 가능하고 원본과 사본을 구별할 수 없으며 얼마든지 위조가 가능하다.

이것의 대안으로 암호기술의 인증기능을 이용해 전자문서에 서명이 가능하게 하는 기술이 전자서명이다. 공개키 암호가 가지는 인증기능을 이용해 서명자의 신분을 인증하고 서명대상인 전자문서의 인증을 동시에 수행하도록 하여 전자서명 효과를 얻을 수 있다.

공개키 암호의 기본적인 가정은 공개키에 대해 쌍이 되는 개인키(비밀키)는 사용자가 안전하고 비밀스럽게 보관한다는 것이다. 이러한 환경에서 개인키를 이용하여 어떤 서명 연산을 한다면 이것은 개인키를 가지고 있는 해당 사용자만이 가능한 작업이며 이 결과는 그 사용자의 서명으로 인정할 수 있는 것이다.

앞에서 설명한 RSA 공개키 암호를 거꾸로 사용하면 전자서명으로 사용할 수 있다.

■ 전자서명
- 메시지 m에 대한 서명은 $s = m^d \bmod n$으로 계산한다.

■ 서명검증
- 서명의 검증은 $s^e \bmod n = m$을 만족하는지 확인한다.

서명자는 자신만이 가지고 있는 개인키 d를 이용하여 메시지 m에 대해 모듈러 승산하여 s를 계산한다. 이러한 s를 계산할 수 있는 사람은 개인키 d를 가지고 있는 해당 사용자뿐이다. 이 서명은 공개키 e를 이용하여 누구나 검증할 수 있다.

[박스원고입니다]
안전한 전자서명 알고리즘의 요구조건
1. 위조 방지: 정당한 서명자 이외의 타인은 서명을 위조할 수 없어야 한다.
2. 사용자 인증: 전자서명으로부터 서명자가 누구인지 확인할 수 있어야 한다.
3. 부인 방지: 서명자는 문서에 서명한 사실을 부인할 수 없어야 한다.
4. 변조 방지: 한번 서명한 문서의 내용을 변경할 수 없어야 한다.
5. 재사용 방지: 하나의 서명은 다른 문서의 서명으로 재사용 할 수 없다.

전자서명이 실제로 전자상거래에 사용될 수 있기 위해서는 이러한 기술적인 요소뿐만 아니라 법적, 제도적인 뒷받침이 필요하다. 오늘날 국내에서 온라인상의 전자상거래가 크게 확산되고 있는 것은 1999년 2월에 제정되고 1999년 7월부터 시행된 전자서명법이 있었기에 가능했던 것이다. 즉 전자문서에 대해 만들어진 전자서명의 유효성을 종이문서에 대한 수기서명과 똑같이 법적으로 인정하게 되었다. 인터넷 뱅킹을 통해 전자자금거래를 하거나 인터넷에서 신용카드결제를 하는 경우 전자서명이 만들어지고 상대방에게 전달되기 때문에 이것이 거래의 유효성에 대한 증거물이 되는 것이다.

공개키 암호는 현대의 전자상거래 환경에 필수불가결한 암호방식이다. 사용자간 키를 안전하게 분배하고 전자서명이 가능하게 하는 등 다양한 정보보호 요구사항을 만족시키기 위해 사용되는 핵심 암호기술이다.

2-5 공개키에 대한 신뢰 부여 : 공개키 인증

2-5-1 인증서와 인증기관 vs 인감증명서와 동사무소

앞에서 공개키 암호는 현대의 전자상거래 환경에서 필요로 하는 다양한 정보보호 요구사항을 만족시키기 위해 사용되는 핵심 암호기술 중의 하나라고 소개했는데 공개키 암호가 실제로 사용될 수 있기 위해서는 공개키의 인증이라는 문제가 먼저 해결되어야 한다. 공개키 인증이란 특정한 공개키가 누구의 키인지 어떻게 확신할 수 있느냐 하는 문제이다.

개인키가 전자서명을 생성하는데 사용될 수 있다고 했는데 나의 개인키로 서명한 것이 나의 서명이라는 것을 남들이 인정하게 하려면 내 개인키와 쌍이 되는 공개키를 남

들도 나의 공개키라고 인정해 줄 수 있어야 한다. 만일 다른 사람이 키 쌍을 임의로 만들고 공개키를 나의 이름으로 등록해서 사용하려고 시도하는 경우 이것을 방지할 수 있어야 한다.

공개키를 분배하는 하나의 방법으로서 초기의 연구자들은 공개키 디렉토리라고 하는 장소에 개인의 공개키를 등록함으로써 분배하는 방법을 생각했다. 그런데 여기에 공개되는 공개키에 대한 무결성과 인증성이 문제가 되었다. 즉 공격자가 타인의 이름으로 공개키를 대신 등록하여 사용한다든가 이미 등록된 공개키를 바꿔치기 한다든가 하는 경우가 발생한다면 공개키 디렉토리에 공개된 공개키를 신뢰할 수 없게 된다.

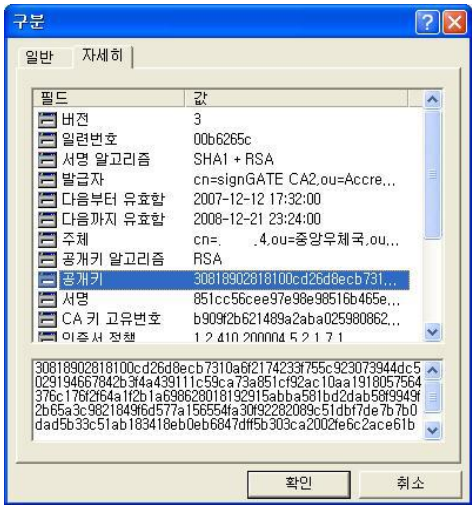
공개키에 대한 무결성과 인증성을 제공하는 현실적인 방법은 모두가 신뢰할 수 있는 권위 있는 인증기관을 도입하고 개인의 공개키에 대해 인증기관이 전자서명을 하도록 하는 것이다. 즉 인증기관이라고 불리는 모두가 신뢰하는 권위 있는 제3자가 나의 신분을 확인한 후 나의 공개키와 나의 신분을 확인할 수 있는 개인정보를 포함하는 문서에 서명함으로써 나의 공개키가 유효한 나의 공개키임을 인정하는 증명서인 인증서를 발급한다. 사용자가 인증기관을 신뢰한다면 인증기관이 발급한 인증서를 신뢰할 수 있고 나의 공개키의 유효성도 인정되는 것이다.

현실세계의 예를 하나 들어보자. 비즈니스 계약서와 같은 중요한 문서에는 인감도장을 찍게 되는데 여기에 찍힌 도장이 나의 인감도장이라는 것을 증명하기 위해 정부에서 발급하는 인감증명서를 첨부하게 되어 있다. 정부에서 발급한 인감증명서를 통해 계약서에 찍힌 도장이 나의 신분을 확인하는 인감도장이라는 것을 남들도 인정하게 된다. 전자상거래 환경에서 공개키는 인감도장에, 공인인증서는 인감증명서에, 인증기관은 정부에 해당된다고 볼 수 있다.

지식 정보화 사회에서 공개키는 인증기관이 발급하는 인증서를 통해 유효성이 확인된다. 국가에서 인정하는 인증기관을 공인인증기관, 이것이 발급하는 인증서를 공인인증서라고 부른다. 반면 개인이나 회사와 같은 조직이 정부의 인증을 받지 않고 자신들의 필요에 따라 인증서를 발급해서 사용할 수도 있는데 이것을 사설인증기관, 사설인증서라고 부른다. 공인인증서는 그 사회 전체에서 유효성을 인정받지만 사설인증서는 해당 그룹에서만 인정받을 뿐 공인인증서와 같이 모두에게 인정받을 수는 없다.

2-5-2 공개키 인증서의 발급과 폐기

공개키 인증서를 발급받기 위해 사용자는 자신의 컴퓨터 내에서 공개키/개인키의 키 쌍을 생성한다. 개인키는 자신의 컴퓨터에 안전하게 보관하고 공개키와 자신의 인증정보를 인증기관에 전송해 인증서 발급을 요청한다. 이때 자신이 개인키를 가지고 있음을 증명하기 위해 자신의 개인키로 서명한 문서도 함께 보낸다. 인증기관은 사용자의 신분을 인증하고 서명문을 확인한 후 사용자의 공개키와 사용자정보, 인증정보를 포함한 문서에 서명하여 인증서를 만들어 사용자에게 발급한다. 사용자는 자신의 개인키/공개키를 사용할 때 공개키가 자신의 것임을 확인시킬 필요가 있을 경우 이 인증서를 제공한다. 사용자의 개인키는 사용자만이 안전하게 보관하므로 비밀문서를 열어보거나 문서에 서명을 하는 것은 사용자만이 할 수 있다.



<그림 > 공개키 인증서의 사례

공개키의 사용에 있어서 마주치는 첫 번째 문제 중의 하나는 이미 발급한 인증서를 더 이상 사용할 수 없도록 하고자 하는 경우 어떻게 무효화 시킬 것인가 하는 것이다. 물론 공개키 인증서에는 유효기간의 정보가 있어서 유효기간이 지나면 자동으로 무효화 되지만 유효기간 이내에 발급을 취소해야 하는 경우를 말하는 것이다.

동사무소에서 주민등록증을 발급하거나 경찰서에서 운전면허증을 발급한 경우 이것을 무효화 시키려면 이것을 반납하면 된다. 주민등록증 위조 사건이 많지만 위조가 안 됐다고 가정할 때 반납과 동시에 주민등록증은 무효화되었다고 볼 수 있다. 그러나 공개키 인증서는 개인정보와 공개키에 대한 인증기관의 전자서명으로서 완전한 전자문서이다. 이것은 얼마든지 복사할 수 있어서 반납이 근본적으로 불가능하다.

공개키 인증서의 발급을 취소해야 하는 사유는 여러 가지가 있을 수 있다. 사용자가 회사를 퇴직해 공개키의 사용자격을 잃은 경우에는 더 이상 전자서명을 사용할 수 없도록 해야 한다. 개인키를 부실하게 관리해 남에게 노출된 것으로 의심될 경우에는 피해가 발생하지 않도록 공개키 인증서 발급을 취소해야 한다. 가장 많은 인증서 취소 사례는 개인이 개인키를 백업하지 않고 잃어버리거나 접근비밀번호를 기억하지 못해 사용할 수 없게 된 경우 인증서를 재발급 받기 위해 기존의 인증서를 취소하는 것이다.

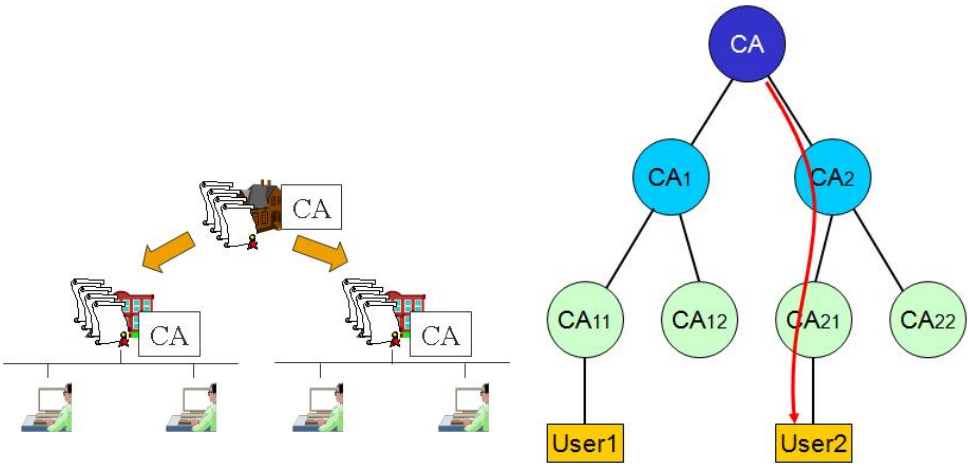
공개키 인증서의 취소를 위해 사용하는 방법은 소위 블랙리스트 방법이다. 즉 인증기관이 인증서취소목록(CRL: Certificate Revocation List)이라고 불리는 취소할 인증서의 리스트를 만들고 이것을 서명한 문서를 배포하는 것이다. 인증서의 사용자들은 최신의 CRL을 획득하여 사용하고자 하는 인증서가 취소되었는지 여부를 확인하고 사용해야 하는 의무를 지닌다. 공개키 인증서의 취소는 인증기관과 사용자 모두의 관점에서 볼 때 매우 부담스러운 업무이다. 인증기관은 인증서의 취소요청에 대해 빠르게 대응해야 하며 이것을 배포해야 한다. 사용자는 인증서 사용 전에 최신의 CRL을 획득하고 점검해야 하는 통신상, 계산상의 부담을 갖는다.

2-5-3 계층적인 인증체계

공개키 인증서를 발급하고 사용함에 있어서 특정 회사, 조직 등 작은 규모의 사회에서는 하나의 인증기관만으로 충분히 커버할 수 있을지도 모른다. 그러나 하나의 국가와

같이 많은 사용자들에게 서비스를 해야 하고 적용분야가 각기 다른 경우 하나의 인증기관으로 모든 인증서비스를 제공하는 것은 어렵기 때문에, 적용분야나 지역별로 별도의 인증기관이 서비스를 해야 한다. 이런 경우 인증기관이 서로 다른 사용자들 간에 상호인증이 가능한가 하는 문제가 있다. 인증기관 간에 인증이 확장되지 않는다면 사용자들은 같은 인증기관에 속한 사용자들끼리만 인증을 사용할 수 있을 뿐 다른 인증기관에 속하는 사용자들은 서로 신뢰할 수 없게 된다.

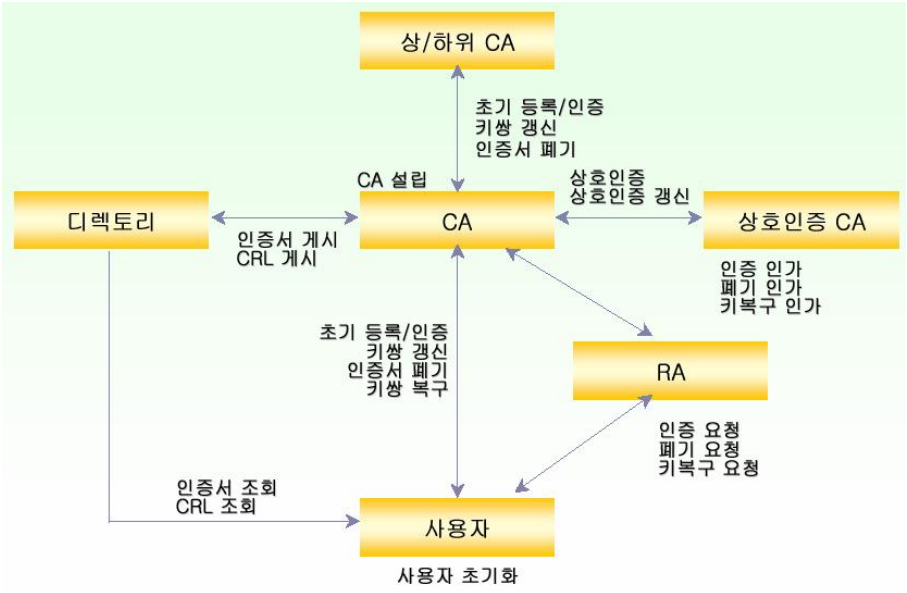
이 문제를 해결하기 위해 사용하는 방법은 계층적인 인증체계를 사용하거나 인증기관 간의 상호인증을 이용하는 것이다. 계층적인 인증체계란 상위의 인증기관이 하위의 인증기관에게 인증서를 발급하는 방식으로 계층적으로 인증을 확장하는 방법이다. 두 사용자가 발급받은 공개키 인증서의 인증기관이 서로 다르지만 이들 인증기관들이 공통의 상위인증기관으로부터 인증을 받았다면 두 사용자들 모두 상위인증기관을 신뢰하므로 서로 인증이 가능한 것이다. 상호인증이란 두 인증기관이 서로 인증서를 주고받음으로써 하위의 사용자들도 서로 인증이 가능하도록 하는 방식이다. 이렇게 복수의 인증기관들이 서로 인증을 확장하는 체계를 공개키 인증체계라고 한다.



<그림 > 계층적 인증체계 (두개의 그림 중 골라서 사용)

2-5-4 인증서비스의 업무흐름

인증기관이 사용자에게 인증서를 발급하여 공개키를 사용할 수 있게 하는 서비스는 많은 서비스 기관들과 이들 사이의 많은 업무들이 포함되는 복잡한 일이다. 인증기관, 등록기관, 사용자, 상위인증기관, 디렉토리 서비스 등 많은 참가자들이 키생성, 신분인증, 등록, 갱신, 폐기, 복구, CRL 발급 등 복잡한 업무절차를 거쳐서 인증서비스가 제공된다. 이러한 인증서비스가 제공되면 인터넷 환경에서 많은 응용분야에 적용할 수 있게 되므로 인증서비스는 지식정보화 사회를 구성하는 주요 기반구조라고 볼 수 있다. 이러한 의미에서 공개키 인증서를 사용할 수 있도록 만들어주는 서비스를 통칭하여 공개키 기반구조(PKI: Public Key Infrastructure) 라고 부른다.



<그림 > 공개키 기반구조에서의 참가자들과 업무흐름

여기서 각각의 기관들이 하는 업무절차들에 대해 간단히 살펴보자. 인증기관(Certification Authority, CA)은 사용자의 신분을 확인하고 사용자의 정보와 공개키에 서명을 함으로써 인증서를 발급해 주는 기관이다. 그런데 사용자의 신분을 확인하는 일은 인증서를 발급하는 일 이상으로 매우 중요하며 시간이 걸리는 일이다. 만일 사용자의 신분확인을 허술하게 하면 공격자가 타인의 명의를 도용하여 인증서를 발급받아 사용하게 될지도 모른다. 그러므로 신분확인은 주민등록번호, 계좌번호, 계좌비밀번호 등 기존의 사용자 인증정보들을 확인하거나, 사용자를 직접 대면 확인하는 등의 복잡한 업무절차가 필요하다. 많은 수의 사용자들에 대해서 하나의 인증기관이 이런 인증작업을 수행하는 것이 어려운 경우 등록기관(Registration Authority, RA)을 둘 수 있다. 은행이나 동사무소와 같이 대민창구에 직원이 있어서 사용자 신분확인을 효율적으로 할 수 있는 기관들이 등록기관으로 사용될 수 있다. 등록기관은 사용자 신분확인을 수행한 후 인증기관에 인증서 발급을 요청하고 발급된 인증서를 사용자에게 전달한다. 인증기관은 인증서 취소 요청이 있을 경우 CRL을 만들어서 디렉토리에 게시한다. 인증기관은 상위 인증기관으로부터 인증서를 발급받는다. 이러한 인증체계에서 가장 상위에 있는 인증기관을 최상위인증기관(Root CA)이라고 한다. 최상위 인증기관은 다른 인증기관으로부터 인증서를 발급받는 것이 아니고 스스로 인증서를 발급하여 게시한다. 인증기관은 다른 인증기관들과 상호인증서를 주고받음으로서 인증을 확장할 수 있다.

2-5-5 공개키를 내 맘대로 정한다 : ID 기반 암호

인증기관을 이용하는 공개키 인증방식은 인증기관으로부터 인증서를 발급받아 사용하는 것으로 암호화된 메시지를 보내거나 상대방의 전자서명을 확인하는 경우 상대방의 인증서를 가지고 있어야 하며 이것의 유효성을 검증해야 한다. 따라서 인증서가 미리 분배되어 있어야 하기 때문에 실제 응용환경에서 단점이 된다.

최근 이러한 인증서의 분배가 필요 없이 사용자의 이름, ID, 이메일주소, IP주소 등

사용자의 ID 정보를 그대로 공개키로 사용할 수 있는 ID기반 암호방식에 대한 연구가 활발히 진행되고 있다. 암호화된 메시지를 보내고자 하는 사람은 상대방의 인증서를 찾아올 필요가 없이 상대방의 ID를 공개키로 사용하여 암호문을 만들어 보내면 된다. 상대방의 전자서명을 받은 경우 상대방의 ID를 공개키로 사용하여 서명을 검증할 수 있다.

이런 장점이 있는 반면 ID기반 암호방식은 개인키를 개인이 스스로 만들 수 있는 것이 아니라 별도의 키 생성 기관이 만들어서 사용자에게 안전하게 전달해 주어야 한다는 단점이 있다. 키 생성 기관은 사용자의 신분을 확인하는 역할을 해야 하므로 인증기관과 비슷한 역할을 한다고 볼 수 있으나 누구나 볼 수 있는 인증서를 만들어 공개하는 것이 아니라 사용자만이 사용할 수 있는 비밀정보인 개인키를 만들어 사용자에게 비밀스럽게 전달해야 한다. 만일 키 생성 기관이 악의를 가지고 있을 경우에는 사용자가 할 수 있는 암호문의 복호화, 전자서명 등 모든 일을 할 수 있다. 그러므로 ID기반 암호시스템은 키 생성 기관을 전적으로 신뢰할 수 있는 소규모 조직에서 효율적인 암호시스템을 구축하고자 하는 경우에 사용할 수 있다.

공개키 기반구조는 공개키를 안전하게 사용하기 위한 신뢰구조로서 지식정보화 사회의 구현을 가능하게 하는 사회의 기반구조라고 생각할 수 있다. 전자상거래 등 개인의 신분인증이 필요하고 당사자의 책임이 수반되는 정보교환에는 공개키 인증이 필수적이다.

2-6 위조문서 식별하기 : 해쉬함수

2-6-1 해쉬함수란?

해쉬함수Hash Function는 임의의 길이를 갖는 메시지를 입력으로 하여 고정된 길이의 해쉬값을 출력하는 함수이다. 현재 사용되고 있는 SHA-1과 같은 표준 해쉬함수들은 160비트 내지 256비트의 해쉬값을 출력한다. 암호 알고리즘에는 키가 사용되지만 해쉬함수는 키를 사용하지 않으므로 같은 입력에 대해서는 항상 같은 출력이 나오게 된다. 이러한 함수를 사용하는 목적은 입력메시지에 대한 변경할 수 없는 증거값을 뽑아냄으로써 메시지의 오류나 변조를 탐지할 수 있는 무결성을 제공하는 목적으로 주로 사용된다.

해쉬함수는 전자서명과 함께 사용되어 효율적인 서명 생성을 가능하게 한다. 긴 메시지에 대해 서명을 하는 경우 전체 메시지에 대해 직접 서명을 하는 것이 아니고 짧은 해쉬값을 계산해 이것에 대해 서명을 하게 된다. 공개키 연산은 많은 계산 량을 필요로 하기 때문에 전체 메시지를 공개키 길이의 블록단위로 나누어 모든 블록에 대해 서명을 하는 것은 매우 비효율적이다. 그러므로 먼저 메시지를 입력으로 하여 160비트 내지 256비트의 짧은 해쉬값을 계산하고 이것에 대해 한번의 서명 연산을 하는 것이다. 이 계산값은 원래의 메시지에 대한 서명으로 인정된다.

해쉬값에 대한 서명이 원 메시지에 대한 서명으로 인정되기 위해서는 같은 해쉬값을 갖는 또 다른 메시지를 찾아내기가 계산적으로 어려워야 한다. 해쉬함수는 임의의 길이의 입력으로부터 짧은 길이의 해쉬값을 출력하므로 입력은 서로 다르지만 같은 출력을 내는 충돌이 반드시 존재한다. 만일 같은 해쉬값을 갖는 다른 메시지를 찾아내기가 쉽

다면 서명자는 자신의 서명에 대해 다른 메시지에 대한 서명이라고 우길 수 있을 것이다. 이것이 가능하다면 전자서명에 대한 신뢰가 불가능하고 전자거래에 사용할 수 없게 될 것이다. 그러므로 안전한 해쉬함수로 사용될 수 있기 위해서는 충돌을 찾아내기 어렵다는 특성을 가져야 한다.

2-6-2 해쉬함수의 안전성

안전한 해쉬함수가 가져야 하는 가장 기본적인 요건은 같은 출력을 내는 두 개의 서로 다른 입력, 즉 충돌을 찾기가 계산적으로 어려워야 한다는 것인데 이것을 충돌저항성Collision Resistance이라고 한다. 해쉬함수의 출력 길이가 길수록 충돌을 찾기가 어렵다는 것은 상식적인 이야기겠지만 효율성 측면에서는 출력 길이가 가능하면 짧은 것이 좋기 때문에 과연 해쉬함수의 출력 길이를 얼마로 하는 것이 효율적이면서도 안전한지는 안전성을 정확히 따져봐야 한다.

암호학자들은 해쉬함수의 충돌을 찾아내는 것에 많은 노력을 기울이고 있는데 이것이 해쉬함수의 안전성을 분석하는데 가장 중요한 요소이기 때문이다. 160비트의 출력을 내는 해쉬함수에 대해 공격자는 임의의 메시지들을 입력시키면서 해쉬값을 계산하여 같은 출력값이 있는지 찾아볼 것이다. 과연 어느 정도의 시도 후에 충돌을 찾아낼 수 있을까?

이것을 분석하는 이론 중에 생일공격Birthday Attack이라는 재미있는 이론이 있다. 사람의 생일은 1년이 365일이므로 365개의 가짓수가 있다. 사람이 많이 모이면 생일이 같은 사람이 있을텐데 몇 사람이 모이면 생일이 같은 사람이 존재할 확률이 0.5보다 커질까?

이 문제를 풀기 위해서는 다음과 같은 실험을 생각해본다.

1. 칠판에 365일을 써넣고 한사람씩 자신의 생일을 지워나간다.
2. 첫 번째 사람은 365일 중에서 무조건 자신의 생일을 지울 수 있으므로 365/365, 즉 1의 확률을 가진다.
3. 두 번째 사람은 364일 중에서 자신의 생일을 지울 수 있으므로 364/365의 확률을 가진다.

이러한 과정을 반복하여 생일이 같은 사람이 나타날 확률이 0.5보다 커지는 사람 수를 계산해보면 답은 23명이다. 즉 23명 이상이 모이면 생일이 같은 사람이 존재할 확률이 0.5 이상이 된다. 이정도 사람이 모이는 기회가 있다면 재미삼아 생일을 조사해 보는 것도 좋을 것이다. 이론적으로는 출력의 가짓수가 n이라고 하면 sqrt(n) 정도의 입력에 대해 조사해 보면 충돌이 발생할 수 있다. 비트수로 따지면 160비트의 출력을 내는 해쉬함수에 대해서는 2^80 개 정도의 서로 다른 입력값에 대해 해쉬값을 계산하여 비교해보면 충돌을 발견할 수 있다는 것이다. 현재의 컴퓨터를 이용하여 2^80 개 정도의 해쉬값을 찾아보는 것은 시간이 많이 걸려서 계산적으로 어렵기 때문에 안전한 해쉬함수로 생각하고 사용한다.

[박스원고입니다]

생일공격(Birthday Attack)이란?

생일 모순(birthday paradox)에 근거하여 해쉬 함수를 공격하는 방법. 생일 모순이란 임의로 모인 23명에서 생일이 같은 사람이 있을 확률은 50%이상이나 된다는 것으로, 실제로 그렇게 많은 경우수가 아니더라도 해쉬함수 출력으로부터 해쉬함수 입력을 찾아낼 수 있다는 이론에 근거하여 반복적인 대입으로 충돌이 발생하는 또다른 입력값을 찾아내는 것이다.

2-6-3 해쉬함수의 응용

앞에서 해쉬함수는 전자서명에 사용된다고 했는데 이것은 서명자가 특정 문서에 자신의 개인키를 이용하여 연산함으로써 데이터의 무결성과 서명자의 인증성을 함께 제공하는 방식이다. 메시지 전체에 대해 직접 서명하는 것은 공개키 연산을 모든 메시지 블록마다 반복해야 하기 때문에 매우 비효율적이며 따라서 메시지에 대한 해쉬값을 계산한 후 이것에 대해 서명함으로써 매우 효율적으로 전자서명을 생성할 수 있다. 서명자는 메시지 자체가 아니라 해쉬값에 대해 서명을 하였지만 같은 해쉬값을 가지는 다른 메시지를 찾아내는 것이 어렵기 때문에 이 서명은 메시지에 대한 서명이라고 인정된다.

송신자의 신분에 대한 인증이 필요없고 데이터가 통신 중 변조되지 않았다는 무결성만이 필요한 경우에는 해쉬함수를 메시지인증코드(MAC : Message Authentication Code)라는 형태로 사용할 수 있다. 송신자와 수신자가 비밀키를 공유하고 있는 경우 송신자는 메시지와 공유된 비밀키를 입력으로 하여 해쉬값을 계산하면 메시지인증코드가 된다. 메시지와 함께 메시지인증코드를 함께 보내면 수신자는 메시지가 통신 도중 변조되지 않았다는 확신을 가질 수 있다.

2-7 암호해독 기술과 암호알고리즘의 안전성

2-7-1 암호알고리즘의 안전성 평가 : 암호해독

지금까지 대칭키 암호, 공개키 암호, 전자서명, 해쉬함수 등 널리 사용되는 암호 알고리즘들을 간단히 소개했는데 이 알고리즘들이 과연 얼마나 안전한가 하는 의문이 들것이다. 과거 고대암호, 근대암호 시절에는 암호 알고리즘 자체를 감추고 암호문을 주고받음으로써 메시지의 비밀을 지키고자 하는 경우도 있었지만, 표준화된 컴퓨터, 표준화된 통신시스템을 이용하는 현재는 암호 알고리즘이 표준화되어 공개적으로 사용된다. 그 대신 암호 알고리즘에 사용되는 추가적인 비밀키 정보를 안전하게 보관함으로써 통신의 보안을 유지한다. 그러므로 우리가 사용하는 암호 알고리즘이 얼마나 안전한지 정확한 평가가 필요한 것이다.

영화에서는 가끔 해커들이 암호시스템의 비밀키를 찾아내는 장면이 매우 다이나믹하게 나타나기도 한다. 암호해독 시스템을 통해 비밀키가 한글자씩 순서대로 찾아지는 장면을 그래픽하게 보여주기도 하는데 이것은 실제 암호 알고리즘의 해독과는 매우 다른 비현실적인 장면이다.

암호학Cryptology을 암호설계Cryptography와 암호해독Cryptanalysis으로 분류하여 얘기하기도 하는데 그만큼 암호설계 뿐만 아니라 암호해독도 중요한 역할을 하고 있다는 것을 보여준다. 안전한 암호 알고리즘을 설계하기 위해서는 설계된 암호 알고리즘의

안전성을 정확히 분석할 수 있어야 가능한 것이다. 암호 알고리즘의 안전성을 담보하기 위한 기반기술로서 암호학자들은 암호해독에도 많은 노력을 기울이고 있다. 암호기술은 암호설계자들과 암호해독자들 사이의 경쟁을 통해 발전한다고 볼 수 있다.

2-7-2 적을 알고 나를 안다 : 공격자 모델 파악하기

암호 알고리즘의 안전성을 분석함에 있어서 공격자는 어떤 지식을 가지고 공격에 임하는지 고려할 필요가 있다. 공격자가 공격에 필요한 더 많은 지식을 가질수록 공격은 쉽게 이루어질 수 있다. 암호 알고리즘의 공격자가 가지는 지식의 양에 따라 다음과 같이 구분한다.

- o 암호문단독공격(Ciphertext Only Attack): 공격자는 암호문만을 가지고 있으며 이로부터 평문 또는 키를 찾아내고자 한다. 공격자가 사용자간의 통신을 관찰하면 암호문은 얼마든지 얻을 수 있으므로 이것은 매우 일반적인 공격상황이라고 볼 수 있다.
- o 기지평문공격(Known Plaintext Attack): 공격자는 특정 암호문에 대한 평문을 알고 있는 상황에서 키를 찾아내거나 다른 암호문에 대한 평문을 알아내고자 한다. 공격자는 평문과 암호문의 쌍을 알고 있으므로 이러한 지식을 이용하여 좀 더 유리한 상황에서 공격할 수 있다. 우리가 주고받는 메시지들은 추측하기 쉬운 내용들을 포함하기도 하는데, 예를 들어 메일을 보낼 때 “Dear Mr. Kim,"과 같이 관용적인 인사로 시작하는 경우 공격자는 암호문에 해당되는 평문을 얼마든지 추측할 수 있다. 또한 특정 형식의 파일을 암호화하여 전송하는 경우 파일의 헤더부분은 잘 알려진 메시지 블록들을 포함하는데 공격자는 이러한 정보들을 기지 평문으로 활용할 수 있다.
- o 선택평문공격(Chosen Plaintext Attack): 공격자가 암호장치에 얼마든지 접근할 수 있어서 선택된 평문을 입력하고 그에 대한 암호문을 얻을 수 있는 상황에서 복호화키를 찾아내거나 선택된 암호문에 대한 평문을 찾아내고자 한다. 암호알고리즘이 하드웨어로 구현되어 있고 키는 내부에 내장되어 있는 암호장치를 공격자가 가지고 있다고 가정해보자. 공격자는 원하는 만큼 선택한 평문을 입력시켜보고 그에 대한 암호문을 얻을 수 있다. 공개키 암호 알고리즘의 경우 공개키가 알려져 있으므로 공격자는 선택한 어떤 평문도 암호화하여 암호문을 얻을 수 있다. 이런 평문/암호문 쌍에 대한 지식을 기반으로 공격자는 좀 더 유리한 환경에서 암호시스템을 공격할 수 있다.
- o 선택암호문공격(Chosen Ciphertext Attack): 공격자가 복호화 장치에 접근할 수 있어서 선택한 어떤 암호문에 대해서도 평문을 얻을 수 있는 능력을 가지고 있는 경우에 키를 찾아내거나 선택된 암호문에 대해 평문을 얻고자 하는 공격이다. 복호화 능력을 항상 가지고 있다면 그 암호 시스템은 해독된 것과 마찬가지이겠지만, 이 공격 모델에서는 공격자가 복호화 하고자 하는 목표 암호문은 이런 방식으로 복호화가 허용되지 않는다고 가정할 때 공격자가 복호를 할 수 있느냐 하는 문제이다. 이 공격 모델은 목표 암호문을 제외한 어떤 암호문에 대해서도 평문을 알 수 있는 기회를 주므로 공격자에게 가장 많은 능력을 부여한 것이다.

공격자에게 부여되는 능력은 암호문단독공격이 가장 적고 선택암호문공격이 가장 큰 능력을 부여한 것이다. 만일 어떤 암호 알고리즘이 이러한 선택암호문 공격에 대해서도

안전하다는 것이 증명되면 암호문단독공격에 대해 안전한 암호 알고리즘보다 안전성이 더 높다고 생각할 수 있다. 따라서 높은 능력을 가지는 공격자에게도 안전한 암호 알고리즘을 설계하는 것이 최근의 암호 설계의 방향이다.

공격자의 모델을 능동적 공격자(Active Adversary)와 수동적 공격자(Passive Adversary)로 나누기도 하는데 수동적 공격자는 암호문을 읽기만 함으로써 기밀성에 대한 위협을 하는 반면에 능동적 공격자는 암호문의 변조, 삭제, 첨가 등을 시도함으로써 기밀성 이외에 무결성과 인증에 대한 위협을 가한다.

공격자는 암호 알고리즘을 공격함에 있어서 알고리즘 자체의 설계에 취약점은 없는지 다양한 방식으로 분석한다. 암호분석학자들은 알고리즘의 설계를 면밀히 검토해보고 취약점을 찾기 위해 노력한다. 한편으로는 무지막지한 방법으로 전수조사(Brute-Force Attack)라는 것을 사용하기도 하는데 이것은 목표 암호문에 대해 키 공간 전체를 시도해 봄으로써 의미있는 평문을 찾아낼 수 있는지 조사하여 키를 찾아내려는 방법이다.

암호 알고리즘이 아무리 취약점이 없게 설계되었다 하더라도 키 공간 자체가 작으면 컴퓨터를 이용하여 전수조사를 해봄으로써 쉽게 깨질 수 있다. 또 컴퓨터의 성능이 높아지고 분산컴퓨팅 기법을 사용하는 등 종합적인 컴퓨팅 능력이 높아지면서 과거에는 안전한 암호 알고리즘으로 인정받고 사용해 왔지만 더 이상 안전하지 않게 되는 경우가 있다. DES 알고리즘이 1977년 표준 알고리즘으로 선정된 후 오랫동안 사용되어 왔지만 지금의 컴퓨팅 환경에서는 안전하지 않기 때문에 2001년 AES 알고리즘이 새로운 표준으로 선정된 것이 좋은 예이다.

2-7-3 키 길이를 늘려라 : 블록암호 방식의 안전성

블록암호 알고리즘은 대용량 메시지를 빠르게 암호화하기 위해 오랫동안 사용되어 왔으며 암호학자들은 이것의 안전성을 분석하기 위해서도 많은 노력을 기울여 왔다. 1990년 비함Biham과 샤미르Shamir는 두 개의 평문 블록들의 비트의 차이(예: 1001과 1101의 차이는 1001 ⊕ 1101 = 0100이 됨)에 대응하는 암호문 블록들의 차이를 이용하여 사용된 암호열쇠를 찾아내는 차분해독법Differential Cryptanalysis을 발표했는데 이를 계기로 암호해독에 대한 연구가 크게 발전했다. DES설계자의 한 사람인 코퍼스미스Coppersmith에 의하면 설계 당시부터 이미 차분공격법에 대해 알고 있어, 그에 대한 대비책으로 S-함수 설계조건이 나왔다고 한다. 1993년 비함과 샤미르에 의하면, DES암호는 계산량 2^47의 차분공격으로 해독이 된다. 1993년에는 일본의 마쯔이Matsui가 DES S-함수의 입력 비트와 출력비트 사이의 상관도를 선형근사하여 해독하는 선형근사공격법Linear Cryptanalysis을 발표하여, 2^43개의 기지평문과 12대의 워크스테이션 Workstation으로 50일 만에 해독했다.

DES는 암호 설계의 안전성과는 별도로 키 길이가 56비트로 너무 짧기 때문에 전수공격에 의해 해독되었다. 1993년에 캐나다의 바이너Weiner는 키 검색 장치(Key Search Machine)를 설계하였는데 100만 달러를 들여 칩을 만들어 공격하면 3.5시간 안에 해독할 수 있다고 주장했다. 실제로 컴퓨터의 성능발달과 해독알고리즘의 개량으로, 전수조사에 의한 DES암호 공격이 실행되어 1997년 2월에는 RSA사에서 개최한 DES Challenge I에서 78,000대의 컴퓨터를 이용하여 96일 만에, 1998년 7월에는 DES Challenge II에서 250,000달러의 전용칩을 제작하여 56시간 만에, 1999년 1월 18일 DES Challenge III에서 1만 여대의 컴퓨터와 전용칩을 이용하여 DES암호를 22시간 15분 만

에 해독했다.

<그림 > DES 의 암호해독에 사용되었던 컴퓨터 시스템

이러한 전수공격에 대응하기 위해서는 키 길이를 늘려야 하는데 128비트 이상의 키 길이를 가지는 암호 알고리즘을 사용할 것을 권고하고 있다. 이를 위해 DES의 키 길이를 늘리는 효과적인 방법인 3중 DES암호를 이용하거나 새로운 국제표준 알고리즘인 AES 암호, 국내 표준 알고리즘인 SEED 암호를 사용할 것을 권고하고 있다.

2-7-4 출력값을 늘려라 : 해쉬함수의 안전성

해쉬함수는 메시지의 무결성과 인증성을 제공하기 위해 사용된다고 소개했는데 안전한 해쉬함수는 충돌을 찾기 어려워야 한다. 해쉬함수의 표준 알고리즘으로는 미국 표준인 SHA-1, 국내 표준인 HAS-160 등이 사용되고 있다.

생일공격에 의하면 n비트의 해쉬값을 출력하는 해쉬함수는 n^{1/2}의 계산량으로 충돌을 찾을 수 있다. 그런데 최근 중국의 왕Wang은 유로크립트Eurocrypt2005 학회에서 SHA-1 등의 해쉬함수에서 차분공격법을 사용하여 생일공격보다 빠르게 충돌을 찾을 수 있다는 것을 보여주었다.

이를 계기로 해쉬함수에 대한 연구가 매우 활발하게 진행되고 있으며 160비트 이상의 출력값을 내는 해쉬함수를 사용해야 한다는 방향으로 논의가 진행되고 있다.

2-7-5 어려운 수학문제를 풀어라 : 공개키 암호의 안전성

공개키 암호 알고리즘은 소인수분해 문제, 이산대수 문제 등 어려운 수학적 문제에 기반하여 하나의 키를 공개키로 공개하더라도 그것의 쌍이 되는 개인키는 찾기 어렵다는 특징을 사용하는 암호 방식이다. 이것에 대한 공격은 암호문으로부터 평문을 직접 찾는 방법 보다는 암호시스템의 이론적 기반이 되는 어려운 수학적 문제를 풀어서 공개키로부터 개인키를 찾아내는 방법으로 진행되어 왔다. 물론 공개키 암호 알고리즘의 안전성과 그의 이론적 기반이 되는 어려운 문제의 안전성이 같은 수준인가 하는 문제는 아직도 연구가 계속되고 있는 분야이다.

예를 들어 소인수분해 문제에 기반한 RSA 알고리즘에서는 두 개의 큰 소수 p, q를 선정하여 이들의 곱 n=pq를 계산하고 mod ϕ(n)에 대하여 서로 역수가 되는 e, d를 계산한 후 (n,e)를 공개키로 공개하고 d를 개인키로 비밀히 보관한다. 이때 사용되었던 p, q는 비밀키로 저장하거나 메모리에서 지워버린다. 공격자는 공개키인 (n,e)를 알고 있으며 개인키인 d를 찾아내고자 한다. 만일 공격자가 공개키 n을 소인수분해 할 수 있으면 p, q를 찾아내게 되고 개인키인 d도 계산해낼 수 있다.

암호학자들은 소인수분해 문제를 풀기 위해 노력을 기울여 왔으며 quadratic sieve, number field sieve, general number field sieve 등의 소인수분해 기법들을 연구하고 있다. 앞에서도 소개했지만 RSA-200은 RSA 사에서 내건 소인수분해 공모 문제로서 663비트의 수, 십진수로 200자리의 큰 숫자의 소인수분해를 구하라는 문제인데, 2005년 5월 9일 F. Bahr, M. Boehm, J. Franke, T. Kleinjung가 GNFS(General Number Field Sieve)라는 방법을 이용하여 네트워크로 연결된 수많은 컴퓨터들의 계산량을 동원하여 2.2 GHz Opteron CPU 컴퓨터의 55년치 계산량에 해당하는 계산 후에 소인수를 찾아냈다.

이 사례는 663비트, 200자리의 숫자를 소인수분해 하는 문제는 더 이상 불가능한 문제가 아니며 많은 계산량을 동원하면 풀릴 수도 있기 때문에 안전한 암호 시스템으로 사용할 수 없다는 것을 말해준다. 또한 소인수분해 문제를 풀기 위해서는 이만큼의 많은 계산량이 소요된다는 것을 보여주고 있다.

현재는 1024비트 이상의 공개키를 사용할 것을 권고하고 있다. 우리나라의 최상위 인증기관인 전자서명인증센터는 최고의 안전성을 추구해야 하는 기관인데 2048비트의 RSA 공개키를 사용하고 있다. 공개키 암호의 안전성을 높이기 위해서는 키 길이를 늘리는 것이 첫 번째 대응책인데 키 길이를 늘리면 계산량도 따라서 늘어난다는 단점이 있다.

한편 이산대수 문제에 기반한 공개키 알고리즘들도 널리 사용되고 있는데 이산대수 문제를 푸는 것은 소인수분해 문제를 푸는 것과 비슷한 수준의 계산량이 필요한 것으로 널리 인식되고 있다.

2-7-6 안전성을 수학적으로 증명한다 : 증명 가능한 안전성

암호 알고리즘의 안전성을 평가함에 있어서 오랫동안 전통적으로 사용해온 방법은 공격자에 의한 공격에 얼마나 오래 견디고 살아남느냐 하는 것이다. 즉 공격자에게 취약성이 발견되고 애초 설계했던 기준보다 빠르게 암호를 해독할 수 있다면 그 알고리즘은 안전하지 않은 것으로 평가되어 폐기된다. 반면 오랜 기간 동안 효율적인 공격방법이 발견되지 않으면 안전한 알고리즘으로 평가해 왔다. 그러나 이런 방법은 체계적인 평가방법이라고 볼 수 없으며 암호학자들은 안전성을 수학적으로 증명하는 방법에 관심을 가지고 있다. 이것을 증명 가능한 안전성(provable security)라고 한다.

이 방법은 제시된 암호 알고리즘이 어떤 보안목표를 가지는지 엄밀하게 정의하고 또한 공격자에게 어떤 능력을 허용하는지 정의한 후, 이러한 환경에서 제시된 암호 알고리즘이 안전하다는 것을 수학적으로 증명하는 것이다. 공격자에게 가능한 한 많은 능력을 부여해도 암호 알고리즘의 보안 목표가 달성될 수 있다는 것을 보여주고자 하는 것이다. 공격자에게 능력을 제공하는 모델로서 오라클(oracle)이라고 하는 개념을 도입한다.

예를 들어 공개키 암호 알고리즘의 안전성을 분석하기 위해서는 질의자(challenger)

와 공격자(attacker) 사이의 다음과 같은 게임을 생각해본다. 공격자는 공개키와 쌍이 되는 개인키를 가지고 있지 않지만 어떤 암호문에 대해서도 그것을 복호화하여 평문을 출력해주는 복호화 오라클에게 접근할 수 있는 권한을 가지고 있다.

[박스원고입니다]

오라클을 이용하는 공격자와 질의자 사이의 게임

1. 먼저 공격자는 복호화 오라클을 마음대로 시험해 볼 수 있다. 어떤 암호문이라도 복호화 오라클에게 질의를 하면 그에 해당되는 평문을 얻어 볼 수 있다.
2. 다음 단계로 공격자는 두 개의 동일 길이의 메시지 m_0, m_1을 선택하여 질의자에게 제시한다.
3. 그러면 질의자는 그 중 임의로 하나를 선택해서 그것을 공개키로 암호화하여 암호문을 만들어 공격자에게 제시한다. 두 개의 메시지 중에서 어느 것을 선택했는지는 공격자에게 알려주지 않는다.
4. 공격자는 복호화 오라클에게 암호문을 질의할 수 있는데, 단, 위의 단계에서 질의자가 제시한 암호문만은 질의할 수 없다.
5. 공격자가 최종적으로 질의자가 어떤 메시지를 선택했는지를 맞춘다면 공격자가 게임에서 이기게 된다.

이러한 게임에서 공격자가 이길 수 있는 확률은 얼마나 될까? 공격자는 복호화 오라클에게 질의할 수 있다는 능력을 이용하여 최선의 전략으로 유리한 질문들을 해 봄으로써 이 게임에서 이기려고 할 것이다. 공격자는 질의를 진행하면서 얻는 경험을 바탕으로 다음번에는 좀 더 유리한 질의를 만들 수 있다. 이러한 공격을 적응적 선택 암호문 공격(adaptive chosen ciphertext attack)이라고 한다. 암호 알고리즘이 안전하다면 이러한 게임에서 공격자가 암호문으로부터 평문이 둘 중 어느 것인지 구분할 수 없어야 한다. 이러한 안전성의 개념을 구별 불가능성(indistinguishability)라고 한다.

공격자에게 질의된 암호문을 직접 해독할 수 있는 능력을 제외한 가능한 한 많은 능력을 제공하더라도 공격자는 두 개의 메시지 중에서 어느 것을 암호화했는지 구별해낼 수 없다면 공개키 암호 알고리즘으로서 매우 안전한 알고리즘이라고 평가할 수 있다. 이러한 증명 가능한 안전성의 개념은 안전한 암호 알고리즘을 설계하기 위해 필수적인 안전성 평가 방식이다.

2-8 사용자 신분을 확인하라 : 인증

2-8-1 인터넷 통신을 위협하는 요소들

인터넷과 같은 개방형 네트워크를 통해 메시지를 주고받는 것은 마치 모든 사람이 들을 수 있는 시장과 같은 공개된 장소에서 큰 소리로 서로 얘기를 주고받는 것과 같다. 또한 상대방을 확인할 수 있는 대면상태가 아니므로 상대방이 맞는지 확신을 가지기 어렵다. 이런 환경에서의 통신에서는 공격자가 다음과 같은 여러 가지 불법적인 위협을 가할 수 있다.

[박스원고입니다]

인터넷 통신의 위협요소

o 메시지 노출: 메시지의 내용이 상대방이 아닌 공격자에게 노출될 수 있다.

o 위장: 공격자가 정당한 송, 수신자인 것처럼 행동하여 불법적인 메시지를 주고받을 수 있다.

o 내용 변조: 공격자가 메시지의 내용 일부 또는 전체를 삽입, 삭제, 변경 등을 할 수 있다.

o 순서 및 시간 변경: 공격자가 메시지의 송신을 지연시키거나 순서를 다르게 보내거나 하여 송신자의 의도와는 다른 결과를 초래할 수 있다.

o 트래픽 분석: 공격자가 사용자들 사이의 트래픽 형태를 분석함으로써 유용한 정보를 얻을 수 있다.

o 부인: 공격자의 불법적인 활동에 따라 메시지의 송, 수신 사실이 부인될 수 있다.

그러므로 개방형 네트워크에서 신뢰할 수 있는 통신을 하기 위해서는 기밀성뿐만 아니라 통신 당사자가 상대방의 신분에 대한 확신을 가질 수 있어야 하고 또한 전송되는 메시지가 변조되지 않았다는 확신을 가질 수 있는 방법이 제공되어야 한다.

<그림 > 비대면 통신에서의 상호 신뢰?

2-8-2 인증이란?

인증Authentication이란 정보의 주체가 되는 송신자와 수신자 간에 교류되는 정보의 내용이 변조 또는 삭제되지 않았는지, 그리고 주체가 되는 송, 수신자가 정당한지를 확인하는 방법을 말한다. 인증이라 하면 사용자 인증과 메시지 인증으로 나눌 수 있다. 사용자 인증(user authentication)이란 네트워크 상에서 사용자가 자신이 진정한 사용자라는 것을 상대방에게 증명할 수 있도록 하는 기능을 말한다. 반면 제삼자가 위장을 통해 사용자 행세를 하는 것이 불가능해야 한다. 메시지 인증(message authentication)이란 전송되는 메시지의 내용이 변경이나 수정되지 않고 본래의 정보를 그대로 가지고 있다는 것을 확인하는 것을 말한다. 전송되는 메시지에 대해 메시지가 변경되지 않았다는 사실과 누가 보낸다는 사실을 함께 증명하는 것은 전자서명을 통해 이룰 수 있다.

사용자 인증은 신원확인(identification)이라고도 하는데 서버에 로그인하는 경우 등

사용자의 신분을 확인하고 정보서비스를 이용할 수 있는 권한을 부여해주기 위해 사용된다. 일단 사용자 인증을 통과하면 원격 접속자가 해당 사용자로서의 모든 권한을 수행할 수 있기 때문에 중요한 서비스를 제공하는 서버의 입장에서는 원격 접속자에게 엄격한 인증절차를 거치도록 요구해야 한다.

2-8-3 사용자 인증을 위한 접근방법

사용자가 서버에게 자신의 신원을 증명하는 방법으로는 사용자가 가지고 있는 지식을 이용하는 방법, 사용자가 가진 물건을 이용하는 방법, 사용자 자신의 신체적인 특징을 이용하는 방법 등이 있다.

사용자가 알고 있는 것을 이용하는 방식은 패스워드를 이용하는 방식이 대표적인 사례이다. 이것은 사용자의 기억에 의존하기 때문에 값싸고 편리하게 사용할 수 있지만 패스워드를 안전하게 관리하는 것이 어렵고 패스워드가 공격자에게 누출되면 공격자도 사용자와 똑같은 행위를 할 수 있다. 질의-응답형 인증프로토콜을 이용하는 방식은 서버에서 사용자에게 사용자만이 대답할 수 있는 어려운 질문을 하고 사용자가 이에 맞는 대답을 하게 되면 인증하는 방식이다.

흔히 포털 사이트에서 사용자 등록시 질문과 그에 대한 대답을 등록하도록 되어 있는데 패스워드를 잃어버린 경우 서버는 이 질문을 하게 되고 사용자가 등록된 대답을 입력하면 인증을 통과하여 패스워드 변경 페이지로 접근할 수 있게 된다. 은행에서 사용하는 보안카드 방식은 이 두 가지 방식이 결합된 방식이라고 볼 수 있다. 넓은 의미에서 암호를 사용하는 인증방식은 이러한 범주에 들어간다고 볼 수 있다.

사용자가 소유하고 있는 것을 이용하는 방식은 주민등록증, 여권, IC카드 등 사용자가 소유하고 있으며 위조하는 것이 어려운 보안토큰을 이용하는 방식이다. 서버의 인증요구에 대해 사용자는 이런 보안토큰을 소유하고 있음을 보여줌으로써 인증을 얻을 수 있다. 이 방식은 사용자에게 특정 보안토큰을 발급해야 하기 때문에 경제적인 부담이 있고 사용자는 이것을 안전하게 보관해야 하는 의무가 있다.

사용자 자신의 신체적 특징을 이용하는 방식은 지문인식, 홍채인식, 얼굴인식, 음성인식, 동적서명인식 등 소위 생체인식(biometrics)을 이용하는 방식이다. 이러한 신체적 특징은 사용자간 구별이 쉬우며 반면 위조하기는 어려워야 한다. 이러한 방식은 값비싼 하드웨어 시스템을 도입해야 하는 경제적인 부담이 있다. 또한 정당한 사용자를 인식하지 못하거나 불법사용자를 정당한 사용자로 인식하는 등 오차가 있을 수 있다는 것을 고려해야 한다.

여기서 소개한 인증 방식들은 각각 장단점이 있어서 어느 한 가지 방식에 전적으로 의존하는 것보다 여러 가지 방식들을 결합해서 사용하는 사례들이 많이 있다. 이러한 방식을 멀티팩터 인증시스템이라고 하는데 예를 들면 IC카드를 사용하는 시스템에서 사용자에게 패스워드 입력을 요구하도록 하는 것이다. IC카드는 위조가 어려운 안전한 하드웨어 시스템이라고 할 수 있지만 사용자의 IC카드가 분실되어 타인에게 넘어간 경우 아무나 사용할 수 있다면 큰 문제가 된다. 이럴 때 패스워드 입력이라는 추가 단계가 있다면 타인에 의한 불법적인 사용을 방지할 수 있다.

2-8-4 암호학적 인증기법

앞에서 구분한 인증기법들은 인증에 사용하는 정보, 수단 등을 기준으로 한 단순한

분류라고 볼 수 있는데 이를 더욱 안전하게 사용하기 위하여 암호 기술이 여러 가지 방식으로 결합되어 적용되고 있다.

패스워드를 이용한 인증방식에서 패스워드가 서버에 그대로 저장된다면 서버가 해킹되는 경우 모든 패스워드가 해커에게 넘어갈 수 있기 때문에 서버에는 패스워드의 해쉬값을 저장하는 쉐도우패스워드shadow password 방식을 이용하기도 한다.

항상 고정된 패스워드가 사용되는 경우 통신망을 감시하고 있는 공격자가 통신내용을 저장해 놓았다가 추후의 인증에 재사용replay하려고 시도할 수 있는데 이를 방지하기 위해 일회용 패스워드 시스템이 사용되기도 한다. 사용자와 서버는 특정한 암호학적 관계를 가지는 정보를 공유하고 세션정보를 유지하며 이를 이용하여 일회용 패스워드를 생성하여 사용할 수 있다. 패스워드는 한번만 사용되고 바뀌므로 공격자는 통신망을 감시하고 현 세션의 패스워드를 알아내더라도 인증에 사용할 수 없다.

공개키 인증서를 이용한 인증에서는 사용자가 서버에게 전자서명을 제출하고 서버는 전자서명의 유효성을 검증함으로써 사용자의 신분을 인증할 수 있다. 공개키 인증이 확산되면서 이러한 인증방식이 확산되고 있는데 인터넷뱅킹 서비스에 로그인시 이런 방식이 이용되고 있다. 전자서명은 암호학적 안전성을 가지고 있으며 일반적 의미의 해커가 위조할 수 없다.

신분증, 출입증을 IC카드 형태로 발급하여 인증에 사용하는 사례가 증가하고 있는데 이 경우 사용자의 개인정보, 암호 등은 IC카드 내에 안전하게 저장되어 복제가 어렵게 된다. IC카드와 단말기는 질의-응답 프로토콜 형태로 신분확인을 할 수 있는데 이때 사용자의 비밀키는 IC카드 내부에서만 사용될 뿐 단말기 쪽으로 전송되지 않는다.

2-8-5 생체인식에 기반한 인증 방법

생체인식 기술은 살아있는 사람의 생리학적 특징 또는 행동적 특징을 기반으로 사용자를 인증하거나 인식하는 기술을 말한다. 널리 사용되는 생체의 특징으로는 다음과 같은 것들이 있다.

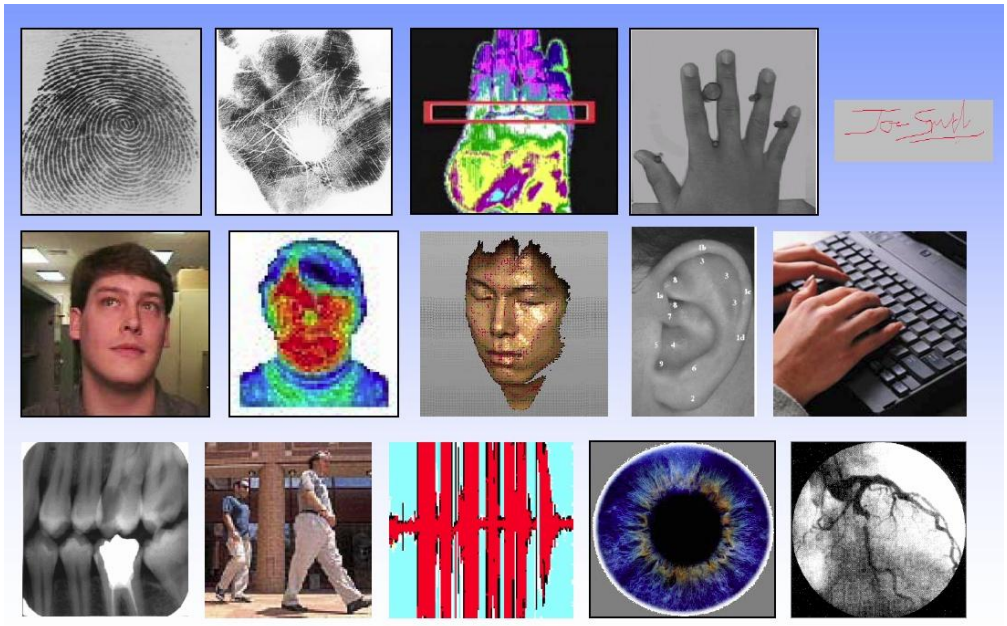
[박스원고입니다]

다양한 생체인식 기법들

- o 지문(fingerprint): 지문은 사람마다 다르고 평생 변화하지 않는다는 특징이 있는데 미뉴시아라고 불리는 분기점, 종단점 등의 특징점에 의해 다른 사람들과 식별한다. 데이터량이 적고 정밀한 인증이 가능하다.
- o 손모양: 손의 각 부분의 길이, 두께를 특징으로 사용자를 인증한다.
- o 얼굴: 사람은 얼굴모양을 보고 타인을 인식하는 만큼 가장 일반적인 인증방법이지만 기계가 영상처리를 통해 인증하는 것은 아직까지 어려운 점이 많다. 사진을 이용한 인증, 변장에 대한 대처 등도 필요하다.
- o 홍채(iris): 사람의 눈에 있는 홍채의 주름을 기반으로 인증하는 방법인데 매우 정밀한 인증이 가능하다. 반면 장비의 가격이 비싼 것이 단점이다.
- o 음성: 음성신호의 주파수 성분으로부터 성문데이터를 추출하여 인증하는 방법이다. 그러나 녹음에 의한 대리인증, 병이나 피로에 의한 음성변화, 잡음에 대한 취약성 등 아직 문제가 많다.
- o 동적서명특성: 필순, 필압, 쓰는 속도, 펜들 들어올릴 때의 운동 등과 같은 동적인

필적을 이용하여 식별하는 방법이다. 정확도가 높지 않다.

이들 생체인식 기술은 사용하는 기술에 따라 인식의 정확도, 비용, 응용환경이 다르며 상용화 할 수 있는 기술수준도 다르다. 가장 널리 보급된 기술은 지문인식 기술로 일반 가정의 출입문 통제에도 널리 사용되고 있다. 그밖에 자동차의 운전자 인식, 휴대폰의 지문인식, 마우스형 지문인식기 등이 있다. 홍채인식을 이용한 출입관리도 보급이 확산되고 있다.



<그림 > 여러 가지 생체인식기술

2-9 재미있는 암호 프로토콜 이야기

지금까지 암호기술들의 여러 가지 측면들을 소개했는데 실제 우리들의 온라인 생활에서의 정보를 보호하기 위해서는 어느 한 가지 암호기술만 사용되는 것보다 목적에 따라 여러 가지 암호기술들이 복합적으로 사용된다. 온라인에서의 특정 작업은 상호 신뢰하기 어려운 여러 참가자들의 복잡한 상호작용으로 이루어지기 때문이다. 여기에서는 암호기술이 현실 세계의 프로토콜과 결합되어 사용되는 방식들에 대해 소개한다.

2-9-1 프로토콜이란?

프로토콜이란 용어는 외교 분야에서 의정서(議定書), 의전(儀典) 등의 의미로 쓰이는데 외교행사에서는 엄격한 의전절차Protocol를 지켜야 한다고 말한다. 프로토콜이란 용어는 특히 통신에서 많이 사용하는데 통신 당사자들 사이의 메시지 전송규약을 의미한다.

현재의 표준 인터넷 프로토콜인 IP(Internet Protocol)는 인터넷으로 연결된 컴퓨터 사이의 패킷 전송시 송신자와 수신자의 인터넷 주소, 전송 서비스의 특성 등을 포함하

여 패킷의 헤더를 구성하는 규칙, 이들 정보에 기반하여 통신장비들이 제공해야 하는 서비스 등을 정의하고 있는데, 간단히 얘기하면 송신측과 수신측의 컴퓨터 사이에 메시지를 원활히 전달하기 위한 상호 약속을 의미한다. 송신측과 수신측이 같은 프로토콜을 사용해야 서로 통신이 가능한데, 비유해서 얘기하자면 송신측은 한글로 보내는데 수신측은 영문으로 이해하려고 한다면 서로 통신이 이루어지지 않을 것이다.

프로토콜의 의미를 좀 더 확장해보면 특정 작업을 수행하기 위한 다자간의 약속된 절차를 의미한다고 볼 수 있다. 우리들은 현실세계에서 다양한 프로토콜들을 수행하며 살아가고 있다. 상점에 가서 물건을 고르고 흥정하고 구입하는 것도 따지고 보면 매우 복잡한 프로토콜이다. 판매원과 고객이 적절한 프로토콜에 따라 흥정이 원만하게 이루어지면 거래가 이루어지겠지만 어느 한쪽이 마음이 상한다거나 손해 본다는 느낌이 든다면 거래가 이루어지기 힘들 것이다. 판매성적이 우수한 영업사원은 고객의 마음을 꿰뚫어보고 욕구를 만족시키는 능력이 탁월한 사람이다. 남녀간의 연애에도 프로토콜이 있다. 어느 한쪽의 일방적인 행동은 때로 상대방에게 거부감을 주고 아픔을 주고 관계를 끝내게 하기도 한다. 양쪽이 모두 상대방을 아껴주고 배려하고 사랑한다면 좋은 결실을 맺을 수 있을 것이다.

2-9-2 공정하게 빵 나누기 : 대면 프로토콜과 비대면 프로토콜

두 명의 어린이가 있는데 하나의 빵을 나눠먹으려고 한다. 그런데 빵을 나누는데 있어서 손해를 보아도 좋다고 생각하는 어린이는 별로 없을 것이다. 그럼 빵을 어떻게 나누면 서로 공평할까? 가장 쉬운 방법 중의 하나는 한 어린이로 하여금 빵을 나누게 하고 대신 다른 어린이가 먼저 선택하도록 하는 것이다. 이렇게 하면 빵을 나누는 어린이는 가능한 한 공평하게 나누려고 노력하게 될 것이다. 이것은 대표적인 대면 프로토콜의 하나이다.

<그림 > 공평하게 빵 나누기

우리가 현실세계에서 남들과 직접 만나 대면상태로 수행하는 여러 가지 작업들을 생각해보자. 상점에 가서 물건을 고르고 흥정을 하고 구입하는 일은 직접 눈으로 확인하고 비교해보아 최종 선택하며, 물건을 받는 것과 함께 지불을 하는데, 우리는 이미 익숙해 있지만 사실은 매우 복잡한 측면이 있는 대면 프로토콜이다. 이러한 쇼핑 과정에서 어떤 문제가 발생하거나 공평하지 않다고 생각되면 언제든지 프로토콜을 중단할 수 있다. 이것은 구매자와 판매자가 같은 시간에 같은 장소에 존재하기 때문이다. 대면해서 프로토콜을 수행하게 되면 프로토콜의 공정성을 엄격하게 감시할 수 있다.

대면 프로토콜에서도 문제가 발생하는 경우가 있는데 물건을 강매하는 경우도 있고,

물건을 교묘하게 속이기도 한다. 한 사람이 이미 약속된 내용을 지키지 않거나 하면 다른 사람에게는 피해가 발생하는데 이런 것을 사기라고 한다. 약속을 지키지 않으면 남들에게서 신뢰를 잃게 되고 더 이상 거래를 하지 못하게 될 수 있다.

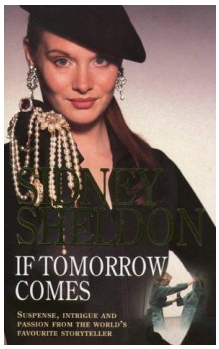
그러면 비대면의 통신을 통해서도 이러한 복잡한 사회활동을 하는 것이 얼마나 가능할까? 반대로 얘기하면 이러한 복잡한 사회활동을 비대면의 통신을 통해서도 가능하게 하려면 무엇이 필요할까? 통신을 통해 상거래와 같은 중요한 업무를 수행하는데 있어서 어떤 걱정들이 있을 수 있는지 살펴보자.

- 물건을 구입하기로 하고 돈을 지불했는데 물건이 배달되지 않는다면?
- 구매자가 지불한 돈이 위조지폐라면?
- 통신을 통해 지불되는 돈을 타인이 가로챈다면?
- 구매자가 어떤 물건들을 구입하는지 남들에게 쉽게 노출된다면?
- 현재 통신하고 있는 상대방을 신뢰할 수 있는가?
- 거래 진행 중에 예기치 않게 중단되는 경우 피해를 보지는 않을까?

이 외에도 많은 문제들이 제기될 수 있겠는데 대면 프로토콜에서는 직접 눈으로 확인을 함으로써 많은 부분들을 해결할 수 있다. 그러나 비대면 프로토콜에서는 이런 걱정들을 다른 방법으로 해결하여 참가자들이 프로토콜의 진행과정을 신뢰할 수 있어야 실제 사용될 수 있는 것이다.

2-9-3 통신 상대방을 믿을 수 있을까?

세계적인 베스트셀러 작가인 시드니 셀던(Sydney Sheldon)의 소설 “내일이 오면(If Tomorrow Comes)"에는 제프와 트레이시라는 두 사기꾼들의 이야기가 나온다. 이들은 세계 랭킹 1, 2위를 다투는 체스 선수들인 멜리코프와 네굴레스코 두 사람과 같은 유람선을 탔는데, 제프는 이들에게 트레이시가 실력있는 체스선수라고 소개하고 두 사람과 동시에 경기하면 적어도 한사람에게는 이기거나 비길 수 있다고 주장했다.



<그림 > 시드니 쉘던의 소설 “내일이 오면”

자신들의 자존심이 침해당했다고 분해하는 두 체스선수들은 드디어 많은 관객들의 내기 속에 제프가 설계한 체스경기에서 트레이시와 대결하게 되었다. 체스 초보로서 제프에게 한두시간 배운 것밖에 없는 트레이시가 사용한 방법은 두 방을 왔다 갔다 하면서 한사람의 움직임을 다른 사람에게 그대로 옮겨놓는 간단한 방법이었는데 결국 경기는 비기고 제프와 트레이시는 내기돈을 챙기게 되었다는 얘기이다.

이러한 공격을 암호학에서는 중간자 공격(man-in-the-middle attack)이라고 하는데

통신망의 중간에 있는 사람이 양쪽을 속이는 공격 방식이다. 또한 재전송공격(replay attack)이라고도 볼 수 있는데 한쪽의 통신내용을 저장했다가 다른 쪽에 그대로 써먹기 때문이다. 소설에서 트레이시는 두 방을 왔다 갔다 하면서 경기를 진행했는데 눈치 빠른 사람이라면 금방 속임수를 알아챌 수 있었을 것이다. 그런데 만일 이러한 경기가 통신을 통해서 동시에 이루어진다면 속임수를 알아채기 어렵다. 그만큼 게임을 신뢰할 수 없게 되고 통신을 통해서는 이렇게 돈이 걸린 중요한 게임을 하지 않을 것이다. 독자 여러분 중에서 혹시 인터넷 바둑을 통해 돈내기 게임하는 사람이 있다면, 상대방 뒤에 고수가 훈수하고 있을지 모른다는 생각을 해보라.

우리는 사람들이 많이 다니는 시장바닥에서 야바위꾼들이 현란한 속임수로 순진한 행인들의 주머니를 털어내는 장면을 많이 보아 왔다. 야바위꾼들은 세밀한 기술로 속임수를 쓰는데 순진한 눈으로는 알아채기 어렵다. 최근에 흥행한 "타짜"라는 영화에서는 화투에서도 정밀한 프로토콜을 통해 속임수가 이루어지는 장면들을 잘 소개하고 있다. 한사람의 물주로부터 돈을 털어내기 위해 나머지 사람들이 모두 짜고 치는 고스톱을 하기도 한다. 사행성 오락 게임장에서 승률을 조작하여 고객들에게 큰 피해를 입혔다는 뉴스를 자주 들어보았을 것이다. 마술이라고 하는 것은 드러내놓고 관객들을 속이는 기술이며 관객들은 속는 것에 즐거움을 느낀다.

이와 같이 눈으로 확인할 수 있는 대면 프로토콜에서도 속임수가 있는데 비대면의 통신상에서 이러한 게임을 한다면 당신은 그 게임을 신뢰하고 참여할 수 있는가? 이렇게 신뢰가 부족한 것은 프로토콜이 정당하고 올바르게 실행되고 있다는 것을 확인할 수 있는 방법이 없기 때문이다.

그러나 암호기술을 이용하면 이러한 비대면 게임에 대해서도 신뢰성을 보장하도록 할 수 있다. 사기를 치려는 참가자가 있다면 드러나게 하고 증거를 남김으로써 모든 참가자들이 건전하게 행동하도록 강제할 수 있다. 반대로 말하면 비대면의 통신 프로토콜에서는 암호기술을 이용하지 않고는 신뢰성을 제공하기 어렵다고 말할 수 있다.

2-9-4 만나지 않고도 믿고 통신할 수 있다 : 암호 프로토콜

암호 프로토콜이란 다자간의 비대면 통신 프로토콜에서 정보보호 기능을 제공하기 위하여 암호기술을 적용하는 프로토콜을 말한다. 이를 위해서는 목표로 하는 프로토콜에서는 어떤 정보보호 요구사항을 만족시켜야 하는지 좀 더 자세히 분석해 보아야 하며 이들 성질들을 어떻게 만족시킬 수 있는지 고려해야 한다. 목표 프로토콜에 따라 정보보호 요구사항은 달라지겠지만 일반적으로 다음과 같은 보안특성이 요구된다.

[박스원고입니다]

- o 공정성: 비대면 프로토콜에 참여하는 참여자들 사이에 어느 한쪽이 유리하거나 불리하지 않도록 공정성을 보장해야 한다. 어느 한쪽이 불리하다는 것이 알려지면 이런 비대면 프로토콜을 사용하려고 하지 않을 것이다.
- o 인증성: 비대면 프로토콜에서 상대방의 신분이 맞는지 주고받는 메시지가 위조되지는 않는지 확인할 수 있어야 한다. 상대방에 대한 신뢰는 프로토콜을 수행할 수 있는 가장 기본적인 전제가 된다.
- o 정확성과 검증성: 통신 프로토콜을 통해 약속된 절차가 맞게 이루어지고 있는지 정확성을 검증해 볼 수 있어야 한다.

o 프라이버시: 프로토콜 수행에 관한 정보가 제삼자에게 불필요하게 노출되지 않아야 한다. 이를 위해 암호화가 사용될 수 있다.

이러한 보안 요구사항들을 만족시키기 위해서는 암호기술에 의존해야 하며 암호기술을 사용하지 않고 이런 특성들을 제공하는 것은 불가능하다고 할 수 있다. 암호 프로토콜에 관한 연구가 매우 활발하게 이루어지고 있는데, 이것의 궁극적인 목표는 비대면의 통신에서도 대면세계 이상으로 참여자들 간에 신뢰성 있게 활동이 가능하도록 제반 관련기술을 제공하는 것이다.

2-9-5 인터넷으로 동전던지기 게임을 한다면?

암호 프로토콜에 대한 가장 첫 번째 이야기로서 우리가 흔히 즐기는 동전던지기 게임을 인터넷을 통해 할 수 있겠는가 하는 질문을 해볼 수 있다. 동전던지기 게임은 누가 동전을 던지든지 확률이 1/2인 게임으로 공평하게 승부를 가릴 때 사용되는 게임이다. 축구경기에서는 심판이 동전을 던져서 선공을 결정하는데 선공이 될 확률은 1/2이다. 이것은 양팀과 심판이 같은 시간과 공간에 존하기 때문에 눈으로 직접 확인할 수 있는 매우 공평하고 이해하기 쉬운 게임이다. 그러나 인터넷을 통해 통신하고 있는 두 사람이 동전던지기와 같은 게임을 할 수 있을까? 어느 한쪽이 속임수를 쓸 수 있는 가능성이 있거나 절차가 불공평하다면 인터넷을 통해서는 절대 동전던지기를 하려고 하지 않을 것이다.

이것을 가능하게 하기 위해서는 블랙박스와 같은 특징을 갖는 장치가 필요하다. 이것은 한번 집어넣으면 마음대로 열어볼 수 없고 내용을 바꿀 수도 없어야 하며 반면 나중에 열어볼 수 있어야 한다. 인터넷상에 이러한 블랙박스가 존재한다면 두 사람 A와 B는 다음과 같이 동전던지기 게임이 가능하다.

1. A는 임의의 숫자를 종이에 써서 블랙박스에 넣는다. 이 박스를 열 수 있는 키는 A만이 가지고 있다. 그리고 이 박스를 B에게 전송한다.
2. B는 이 박스에 들어있는 숫자가 홀수인지, 짝수인지 추측하여 A에게 말한다.
3. A는 블랙박스를 열 수 있는 키를 B에게 제공한다.
4. B는 블랙박스를 열어서 숫자를 확인한다. 홀수/짝수를 B가 맞추었으면 B가 이기는 것이고 아니면 A가 이기는 것이다.

이러한 특성을 갖는 블랙박스는 해쉬함수와 같은 일방향 함수를 이용하여 구현 가능하다. 해쉬함수는 그 특성상 입력 메시지로부터 해쉬값을 계산하는 것은 쉽지만 해쉬값이 주어졌을 때 그 값을 출력할 수 있는 입력 메시지를 찾아내는 것은 계산적으로 어렵다. 위의 동전던지기 게임을 해쉬함수 H를 이용하여 다시 정리해 보자.

1. A는 임의의 정수 x를 선택하여 이것의 해쉬값 H(x)를 계산하고 이 값을 B에게 전송한다.
2. B는 정수 x가 홀수인지, 짝수인지 추측하여 A에게 말한다.
3. A는 원래의 정수 x를 B에게 전송한다.
4. B는 H(x)를 계산하여 앞에서 받은 값과 같은지 확인한다. 홀수/짝수를 B가 맞추

었으면 B가 이기는 것이고 아니면 A가 이기는 것이다.

이렇게 암호기술을 이용하면 두 사람 A와 B가 서로 사기를 칠 수 없고 결과의 정확성을 검증할 수 있는 방식으로 비대면 통신상으로도 공정한 게임을 할 수 있다.

2-9-6 여러 가지 암호 프로토콜

암호기술을 이용해 여러 가지 유용한 프로토콜들을 안전하게 수행하는 재미있는 사례들이 많이 있다. 여기에서는 몇 가지 대표적인 사례들에 대해 간단히 소개하고자 하는데 비대면 통신 프로토콜에서 통상적으로 이루기 어려운 문제들을 암호학을 이용하면 어떻게 구현 가능한지 알 수 있다.

2-9-6-1 만나지 않고도 비밀키를 공유할 수 있다 : 키합의

A와 B 두 사람은 인터넷을 통해 안전한 통신을 하려고 하는데 이를 위해 안전한 암호알고리즘으로 암호화해서 메시지를 전송하기로 하였다. 그런데 이를 위해서는 먼저 암호 알고리즘에 사용할 비밀키를 공유해야 한다. 직접 만나서 비밀키를 합의한다면 문제가 해결되겠지만 요즘 같은 세상에 직접 만나서 비밀키를 합의한다는 것은 귀찮고 경쟁력에 뒤떨어지는 방법이다. 이런 목적에 사용할 수 있는 방법이 키합의**(key agreement)** 프로토콜이다. Diffie-Hellman 프로토콜을 이용한 키합의 방식이 대표적인데 이것은 이산대수 문제를 푸는 것이 어렵다는 사실에 기반하여 안전하게 키를 합의하는 방식이다. mod p를 사용하는 이산대수계에서 생성자를 g라고 할 때,

1. A는 임의의 난수 a를 선택하고 $y_a = g^a \bmod p$를 계산하여 B에게 전송한다.
2. B는 임의의 난수 b를 선택하고 $y_b = g^b \bmod p$를 계산하여 A에게 전송한다.
3. A와 B는 자신의 난수와 상대방으로부터 받은 정보를 이용하여 똑같은 정보 K를 계산할 수 있다.

   A: $y_b^a = (g^b)^a \bmod p = g^{ab} \bmod p \rightarrow K$

   B: $y_a^b = (g^a)^b \bmod p = g^{ab} \bmod p \rightarrow K$

여기에서 계산된 K는 A와 B만이 계산할 수 있는 비밀정보로서 두사람은 이것을 비밀키로 사용할 수 있다. A와 B 이외의 공격자들은 전송되는 정보인 y_a와 y_a를 얻을 수 있겠지만 이것으로부터 $K = g^{ab} \bmod p$를 계산할 수 없다. 이것을 계산적디피헬만문제(Computational Diffie-Hellman problem)라고 하는데 어려운 문제로 널리 알려져 있다.

이러한 키합의 방식은 SSL(secure socket layer)과 같은 세션암호화에 사용된다. 웹브라우저와 웹서버간에 결제정보와 같은 중요 정보를 전송하고자 하는 경우 이것이 공격자에게 노출되면 안되기 때문에 암호화해서 전송하게 된다. 그런데 암호화에 사용되는 비밀키를 결정하기 위해 웹브라우저와 웹서버는 서로 만날 수 없고, 이러한 키합의 방식을 이용하여 y_a와 y_a를 주고받은 후 비밀키 K를 계산하여 암호화를 할 수 있게 된다.

2-9-6-2 눈감고 서명하기 : 은닉서명

은닉서명**(blind signature)**은 사용자 A가 서명자 B에게서 자신의 메시지를 보여주지 않고 서명을 받을 수 있는 방법으로서 메시지에 대한 비밀성을 지키면서도 그것에 대한 타인의 인증을 받고자 하는 경우에 사용할 수 있다. 이것은 사용자 A가 자신의 종이문서 위에 먹지를 올려놓고 서명자 B에게 서명을 요구하여 서명을 받는 것과 같다. 은닉서명 프로토콜은 암호화 기술 전문가인 데이비드 촘David Chaum에 의해 제안되었는데 그는 이것을 전자화폐, 전자투표 등에 적용하여 사용자의 익명성, 프라이버시를 제공할 수 있다는 것을 제안했다.

먼저 RSA 암호를 이용한 은닉서명 프로토콜을 설명해보자. 서명자 B의 공개키는 (n, e), 비밀키는 d 라고 하자. 사용자 A는 메시지 M에 대해 서명자 B의 서명을 얻고자 한다.

1. 사용자 A는 임의의 난수 r을 선택하고 은닉된 메시지 $r^e M \bmod n$을 계산하여 B에게 전송한다.
2. 서명자 B는 A로부터 받은 메시지에 대해 서명하고 이것을 A에게 전송한다. $(r^e M)^d \bmod n = rM^d \bmod n$
3. A는 난수 r을 제거하여 서명자 B의 서명을 얻는다. $rM^d / r \bmod n = M^d \bmod n$

서명자 B는 은닉된 메시지에 서명하고 사용자 A는 은닉된 난수를 제거하여 최종서명을 얻게 되므로, 서명자는 자신이 실제로는 어떤 메시지에 서명을 했는지, 어떤 서명을 누구에게 해주었는지 구분할 수 없다.

<그림 > 데이비드 촘

이러한 은닉서명은 은행이 화폐를 발행하는 용도에 사용할 수 있다. 화폐라는 것은 발권은행의 서명이 있어야 유효성을 인정받을 수 있는데 은행이 일반적인 전자서명을 이용하여 전자화폐를 발행하면 그것을 사용하는 사용자는 어디에 사용했는지 모두 추적될 수 있다. 현실세계에서 사용하는 지폐는 유일성을 나타내는 일련번호도 있고 하지만 일반적으로 그것을 기록하면서 사용하지 않기 때문에 화폐를 어디에 사용했는지 프라이버시가 보장된다고 볼 수 있다.

그러나 전자화폐는 사용자의 컴퓨터에 저장되고, 이것을 받은 상점은 은행에서 환전하기 전까지는 데이터베이스에 안전하게 보관할 것이며, 은행에서도 또한 전자화폐의 사용 기록을 남기기 위해 데이터베이스에 보관할 것이다. 이렇게 되면 사용자가 전자화폐를 어디에 사용했는지에 관한 개인정보가 모두 추적될 수 있게 된다.

이러한 용도에 위의 은닉서명 방식이 사용될 수 있는데, 사용자는 은행에게 전자화

폐 인출을 요구하고 은닉서명 프로토콜을 이용하여 전자화폐를 발권 받으면 사용자는 유효한 전자화폐를 얻게 되지만 은행은 이 전자화폐를 누구에게 발권했는지 추적할 수 없게 된다.

은닉서명은 전자투표에도 사용될 수 있다. 현실세계에서 수행하는 투표용지 방식의 투표에서는 선거관리위원회가 도장을 찍은 유효한 투표용지를 제공하고 투표자가 여기에 기표를 하여 투표함에 넣게 되지만, 전자적인 방식의 투표에서는 이런 방식으로 투표용지를 만들 수 없다.

일반적인 전자서명으로 투표용지를 만들면 이것을 누구에게 제공했는지 기록으로 남기 때문에 투표자가 어떤 후보자에게 투표를 했는지 추적될 수 있다. 은닉서명 방식으로 전자투표를 구현하는 방식은 먼저 투표자가 기표를 한 메시지를 선관위에게 은닉하여 전송하고 선관위가 여기에 서명을 하여 돌려주면 투표자는 은닉을 해제하여 선관위가 서명한 투표용지를 계산해 내는 방식이다. 이런 방식으로 투표가 이루어진다면 투표자의 선택이 드러나지 않아 비밀투표의 원칙을 지킬 수 있다.

은닉서명은 메시지의 비밀성을 보장하면서도 메시지에 대한 타인의 인증이 필요한 경우 사용할 수 있는 유용한 암호 프로토콜이다.

2-9-6-3 비밀 정보를 안전하게 보관한다 : 비밀분산과 문턱암호

소수의 허가받은 사람만이 사용해야 하는 아주 중요한 정보가 있다고 하면 이것을 어떻게 해야 안전하게 보관할 수 있을까? 예를 들면 핵무기를 발사할 수 있는 장치를 동작시키는 키가 있다고 하자. 이 키를 잃어버리게 되면 핵무기를 발사할 수 없으므로 잃어버리지 않도록 안전하게 보관해야 한다. 그렇다고 키의 복사본을 여러 개 만들어 보관하는 것은 그 중에 하나라도 공격자들의 손에 넘어간다면 핵무기가 불법적으로 사용될 수도 있기 때문에 조심해야 한다.

또한 핵무기의 발사는 매우 중요한 결정사항이기 때문에 어떤 한사람의 결정만으로 발사할 수 있도록 해서는 안된다. 이런 경우에 비밀정보를 여러 개로 나누어 각자 다른 사람들에게 분산시켜 보관하고 비밀정보를 사용할 경우에는 분산된 정보를 모아서 사용하도록 하는 것을 비밀분산이라고 한다.

비밀분산(secret sharing)과 함께 문턱암호(threshold cryptography) 기법이 사용되기도 한다. 비밀정보를 여러 개로 나누어 보관하고 있는데 이 모든 정보가 모여져야만 사용할 수 있다면 분산 정보의 안전한 보관에 어려움이 있을 수 있다. 분산정보 중에서 일부가 훼손될 수도 있고 그것을 보관하는 사람이 협조하지 않을 수도 있다. 문턱암호란 비밀이 분산되어 보관되는 경우 그 중에서 몇 개 이상의 비밀정보가 모여지면 비밀정보를 복구할 수 있도록 하는 암호기법을 말한다. 예를 들어 비밀정보를 n개로 나누고 그 중에서 t개 이상이 모이면 원래의 비밀정보를 복구하여 사용할 수 있다면 이것을 (t, n) 문턱암호라고 한다.

영화에서도 핵무기를 발사하는 장면에서는 키를 나누어 보관하고 있던 여러 사람이 함께 동의하고 키를 모두 꽂아야만 비로소 핵무기가 발사되도록 하는 것을 보았을 것이다. 실제로 러시아, 미국의 핵무기 시스템은 이렇게 비밀이 분산되어 보호되는 시스템을 사용한다는 것이 알려져 있다.

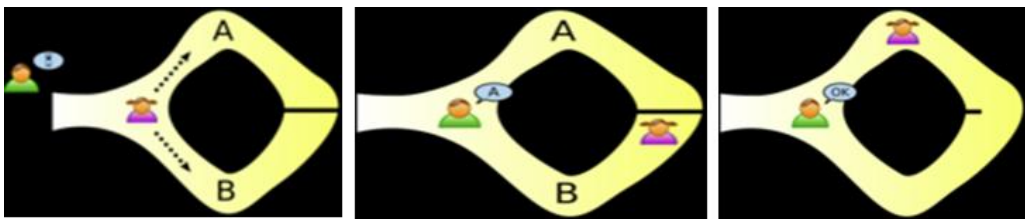
2-9-6-4 비밀을 보여주지 않고 남을 설득하기 : 영지식증명

영지식증명(zero knowledge proof)은 암호학에서 매우 중요한 역할을 하는 프로토콜이다. 수학책에서 배우는 정리(theorem)와 그것에 대한 증명(proof)을 생각해 보자. 수학자는 자신이 주장하는 정리가 옳다는 것을 남들에게 보여주기 위해 자신의 지식을 동원하여 엄밀한 수학적인 증명을 한다. 여기에서 사용되는 증명은 모든 과정과 정보를 공개하는 공개적인 증명이다.

그런데 우리의 현실세계에서는 정보를 공개하지는 못하지만 자신이 그것을 알고 있다는 것을 증명할 필요가 있는 경우가 많다. 자신이 가진 비밀정보를 보여주면 물론 그 비밀정보를 알고 있었다는 것을 증명할 수 있겠지만 한번 노출시키면 그 정보는 더 이상 비밀정보가 아니다. 자신의 정보를 노출시키지 않으면서 그것을 알고 있다는 것을 증명하는 것을 영지식 증명이라고 한다.

아라비안나이트에 나오는 "알리바바와 40인의 도적" 이야기를 알고 있을 것이다. 알리바바는 보물이 가득한 동굴의 문을 열 수 있는 "열려라 참깨"라고 하는 주문을 알고 있다. 물론 비밀정보를 혼자서만 기억하고 오랫동안 보물을 혼자서만 쓴다면 좋겠지만 알리바바는 입이 근질거려서 이것을 남들에게 자랑하고 싶어졌다. 그러나 이 주문을 남들에게 알려준다면 보물은 더 이상 자신만의 것이 아니다. 알리바바는 이 주문을 알려주지 않고 어떻게 하면 자신이 이것을 알고 있다는 것을 남들에게 자랑할 수 있을까?

<그림 >의 동굴 그림은 영지식 증명을 설명하기 위해 사용되는 대표적인 예이다. 갑순이는 동굴의 입구로부터 반대쪽에 있는 비밀문을 열 수 있는 암호를 알고 있다. 갑순이는 이 암호를 을돌이에게 알려주지 않고 이것을 알고 있다는 것을 증명하려고 한다. 어떻게 하면 이것이 가능할까?



(1)　　(2)　　(3)

<그림 > 영지식 증명을 설명하는 동굴 예제

갑순이와 을돌이는 다음과 같은 게임을 한다.

1. 을이는 동굴 입구에서 기다리게 하고 갑순이는 갈라지는 지점에서 A 또는 B의 방향으로 임의로 선택하여 들어가서 비밀문 앞으로 간다. 이때 을돌이는 갑순이가 어느 방향으로 갔는지 알 수 없다.
2. 을돌이는 동굴이 갈라지는 지점까지 와서 갑순이에게 A 또는 B의 어느 한 방향으로 나오도록 큰 소리로 말한다.
3. 갑순이는 을돌이의 요구에 따라 A 또는 B의 방향으로 을돌이에게 간다. 갑순이는 필요한 경우 암호를 이용하여 비밀문을 열고 반대편으로 갈 수 있다.
4. 1~3의 게임을 을돌이가 동의할 때까지 n회 반복한다.

여기에서 갑순이는 비밀문을 열 수 있는 암호를 알고 있으므로 을돌이가 원하는 방

향으로 항상 나타날 수 있다. 그러면 을돌이는 자신이 요구하는 방향에서 갑순이가 나타났을 경우 갑순이가 암호를 알고 있다는 것을 얼마나 믿을 수 있을까? 만일 갑순이가 비밀문의 암호도 모르면서 A 또는 B의 방향을 임의로 선택하여 갔다면 을돌이의 임의의 요구에 맞춰서 나타날 수 있는 확률은 1/2 밖에 되지 않을 것이다.

그러나 이런 게임을 n회 반복하여 갑순이가 모두 성공적으로 응답을 했다면, 갑순이가 암호를 모르면서 성공할 확률은 n번 모두 우연히 을돌이의 선택과 똑같은 선택을 하는 경우이며 확률은 1/2^n 밖에 되지 않는다. 그러므로 n번 모두 갑순이가 성공했다면 을돌이는 갑순이가 암호를 알고 있다는 것을 높은 확률로 확신할 수 있다.

이러한 영지식 증명 프로토콜은 통신 프로토콜에서 사용자가 자신의 비밀정보를 알려주지 않으면서 그것을 알고 있다는 것을 보여주는 목적으로 사용될 수 있다. 비밀정보를 알려주지 않으면서 증명을 할 수 있으면 같은 비밀정보를 반복해서 사용할 수 있게 된다. 사용자가 공개키 암호 시스템을 사용하는 경우 개인키는 자신이 안전하게 보관하면서 공개키는 인증기관으로부터 인증을 받아 오랜 기간동안 사용하게 되는데 자신의 비밀키는 어떤 경우라도 남에게 보여주지 않아야 한다. 비밀키가 노출된다면 발급된 인증서를 취소하고 새로운 키쌍을 만들고 새로 인증서를 받아서 사용해야 하는데 이것은 관리적 측면에서도 매우 부담이 큰 과정이다.

원격 서버에 로그인하는 경우 사용자는 패스워드를 직접 전송하는 것이 아니라 공개키 암호 시스템을 이용하여 다음과 같은 질의-응답(challenge-response) 방식으로 자신의 신분을 인증할 수 있다.

1. 사용자는 서버에 인증을 요청한다.
2. 서버는 사용자에게 사용자의 개인키를 이용하여 특정 계산을 해서 응답할 것을 요구한다.
3. 사용자는 자신의 개인키를 이용하여 요청된 계산을 수행하여 그 결과를 전송한다.
4. 서버는 사용자의 계산이 맞는지 사용자의 공개키를 이용하여 검증한다. 검증이 맞으면 로그인을 허용한다.

사용자는 자신의 개인키를 노출하는 것이 아니므로 반복하여 인증에 사용할 수 있다. 서버는 인증기관에 의해 인증된 공개키를 이용하여 검증을 하므로 사용자의 신분에 대해 확신을 가질 수 있다.

영지식 증명 기법은 두 사용자간의 상호작용을 통하여 비밀정보를 노출하지 않고도 그 정보를 가지고 있다는 것을 상대방에게 증명하는 방법으로서, 복잡한 과정을 거쳐야 하는 프로토콜을 수행함에 있어서 매 단계가 원래의 약속대로 잘 진행되고 있다는 것을 확신하게 하는데 이용할 수 있으며 프로토콜의 건전성, 검증성, 신뢰성을 보장하기 위한 용도에 사용된다.

2-9-7 암호 프로토콜의 응용

이와 같은 암호 프로토콜에 관한 연구는 실생활에서 사용되는 복잡한 프로토콜을 비대면 통신상에서 구현하기 위한 목적으로 사용될 수 있다. 현실세계의 화폐를 전자적인 방식으로 구현하는 전자화폐, 투표 및 개표행위를 전자적으로 구현하는 전자투표, 경매를 통신망을 통해 전자적인 방법으로 구현하는 전자경매 등은 좋은 사례인데, 다수의

신뢰기관, 중개기관, 사용자들 간의 엄밀한 절차를 따르는 복잡한 통신행위를 통해 원하는 목표를 이루게 된다.

이러한 프로토콜들은 작업목표에 따라 만족시켜야 하는 다양한 보안요구사항들이 있다. 예를 들면 안전한 전자화폐 시스템이라고 하면 사용자들이 한번 발급받은 화폐를 여러 번 사용하거나 하는 불법적인 행위를 방지할 수 있어야 한다. 안전한 전자투표 시스템이라고 하면 투표자가 어떤 투표를 했는지 노출되지 않아서 비밀성을 보장하여야 하고 투표값들이 정확히 개표되어야 할 것이다. 안전한 비밀경매 시스템이라고 하면 경매자의 입찰가격이 남들에게 드러나지 않으면서도 최고가를 제출한 낙찰자를 정확히 판정할 수 있어야 하며 최종결과에 대해 모두 인정할 수 있어야 할 것이다. 이런 목표 요구사항을 어떻게 만족시킬 수 있을까? 이를 위해서는 암호기술, 암호 프로토콜 기술을 이용해야 하며 이것을 사용하지 않고 이런 보안요구사항을 만족시키는 방법은 없다고 생각할 수 있다.

이미 우리는 이러한 복잡한 암호 응용 시스템들을 사용하고 있다. 버스카드, 지하철 카드 등은 일종의 전자화폐라고 볼 수 있는데 이것을 사용하는 절차를 살펴보면 암호기술이 사용되는 복잡한 프로토콜이라는 것을 알 수 있다. 인터넷 쇼핑몰을 통해 물건을 구입하고 결제하는 과정을 살펴보면 이것도 암호기술이 사용되는 복잡한 프로토콜이라는 것을 알 수 있다.

전자투표에 대해서도 많은 연구개발이 이루어지고 있는데 중앙선거관리위원회에서는 오래전부터 전자투표를 도입하려는 장기적인 계획을 가지고 추진해 왔다. 전자투표가 도입되면 편리성, 효율성, 빠른 개표, 투표 참여 확대 등 많은 장점이 있지만 선거 결과의 위조 가능성, 매수행위의 가능성, 결과의 검증성 등 문제점을 제기하기도 한다. 그러나 전자투표의 도입이 늦어지고 있는 이유는 이런 전자투표의 문제점들보다도 선거라고 하는 행위가 정당, 정치인들의 이해가 첨예하게 엇갈리는 문제라서 쉽게 합의되거나 추진이 되지 못하고 있는 것으로 보인다. 국가적인 규모로 운영되는 대통령선거, 국회의원 선거, 지방자치선거 등에서는 전자투표가 아직 도입되지 않고 있지만, 정당 내부의 경선에서는 전자투표가 활발하게 이용되는 것이 그 좋은 사례라고 볼 수 있다.

위에서 본 것처럼 암호 프로토콜은 다자간의 비대면 통신 프로토콜에서 정보보호 기능을 제공하기 위하여 암호기술을 적용하는 프로토콜을 말한다. 이러한 연구들은 현실세계에서 요구되는 다양한 비대면 활동들에 대해 신뢰성, 공정성을 제공하기 위해 사용될 수 있다. 앞으로 사람들의 활동이 비대면 사이버 세상에서도 더욱 활발하게 이루어지는 세상이 다가올 것이며 암호 프로토콜 기술은 이것을 구현하기 위한 핵심 기술이다.