

Computer Security DD2395

<http://www.csc.kth.se/utbildning/kth/kurser/DD2395/dasakh10/>

Fall 2010

Sonja Buchegger

buc@kth.se

Lecture 2, Oct. 27, 2010

Cryptography

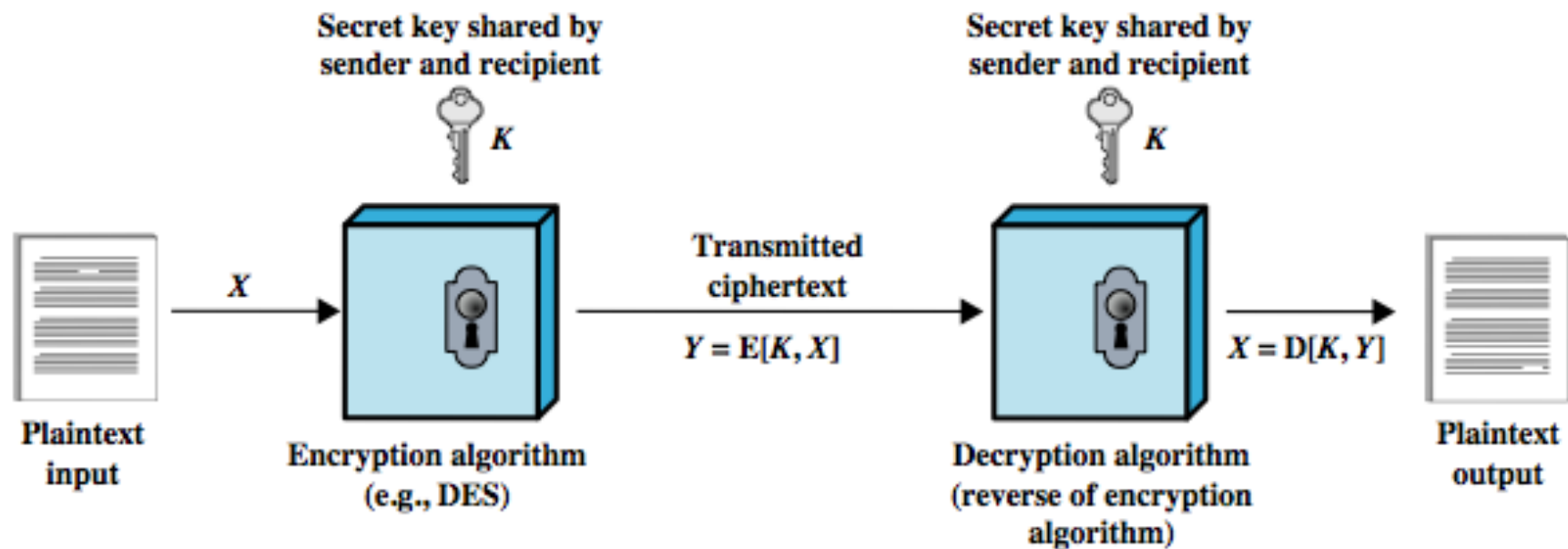
Questionnaire Results

- Prior security knowledge:
 - Most low to medium, a few higher
- Expectations:
 - Most quite high
- Some questions:
 - Partners
 - Labs, ECTS
 - Book
 - Exam, date
 - CSN
 - Theory/practice/depth applicability/jobs

Cryptographic Tools

- cryptographic algorithms important element in security services
- review various types of elements
 - symmetric encryption
 - public-key (asymmetric) encryption
 - digital signatures and key management
 - secure hash functions
- example is use to encrypt stored data

Symmetric Encryption

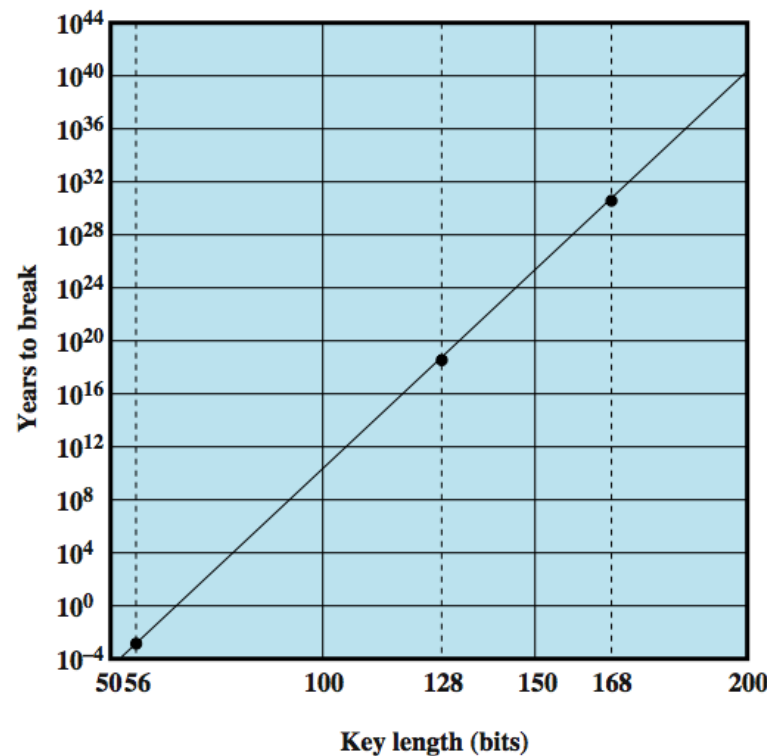


Attacking Symmetric Encryption

- cryptanalysis
 - rely on nature of the algorithm
 - plus some knowledge of plaintext characteristics
 - even some sample plaintext-ciphertext pairs
 - exploits characteristics of algorithm to deduce specific plaintext or key
- brute-force attack
 - try all possible keys on some ciphertext until get an intelligible translation into plaintext

Exhaustive Key Search

Key Size (bits)	Number of Alternative Keys	Time Required at 1 Decryption/ μ s	Time Required at 10^6 Decryptions/ μ s
32	$2^{32} = 4.3 \times 10^9$	$2^{31} \mu\text{s} = 35.8 \text{ minutes}$	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	$2^{55} \mu\text{s} = 1142 \text{ years}$	10.01 hours
128	$2^{128} = 3.4 \times 10^{38}$	$2^{127} \mu\text{s} = 5.4 \times 10^{24} \text{ years}$	$5.4 \times 10^{18} \text{ years}$
168	$2^{168} = 3.7 \times 10^{50}$	$2^{167} \mu\text{s} = 5.9 \times 10^{36} \text{ years}$	$5.9 \times 10^{30} \text{ years}$
26 characters (permutation)	$26! = 4 \times 10^{26}$	$2 \times 10^{26} \mu\text{s} = 6.4 \times 10^{12} \text{ years}$	$6.4 \times 10^6 \text{ years}$



Symmetric Encryption Algorithms

	DES	Triple DES	AES
Plaintext block size (bits)	64	64	128
Ciphertext block size (bits)	64	64	128
Key size (bits)	56	112 or 168	128, 192, or 256

DES = Data Encryption Standard

AES = Advanced Encryption Standard

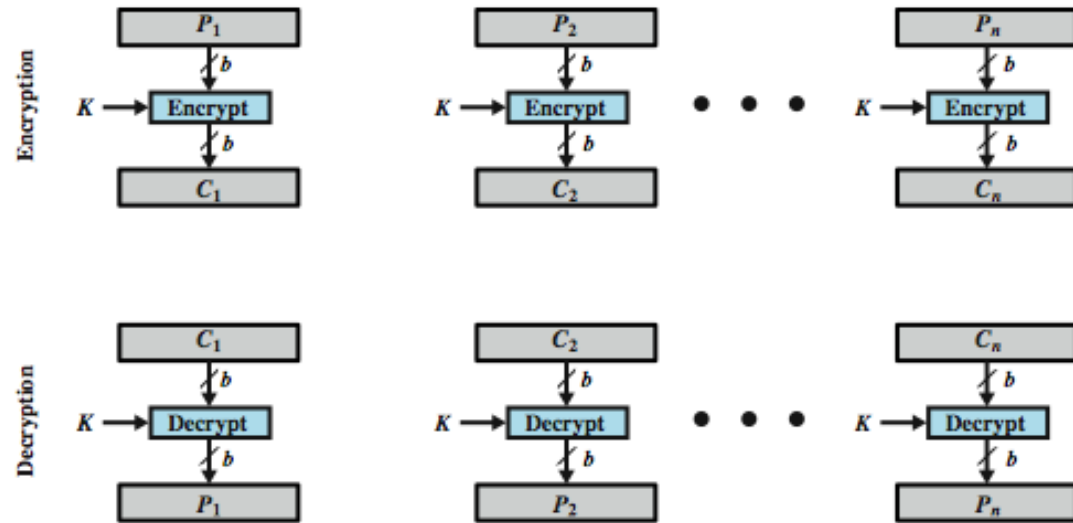
DES and Triple-DES

- Data Encryption Standard (DES) is the most widely used encryption scheme
 - uses 64 bit plaintext block and 56 bit key to produce a 64 bit ciphertext block
 - concerns about algorithm & use of 56-bit key
- Triple-DES
 - repeats basic DES algorithm three times
 - using either two or three unique keys
 - much more secure but also much slower

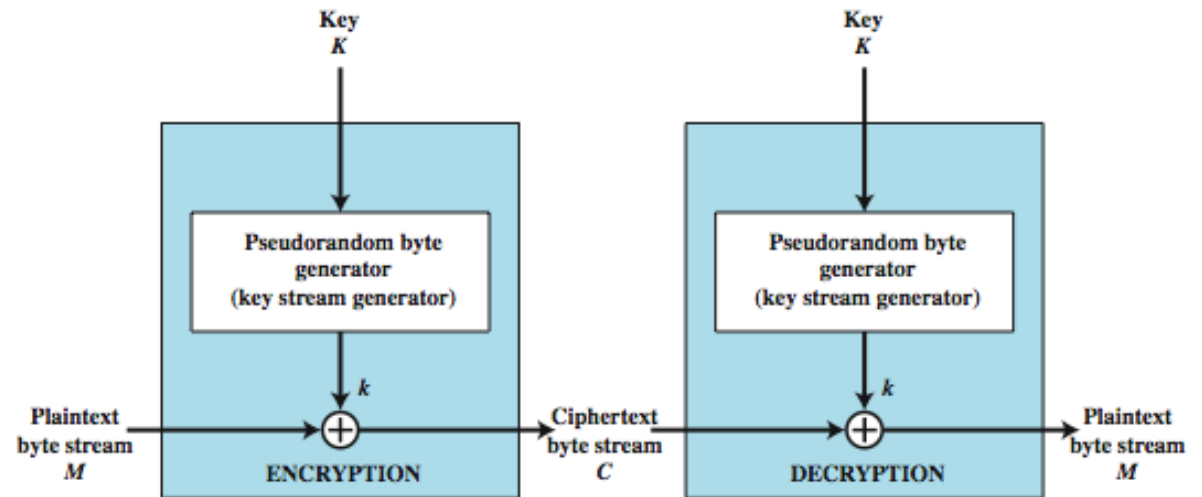
Advanced Encryption Standard (AES)

- needed a better replacement for DES
- NIST called for proposals in 1997
- selected Rijndael in Nov 2001
- published as FIPS 197
- symmetric block cipher
- uses 128 bit data & 128/192/256 bit keys
- now widely available commercially

Block verses Stream Ciphers



(a) Block cipher encryption (electronic codebook mode)

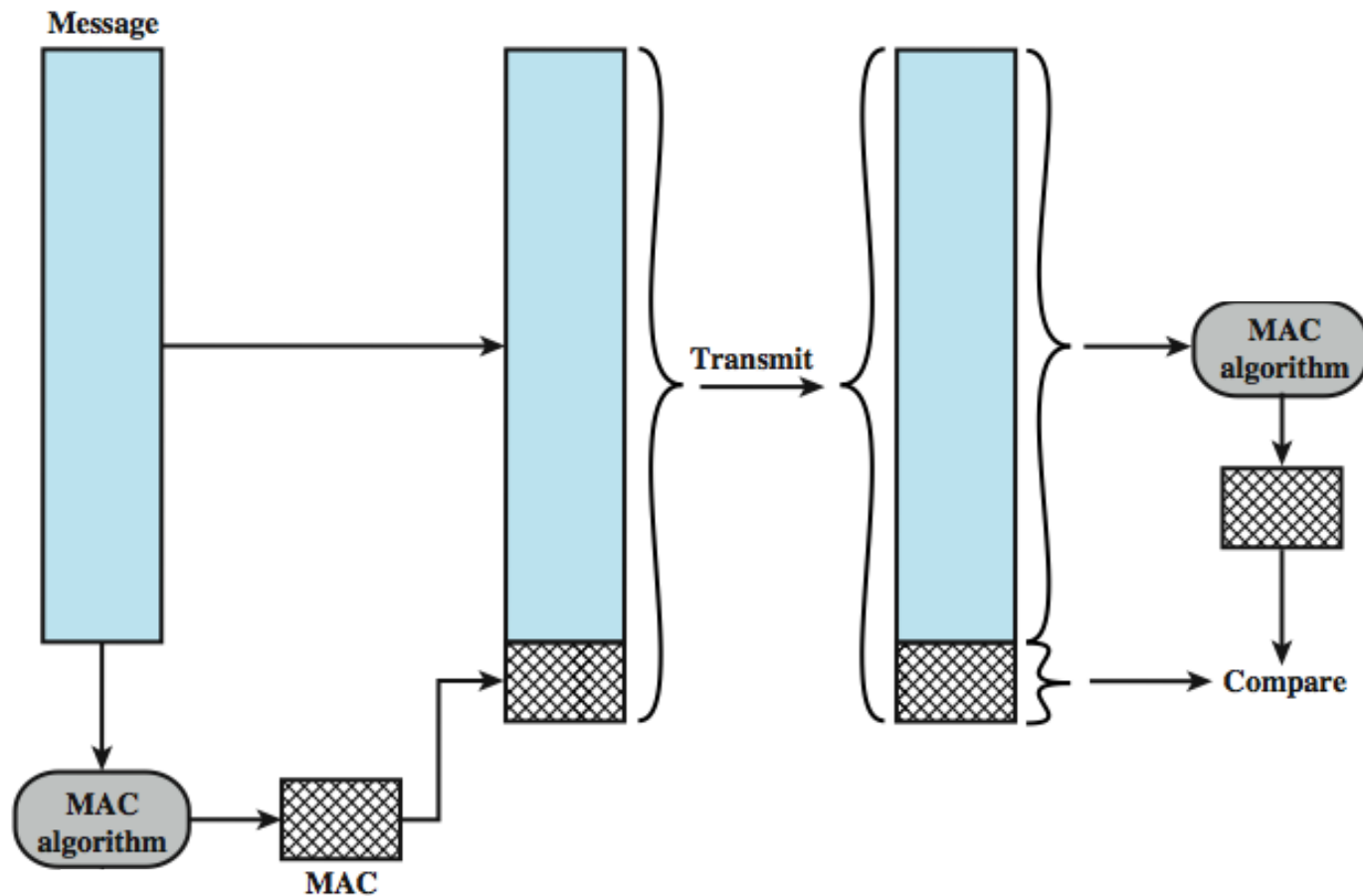


(b) Stream encryption

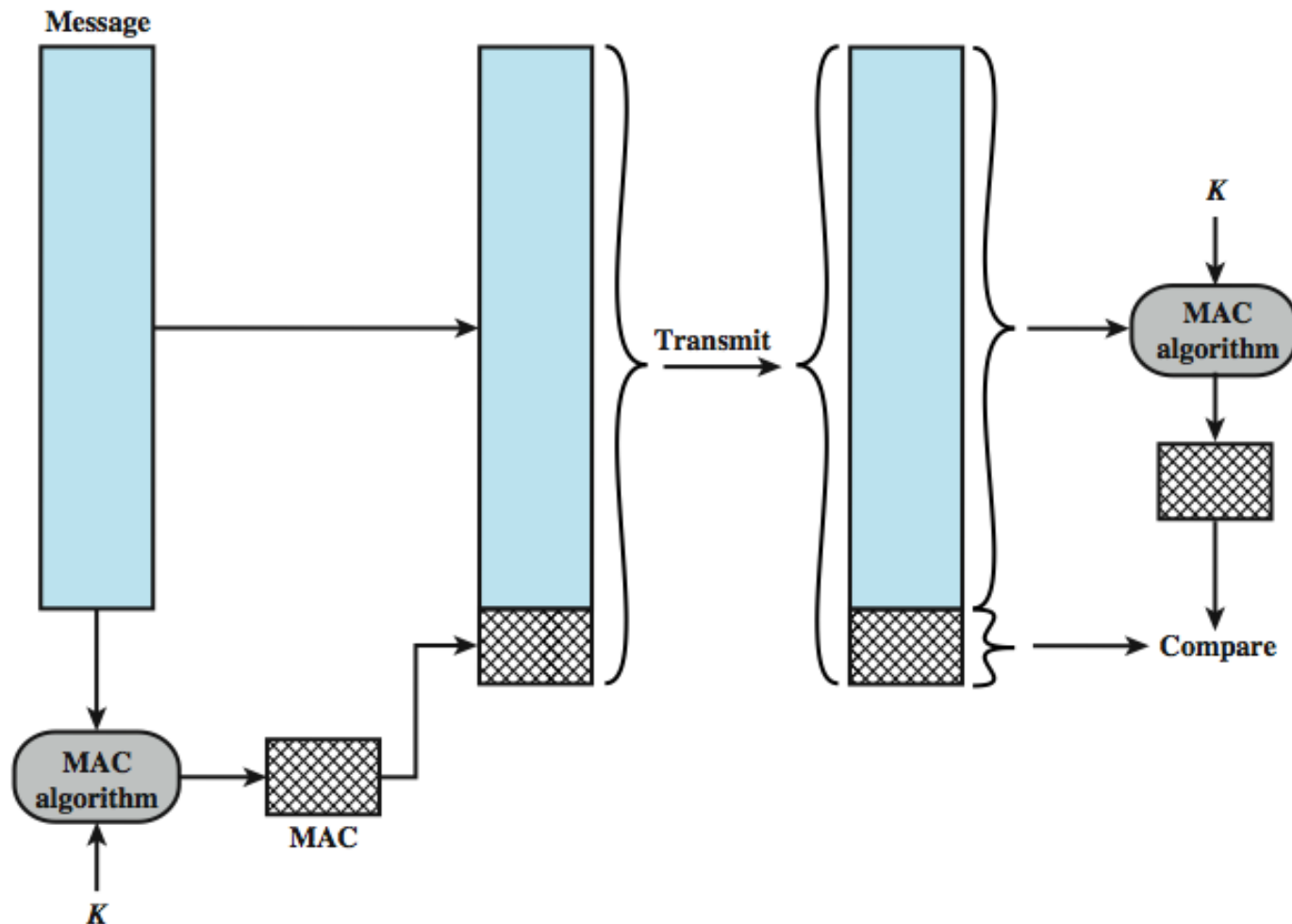
Message Authentication

- protects against active attacks
- verifies received message is authentic
 - contents unaltered
 - from authentic source
 - timely and in correct sequence
- can use conventional encryption
 - only sender & receiver have key needed
- or separate authentication mechanisms
 - append authentication tag to cleartext message

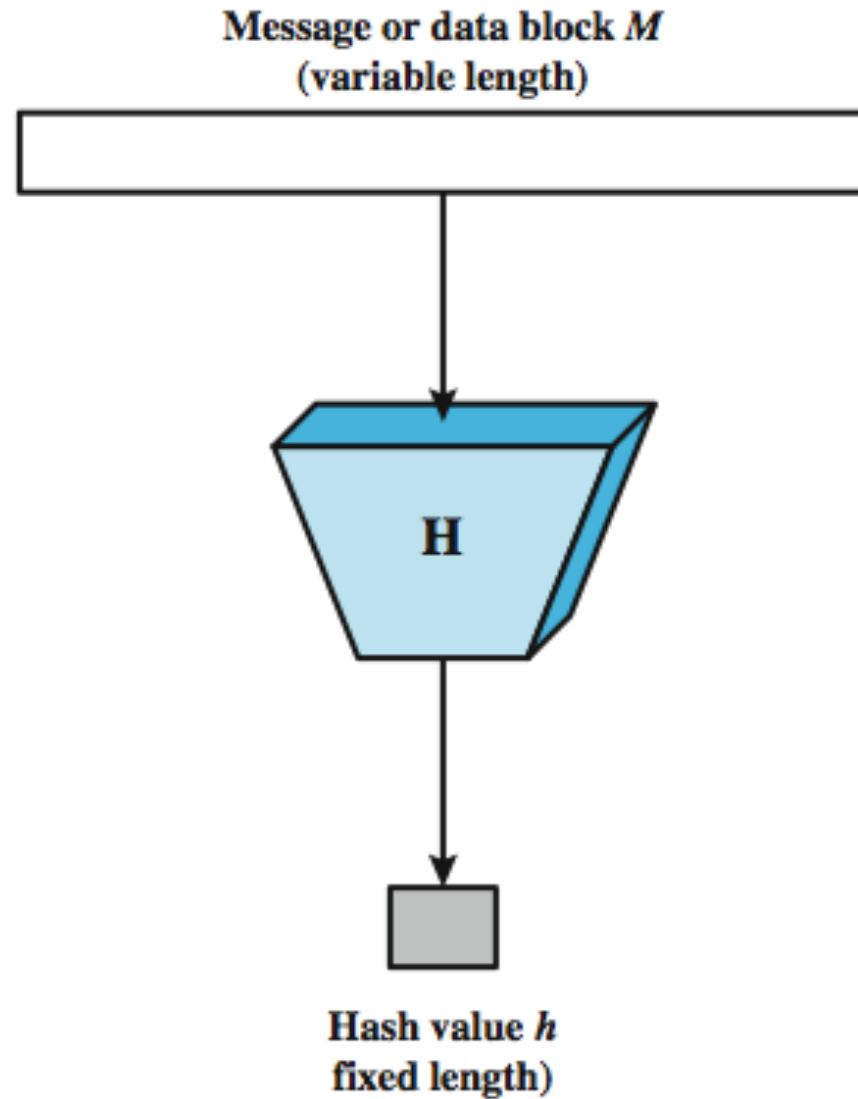
Message Authentication Codes 1



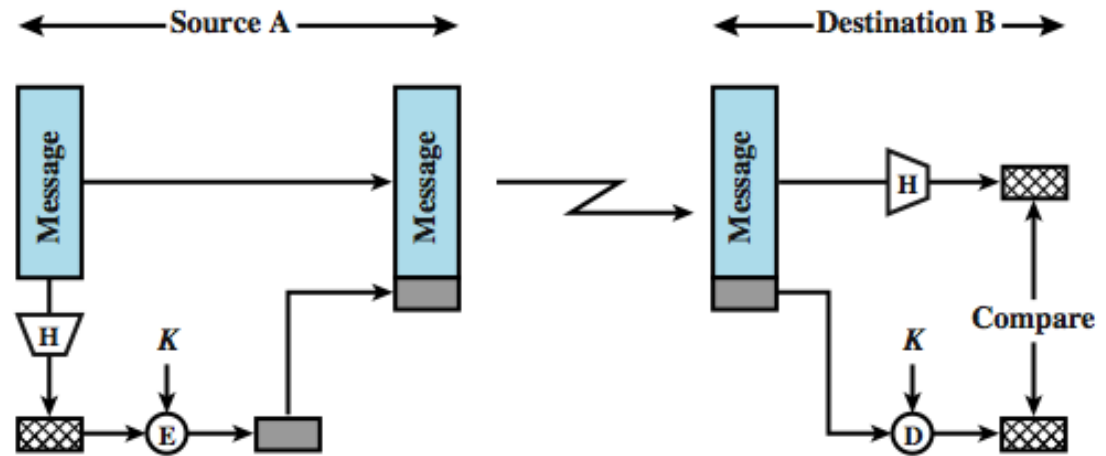
Message Authentication Codes



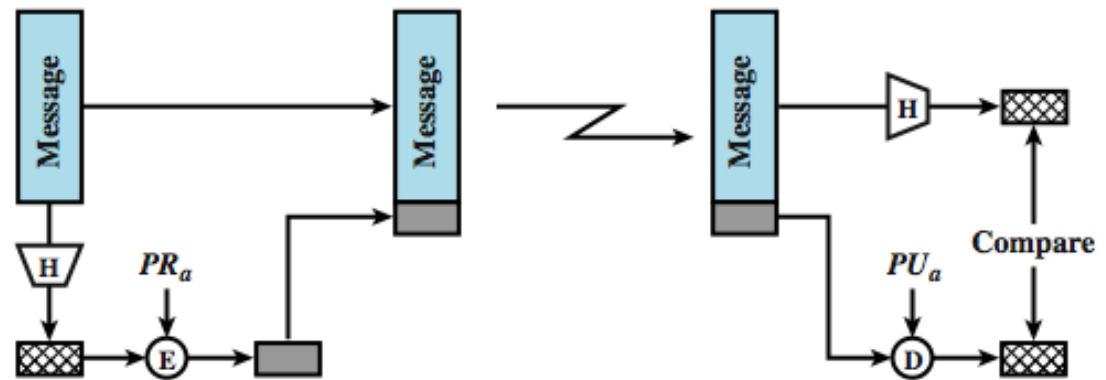
Secure Hash Functions



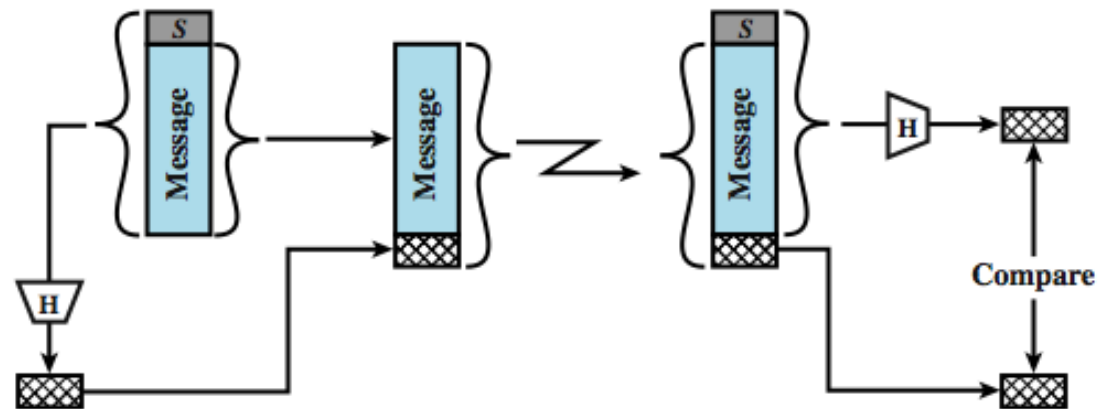
Message Auth



(a) Using conventional encryption



(b) Using public-key encryption



(c) Using secret value

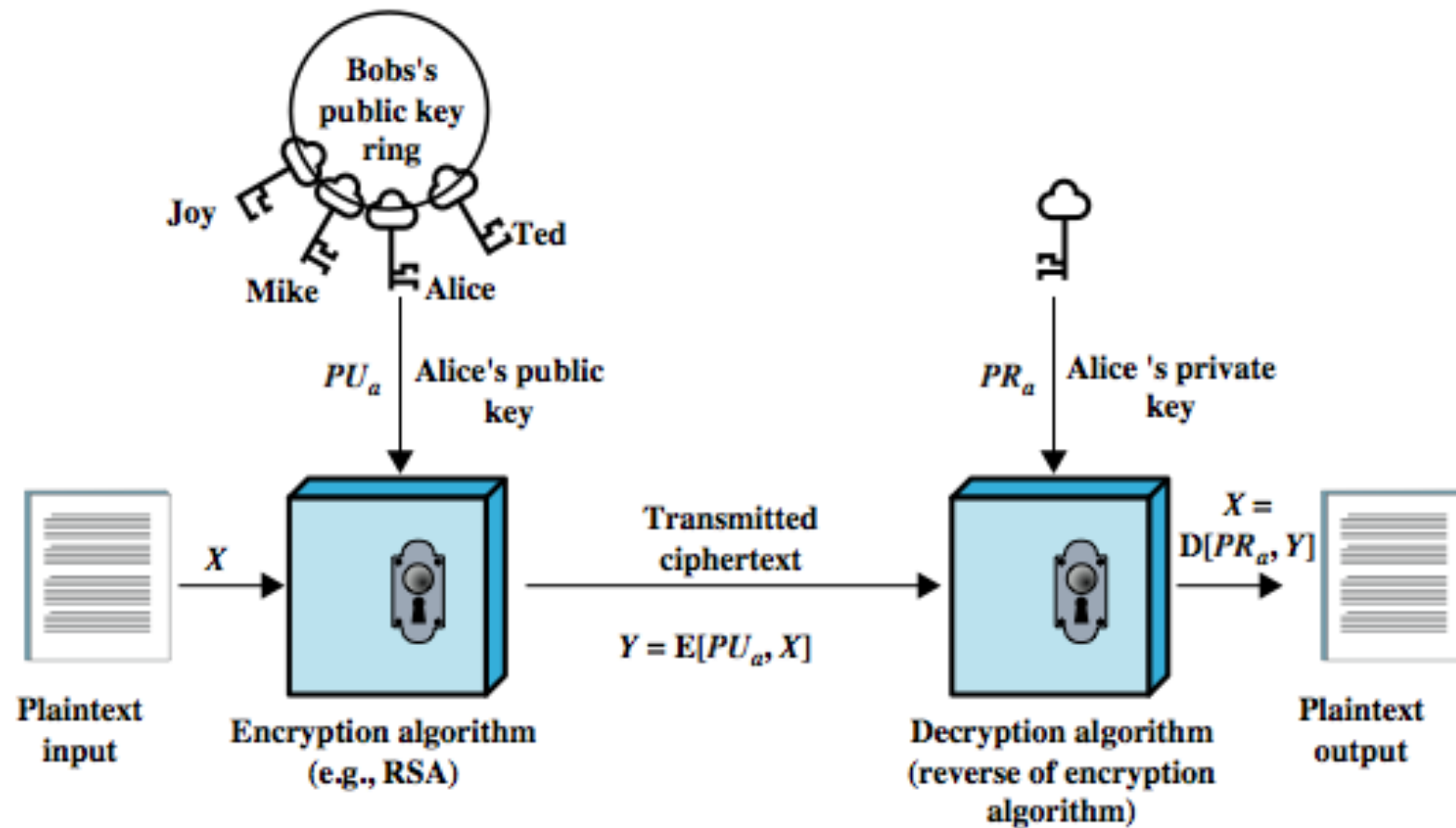
Hash Function Requirements

- applied to any size data
- H produces a fixed-length output.
- $H(x)$ is relatively easy to compute for any given x
- one-way property
 - computationally infeasible to find x such that $H(x) = h$
- weak collision resistance
 - computationally infeasible to find $y \neq x$ such that $H(y) = H(x)$
- strong collision resistance
 - computationally infeasible to find any pair (x, y) such that $H(x) = H(y)$

Hash Functions

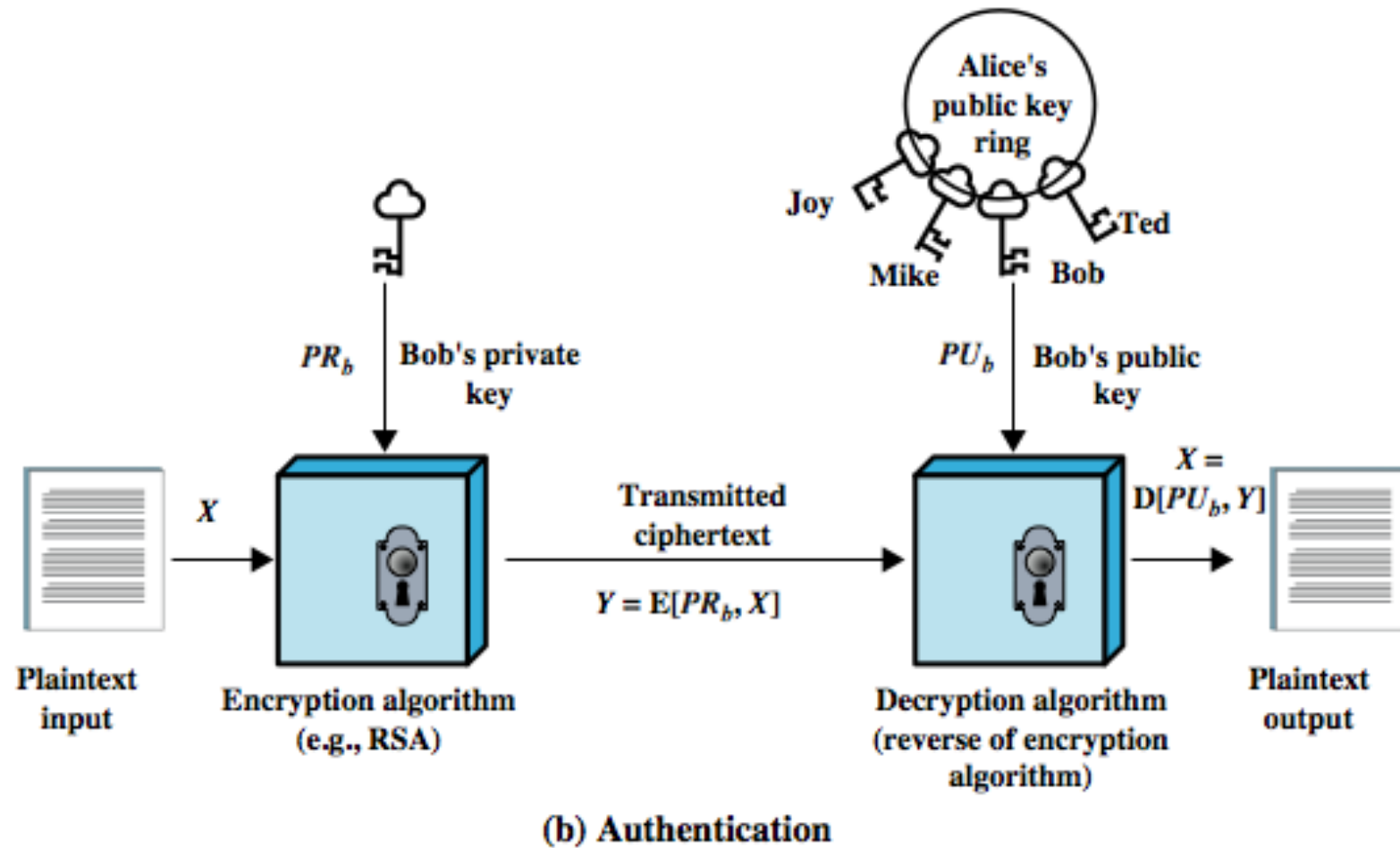
- two attack approaches
 - cryptanalysis
 - exploit logical weakness in alg
 - brute-force attack
 - trial many inputs
 - strength proportional to size of hash code ($2^{n/2}$)
- SHA most widely used hash algorithm
 - SHA-1 gives 160-bit hash
 - more recent SHA-256, SHA-384, SHA-512 provide improved size and security

Public Key Encryption



(a) Confidentiality

Public Key Authentication



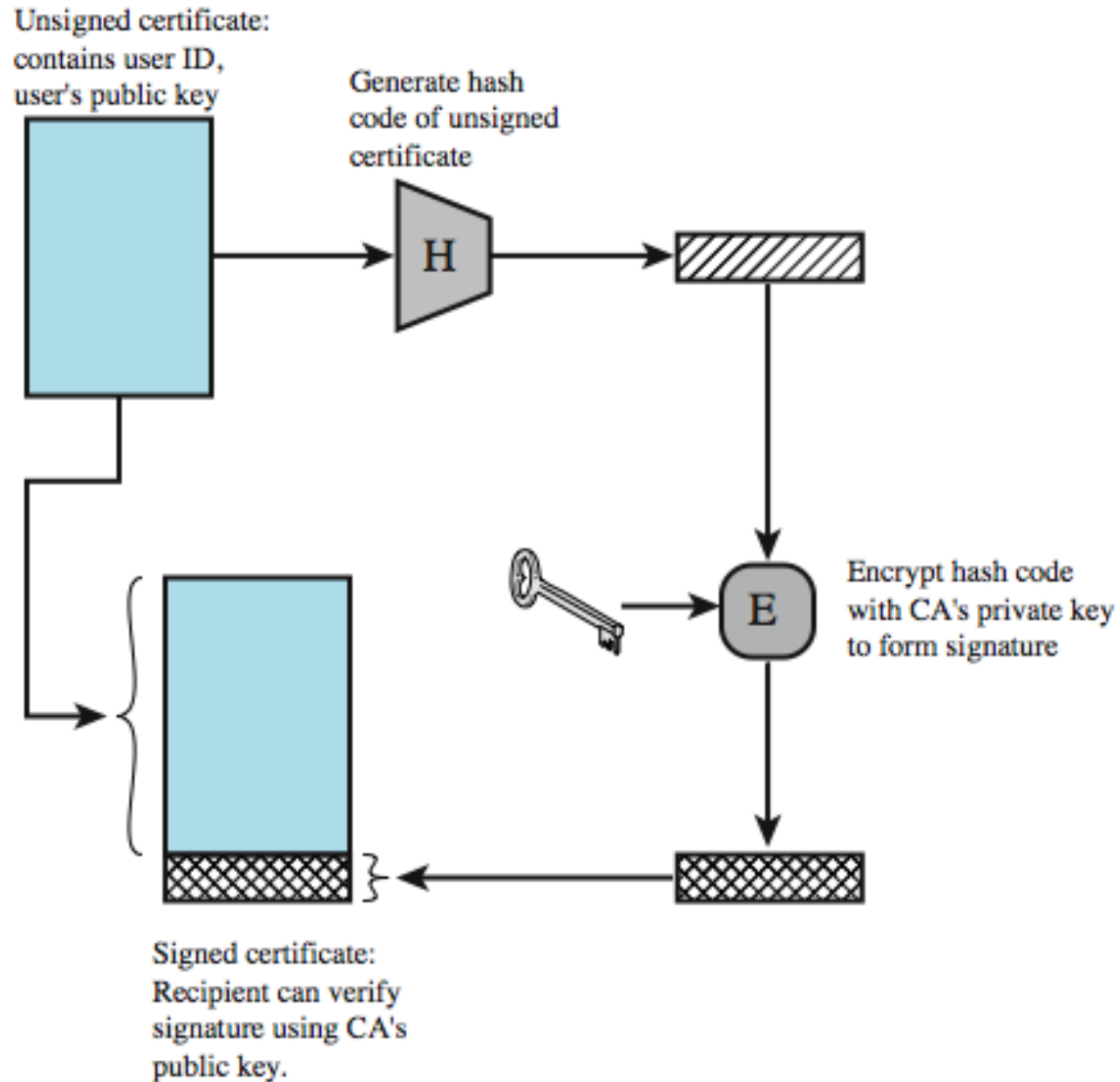
Public Key Requirements

1. computationally easy to create key pairs
2. computationally easy for sender knowing public key to encrypt messages
3. computationally easy for receiver knowing private key to decrypt ciphertext
4. computationally infeasible for opponent to determine private key from public key
5. computationally infeasible for opponent to otherwise recover original message
6. useful if either key can be used for each role

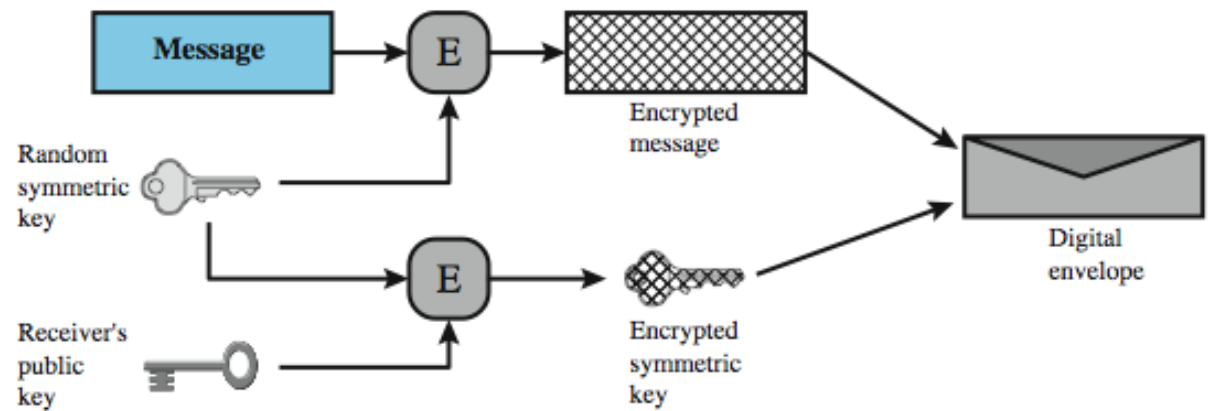
Public Key Algorithms

- RSA (Rivest, Shamir, Adleman)
 - developed in 1977
 - only widely accepted public-key encryption alg
 - given tech advances need 1024+ bit keys
- Diffie-Hellman key exchange algorithm
 - only allows exchange of a secret key
- Digital Signature Standard (DSS)
 - provides only a digital signature function with SHA-1
- Elliptic curve cryptography (ECC)
 - new, security like RSA, but with much smaller keys

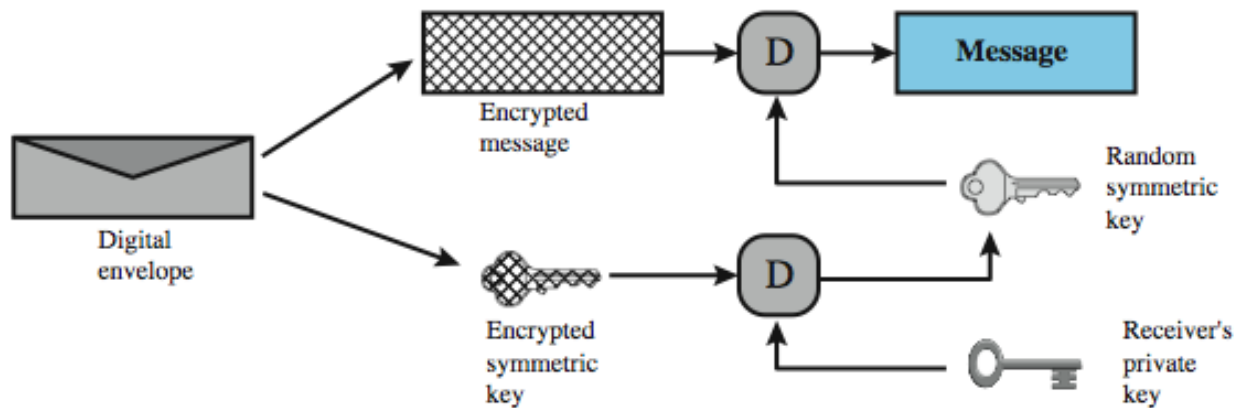
Public Key Certificates



Digital Envelopes



(a) Creation of a digital envelope



(b) Opening a digital envelope

Random Numbers

- random numbers have a range of uses
- requirements:
- randomness
 - based on statistical tests for uniform distribution and independence
- unpredictability
 - successive values not related to previous
 - clearly true for truly random numbers
 - but more commonly use generator

Pseudorandom versus Random Numbers

- often use algorithmic technique to create pseudorandom numbers
 - which satisfy statistical randomness tests
 - but likely to be predictable
- true random number generators use a nondeterministic source
 - e.g. radiation, gas discharge, leaky capacitors
 - increasingly provided on modern processors

Practical Application: Encryption of Stored Data

- common to encrypt transmitted data
- much less common for stored data
 - which can be copied, backed up, recovered
- approaches to encrypt stored data:
 - back-end appliance
 - library based tape encryption
 - background laptop/PC data encryption

Summary

- introduced cryptographic algorithms
- symmetric encryption algorithms for confidentiality
- message authentication & hash functions
- public-key encryption
- digital signatures and key management
- random numbers

Public-Key Cryptography and Message Authentication

- now look at technical detail concerning:
 - secure hash functions and HMAC
 - RSA & Diffie-Hellman Public-Key Algorithms

Simple Hash Functions

- a one-way or secure hash function used in message authentication, digital signatures
- all hash functions process input a block at a time in an iterative fashion
- one of simplest hash functions is the bit-by-bit exclusive-OR (XOR) of each block

$$C_i = b_{i1} \oplus b_{i2} \oplus \dots \oplus b_{im}$$

- effective data integrity check on random data
- less effective on more predictable data
- virtually useless for data security

SHA Secure Hash Functions

- SHA originally developed by NIST/NSA in 1993
- was revised in 1995 as SHA-1
 - US standard for use with DSA signature scheme
 - standard is FIPS 180-1 1995, also Internet RFC3174
 - produces 160-bit hash values
- NIST issued revised FIPS 180-2 in 2002
 - adds 3 additional versions of SHA
 - SHA-256, SHA-384, SHA-512
 - with 256/384/512-bit hash values
 - same basic structure as SHA-1 but greater security
- NIST intend to phase out SHA-1 use

Other Secure Hash Functions

- most based on iterated hash function design
 - if compression function is collision resistant
 - so is resultant iterated hash function
- MD5 (RFC1321)
 - was a widely used hash developed by Ron Rivest
 - produces 128-bit hash, now too small
 - also have cryptanalytic concerns
- Whirlpool (NESSIE endorsed hash)
 - developed by Vincent Rijmen & Paulo Barreto
 - compression function is AES derived W block cipher
 - produces 512-bit hash

RSA Public-Key Encryption

- by Rivest, Shamir & Adleman of MIT in 1977
- best known & widely used public-key alg
- uses exponentiation of integers modulo a prime
- encrypt: $C = M^e \bmod n$
- *decrypt*: $M = C^d \bmod n = (M^e)^d \bmod n = M$
- both sender and receiver know values of n and e
- only receiver knows value of d
- public-key encryption algorithm with
 - public key $PU = \{e, n\}$ & private key $PR = \{d, n\}$.

RSA Algorithm

Key Generation

Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calculate $\phi(n) = (p - 1)(q - 1)$	
Select integer e	$\gcd(\phi(n), e) = 1$; $1 < e < \phi(n)$
Calculate d	$de \bmod \phi(n) = 1$
Public key	$KU = \{e, n\}$
Private key	$KR = \{d, n\}$

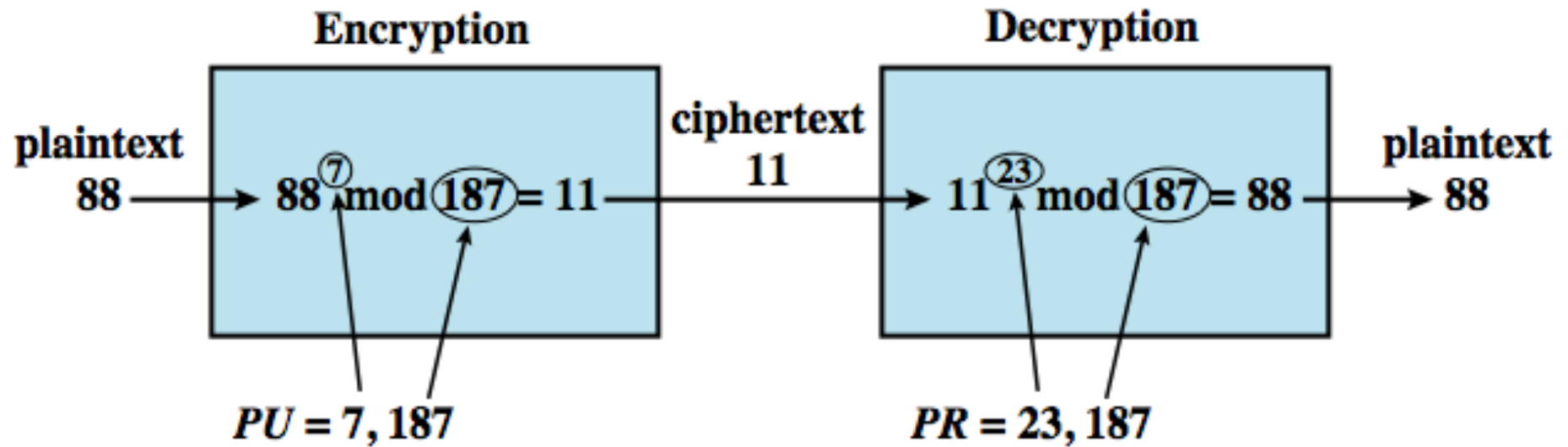
Encryption

Plaintext:	$M < n$
Ciphertext:	$C = M^e \pmod{n}$

Decryption

Ciphertext:	C
Plaintext:	$M = C^d \pmod{n}$

RSA Example



Attacks on RSA

- brute force
 - trying all possible private keys
 - use larger key, but then slower
- mathematical attacks (factoring n)
 - see improving algorithms (QS, GNFS, SNFS)
 - currently 1024-2048-bit keys seem secure
- timing attacks (on implementation)
 - use - constant time, random delays, blinding
- chosen ciphertext attacks (on RSA props)

Diffie-Hellman Key Exchange

- first public-key type scheme proposed
- by Diffie & Hellman in 1976 along with the exposition of public key concepts
 - note: now know that Williamson (UK CESG) secretly proposed the concept in 1970
- practical method to exchange a secret key
- used in a number of commercial products
- security relies on difficulty of computing discrete logarithms

Diffie-Hellman Algorithm

Global Public Elements

q prime number
 α $\alpha < q$ and α a primitive root of q

User A Key Generation

Select private X_A $X_A < q$
Calculate public Y_A $Y_A = \alpha^{X_A} \bmod q$

User B Key Generation

Select private X_B $X_B < q$
Calculate public Y_B $Y_B = \alpha^{X_B} \bmod q$

Generation of Secret Key by User A

$$K = (Y_B)^{X_A} \bmod q$$

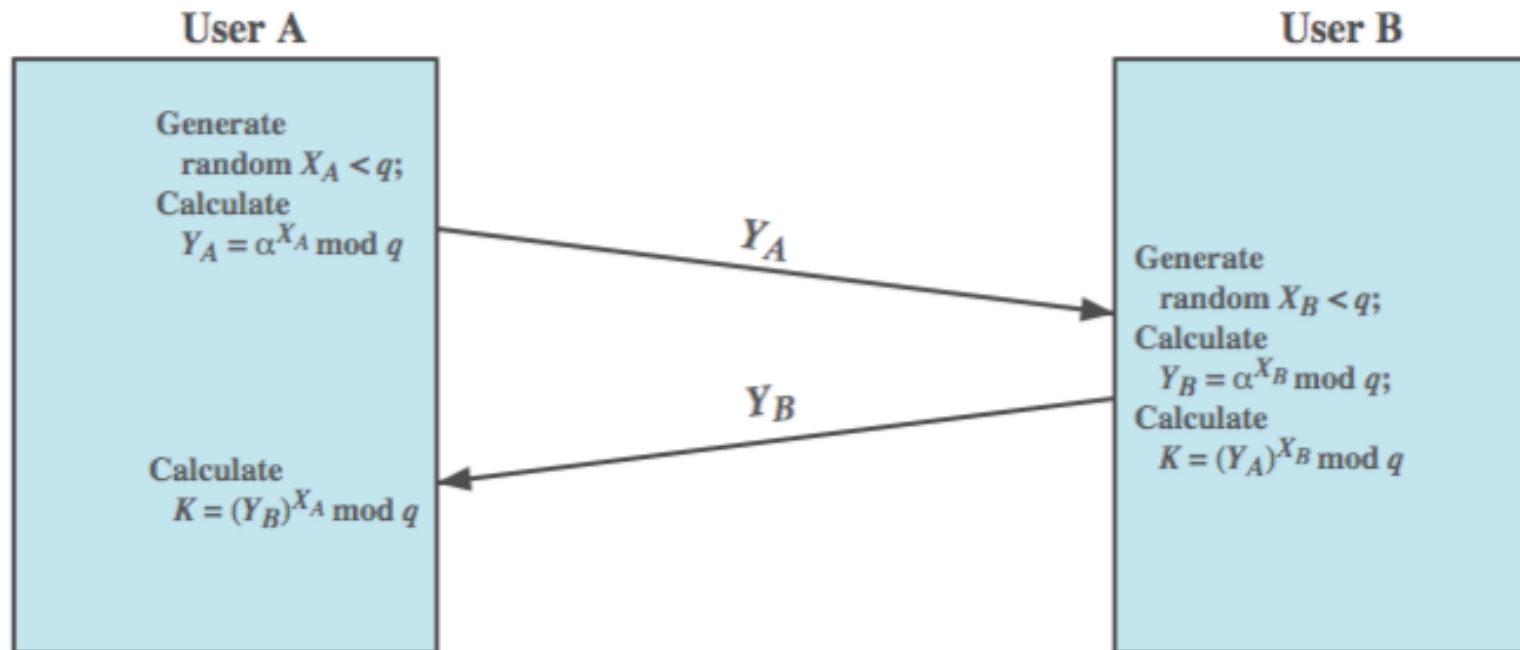
Generation of Secret Key by User B

$$K = (Y_A)^{X_B} \bmod q$$

Diffie-Hellman Example

- have
 - prime number $q = 353$
 - primitive root $\alpha = 3$
- A and B each compute their public keys
 - A computes $Y_A = 3^{97} \bmod 353 = 40$
 - B computes $Y_B = 3^{233} \bmod 353 = 248$
- then exchange and compute secret key:
 - for A: $K = (Y_B)^{X_A} \bmod 353 = 248^{97} \bmod 353 = 160$
 - for B: $K = (Y_A)^{X_B} \bmod 353 = 40^{233} \bmod 353 = 160$
- attacker must solve:
 - $3^a \bmod 353 = 40$ which is hard
 - desired answer is 97, then compute key as B does

Key Exchange Protocols



Man-in-the-Middle Attack

- attack is:
 1. Darth generates private keys X_{D1} & X_{D2} , and their public keys Y_{D1} & Y_{D2}
 2. Alice transmits Y_A to Bob
 3. Darth intercepts Y_A and transmits Y_{D1} to Bob. Darth also calculates K2
 4. Bob receives Y_{D1} and calculates K1
 5. Bob transmits X_A to Alice
 6. Darth intercepts X_A and transmits Y_{D2} to Alice. Darth calculates K1
 7. Alice receives Y_{D2} and calculates K2
- all subsequent communications compromised

Other Public-Key Algorithms

- Digital Signature Standard (DSS)
 - FIPS PUB 186 from 1991, revised 1993 & 96
 - uses SHA-1 in a new digital signature alg
 - cannot be used for encryption
- elliptic curve cryptography (ECC)
 - equal security for smaller bit size than RSA
 - seen in standards such as IEEE P1363
 - still very new, but promising
 - based on a mathematical construct known as the elliptic curve (difficult to explain)

Summary

- discussed technical detail concerning:
 - secure hash functions and HMAC
 - RSA & Diffie-Hellman Public-Key Algorithms