University of Houston

Foundations of Security

COSC 6347

Midterm Review

Author
K.M. Hourani

Based on Notes By
Dr. Aron Laszka

October 1, 2020

1 Introduction to Security

1.1 Objectives

	Term	Definition
CIA	Confidentiality	not available to unauthorized entities
	Integrity	cannot be altered by unauthorized entities
	Availability	available to authorized entities
	Non-repudiation Accountability	actions can be provably traced back to an entity
	Privacy	individuals have control over information related to them

1.2 Challenges

Weakest link – principle that the defender needs to find and fix all vulnerabilities, but attacker needs to find only a single vulnerability

Security is a process, not a product – attackers continuously looking for new vulnerabilities, so systems must be regularly updated and continuously monitored.

Tension between security and

- usability
- functionality
- efficiency
- time-to-market
- development cost

Value of security often only perceived when there is a security failure

Can be measured by

- checking compliance
- pentesting

2 Introduction to Cryptography

2.1 Attacker Modeling Principles

Security is defined with respect to an attacker model – what the attacker

- can do
- knows
- wants to achieve

Generally better to overestimate the attacker's capabilities, knowledge, and determination.

Safe to assume attacker knows

- algorithms
- system design
- implementation
- configuaration

but the attacker cannot know *truly* random values.

2.2 Security by Obscurity

Security by obscurity – providing security by keeping the design or implementation of a system secret

Generally rejected by security experts, researchers, standard bodies, i.e., everyone.

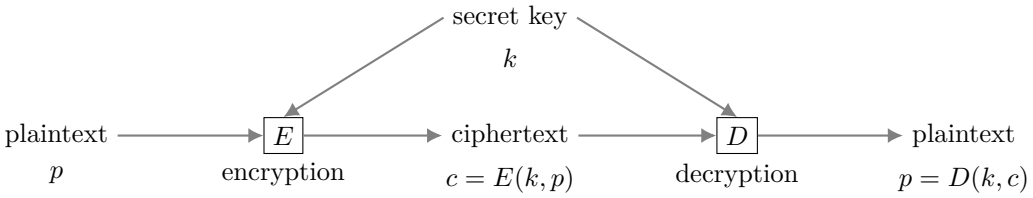
Obscurity can slow down, but not stop, an attack:

- if we thought of something, attacker might also
- attacker might try attack for many possible design/implementation choices

Can create false sense of security.

2.3 Symmetric-Key Ciphers

Sender and receiver share a secret key k



Types of attacks:

Acronym	Attack	Description
COA	ciphertext only	only the algorithms used and the ciphertext are known
KPA	known plaintext	one or more plaintext-cipher pairs is known
CCA	chosen ciphertext	one or more *chosen* plaintext-cipher pairs is known
CPA	chosen plaintext	can obtain the ciphertext for any plaintext
CTA	chosen text	both chosen ciphertext and chosen plaintext
	brute-force	every possible key is tried
	cryptanalytic	relies on the nature of the algorithm/characteristics of the plaintext

2.4 Kerckhoffs's Principle

Kerckhoffs's Principle – a cryptographic system should be secure, even if all of its details, except for the key, are publicly known. Rejection of security by obscurity

3 Stream Ciphers

3.1 Perfect Security

Perfect security – attacker gains no information about the plaintext from observing the ciphertext, formally,

$$\mathbb{P}(P = p) = \mathbb{P}(P = p \mid E(K, P) = c)$$

i.e., that the plaintext and ciphertext are independent

One-time pad – perfect security in which a single-use encryption key at least as long as the plaintext is chosen randomly and used to encrypt only a single message

3.2 Semantic Security

Semantic security – attacker advantage for any efficiently computable guess is negligible over random guessing

Many-time pad: reusing the one-time key for multiple plaintext. Attacker can recover $p_1 \oplus p_2$:

$$\begin{aligned} c_1 \oplus c_2 &= (p_1 \oplus k) \oplus (p_2 \oplus k) \\ &= (p_1 \oplus p_2) \oplus (k \oplus k) \\ &= p_1 \oplus p_2 \end{aligned}$$

and if attacker knows p_1, can recover p_2:

$$\begin{aligned} p_1 \oplus (c_1 \oplus c_2) &= p_1 \oplus (p_1 \oplus p_2) \\ &= (p_1 \oplus p_1) \oplus p_2 \\ &= p_2 \end{aligned}$$

3.3 General Model of Stream Ciphers

Make one-time pad practical by securely extending the key.

Pseudorandom Number Generator

pseudorandom number generator (PRNG) – takes fixed-length seed and generates a sequence of bits using a deterministic algorithm

Requirements:

- performance – generates key as long as plaintext, so must be computationally efficient
- security – generated sequence must be indistinguishable from true randomness
  - cryptanalytic attack
    - uniform distribution – 0s and 1s occur with approximately same frequency
    - independence – no subsequence can be inferred from another, disjoint subsequence
  - brute-force attack
    - n bit key has 2^n possible values – attacker can try all
    - key must be sufficiently long – in 2014, NIST recommends 112-bits
    - as computers become faster, key length must be increased

How Stream Cipher Works

stream cipher – takes fixed-length seed and uses a PRNG to produce sequence of bits as long as the plaintext then encrypts with XOR

Use PRNG to generate the sequence up to the length of the plaintext, then to

encrypt — XOR plaintext with key

decrypt — XOR ciphertext with key

3.4 Key-Reuse Problem

If attacker learns $p_1 \oplus p_2$, $p_2 \oplus p_3$, $p_1 \oplus p_3$, ..., they can recover other plaintexts. Solutions:

- one continuous sequence that allows seeking to any position in the key
- nonce – number used once
  - xor key with nonce for each plaintext to produce different key

3.5 RC4

Old WiFi and Web Security standard

RC4 Advantages

- variable key length (from 8 to 2048 bits)
- very simple, uses byte-oriented operations:
  - only 8 to 16 machine operations required per output byte

Applications

- Wifi: WEP and WPA
  - broken in 2001, deprecated in 2004
- Web Security (HTTPS): SSL and TLS
  - broken in 2013, deprecated in 2015

RC4 has been retired.

Advantages

- fast software implementation (simple 32-bit operations)
- can seek to any position in output sequence
- 64-bit nonce part of algorithm to prevent key-reuse

currently, no attacks better than brute-force attack known.

Algorithm

- Output in blocks of 16×32 bits
- internal state: 16×32 bits
  - initialized using key, nonce, and seek position
- State updated with XOR, 32-bit addition mod 2^{32}, and rotating 32 bit values
- Performs 20 rounds of XOR-add-rotate, each of which updates all values in state
- State added to original state to obtain output

3.6 Salsa20/ChaCha20

State of the Art Stream Cipher Salsa20 (and more secure, more efficient variant ChaCha20)

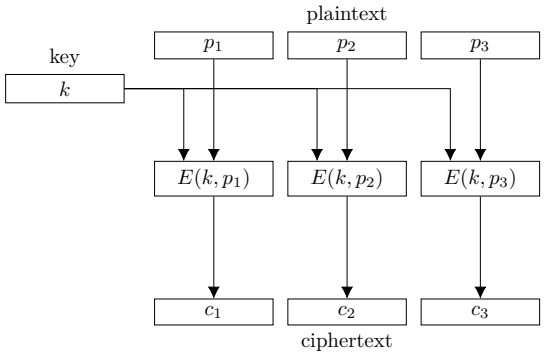
Key length is 128 or 256 bits.

4 Block Ciphers

Unlike stream ciphers, block ciphers have different encryption and decryption operations. A block cipher encrypts plaintext in fixed-length blocks

4.1 Design Considerations

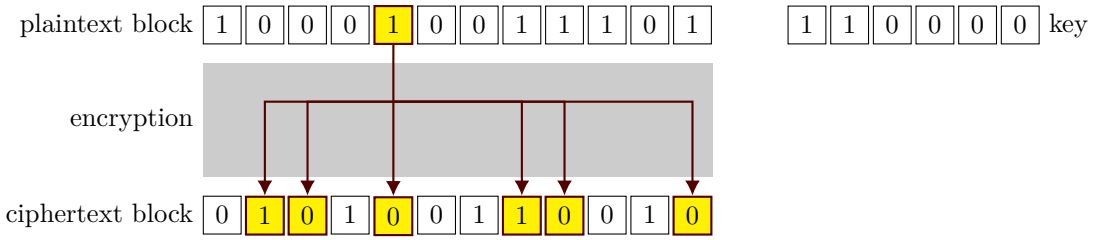
- Key Size
  - number of possible k-bit keys is 2^k
  - k must be sufficiently large to prevent brute-force attacks
- Block Size
  - too short $\rightarrow$ does not hide patterns in plaintext
    - e.g. $n = 8$ bits is 1 character
    - same as substitution cipher
  - too long – impractical, wasteful
- encryption must be invertible
  - different input blocks must be transformed into different output blocks
  - can be viewed as a permutation on all n-bit blocks
  - $(2^n)!$ possible permutations



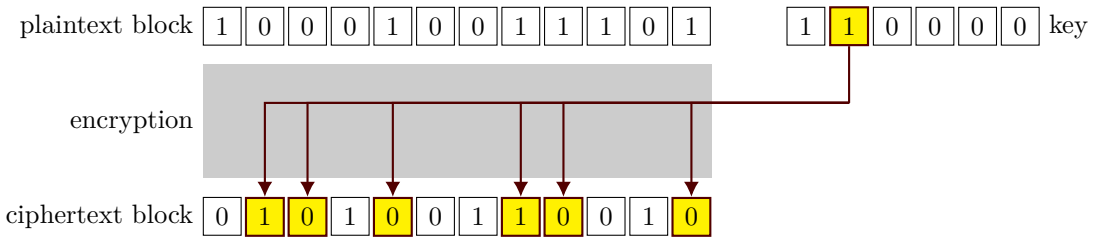
4.2 Secure Block Cipher

An n-bit block cipher is secure (for a computationally bounded attacker) if it is indistinguishable from a random permutation of n-bit blocks.

diffusion – each plaintext bit should affect the value of many ciphertext bits



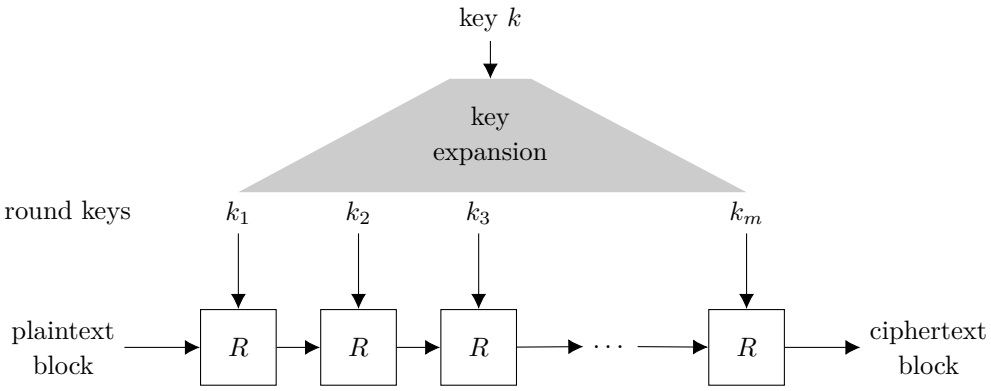
confusion – each bit of the ciphertext should depend on many bits of the key



4.3 Iterated Block Ciphers

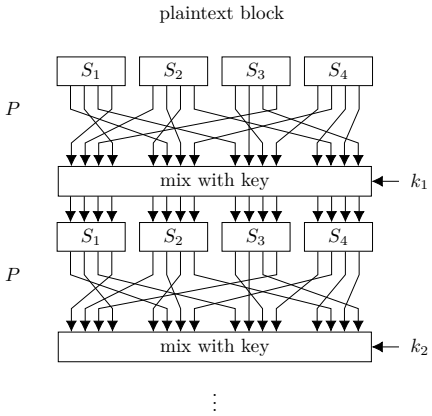
Hard to design a single invertible function that satisfies diffusion and confusion. Use a round function

- R – round function
  - relatively weak transformation that introduces diffusion and confusion
  - by iterating, builds strong block cipher



4.4 Substitution-Permutation Ciphers

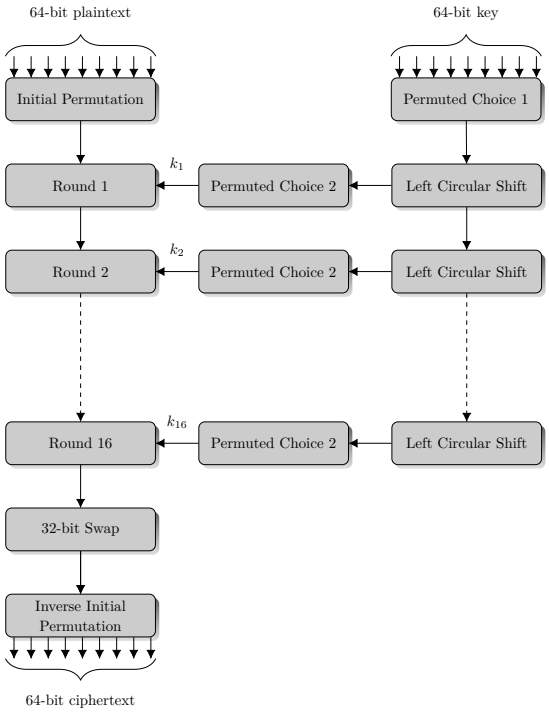
Common subtype of iterated block cipher, each round R consists of



- Substitution S
  - substitutes small block with another small block
  - ideally, changing one input bit changes half of output bits
- Permutation P
  - permutation of all bits

4.5 DES

Data Encryption Standard (DES)



- block size – 64 bits
- key size – 56 bits
  - 56 bit random
  - 8 bit parity check
- iterated substitution cipher of 16 rounds
- initial permutation
  - no cryptographic significance
  - facilities loading blocks in and out of 8-bit hardware
- key permutation
  - discards parity bits
  - no cryptographic significance

Advantages

- relatively secure against cryptanalytic attacks – best attack in 2^{43} steps
- thoroughly studied and widely supported

Disadvantages

Vulnerable to brute-force attacks – 56-bit key $\rightarrow$ 2^{56} possible keys.

4.6 Feistel Network

Consists of encryption and decrpytion round

- Encryption round
  - input – block from previous round (or plaintext)
  - divide input in half – L_i and R_i
  - derive round key k_i from secret key (different each round)
  - output

$$L_{i+1} = R_i$$
$$R_{i+1} = L_i \oplus F(k_i, R_i)$$

- Decryption round
  - we can invert encryption without inverting F

$$R_i = L_{i+1}$$
$$L_i = R_{i+1} \oplus F(k_i, L_{i+1})$$
$$= R_{i+1} \oplus F(k_i, R_i)$$

4.7 AES

Advanced Encryption Standard (AES)

- Substitution-permutation
  - but **not** a Feistel network
- each round must be invertible for decrpytion
- key expansion and schedule – generates different round key each round
- number of rounds n depends on key size k

k	n
128	10
192	12
256	14

4.8 AES Round

- input
  - 128-bit state from previous round (or plaintext) as 4×4 byte matrix
  - 128-but round key from key schedule
- output – 128-bit state
- each round consists of multiple steps
  - AddRoundKey – xor round key to state
  - 128-but round key from key schedule
  - substitution and permutation
    - SubBytes
    - ShiftRows
    - MixColumns

SubBytes

- Each byte is replaced using an 8-bit substitution box (S-box)
  - defined using mathematical operations: multiplicative inverse over a finite field + affine transformation
- designed to resist cryptanalysis
  - minimize correlation to linear functions
  - minimize difference propagation

ShiftRows

- Cyclically shifts 2nd, 3rd, and 4th rows left

row	shift
2nd	1
3rd	2
4th	3

- ensures the 4 bytes of each column are spread to 4 different columns → provides diffusion
  - without this step each input byte would only affect a single column

MixColumns

- Each column is multiplied by a fixed matrix
  - invertible linear transformation
- good mixing among bytes of each column → provides diffusion
  - in conjunction with ShiftRows, ensures each output bit depends on every input bit after a few rounds

4.9 AES Decryption

- each step is invertible
  - InvertMatrixColumns – multiply by matrix inverse
  - InvertShiftRows – shift rows cyclically to right
  - InvertSubBytes – invert affine transformation and multiplicative inverse
  - InvertAddRoundKey – xor round key to state

- Round keys are used in reverse order

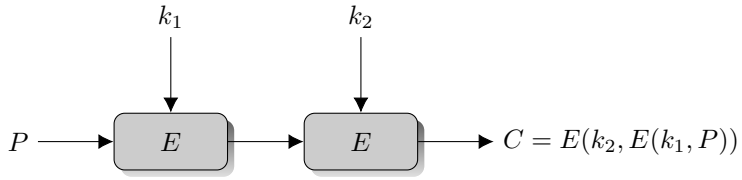
4.10 AES Performance and Security

- Operations on bytes and 32-bit words
  - most operations can be precomputed
- Supported by hardware – AES instruction set for CPUs
- very secure – best known attack takes 2^{126} steps, only 4x faster than brute-force attack

4.11 Multiple Encryption

Use same encryption algorithm multiple times, each time with a different key

2DES

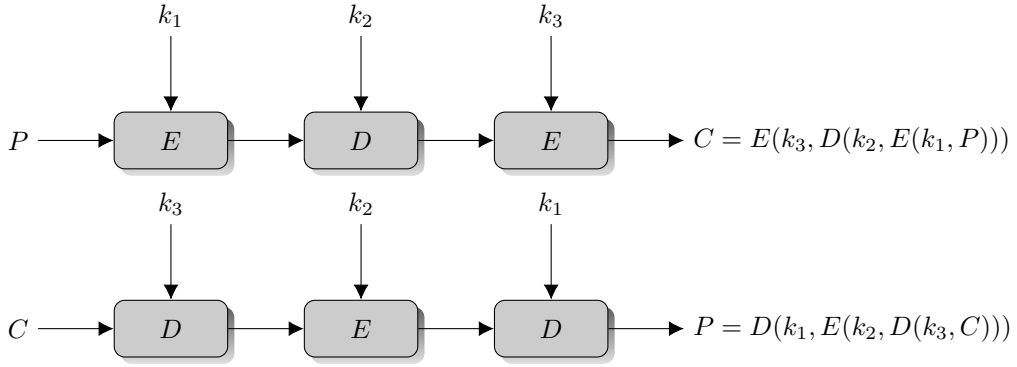


4.11.1 Meet in the Middle

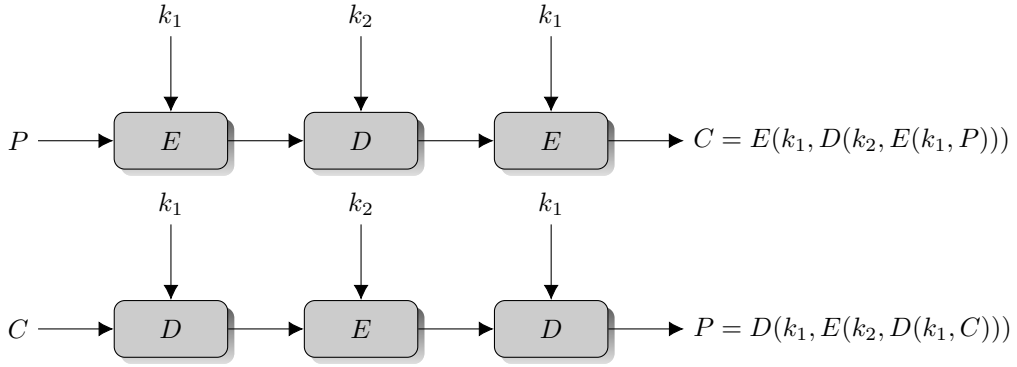
meet-in-the-middle attack – trade time for storage

- brute-force attack requires 2^{112} steps
- store $\sqrt{2^{112}} = 2^{56}$ values, $\approx 2^{56}$ steps
- generally, storing $2^{56-m} \rightarrow \approx 2^{56+m}$ steps

3DES Using 3 keys instead of 2. Naive implementation suffers same vulnerability to meet-in-the-middle attack as 2DES. Instead use EDE – Encryption-Decryption-Encryption



Above has 3 keys, but vulnerable to more sophisted MITM attack – effectively only 112-bit security. Taking $k_1 = k_3$ provides 80-bits of effective security.



5 Block Cipher Modes of Operation

Orientation	Mode		Use
Block	ECB	Electronic Code Book	single block
	CBC	Cipher Block Chaining	commonly used
Stream	OFB	Output Feedback	no random access
	CFB	Cipher Feedback	self-synchronized stream cipher
	CTR	Counter Mode	very efficient, very commonly used

Mode	Advantages	Disadvantages
ECB	blocks can be encrypted/decrypted in parallel	identical plaintext → identical ciphertext
		attacker can rearrange or remove blocks from ciphertext
CBC	hides patterns in the plaintext	blocks cannot be encrypted in parallel
	blocks can be decrypted in parallel	attacker might be able to rearrange or remove blocks from ciphertext
		IV needs integrity protection
		attacker might be able to tamper with bits of the plaintext
OFB	bit errors do not propagate	blocks cannot be encrypted or decrypted in parallel
	pre-computation is possible	attacker can tamper with the bits of the plaintext
CFB	blocks can be decrypted in parallel	blocks cannot be encrypted in parallel
	self-synchronizing stream cipher	attacker might be able to tamper with the bits of the plaintext
		attacker might be able to rearrange or remove blocks
CTR	blocks can be encrypted and decrypted in parallel	attacker can tamper with bits of the plaintext
	bit errors do not propagate	
	pre-computation is possible	

6 Public-Key Cryptography

public-key cryptography, also called asymmetric-key cryptography

Use a pair of keys, one public, one private. Solves

- public-key encryption → key exchange
- digital signature → non-repudiation

6.1 Public-Key Encryption

- everyone knows public key → sender can encrypt
- received knows private key → receiver can decrypt
- attacker does not know private key → cannot decrypt

- public key can be published

3 algorithms:

- key generation $G \to (PU, PR)$
- encryption – $E(PU, M) \to C$
  - takes public key PU and plaintext M and outputs ciphertext C
- encryption – $D(PR, C) \to P$
  - takes private key PR and ciphertext C and outputs plaintext P

Unlike symmetric-key, requires largest keys and is much slower

	Symmetric	Asymmetric
Typical Design	series of subs. and perms.	hard mathematical problems
Key	completely random	special structure, expensive to generate
Rec. Key Size	128 - 256 bits	2048 - 15360 bits
Performance	fast	slow

6.2 RSA

Choose two large primes p and q, set $n = pq$ and choose e such that $\gcd(e, \varphi(n)) = 1$. Set $d = e^{-1} \bmod \varphi(n)$, then

$$PU = (e, n)$$
$$PR = (d, n)$$

Encrypt plaintext M with $C = M^e \bmod n$ and decrypt with $M = C^d \bmod n$.

6.3 Security of RSA

Security comes from difficulty of determining $C^{1/e} \bmod n$ efficiently. Best known algorithm is to factor $n = pq$ and compute $e^{-1} \bmod \varphi(n)$. Integer factorization is assumed to be hard but this is unproved.

Very slow encryption, so commonly used to encrypt a secret key for use with symmetric-key encryption. Comparable symmetric key security (number of bits):

Symmetric	RSA
80	1024
128	3072
256	15360

6.4 ElGamal Encryption

Choose a large prime q, primitive root α of q, and $X \in \{1, 2, \ldots, q-1\}$. Set $Y = \alpha^x \bmod q$ and

$$PU = (q, \alpha, U)$$
$$PR = (q, \alpha, X)$$

Encryption:

Choose random $k \in \{0, 1, \ldots, q-2\}$ and set $K = y^k \bmod q$. Then return (C_1, C_2) where

$$C_1 = \alpha^k \bmod q$$
$$C_2 = KM \bmod q$$

Decryption:

Set $K = C_1^X \bmod q$, then $M = C_2 K^{-1} \bmod q$.

6.5 Security of ElGamal

Security comes from difficulty of discrete logarithm, widely believed to be computationally hard.

Recover X requires computing discrete-$\log_\alpha$ of $Y \bmod q$.

Recover k requires computing discrete-$\log_\alpha$ of $C_1 \bmod q$.

6.6 Elliptic Curve Cryptography

Elliptic Curve is a set of points (x, y) such that

$$y^2 = x^3 + ax + b$$

Binary operation $P + Q$: draw line through P and Q, find where it intersects the curve, call this R. Then $P + Q = -R$. Combined with a point at infinity (the identity), forms an abelian (commutative) group. Can naturally define $kP = \underbrace{P + P + \cdots + P}_{k-\text{times}}$ and so can implement ElGamal in a straightforward fashion.

160-bit ECC is comparable in security to 1024-bit RSA key.

7 Hash Functions

A hash function H maps a variable-length input to a fixed-length hash value. It must be

- efficient – computing $H(M)$ is easy
- one-way – finding an input for which the output is a given hash-value is hard
- collision-resistant – finding two inputs for which the hash-values are the same is hard
- pseudorandom

7.1 Security Requirements

Requirement	Definition
preimage resistance one-way property	given a hash value h, it is computationally infeasible to find an input y such that $H(y) = h$
second preimage resistance weak collision resistance	given input x, it is computationally infeasible to find y such that $x \neq y$ but $H(x) = H(y)$
collision resistance strong collision resistance	computationally infeasible to find any pair of inputs (x, y) such that $x \neq y$ but $H(x) = H(y)$; implies weak collision resistance

7.2 Brute-Force Attacks

Try random inputs until a collision is found. If output is m bits, then probability of success for a single try is 2^{-m}. Expected number of tries until success is 2^m.

Collision resistance attacks are a consequence of the birthday paradox:

Theorem 7.1 ▶ The Birthday Paradox

The probability that a collision occurs given a random map from $\{1, 2, \ldots, n\}$ to $\{1, 2, \ldots, m\}$ is

$$1 - \prod_{i=1}^{n-1} 1 - \frac{i}{m}$$

Since $1 - x \approx e^{-x}$, we have

$$1 - \prod_{i=1}^{n-1} 1 - \frac{i}{m} \approx 1 - e^{-\frac{n^2}{2m}}$$

so when $n = \Omega(\sqrt{m})$, a collision is likely.

8 Message Authentication

9 Digital Signatuers

10 Key Distribution

11 Public-Key Distribution