Part 1. 보충자료

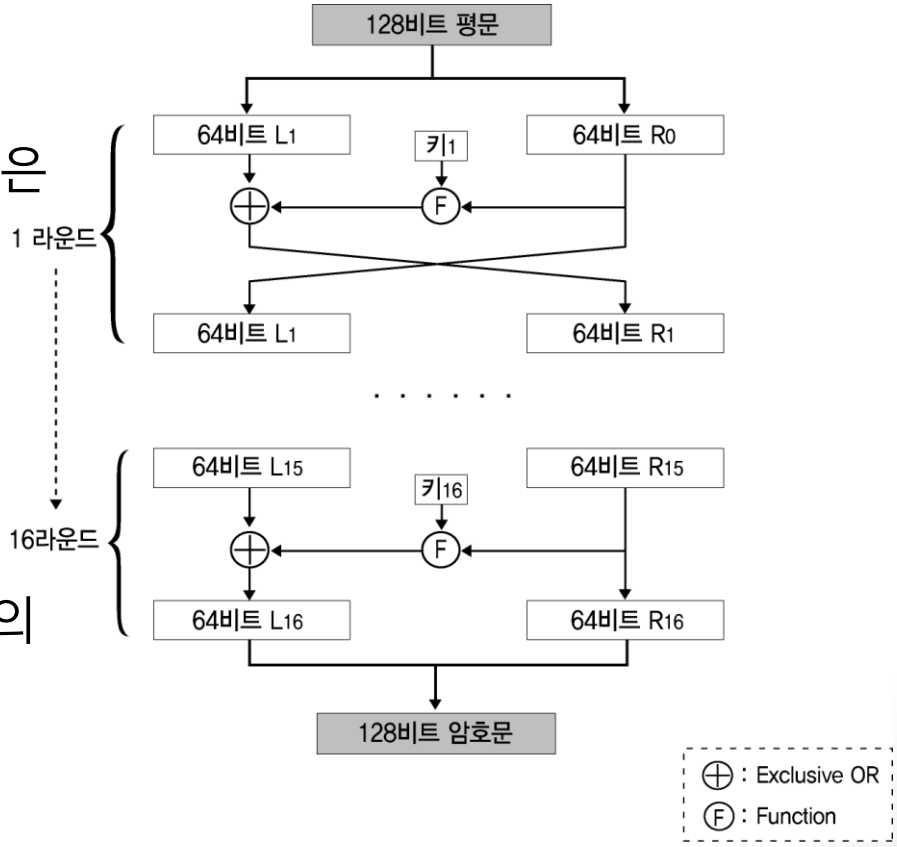
제2장. 대칭 암호와 메시지 기밀성
제3장. 공개키 암호와 메시지 인증

SEED (1)

- 개요
  - 1999년 한국정보보호진흥원(KISA)에 의해 개발된 국내 대칭키 기반 블록 암호 알고리즘
  - 1999년 한국정보통신협회(TTA)에 의해 국내 표준으로 채택
  - 현재 전자상거래, 전자 메일, 인터넷 뱅킹, 데이터베이스 암호화, 가상 사설망(VPN), 지적재산권 보호 등의 다양한 분야에서 사용
  - 대칭키 기반 블록 암호 알고리즘
  - 128비트의 키를 사용하는 128비트 블록 단위로 메시지를 암호화
  - 16라운드의 Feistel 구조로 구성
  - 2개의 S 박스 사용
  - DES, MISTY와 비교하였을 때 우수한 내부 함수를 내장
  - 차분 공격 및 선형 공격에 강함

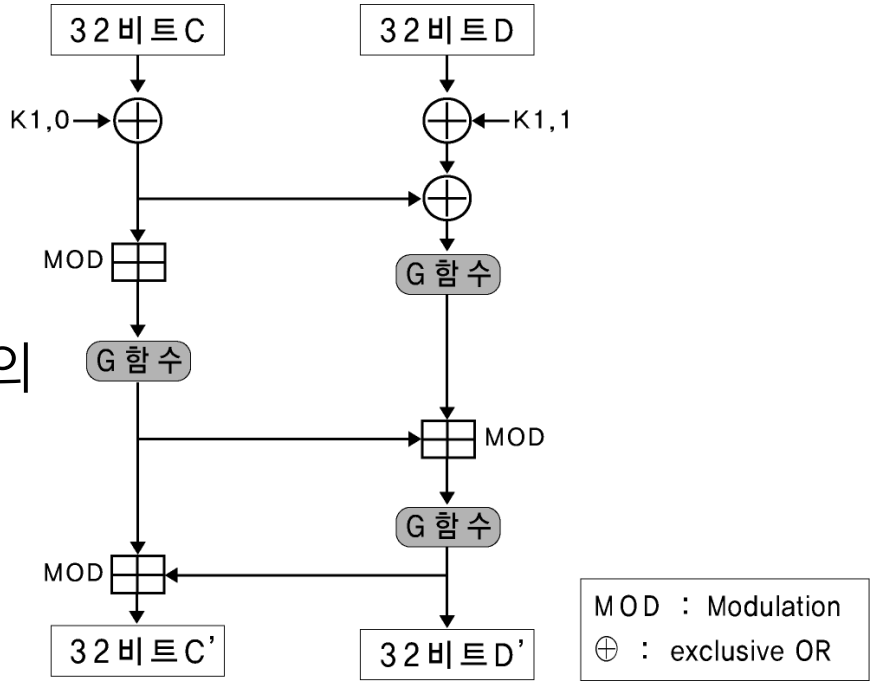
SEED (2)

- 구조
  - 128비트 단위 블록으로 구성된 평문 메시지를 16라운드의 Feistel 구조를 거쳐 암호화
  - 128비트의 평문 메시지 블록은 두 개의 64비트 메시지 블록으로 분할
  - 분할된 메시지 블록들은 F 함수, XOR(exclusive OR) 연산 등을 실행하는 16회 반복
  - 16 라운드 이후 반복 메시지 블록들은 통합되어 128비트의 암호문 메시지 블록이 됨
  - 각 라운드에 대해서는 서로 다른 64비트의 키가 적용 됨



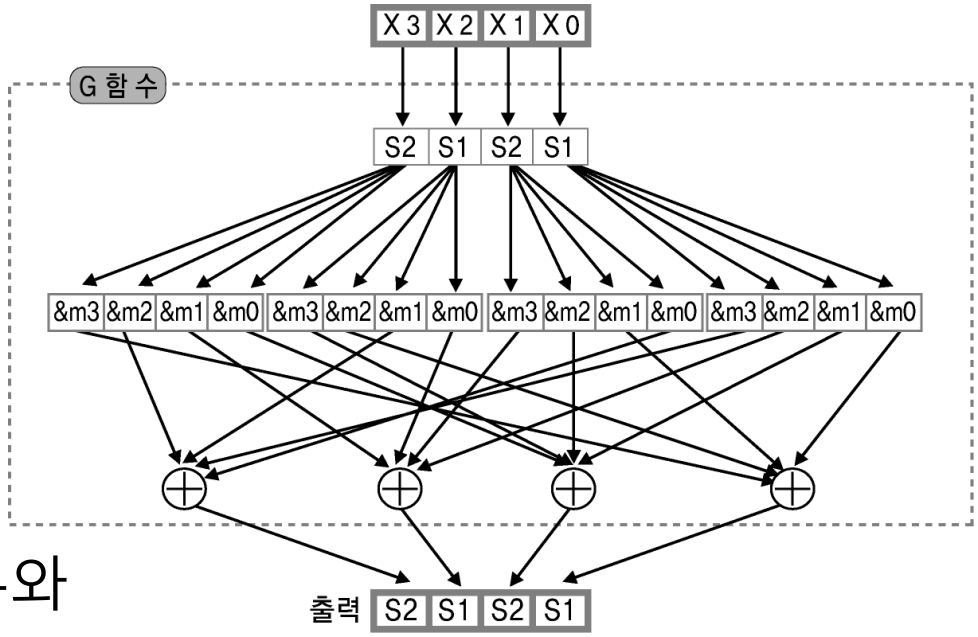
SEED (3)

- F 함수
  - SEED에서 블록 단위 메시지를 암호화하는 함수
  - 수정된 Feistel 구조로 구성
  - 64비트 블록을 분할한 32비트 블록 2개(C, D)와 64비트 키에서 분할된 2개의 라운드 키 Ki, 0와 Ki,1을 입력받아 XOR 연산, MOD (modulation) 연산, G 함수 등을 거친 후 32비트 블록 2개를 출력
  - 각 라운드 내에서 사용되는 F 함수의 구조



SEED (4)

- G 함수와 S 박스
  - F 함수 내에서 사용되는 G 함수는 4바이트 입력 데이터를 2개의 S-box를 이용하여 전치하여 4바이트의 출력 값으로 만드는 기능을 제공
  - S-box는 DES의 경우와 마찬가지로 수열로 구성
  - G 함수의 구조



ElGamal (1)

- 이산대수(discrete logarithm) 문제를 근간으로 만들어진 공개키 암호 알고리즘 방식
- 이산 대수 문제
  - 큰 소수 p로 만들어진 집합 Zp상에서의 원시 원소를 g라 할 때 $g^x \equiv y \bmod p$의 g와 y값을 알고 있어도 $\log_g y \equiv x$를 구하는 것이 어려움
  - g를 알고 있는 사용자가 y를 계산하는 것은 간단
- 사용자는 큰 소수 p를 선정하여 Zp상의 원시 원소 g와 함께 p를 공개
- 송신자 A
  - Zp상의 임의의 원소 x_A를 비밀 정보로 선택하여 $y_A \equiv g^{xA} \bmod p$의 공개 정보 yA을 계산함
- 송신자 B
  - Zp 상의 임의의 원소 x_B를 비밀 정보로 선택하여 $y_B \equiv g^{xB} \bmod p$의 공개 정보 yB를 계산함
- 송신자 A와 수신자 B의 y_A, y_B, p, q를 공개 목록에 등록함
  - y_A 와 y_B가 송신자 A와 수신자 B의 공개 암호화 키 Ke이고 x_A와 x_B가 송신자 A와 수신자 B의 비밀 복호화 키 Kd가 됨

ElGamal (2)

- 송신자 A가 평문 M을 암호화하여 암호문 C를 수신자 B에게 전송하기 위해서, Z_p상에서 임의의 난수 $r \in Z_{p-1}$을 선정하여 수신자 B의 공개 암호화 키 y_B로 $K \equiv y^r \bmod p_B$를 계산
- 암호문 C
- $C_1 \equiv y^r \bmod p$와 $C_2 \equiv KM \bmod p$을 계산한 다음 $C = (C_1, C_2)$가 됨
- 수신자 B의 평문 복호화 과정은 암호문 C_1에 수신자 B자신의 비밀 복호화 키x_B를 누승하여 $K \equiv C^x_{b \bmod p1}$를 구한 다음 $M \equiv C_2/K \bmod p$로 평문을 구함

타원곡선(EC)

- 대개 실수와 유리수와 같은 유한대 영역에 대해 정의되고 이산대수 문제에 대한 아날로그를 구현
- 하나의 곡선
  - 무한한 싱글 포인트 O를 갖는 $y^2 = x^3 + ax + b$ 타원곡선의 공간들
  - 덧셈은 모듈러 곱셈의 카운터 파트
  - 곱셈은 모듈러 멱승 연산의 카운터파트
- 하나의 타원 곡선 상에 주어진 두 지점 P와 R에 대해 K=PR을 만족하는 K를 찾아낸다는 것은 타원곡선 이산대수 문제로 알려진 어려운 문제임
- 작은 키 값을 갖고도 높은 보안 수준을 이룰 수 있음

공개키 암호화 방식의 예

구분	RSA	ECC	LUC	GH	NTRU	EPOC	XTR
개발국	미국 (RSA사)	캐나다 (Certicom)	호주 (Lucent)	미국 (MIT)	일본 (Braun Univ.)	일본 (NTT)	미국 (Citybank)
개발자	Rivest Shamit Adleman	Koblita Miller	Smith	Goldreich Goldwasser Halevi	Horrstein Pipher Silverman	Okamoto Uchivama Fujisaki	Lenstra Verheul
개발 시기	1978년	1985년	1993년	1996년	1996년	1998년	2000년
기반 문제	IFP of n=pq	ECDLP	IFP of m=pq	LRP(CVP)	LRP(SVP)	IFP of n=pq	DLP
특징	• 현 상용 시스템	• RSA에 상대적 고비도 • 스마트카드에 적합	• RSA의 지수함수 대신 Lucas 수열 이용	• McEiece스킴을 Lattice에 적용	• 확률론적 암호시스템 • 암복호화 속도 빠름	• 안전성이 증명 가능한 확률론적 암호	• 안전성 및 효율성에서 ECC와 필적

공개키 암호알고리즘 비교

	RSA	ElGamal	ECC
수학적 문제	소인수 분해	이산대수	타원곡선 이산대수
키 크기	크다	크다	작다
속도	비교적 느리다	비교적 느리다	빠르다
암호문 크기	-	평문의 두배	-
메모리	ElGamal에 비해 적음	가장 많이 차지	가장 적게 차지
비용	많이 소요	많이 소요	적게 소요
통신	유선	유선	무선

PKCS(Public Key Cryptography Standards)

- 미국의 RSA사가 개발한 암호 작성 시스템
- 애플, 마이크로소프트, DEC, 로터스, 선, MIT 등 컨소시엄 공동으로 개발
- PKCS는 인터넷 상에서 안전한 정보 교환을 이루기 위해 산업계 내부에서 사용되는 일련의 비공식 표준 프로토콜
- PKCS#1 ~ PKCS#15
  - PKCS#1 : RSA Cryptography Standard.
  - PKCS#2 : PKCS#1에 통합됨.
  - PKCS#3 : Diffie-Hellman Key Agreement Standard.
  - PKCS#4 : PKCS#1에 통합됨.
  - PKCS#5 : Password-based Encryption Standard.
  - PKCS#6 : Extended-Certificate Syntax Standard.
  - PKCS#7 : Cryptographic Message Syntax Standard.
  - PKCS#8 : Private-Key Information Syntax Standard.
  - PKCS#9 : Selected Attribute Types.
  - PKCS#10 : Certification Request Syntax Standard.
  - PKCS#11 : Cryptographic Token Interface Standard.
  - PKCS#12 : Personal Information Exchange Syntax Standard.
  - PKCS#13 : Elliptic Curve Cryptography Standard.
  - PKCS#14 : Pseudo-random Number Generation.
  - PKCS#15 : Cryptographic Token Information Format Standard

대칭키와 공개키 암호 방식의 비교 (1)

- 공개키(public key) 암호의 주요한 장점
  - 강화된 보안성과 편리함
  - 전자서명 기법을 제공
  - 부인방지(Non- Reputation)
  - 공개키 인증에서는 사용자가 스스로 자신의 개인키 보호에 대한 전적인 책임을 짐
- 공개키 암호법의 단점
  - 암호화 속도
    - 전자 봉투(Digital Envelope) : 공개키 시스템은 대형 파일이나 메시지를 암호화하는데 사용되는 비밀키의 암호화에 사용
  - 가장공격(Impersonation)에 취약
    - 침입자는 인증기관(Certification Authority)을 공격하여 획득한 공개키 인증서를 사용해 다른 사용자인 척 함

대칭키와 공개키 암호 방식의 비교 (2)

항목	대칭키 암호 방식	공개키 암호 방식
키의 상호관계	암호화키 = 복호화키	암호화키 ≠ 복호화키
암호화 키	비밀	공개
복호화 키	비밀	비밀
암호알고리즘	비밀/공개	공개
대표적인 예	DES	RSA
비밀 키 전송	필요	불필요
키 개수	n(n-1)/2	2n
안전한 인증	곤란	용이
암호화 속도	고속	저속
경제성	높다	낮다
전자서명	복잡	간단

국산 암호 알고리즘 (1)

- SEED
  - 전자상거래, 금융, 무선통신 등에서 전송되는 개인정보와 같은 중요한 정보를 보호하기 위해 1999년 2월 한국인터넷진흥원과 국내 암호전문가들이 순수 국내기술로 개발한 128비트 블록 암호 알고리즘
  - 1999년에는 128비트 키를 지원하는 SEED 128을 개발
  - SEED 128은 1999년 9월 정보통신단체표준(TTA)으로 제정
  - 2005년에는 국제 표준화 기구인 ISO/IEC 국제 블록암호알고리즘, IETF 표준으로 제정
  - 2009년 암호 알고리즘 보안 강화를 위해 256 비트 키를 지원하는 SEED 256을 개발

국산 암호 알고리즘 (2)

- HIGHT(HIGh security and light weigHT)
  - RFID, USN 등과 같이 저전력・경량화를 요구하는 컴퓨팅 환경에서 기밀성을 제공
  - 2005년 KISA, ETRI 부설연구소 및 고려대가 공동으로 개발한 64비트 블록암호 알고리즘
  - 소프트웨어 및 하드웨어 구현 성능 비교 결과

알고리즘	CPU/OS/Complier	
	P3/WinXP/VC	P4/WinXP/VC
HIGHT	88.882	72.413
AES1	329.452	234.765

  - 2006년 12월 정보통신단체표준(TTA)으로 제정
  - 2010년 12월 ISO/IEC 국제 블록 암호 알고리즘 표준으로 제정

국산 암호 알고리즘 (3)

- ARIA
  - ARIA라는 이름은 Academy(학계), Research Institute(연구소), Agency(정부 기관)의 첫 글자들을 딴 것
  - 경량 환경 및 하드웨어 구현을 위해 최적화된, Involutional SPN 구조를 갖는 범용 블록 암호 알고리즘
  - ARIA가 사용하는 대부분의 연산은 XOR과 같은 단순한 바이트 단위 연산으로 구성
  - 주요 특성
    - 블록 크기 : 128비트
    - 키 크기 : 128/192/256비트 (AES와 동일 규격)
    - 전체 구조 : Involutional Substitution-Permutation Network
    - 라운드 수 : 12/14/16 (키 크기에 따라 결정됨)
  - 2004년에 국가표준기본법에 의거, 국가표준(KS)으로 지정

국산 암호 알고리즘 (4)

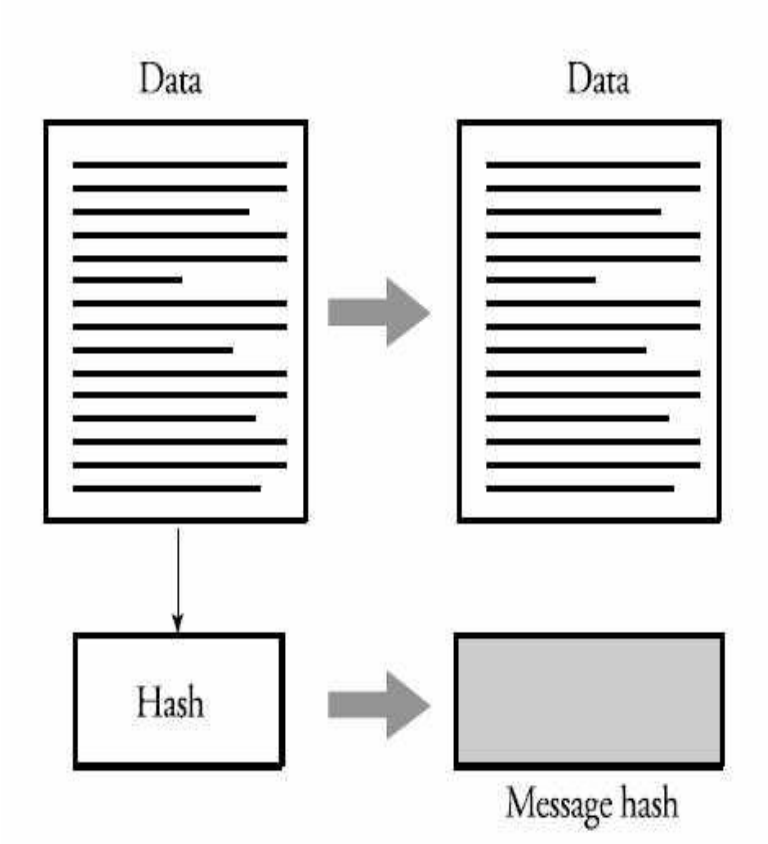
- LEA(Lightweight Encryption Algorithm)
  - 빅데이터, 클라우드 등 고속 환경 및 모바일기기 등 경량 환경에서 기밀성을 제공하기 위해 개발된 128비트 블록암호 알고리즘
  - 주요 특성
    - 개발연도 : 2013년
    - 알고리즘 구분 : 128비트 블록암호
    - 키 길이 : 128비트, 192비트 또는 256비트
    - 구조 : ARX(Addition, Rotation, Xor) 기반 GFN(Generalized Feistel Network)
  - 성능 : 다양한 SW 환경에서 국제 표준암호 AES 대비 1.5배 ~ 2배 성능
  - LEA 규격 및 운영모드는 국내 TTA 표준으로 제정
  - 2015년 6월 암호모듈 검증제도 검증 대상 알고리즘에 포함

국산 암호 알고리즘 (5)

- 해쉬함수 LSH(Lightweight Secure Hash)
  - 메시지 인증, 사용자 인증, 전자서명 등 다양한 암호 응용 분야에 활용 가능한 암호학적 해시 함수
  - 개발연도 : 2014년
  - 알고리즘 구분 : 해시 함수
  - 출력 길이 : 224비트, 256비트, 384비트 또는 512비트
  - 구조 : Wide-pipe Merkle Damgård 구조
  - 다양한 SW 환경에서 국제 표준(SHA2/3) 대비 2배 이상 성능
  - 국내 TTA 표준으로 제정

정보보호에서의 해쉬함수 이용

- 데이터 무결성을 제공하는 알고리즘 중 하나
  - 메시지 인증 알고리즘
  - 단방향 해쉬함수
- 임의의 길이의 메시지를 받아들여 특정 길이의 출력 값 생성
- 출력 값 비교를 통해 무결성 확인



해쉬함수의 조건

- H는 임의의 크기의 입력 M을 적용할 수 있어야 한다.
- H는 일정 크기의 출력 h = H(M)을 만들어야 한다.
- H와 M이 주어졌을 때 h = H(M) 계산이 쉬워야 한다
- H와 h가 주어졌을 때 M을 구하는 계산이 거의 불가능해야 한다.(one way property)
- H가 주어졌을 때 같은 출력을 갖는 두 입력을 착지 어려워야 한다. (충돌 회피성)(weak collision resistance)

전자서명의 조건

- 위조 불가
  - 서명자만이 서명 생성 가능
- 서명자 인증
  - 서명자의 신분 확인 가능
- 재사용 불가
  - 다른 문서의 서명으로 사용 불가능
- 변경 불가
  - 서명된 문서 내용 변경 불가
- 부인 불가
  - 서명한 사실 부인 불가

디지털 서명 알고리즘

- 공개키 암호방식을 이용한 서명 방식
- 서명자가 비밀키로 서명을 생성하고, 검증자가 공개키로 확인하는 시스템
- 직접 서명 방식
  - 송신자와 수신자 간에 직접 서명 및 검증
- 중계 서명 방식
  - 중재자를 통해 확인
  - 통신 전에 정보 공유가 필요 없고, 외부로부터 공격에 강하며, 시간 확인까지 가능

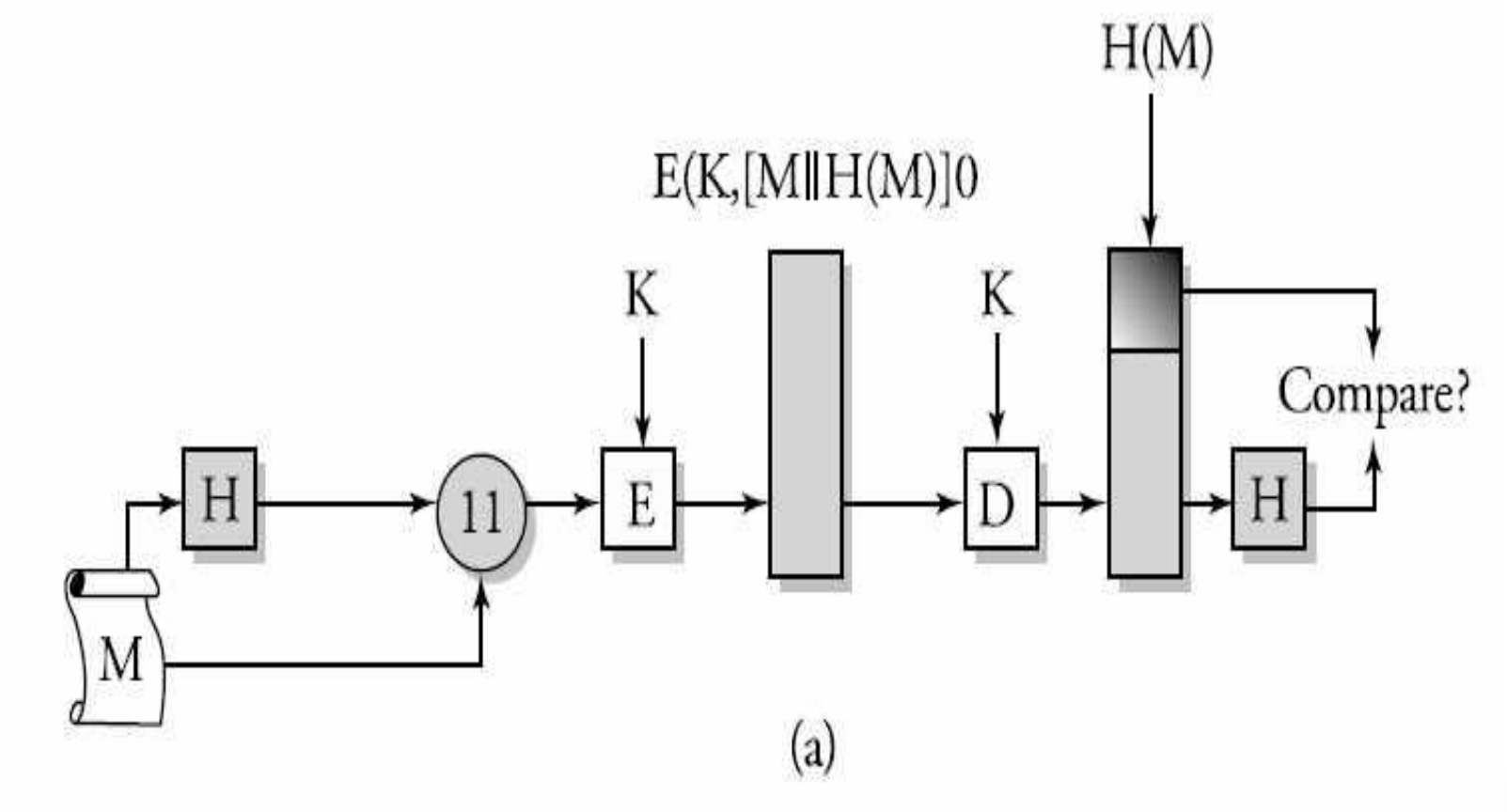
RSA 서명 방식

- 가장 먼저 실용화
- 서명 알고리즘의 안전도는 RSA 암호 방식 안전도와 동일
- 가장 보편적으로 사용
- 알고리즘
  - 키 생성 단계
    - RSA 암호 알고리즘과 동일
  - 서명 단계
    - 메시지 M의 해쉬값 H를 구하여 서명값 계산
    - $S = H^{dA} \bmod n_A$
    - 원본 메시지 M과 같이 전송
  - 검증 단계
    - 원본 메시지에 대한 H' 계산
    - 공개키로 S의 H 계산 후 비교

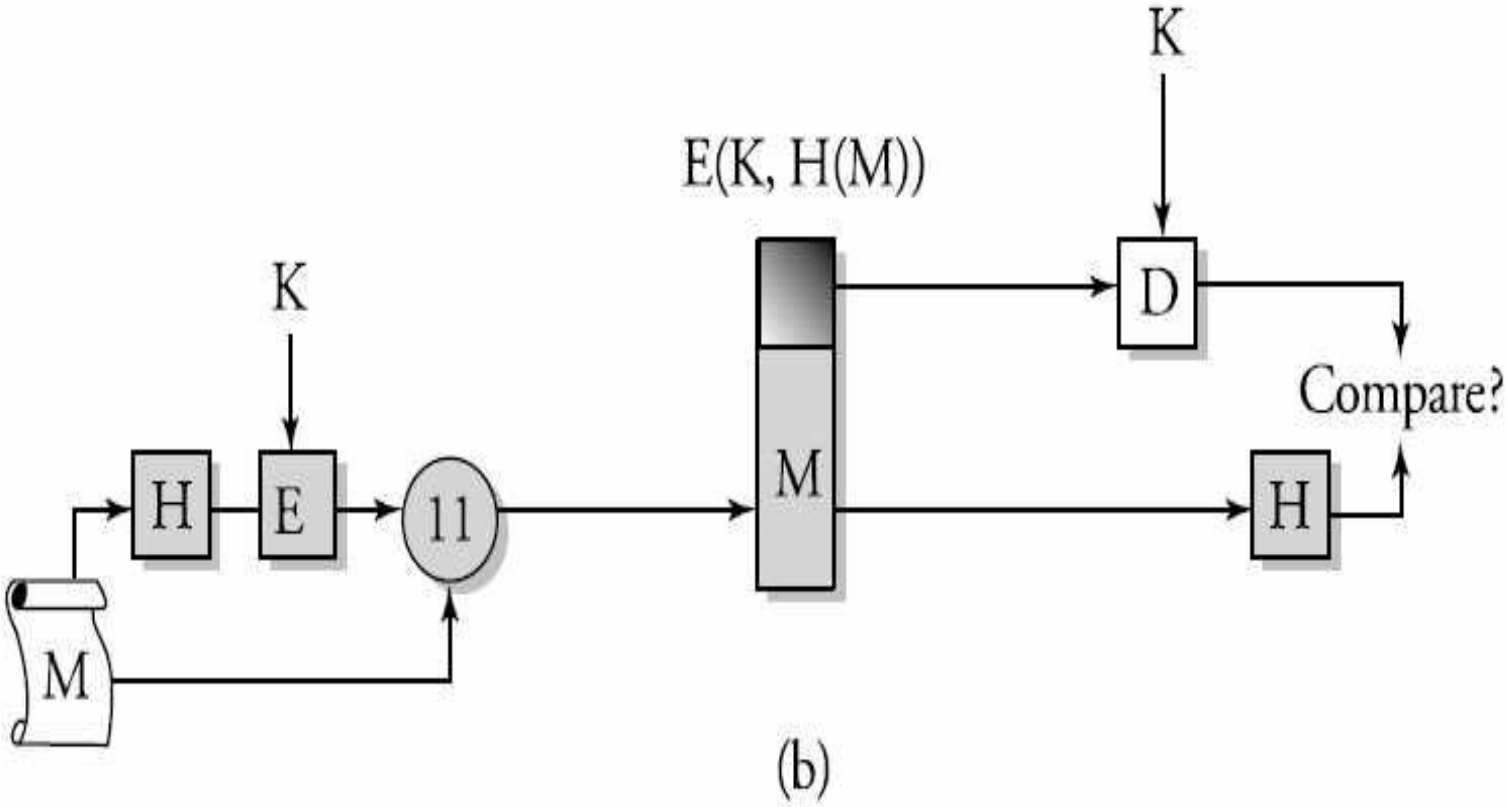
그외 서명 방식

- ElGamal 서명 알고리즘
  - 키 생성
  - 서명
  - 서명 검증
- DSS(Digital Signature Standard) 서명 알고리즘
  - ElGamal 서명 기술 응용
  - NIST에 의해 1994년 12월 표준 채택
  - ElGamal(512비트)보다 짧은 서명값(160비트)
  - 서명
  - 서명 검증 단계
- 타원곡선 디지털 서명 알고리즘
  - EC-DSA
    - 1985년 N.Koblitz와 V.S Miller가 RSA 방식의 대안으로 제시
    - 동일한 안전성을 실현하는데 RSA 1024 비트, ECC 160 비트
  - EC-KCDSA

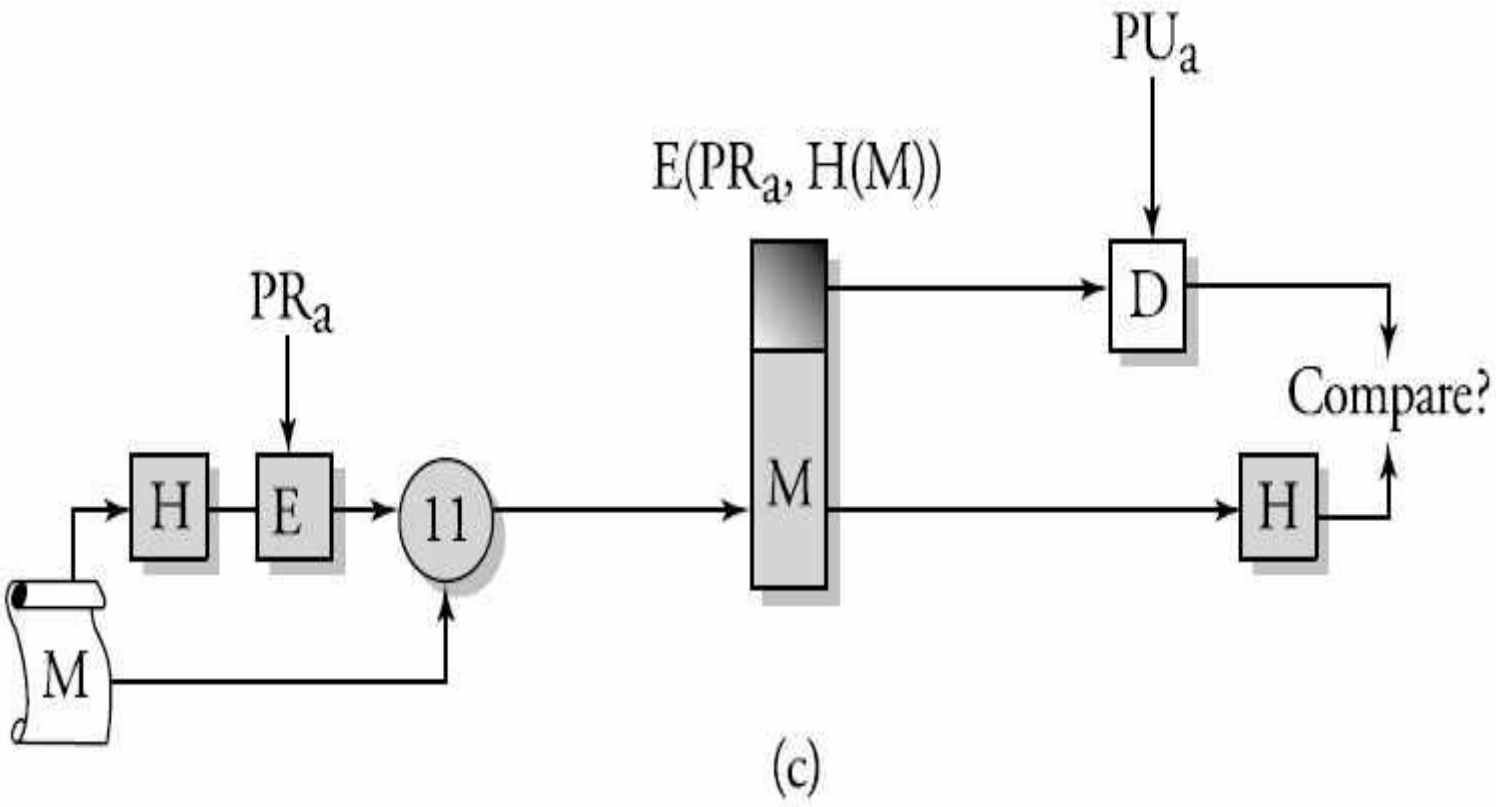
해쉬함수의 응용 (1)



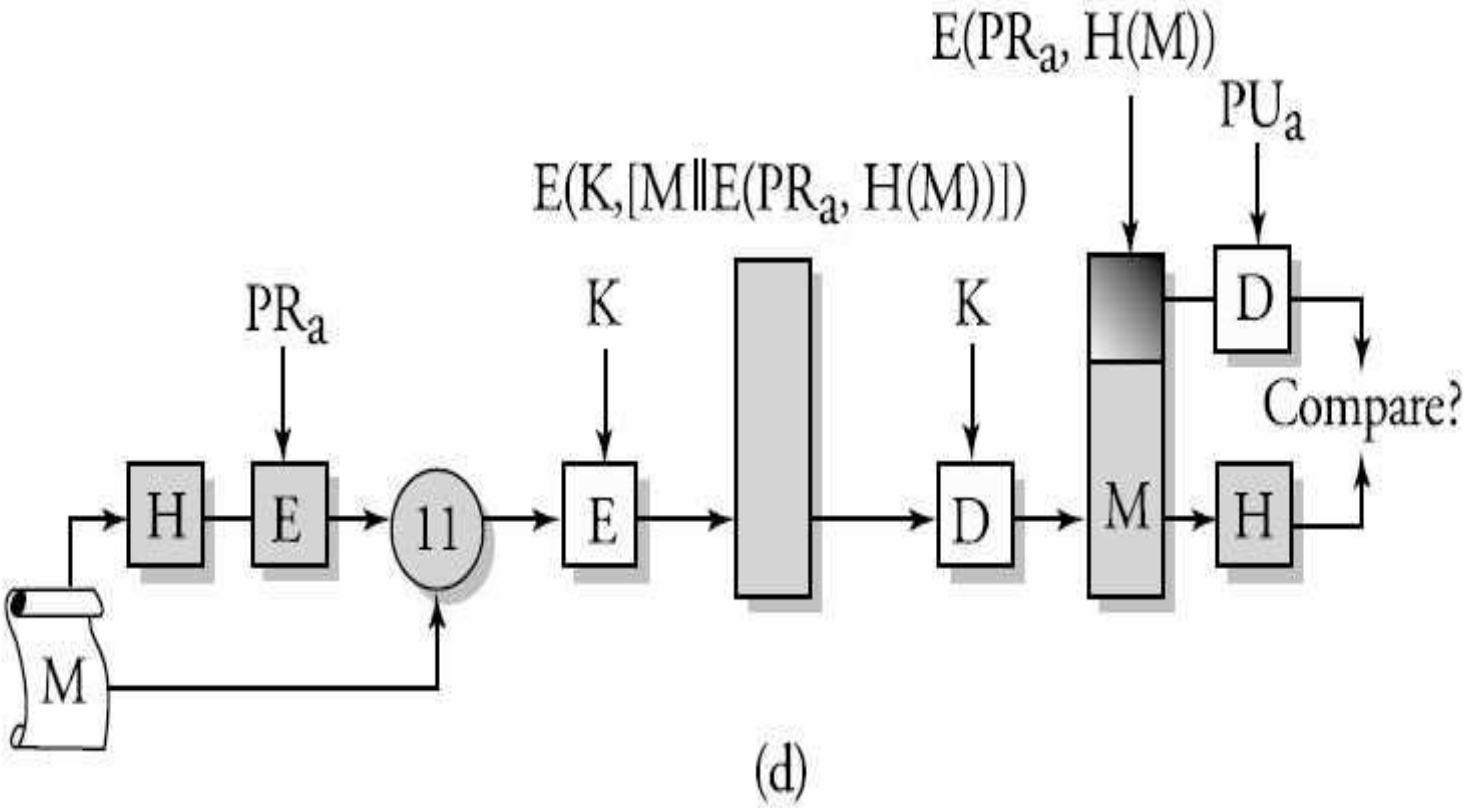
해쉬함수의 응용 (2)



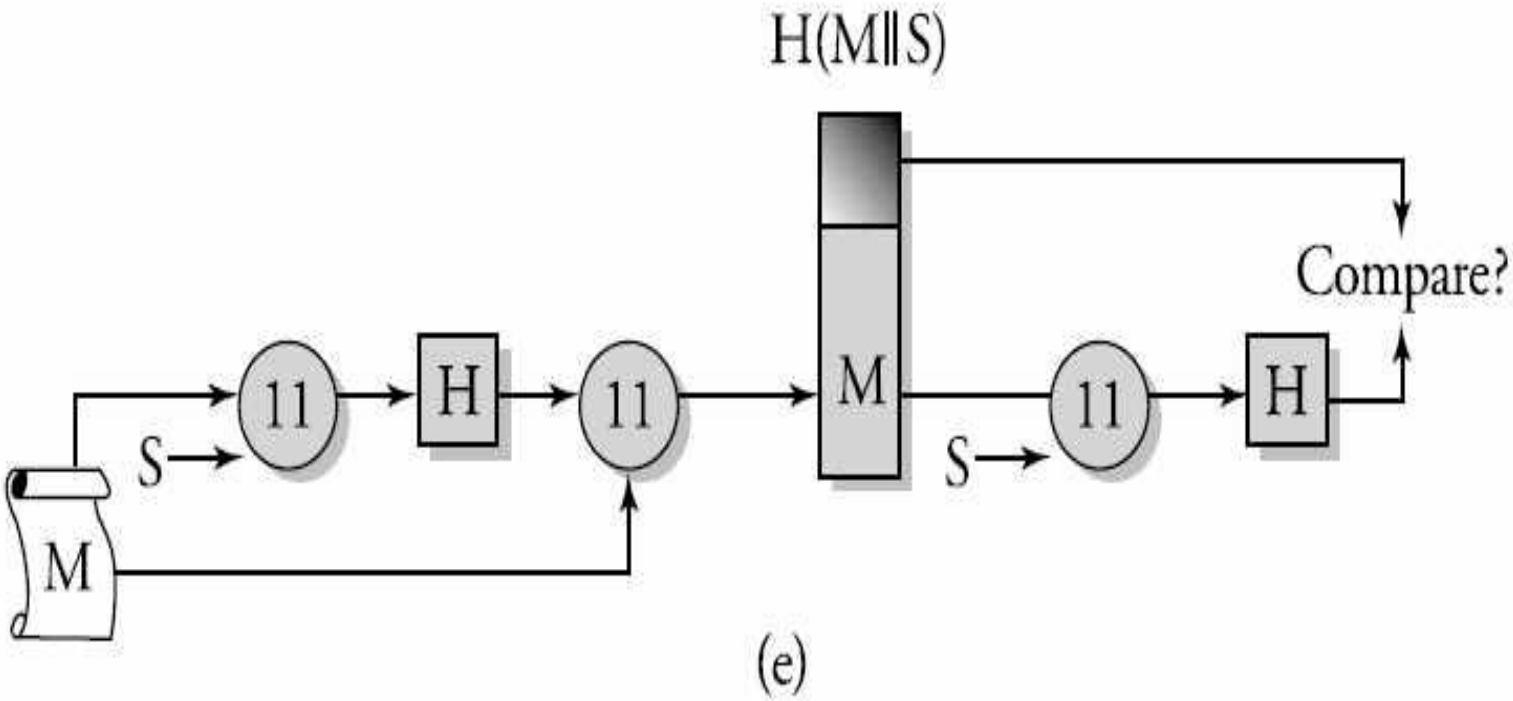
해쉬함수의 응용 (3)



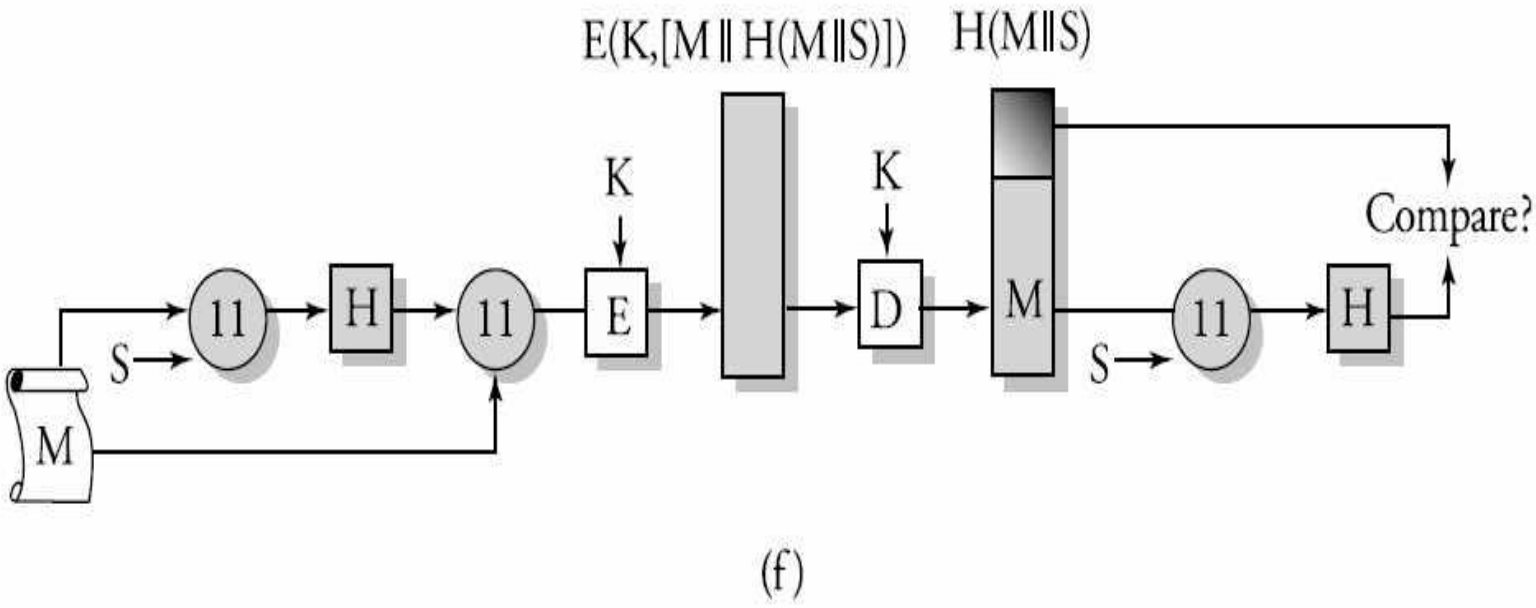
해쉬함수의 응용 (4)



해쉬함수의 응용 (5)



해쉬함수의 응용 (6)



해쉬함수 (1)

- MD5 (Message Digest Version 5)
  - 512비트 입력 128비트 출력
  - 충돌회피성에 대한 문제로 인해 기존 응용과 호환으로만 사용 제한
- MD4 (Message Digest Version 4)
  - 1990년 Rivest가 개발
  - 메시지를 128비트로 압축
  - MD5보다 약간 빠르고, 안전성 측면에서는 다소 떨어짐
- MD2 (Message Digest Version 2)
  - PEM 프로토콜에 응용
  - 보안성은 바이트의 random permutation에 달려있음
  - 다른 함수보다 다소 느리지만 취약 부분은 아직 발견되지 않음

해쉬함수 (2)

- RIPE-MD
  - 유럽의 RIPE 프로젝트에서 개발된 해쉬함수
  - SHA-1 못지않은 안전성을 가진 것으로 평가되며, 128, 160, 256, 320 비트 해쉬 출력값을 제공
- HAVAL
  - 1992년 Zheng에 의해 개발
  - 출력값 길이가 가변
  - MD5를 변형하여 1024비트 블록으로 수행
  - 다양한 라운드 수와 7개의 가변 함수, 128, 160, 192, 224, 256 비트 길이 해쉬값 출력 가능

해쉬함수 (3)

- SHA (Secure Hash Algorithm)
  - NIST에 의해 1993년 FIPS PUB 180으로 표준화
  - MD4와 유사하게 설계
  - 512비트 단위로 메시지를 입력하여 160비트 해쉬값 출력 (입력 전 메시지 길이를 512 비트 정수배로 조정)
- SHA-1(전자서명용)과 MD5 비교
  - 각각의 키 길이의 차이가 있음 (160 : 128)
  - 공격 대응면에서 SHA-1이 더 강하다 (길이 차이)
  - 안전성 : SHA-1이 비교적 더 안전
  - 속도 : 연산이 많아 SHA-1이 다소 늦다
  - 단순성과 간결성 : 두 알고리즘 모두 비교적 간단하며 적용 용이
  - 바이트 순서 : (big-endian : little-endian)

해쉬함수 (4)

Hash Function	Output Length	Round	Block Size	Speed
MD4	128	48	512	1.00
MD5	128	64	512	0.68
RIPE-MD-128	128	128	512	0.39
SHA-1	160	80	512	0.28
SHA-256	256	64	512	
SHA-384	384	80	1024	
SHA-512	512	80	1024	
RIPE-MD-160	160	160	512	0.24
RIPE-MD-256	256	128	512	
RIPE-MD-320	320	160	512	
Tiger	192	56	512	

해쉬함수 (5)

- 일반적으로 MD5가 많이 사용되고 있음
  - 취약성이 발견되어 제한적 사용 권고
- SHA-1은 디지털 서명에 사용하도록 제안됨
- AES의 128, 192, 256비트에 적용하도록 SHA256, SHA382, SHA512로 확장
- RIPE-MD-128, RIPE-MD-160, RIPE-MD-256, RIPE-MD-320은 MD5를 대신할 수 있도록 제안
  - RIPE-MD-128은 충돌저항성 문제가 있음
  - RIPE-MD-160은 효율성은 낮지만 높은 안전성으로 널리 사용 중
- Tiger는 64비트 환경에 최적화됨