Chapter 1

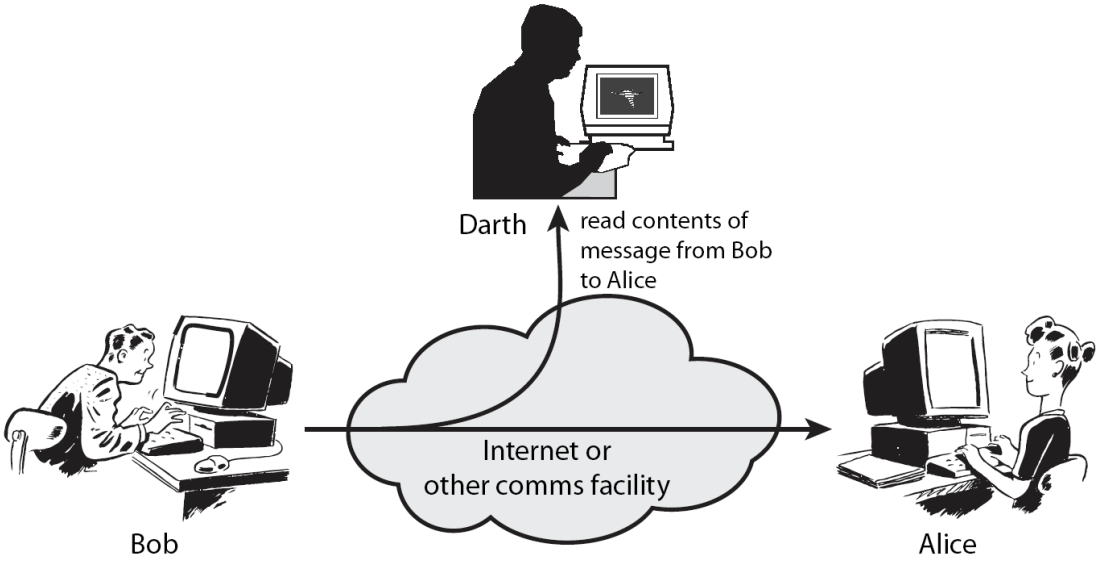
1. (Final 2007) Compare active attack to passive attack, giving an example for each

Sol:

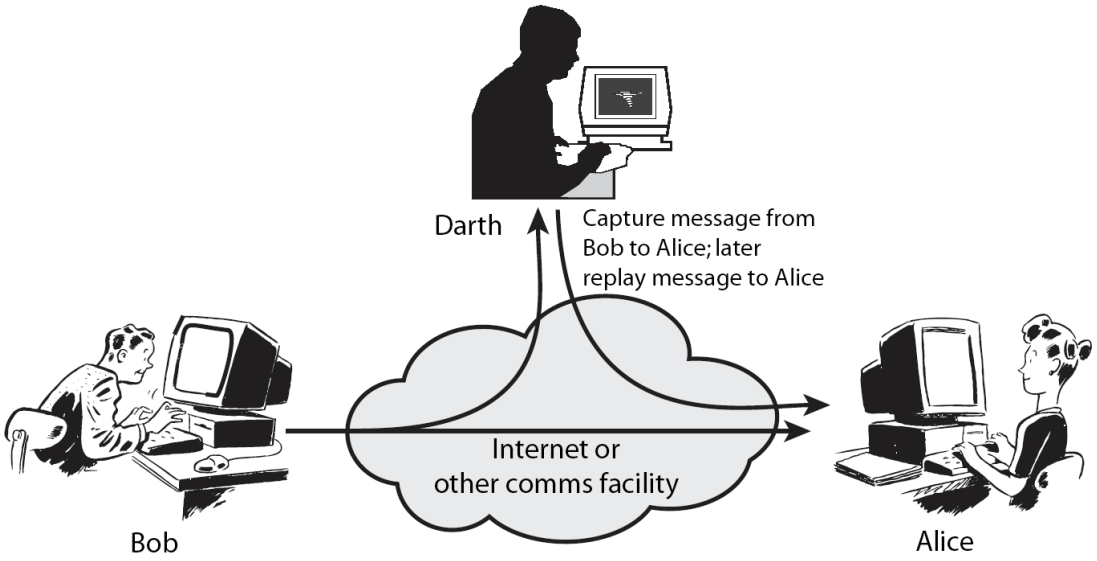
Attacks: Passive and Active

Comparison	Passive Attacks	Active Attacks
Definition	A passive attack attempts to learn or make use of information from the system but does not affect system resources.	Active attacks involve some modification of the data stream or the creation of a false stream
Types or Categories	Two types of passive attacks are: + release of message contents + traffic analysis	Four categories: + masquerade + replay + modification of messages + denial of service
Detection	difficult to detect	Can be detected
Prevention	Measures are available to prevent their success.	It is quite difficult to prevent active attacks absolutely, because of the wide variety of potential physical, software, and network vulnerabilities.
Handling	Prevention	Detection and recovering from disruption or delays caused by them

Example of Passive attacks → release of message contents:



Example of Active attacks → replay previous messages:



Chapter 2

1. (Final 2007) Using this Playfair matrix

M	F	H	I/J	K
U	N	O	P	Q
Z	V	W	X	Y
E	L	A	R	G
D	S	T	B	C

Encrypt this message:
Must see you over Cadogan West. Coming at once.

Sol: → this my solution
Remember the algorithm:

0- Divide them to pairs of letters
1. If repeating plaintext letters in the pair → separate with a filter letter (x), ex. Balxlon
2. If two plaintext letters in the same row →replaced by the letter to <u>right</u>
3. If two plaintext in the same columns → replaced by the letter <u>beneath</u>
4. Otherwise (not same row or same col.), each plaintext letter in pair → is replaced by the letter lies in its own row and column occupied by the other letter (row of the letter itself and col. of the other letter).

Sol of the question:
The message:
mu st se ey ou ov er ca do ga nw es tc om in ga to nc e**x**
(Note: we pad letter x at the end to make a pair)
Encryption of the message:
uz tb dl gz pn nw lg tg tu er ov ld bd uh fp er hw qs rz
(Note: remove spaces in sol.)

---

2. Define Cryptanalysis and Brute-force attacks

Sol:

- **Cryptanalysis:** relies on the <u>nature of the algorithm</u> plus perhaps some knowledge of the general characteristics of the plaintext or even <u>some sample plaintext- ciphertext pairs</u>. This type of attack exploits the characteristics of the algorithm to attempt to deduce a specific plaintext or to deduce the key being used.
  A cryptanalysis is an attack based on weaknesses in a particular cryptographic algorithm.

- **Brute-force attacks:** try every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained. On average,half of all possible keys must be tried to achieve success.
  A brute-force attack does not depend on the specific algorithm but depends only on bit length.

3. (Final 2010) Using a playfair matrix, encode the word "university" using the key "worldcup" and show your steps clearly.

   Sol:

   1. Fill the PlayFair matrix: 1st by the key then the remaining letters. (Note: I and j are together.)
   2. Encrypt using the algorithm(written in previous question)
      Message: un iv er si ty
      CipherText: pm fz gw zs vz

w	o	r	l	d
c	u	p	a	b
e	f	g	h	i/j
k	m	n	q	s
t	v	x	y	z

Chapter 3

1. (Final 2007) What is meant by a timing attack? Compare with brute force attack.

Sol: (Note: that's the comparison that I got)

A timing attack is one in which information about the key or the plaintext is obtained by observing how long it takes a given implementation to perform decryptions on various ciphertexts.
A timing attack exploits the fact that an encryption or decryption algorithm often takes slightly different amounts of time on different inputs.
The AES analysis process has highlighted this attack approach, and showed that it is a concern particularly with smartcard implementations, though DES appears to be fairly resistant to a successful timing attack.

Comparison	Timing Attack	Brute Force
Definition	As above	Try every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained. On average, half of all possible keys must be tried to achieve success.
Countermeasures (Handling this attack)	Although the timing attack is a serious threat, there are simple countermeasures that can be used, including using: - constant exponentiation time algorithms - adding random delays - using blind values in calculations.	It's proportional to key size. The larger the key size, the more computationally infeasible brute force attack.

Chapter 4

1. (Final 2007) Multiply 01010111 by 10000011 in GF(2^8) modulo m(x)=$x^8+x^4+x^3+x+1$

Sol (the book, page 156):

The Advanced Encryption Standard (AES) uses arithmetic in the finite field GF(2^8), with the irreducible polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$. Consider the two polynomials $f(x) = x^6 + x^4 + x^2 + x + 1$ and $g(x) = x^7 + x + 1$. Then

$$\begin{aligned} f(x) + g(x) &= x^6 + x^4 + x^2 + x + 1 + x^7 + x + 1 \\ &= x^7 + x^6 + x^4 + x^2 \end{aligned}$$

$$\begin{aligned} f(x) \times g(x) &= x^{13} + x^{11} + x^9 + x^8 + x^7 \\ &\quad + x^7 + x^5 + x^3 + x^2 + x \\ &\quad + x^6 + x^4 + x^2 + x + 1 \\ &= x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1 \end{aligned}$$

$$\begin{array}{r} x^5 + x^3 \\ x^8 + x^4 + x^3 + x + 1 \overline{)\, x^{13} + x^{11} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + 1} \\ \underline{x^{13} \qquad + x^9 + x^8 \qquad + x^6 + x^5} \\ x^{11} \qquad\qquad\qquad + x^4 + x^3 \\ \underline{x^{11} \qquad + x^7 + x^6 \qquad + x^4 + x^3} \\ x^7 + x^6 \qquad\qquad + 1 \end{array}$$

Therefore, $f(x) \times g(x) \bmod m(x) = x^7 + x^6 + 1$.

Result = 011000001

Chapter 6

1. (Final 2010) Output Feedback (OFB) is one of the DES modes. List the other four modes.

Mode	Description	Typical Application
Electronic Codebook (ECB)	Each block of 64 plaintext bits is encoded independently using the same key.	• Secure transmission of single values (e.g., an encryption key or IV)
Cipher Block Chaining (CBC)	The input to the encryption algorithm is the XOR of the next 64 bits of plaintext and the preceding 64 bits of ciphertext.	• General-purpose block-oriented transmission • Authentication
Cipher Feedback (CFB)	Input is processed s bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of	• General-purpose stream-oriented transmission • Authentication

	ciphertext.	
Output Feedback (OFB)	Similar to CFB, except that the input to the encryption algorithm is the preceding encryption output, and full blocks are used.	• Stream-oriented transmission over noisy channel (e.g. satellite communication)
Counter (CTR)	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	• General-purpose block-oriented transmission • Useful for high-speed requirements

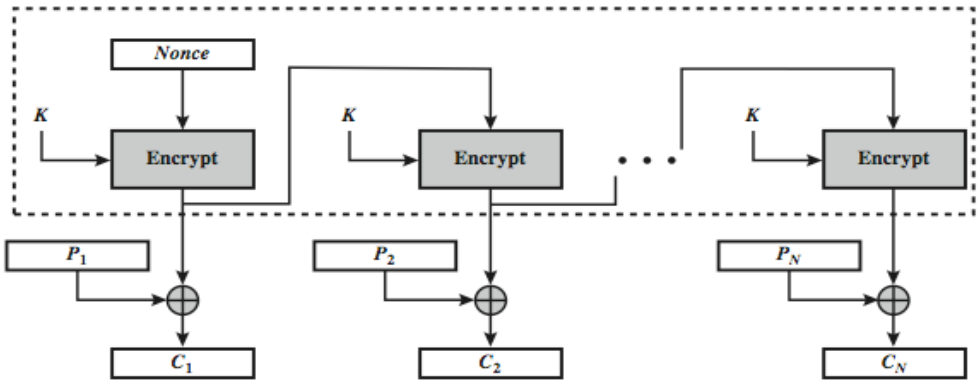
Look at the figures and equations of each one in the book.

Note: OFB and CTR use nonce -> because IV or Counter initial value must be unique for each execution of the encryption operation (be different for all of the messages encrypted using the same key.).

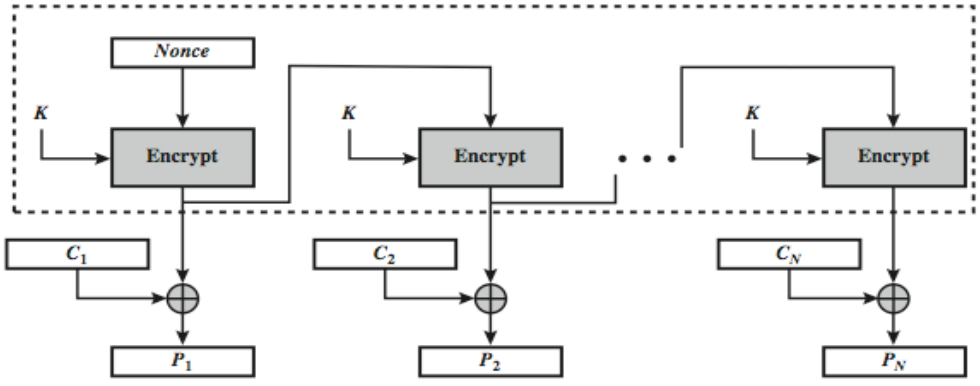
---

2. (Final 2010) Draw the block diagram of the OFB DES mode and illustrate it. Use symbolic forms in your illustration.

   Sol:



(a) Encryption



(b) Decryption

---

3. (Final 2010) Why is it not good to encrypt two plain texts using the same key in stream ciphers?

   Sol:

Don’t know
but I think because this may result in repetition of ciphertext parts (if the plaintext repeated) which will make it easy for cryptanalysis to know patterns.

Chapter 9

1. Draw the public-key cryptosystem that provides both authentication and secrecy. Illustrate it.

   Sol:

   Secrecy Only:

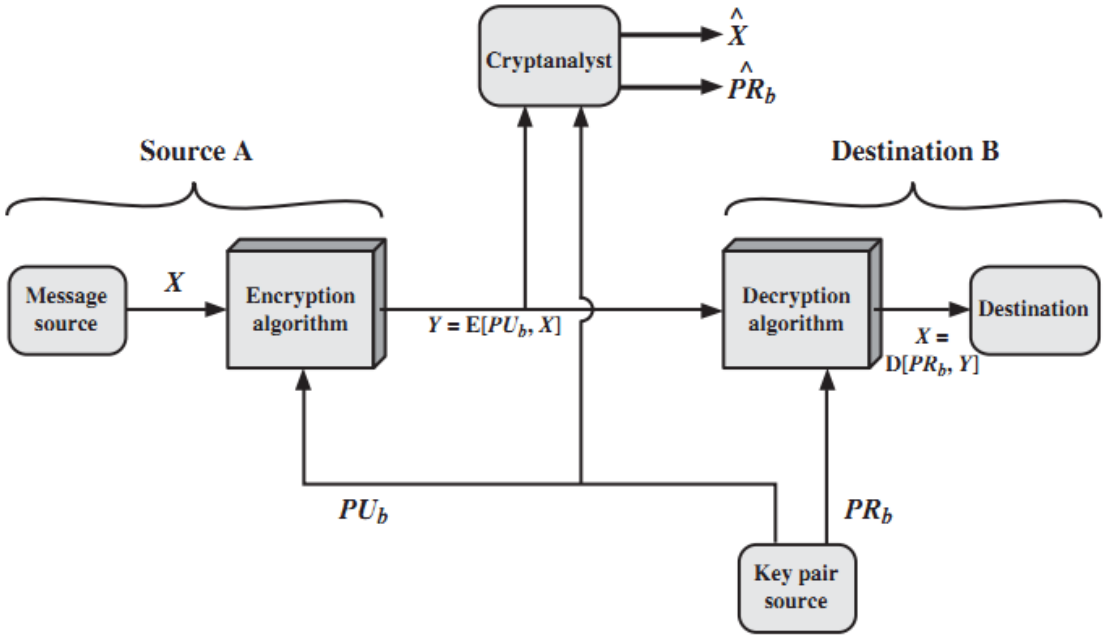


Figure 9.2 Public-Key Cryptosystem: Secrecy

Authentication Only:

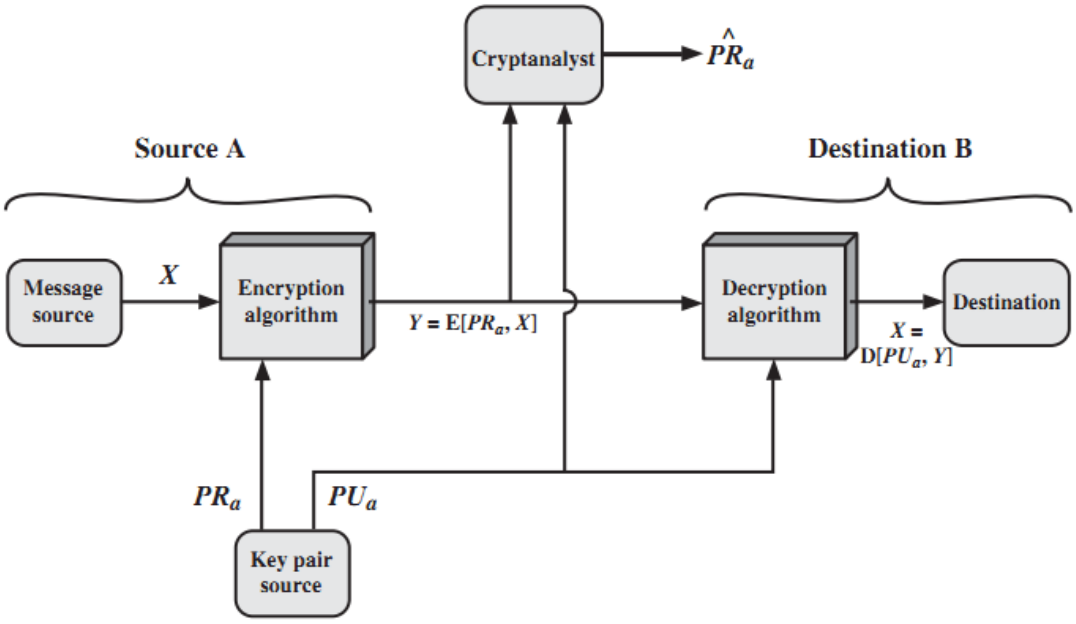


Figure 9.3 Public-Key Cryptosystem: Authentication

Both Authentication and Secrecy:

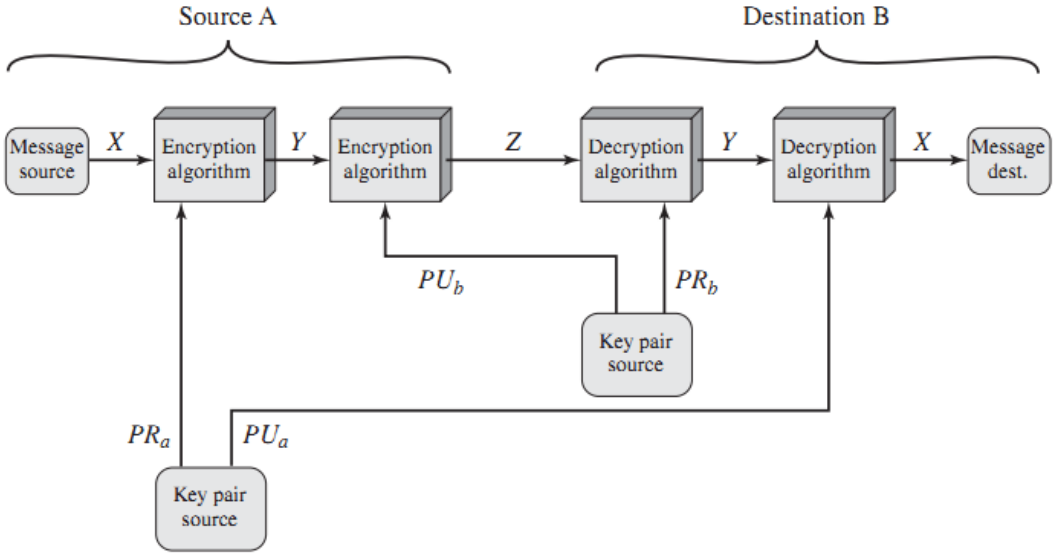


Figure 9.4 Public-Key Cryptosystem: Authentication and Secrecy

---

4. Compare between Convertional and Public-Key Encryption

Conventional Encryption	Public-Key Encryption
Needed to Work: 1- The same algorithm with the same key is used for encryption and decryption. 2- The sender and receiver must share the algorithm and the key.	Needed to Work: 1- One algorithm is used for encryption and decryption with a pair of keys, one for encryption and one for decryption. 2- The sender and receiver must each have one of the matched pair of

Needed for Security: 1. The key must be kept secret. 2. It must be impossible or at least impractical to decipher a message if no other information is available. 3. Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key.	keys (not the same one). Needed for Security: 1. One of the two keys must be kept secret. 2. It must be impossible or at least impractical to decipher a message if no other information is available. 3. Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be insufficient to determine the other key.

Chapter 10

1- (Final 2010) How Diffe-Hellman Works? Show the derivation. What is it used for?

Sol:

It's used for → enabling 2 users to secretly exchange a key that can be used for subsequent encryption of messages between them.

Algorithm:

- There are two publicly known numbers: a prime number **q** and an integer **α** that is a primitive root of q.
- Suppose the users A and B wish to exchange a key.
- User A selects a random integer $X_A < q$
  and computes $Y_A = \alpha^{XA} \bmod q$
- Similarly, user B independently selects a random integer $X_B < q$
  and computes $Y_B = \alpha^{XB} \bmod q$
- Each side keeps the **X** value private and makes the **Y** value available publicly to the other side.
- User A computes the key as
  $K = (Y_B)^{XA} \bmod q$
- and user B computes the key as
  $K = (Y_A)^{XB} \bmod q$
- These two calculations produce identical results (same key)

Derivation (That the two keys are equal):

$$
\begin{aligned}
K &= (Y_B)^{X_A} \bmod q \\
&= (\alpha^{X_B} \bmod q)^{X_A} \bmod q \\
&= (\alpha^{X_B})^{X_A} \bmod q && \text{by the rules of modular arithmetic} \\
&= \alpha^{X_B X_A} \bmod q \\
&= (\alpha^{X_A})^{X_B} \bmod q \\
&= (\alpha^{X_A} \bmod q)^{X_B} \bmod q \\
&= (Y_A)^{X_B} \bmod q
\end{aligned}
$$

Note: The adversary (الراجل الوحش :D) knows: q, α, Y_B, Y_B (Public), but doesn't know XA or XB (Private). To get them, he needs to take discrete logarithm to define the X (which will lead him to K), but this operation (discrete logarithm "dlog") is very difficult.

$$X_B = \text{dlog}_{\alpha,q}(Y_B)$$

---

2. (Final 2010) Two users are using Diffe-Hellman with alpha = 7 and q = 71. If Xa = 5, compute Ya. If Xb = 12, compute Yb. Compute the secret key. (this question is identical to problem 10.1 in the book)

Sol:

$Y_A = \alpha^{XA} \bmod q = 51$

$Y_B = \alpha^{XB} \bmod q = 4$

$K = (Y_B)^{XA} \bmod q = K = (Y_A)^{XB} \bmod q = 30$

Chapter 11

1- (Final 2010) What are the three ways to do authentication?

Sol:
Message authentication: is a mechanism or service used to verify the integrity of a message, by assuring that the data received are exactly as sent.
The three ways to do authentication:
1. Hash Function
2. Message Encryption
3. MAC (Message Authentication Code) -> also known as **keyed hash function**

---

2. (Final 2010) What are the requirements for a **strong** hash function? (I think this question is so important)

Sol:

In general the requirements for a cryptographic hash function H is as follows:

Table 11.1 Requirements for a Cryptographic Hash Function H

Requirement	Description
Variable input size	H can be applied to a block of data of any size.
Fixed output size	H produces a fixed-length output.
Efficiency	$H(x)$ is relatively easy to compute for any given x, making both hardware and software implementations practical.
Preimage resistant (one-way property)	For any given hash value h, it is computationally infeasible to find y such that $H(y) = h$.
Second preimage resistant (weak collision resistant)	For any given block x, it is computationally infeasible to find $y \neq x$ with $H(y) = H(x)$.
Collision resistant (strong collision resistant)	It is computationally infeasible to find any pair (x, y) such that $H(x) = H(y)$.
Pseudorandomness	Output of H meets standard tests for pseudorandomness.

A hash function that satisfies the first five properties in Table 11.1 is referred to as a **weak hash function**.
If the sixth property, **collision resistant**, is also satisfied, then it is referred to as a **strong hash function**.
A strong hash function protects against an attack in which one party generates a message for another party to sign.

Note: the 4th, 5th, and 6th properties are so important.
They're called the 3 Resistant Properties: (Note: x is the preimage of h if h = H(x))

1- **preimage resistant,** is the one-way property: it is easy to generate a code given a message, but virtually impossible to generate a message given a code.
2- **second preimage resistant,** guarantees that it is impossible to find an alternative message with the same hash value as a given message.
3- **collision resistant.**

The following figure shows the relationship between them.

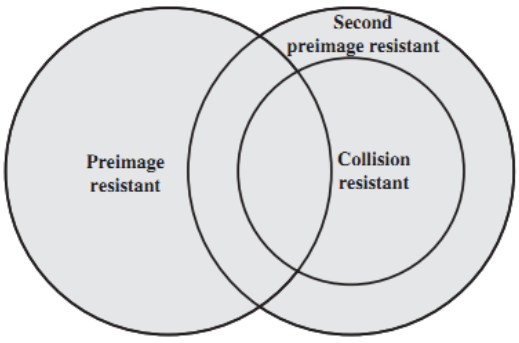


Figure 11.5 Relationship Among Hash Function Properties

---

3. (Final 2010) What is the difference between the weak collision resistance and the strong collision resistance?

Sol:

Comparison	weak collision resistance (second preimage resistant)	strong collision resistance (collision resistant)
Description	For any given block x, it is computationally infeasible to find $y \neq x$ with $H(y) = H(x)$. It is impossible to find an alternative message with the same hash value as a given message.	It is computationally infeasible to find any pair (x, y) such that $H(x) = H(y)$. It protects against an attack in which one party generates a message for another party to sign.
Example	If this property were not true, an attacker would be capable of the following sequence: First, observe or intercept a message plus its encrypted hash code; Second, generate an unencrypted hash code from the message; Third, generate an alternate message with the same hash code.	For example, suppose Bob writes an IOU message, sends it to Alice, and she signs it. Bob finds two messages with the same hash, one of which requires Alice to pay a small amount and one that requires a large payment.Alice signs the first message, and Bob is then able to claim that the second message is authentic.
Effect on Hash Function when satisfied	Weak hash function (not yet strong).	Strong hash function. If this property is satisfied, then also weak collision resistance is satisfied.

Chapter 14

1. (Final 2007) Compare Link Encryption to End-to-End Encryption, and which do you think is more secure and why?

Sol:

Comparison	Link Encryption	End-to-End
Definition	Each link encryption device is going to be exchanging data with only with its partner on the other end of the link.	It's in network or distributed system. Any given host or terminal may need to exchange data with many other hosts and terminals over time. Thus, each device need number of keys supplied dynamically.
Key Distribution	Physical delivery is simplest, because there is one communicating pair. The other methods are also applicable.	Physical Delivery is awkful, because there are many communicating pairs. The most possible ways: - If the two ends have an encryption connection with each other, they can send the new key encrypted with the old key. - is to use trusted third party that both ends have an encryption connection with.
Secure	More	less

Link Encryption is more secure than the End-to-End Encryption, because physical delivery of key, which is simple in Link Encryption, is the most secure key distribution way.

---

2. (Final 2010) Draw the public-key distribution scenario. Illustrate it.
   Sol:

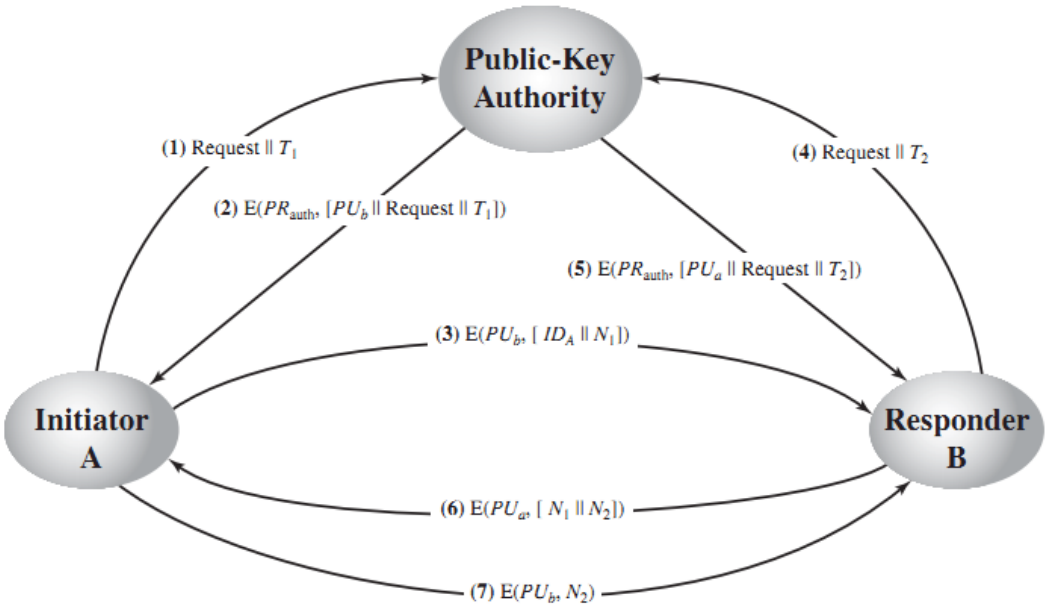


Figure 14.11 Public-Key Distribution Scenario

Illustration: Page in450 and 451 in the book (5th edition)

Public-Key Authority

Stronger security for public-key distribution can be achieved by providing tighter control over the distribution of public keys from the directory. A typical scenario is illustrated in Figure 14.11, which is based on a figure in [POPE79]. As before, the scenario assumes that a central authority maintains a dynamic directory of public keys of all participants. In addition, each participant reliably knows a public key for the authority, with only the authority knowing the corresponding private key. The following steps (matched by number to Figure 14.11) occur.

1. A sends a timestamped message to the public-key authority containing a request for the current public key of B.
2. The authority responds with a message that is encrypted using the authority's private key, PR_{auth}. Thus, A is able to decrypt the message using the authority's public key. Therefore, A is assured that the message originated with the authority. The message includes the following:
   - B's public key, PU_b, which A can use to encrypt messages destined for B
   - The original request used to enable A to match this response with the corresponding earlier request and to verify that the original request was not altered before reception by the authority

- The original timestamp given so A can determine that this is not an old message from the authority containing a key other than B's current public key

3. A stores B's public key and also uses it to encrypt a message to B containing an identifier of A (ID_A) and a nonce (N_1), which is used to identify this transaction uniquely.

4, 5. B retrieves A's public key from the authority in the same manner as A retrieved B's public key.

At this point, public keys have been securely delivered to A and B, and they may begin their protected exchange. However, two additional steps are desirable:

6. B sends a message to A encrypted with PU_a and containing A's nonce (N_1) as well as a new nonce generated by B (N_2). Because only B could have decrypted message (3), the presence of N_1 in message (6) assures A that the correspondent is B.

7. A returns N_2, which is encrypted using B's public key, to assure B that its correspondent is A.

Thus, a total of seven messages are required. However, the initial four messages need be used only infrequently because both A and B can save the other's public key for future use—a technique known as caching. Periodically, a user should request fresh copies of the public keys of its correspondents to ensure currency.