



BAIT1093 Introduction to Computer Security

Chapter 5: Elementary Cryptography

Topics

5.1 Basic Definitions

5.2 Cryptographic Algorithms

5.2.1 Block Ciphers vs Stream Ciphers

5.2.2 Hash Algorithms

- Message Digest (MD)
- Secure Hash Algorithm (SHA)
- Hashing Message Authentication Code (HMAC)

Topics

5.3 Symmetric Encryption Principles

5.3.1 Data Encryption Standard (DES)

5.3.2 Advanced Encryption Standard (AES)

5.3.3 Rivest Cipher (RC)

5.3.4 Location of Symmetric Encryption Devices

5.3.5 Key Distributions

Topics

5.4 Asymmetric Encryption or Public Key Cryptography

5.4.1 RSA Public-Key Encryption

5.4.2 Elliptic Curve Cryptography (ECC)

5.4.3 Digital Signature Algorithm (DSA)

5.4.4 Diffie-Hellman Key Exchange

5.5 New Cryptographic Standard

5.1 Basic Definitions [1]

- Cryptography is the practice of **transforming information** so that it **cannot be understood** by unauthorized parties and, thus, is secure.
- Cryptography is usually accomplished through **“scrambling” the information** so that only approved recipients (either human or machine) can understand it.

5.1 Basic Definitions [1]

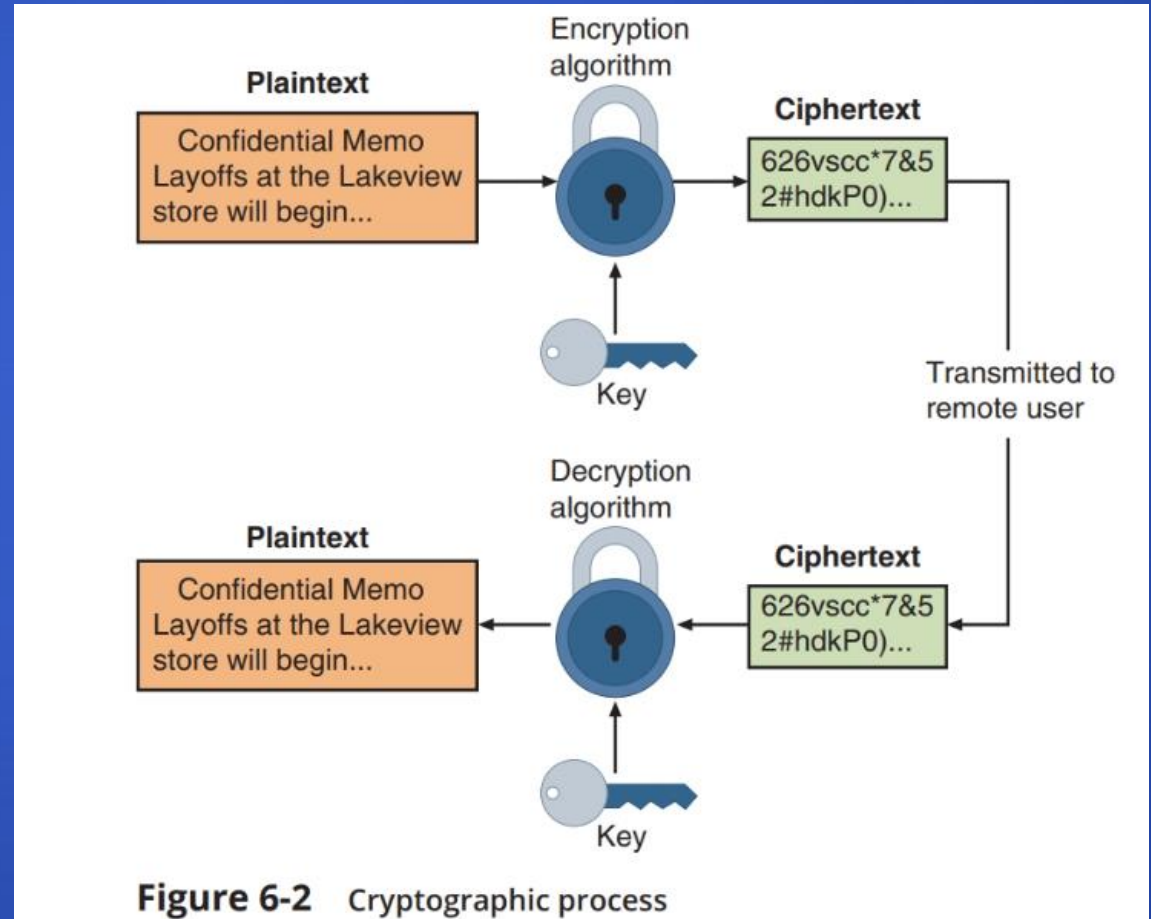
- When using cryptography, the process of changing the original text into a scrambled message is known as **encryption**.
- Changing the message back to its original form is known as **decryption**.

5.1 Basic Definitions [1]

- In addition, the following terminology applies to cryptography:
 - **Plaintext.** Unencrypted data that is input for encryption or is the output of decryption.
 - **Ciphertext.** The scrambled and unreadable output of encryption.
 - **Cleartext.** Unencrypted data that is not intended to be encrypted.
 - **Cipher:** A cryptographic algorithm to encrypt plaintext data (input) using procedures based on a mathematical formula. A key is a mathematical value entered into the algorithm to produce the ciphertext.

5.1 Basic Definitions [1]

- Cryptographic algorithms are public and well known; however, the individualized key for the algorithm that a user possesses must at all costs be kept secret.



5.1 Basic Definitions [4]

Key elements that go into making an effective cryptosystem:

1. Must be **reversible**: A crypto algorithm is of no practical use if once you have scrambled your information, you cannot unscramble it.
2. **Secrecy and length of the key**: The security of the cryptosystem should be dependent on the secrecy and length of the key, **not on the details of the algorithm**. In other words, knowing the algorithm should not make it significantly easier to crack the code (restricted versus unrestricted). If security is dependent on keeping the algorithm secret, then it is considered a restricted algorithm.

5.1 Basic Definitions [4]

Key elements that go into making an effective cryptosystem (cont.):

3. Subjected to **substantial cryptanalysis**: Only those algorithms that have been analyzed completely and at length are trustworthy. The algorithm should contain no serious or exploitable weakness. Theoretically, all algorithms can be broken by one method or another. However, an algorithm should not contain an inherent weakness that an attacker can easily exploit.

5.1 Basic Definitions [4]

Cryptanalysis

- Technically, any method employed to break a cipher or code. However, cryptanalysis here specifically refers to employing mathematical analysis to break a code.
- This method requires a high level of skill and sophistication. It is usually only employed by academics and governments.
- Today, it relies very heavily on the use of ultrafast super computers. Probably the most active and successful organization in the world dedicated to breaking codes is the National Security Agency (NSA). This is the largest spy agency in the United States with tens of thousands of employees.
- It is sometimes referred to as the Puzzle Palace, because the group spends so much time and energy on codes and cipher.

5.1 Basic Definitions [1]

- Ciphers can be separated into several categories:
 - **Substitution cipher**, which exchanges one character for another.
 - **XOR cipher**, which is based on the binary operation eXclusive OR to compare two bits: if the bits are different, a 1 is returned, but if they are identical, then a 0 is returned.

5.1 Basic Definitions [2]

Cipher 1) Substitution cipher

- Caesar Cipher – one of the oldest encryption methods. Choose some number by which to shift each letter of a text.

In this cipher technique, we replace each letter of the alphabet with a letter that is standing three places further:

- In the generalized version of Caesar's cipher each alphabet is replaced with another alphabet after shifting it x times to the right.
- The amount of the **shift (x)** is the encryption key.
- For the decryption process, we reverse the process and replace the ciphertext alphabet with the alphabet after doing a **left shift by x alphabets**.
- The numerical equivalent to each letter is shown below:

A	B	C	D	E	F	G	H	I	J	K	L	M
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
0	1	2	3	4	5	6	7	8	9	10	11	12
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
13	14	15	16	17	18	19	20	21	22	23	24	25

Source

<https://www.educative.io/answers/encryption-using-caesars-cipher>

5.1 Basic Definitions [2]

Cipher [?] 1)

Substitution cipher

- Caesar Cipher

Source

<https://www.educative.io/answers/encryption-using-caesars-cipher>

Note:

- Encryption: $(PT + Key) \bmod 26$
- Decryption: $(CT - Key) \bmod 26$

Let's try to solve Caesar cipher with an example, assuming The given PT=CRYPTOLOGY and Key=3:

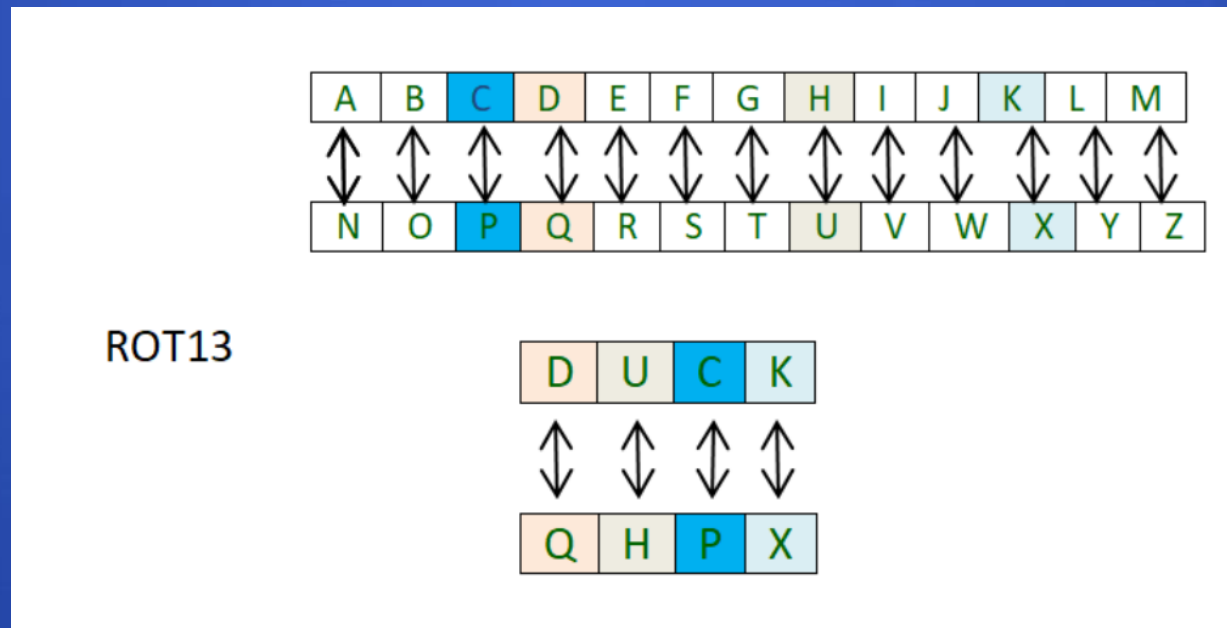
- $C1 = (2+3) \bmod 26 = 5$ that is **F**
- $C2 = (17+3) \bmod 26 = 20$ that is **U**
- $C3 = (24+3) \bmod 26 = 1$ that is **B**
- $C4 = (15+3) \bmod 26 = 18$ that is **S**
- $C5 = (19+3) \bmod 26 = 22$ that is **W**
- $C6 = (14+3) \bmod 26 = 17$ that is **R**
- $C7 = (11+3) \bmod 26 = 14$ that is **O**
- $C8 = (14+3) \bmod 26 = 17$ that is **R**
- $C9 = (6+3) \bmod 26 = 9$ that is **J**
- $C10 = (24+3) \bmod 26 = 1$ that is **B**

The ciphertext for **CRYPTOLOGY** is **FUBSWRORJB**.

5.1 Basic Definitions [1]

Cipher 1) Substitution cipher

- ROT13, in which the entire alphabet is rotated 13 steps (A = N, B = O, etc.) so that the word *security* becomes *frphevgl*.



5.1 Basic Definitions [4]

Cipher → 1) Substitution cipher

- This rudimentary cipher would not be effective at keeping a message secret for long.
- It does not comply with one of the qualities of a truly effective cipher, where knowing the algorithm should not make it significantly easier to crack the code. If it is known that the given ciphertext is Caesar cipher, then a brute-force cryptanalysis is easily performed since there are only 25 combinations to search out for given plaintext.
- This is an example of a restricted algorithm.
- In this case, the cipher is the code.
- Once you know the cipher, you can unscramble any message.

5.1 Basic Definitions [1]

Cipher → 2) XOR Cipher

XOR Cipher Example

Plaintext	s
Decimal	115
Binary	01110011
	XOR
Key	f
Decimal	102
Binary	01100110
Ciphertext	00010101

5.1 Basic Definitions [1]

Cipher → 2) XOR Cipher

XOR Cipher Example

Plaintext	s	e	c	u	r	i	t	y
Decimal	115	101	99	117	114	105	116	121
Binary	01110011	01100101	01100011	01110101	01110010	01101001	01110100	01111001
	XOR							
Key	f	l	a	p	j	a	c	k
Decimal	102	108	97	112	106	97	99	107
Binary	01100110	01101100	01100001	01110000	01101010	01100001	01100011	01101011
Ciphertext	00010101	00001001	00000010	00000101	00011000	00001000	00010111	00010010

5.1 Basic Definitions [1]

Cryptography Use Cases

- Several common use cases (situations) for cryptography can provide a range of security protections:
 - Confidentiality
 - Integrity
 - Authentication
 - Nonrepudiation
 - Obfuscation

5.1 Basic Definitions [1]

Cryptography Use Cases → 1) Confidentiality

- Cryptography can protect the confidentiality of information by ensuring that only authorized parties can view it.
- When private information, such as a list of employees to be laid off, is transmitted across the network or stored on a file server, its contents can be encrypted, which allows only authorized individuals who have the key to read it.

5.1 Basic Definitions [1]

Cryptography Use Cases → 2) Integrity

- Cryptography can protect the integrity of information. Integrity ensures that the information is correct and no unauthorized person or malicious software has altered that data. Because ciphertext requires that a key must be used to open the data before it can be changed, cryptography can ensure its integrity.
- The list of employees to be laid off, for example, can be protected so that no names can be added or deleted by unauthorized personnel.

5.1 Basic Definitions [1]

Cryptography Use Cases → 3) Authentication

- The authentication of the sender can be verified through cryptography.
- Specific types of cryptography, for example, can prevent a situation such as circulation of a list of employees to be laid off that appears to come from a manager but, in reality, was sent by an imposter.

5.1 Basic Definitions [1]

Cryptography Use Cases → 4) Nonrepudiation

- Cryptography can enforce nonrepudiation. Repudiation is defined as denial; nonrepudiation is the inability to deny. In information technology, nonrepudiation is the process of proving that a user performed an action, such as sending an email message.
- The nonrepudiation features of cryptography can prevent managers from claiming they never sent lists of employees to be laid off to an unauthorized third party.

5.1 Basic Definitions [1]

Cryptography Use Cases → 5) Obfuscation

- Obfuscation is making something obscure or unclear.
- Cryptography can provide a degree of obfuscation by encrypting a list of employees to be laid off so that an unauthorized user cannot read it.

5.1 Basic Definitions [1]

Table 6-1 Information protections by cryptography

Characteristic	Description	Protection
Confidentiality	Ensures that only authorized parties can view the information	Encrypted information can only be viewed by those who have been provided the key.
Integrity	Ensures that the information is correct and no unauthorized person or malicious software has altered that data	Encrypted information cannot be changed except by authorized users who have the key.
Authentication	Provides proof of the genuineness of the user	Proof that the sender was legitimate and not an imposter can be obtained.
Nonrepudiation	Proves that a user performed an action	Individuals are prevented from fraudulently denying that they were involved in a transaction.
Obfuscation	Makes something obscure or unclear	By being made obscure, the original information cannot be determined.

5.1 Basic Definitions [1]

- Cryptography can provide protection to data as that data resides in any of three states:
 - **Data in processing.** Also called data in use is data on which actions are being performed by devices, such as printing a report from a device.
 - **Data in transit.** Actions that transmit the data across a network, such as an email sent across the Internet. Sometimes called data in motion.
 - **Data at rest.** Data stored on electronic media.

5.2 Cryptographic Algorithms

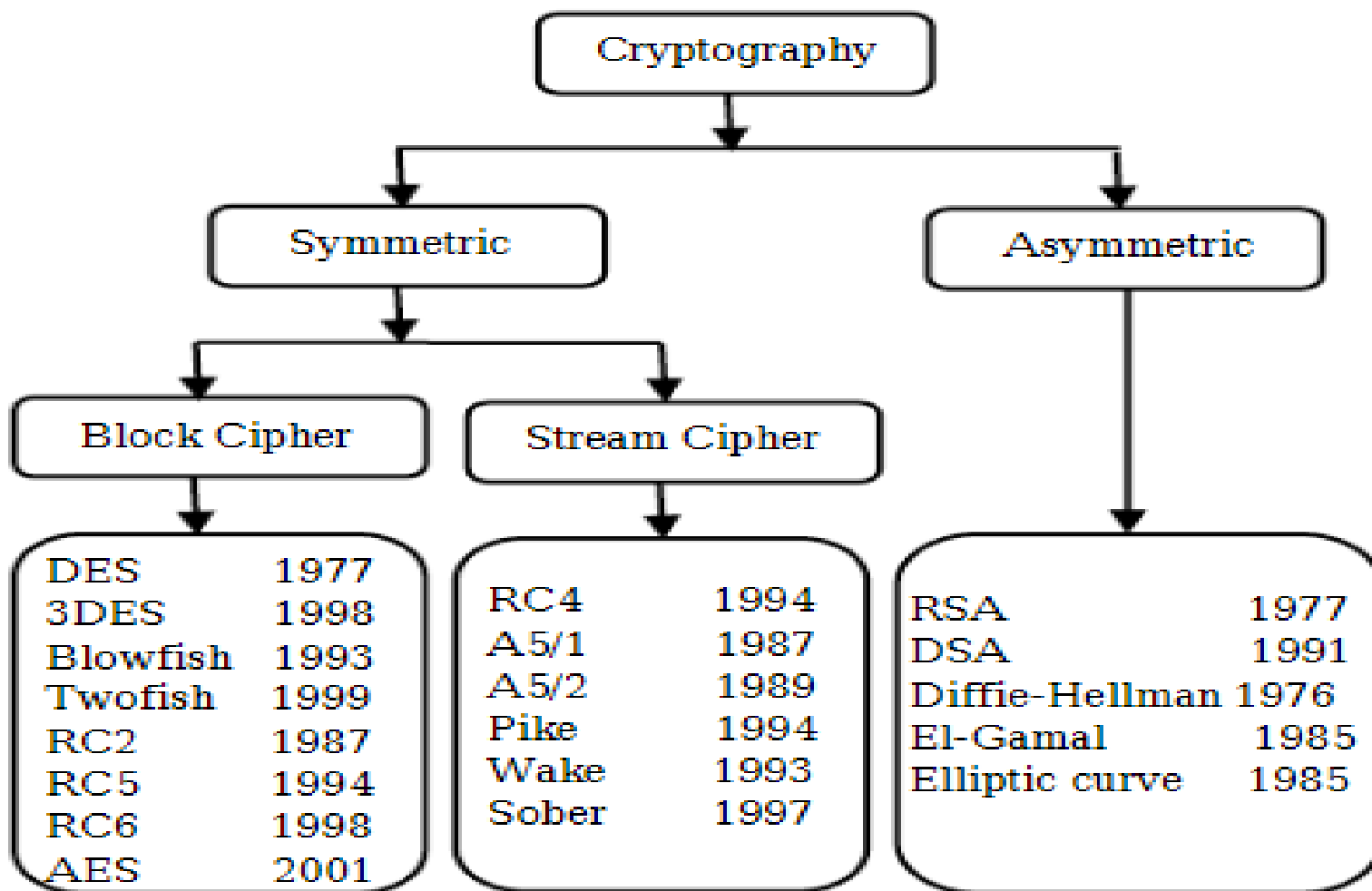


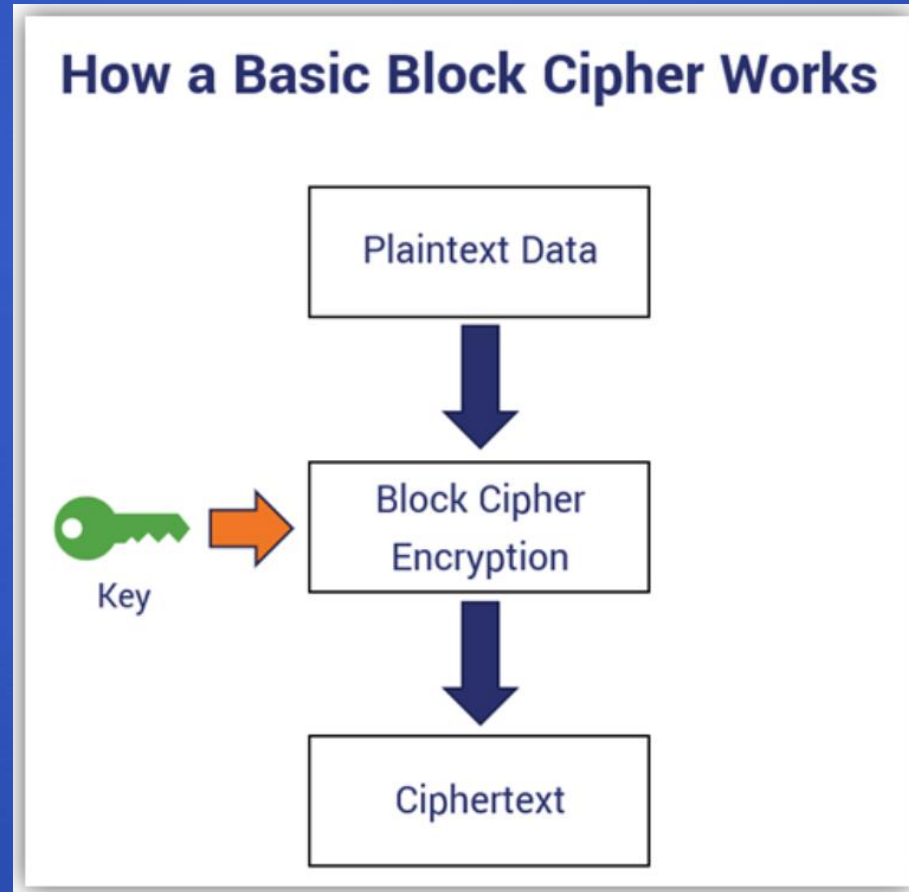
Figure 2: Cryptographic Algorithms Classification

5.2.1 Block Ciphers vs Stream Ciphers [2]

- **Block Cipher:**
 - Convert data in plaintext into ciphertext in fixed-size blocks.
 - A key is applied to blocks (64 bits or 128 bits) at a time.
 - If the plaintext length is not a multiple of 8, the encryption scheme uses padding to ensure complete blocks.
 - For instance, to perform 128-bit encryption on a 150-bit plaintext, the encryption scheme provides two blocks: 1 with 128 bits and one with 22 bits + 106 bits (padding).

5.2.1 Block Ciphers vs Stream Ciphers [2]

- **Block Cipher:**



5.2.1 Block Ciphers vs Stream Ciphers [2]

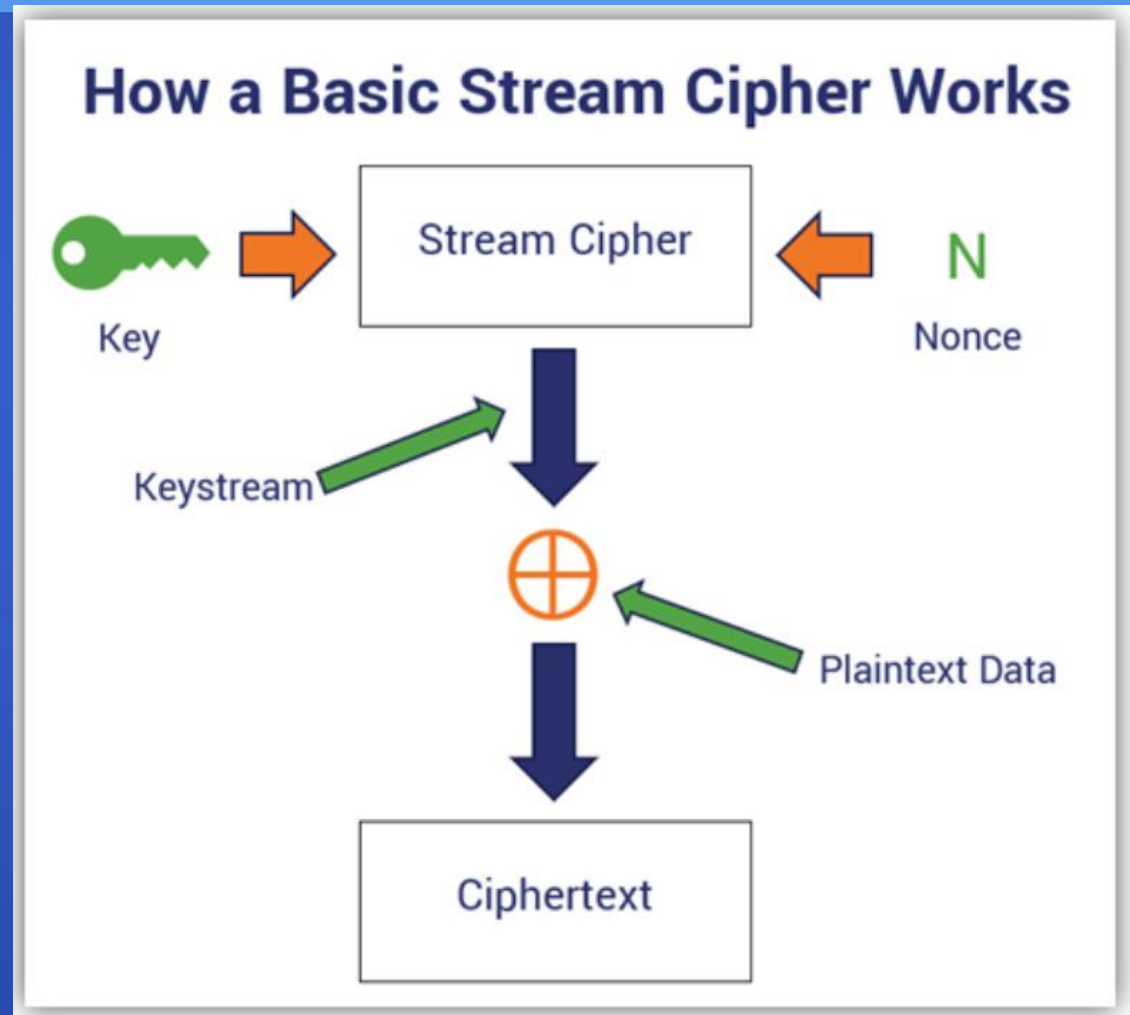
- **Stream Cipher:**
 - Encrypts a continuous string of binary digits by applying time-varying transformations on plaintext data
 - Encryption is done 1 bit at a time using keystreams to generate ciphertext for arbitrary lengths of plaintext messages.

5.2.1 Block Ciphers vs Stream Ciphers [2]

- **Stream Cipher:**
 - The cipher combines a key (128/256 bits) and a nonce (number used only once) digit (64-128 bits) to produce the keystream (a pseudorandom number) XORed with the plaintext to produce ciphertext.

5.2.1 Block Ciphers vs Stream Ciphers [2]

- Stream Cipher:



Source: <https://www.thesslstore.com/blog/block-cipher-vs-stream-cipher/>

5.2.2 Hash Algorithms [1]

- One type of cryptographic algorithm is a one-way hash algorithm.
- A hash algorithm creates a unique “digital fingerprint” of a set of data.
- This process is called hashing, and the resulting fingerprint is a digest (sometimes called a message digest or hash) that represents the contents.
- Hashing is used primarily for comparison purposes.

5.2.2 Hash Algorithms [1]

- Although hashing is a cryptographic algorithm, its purpose is not to create ciphertext that can later be decrypted.
- Hashing is intended to be one-way in that its digest cannot be reversed to reveal the original set of data.
- For example, when 12 is multiplied by 34, the result is 408. If a user were asked to determine the two numbers used to create the number 408, it would not be possible to work backward and derive the original numbers with absolute certainty because there are too many mathematical possibilities (1 x 408, 2 x 204, 3 x 136, 4 x 102, etc.). Hashing is similar in that it is not possible to determine the plaintext from the digest.

5.2.2 Hash Algorithms [1]

- A hashing algorithm is considered secure if it has the following characteristics:
 - **Fixed size.** A digest of a short set of data should produce the same size as a digest of a long set of data. For example, a digest of the single letter *a* is *86be7afa339d0fc7cfc785e72f578d33*, while a digest of **one million** occurrences of the letter *a* is *4a7f5723f954eba1216c9d8f6320431f*, the same length.

5.2.2 Hash Algorithms [1]

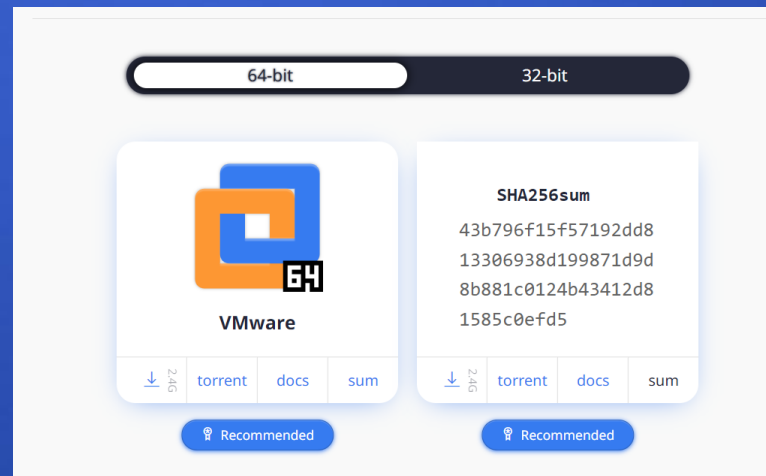
- A hashing algorithm is considered secure if it has the following characteristics (cont.):
 - **Unique.** Two different sets of data cannot produce the same digest. Changing a single letter in one data set should produce an entirely different digest. For example, a digest of *Sunday* is **0d716e73a2a7910bd4ae63407056d79b** while a digest of *sunday* (lowercase s) is **3464eb71bd7a4377967a30da798a1b54**.

5.2.2 Hash Algorithms [1]

- A hashing algorithm is considered secure if it has the following characteristics (cont.):
 - **Original.** It should not be possible to produce a data set that has a desired or predefined hash.
 - **Secure.** The resulting hash cannot be reversed to determine the original plaintext.

5.2.2 Hash Algorithms [1]

- Hashing is often used as a check to verify that the original contents of an item have not been changed.
- After downloading any file from the Internet, users can create their own digest on the file and then compare it with the digest value posted on the website. A match indicates the original file did not change while it was being downloaded.



5.2.2 Hash Algorithms [1]

- Common Hash Algorithms
 - **Message Digest (MD)**
 - One of the earliest hash algorithms is a “family” of algorithms known as Message Digest (MD). Versions of MD hashes were introduced over almost 20 years, from MD2 (1989) to MD6 (2008).
 - The most widely used of these algorithms is MD5.
 - Mainly used to check the integrity of files or to encode smaller strings like passwords, credit card numbers or sensitive data in the database.

5.2.2 Hash Algorithms

- Common Hash Algorithms
 - **Message Digest (MD)**
 - MD5 is expressed as a 32 digit hexadecimal number.
 - Different messages or file content will have different message digests.

```
MD5(There is $1500 in the blue bo)    = f80b3fde8ecbac1b515960b9058de7a1
MD5(There is $1500 in the blue box)    = a4a5471a0e019a4a502134d38fb64729
MD5(There is $1500 in the blue box.) = 05f8cfc03f4e58cbee731aa4a14b3f03
MD5(There is $1500 in the blue box!) = 4b36807076169572b804907735accd42
MD5(There is $1500 in the blue box..)= 3a7b4e07ae316eb60b5af4a1a2345931
```

Source:

<http://etutorials.org/Linux+systems/unix+internet+security/Part+II+Security+Building+Blocks/Chapter+7.+Cryptography+Basics/7.4+Message+Digest+Functions/>

5.2.2 Hash Algorithms [1]

- Common Hash Algorithms
 - **Message Digest (MD)**
 - Serious weaknesses have been identified in MD5, and it is no longer considered suitable for use.
 - MD5 faces a problem called collision where there is a finite chance that two files will have the same MD5 code, although the chances are slim.
 - For a message digest function to be secure, it should be computationally infeasible to find or produce these collisions.

Source:

<http://etutorials.org/Linux+systems/unix+internet+security/Part+II+Security+Building+Blocks/Chapter+7.+Cryptography+Basics/7.4+Message+Digest+Functions/>

5.2.2 Hash Algorithms [1]

- Common Hash Algorithms
 - **Secure Hash Algorithm (SHA)**
 - SHA-1 was developed in 1993 but is no longer considered suitable for use.
 - SHA-2 has six variations, the most common are SHA-256, SHA-384, and SHA-512 (the last number indicates the length in bits of the digest that is generated) and is currently considered a secure hash.
 - In 2015, after eight years of competition between 51 original entries, SHA-3 was announced as a new standard.
 - One design goal of SHA-3 was to make it dissimilar to previous hash algorithms to prevent threat actors from building on earlier work of compromising the algorithms.

5.2.2 Hash Algorithms [1]

Table 6-4 illustrates the digests generated from several one-way hash algorithms using the word *Cengage*.

Table 6-4 Digests generated from one-time hash algorithms

Hash	Digest
MD2	b365b3f6ca8b35460782a658f2e82009
MD4	4fe043f7a0cde169a5d069b46bcfc0f7
MD5	7e169c6f44088e315c7b1f6513c1b0f7
SHA-1	963ea98f0af1927e02ed0f13786162a5b8e713c0
SHA-256	f6c8a86bf6a5128cbaf2ad251b0beaa3604c11c51587de518737537800098d76
SHA-512	a2221473f8c4737d97f44265b3731a61127cc9521d4e07d18b1be357df4f04f8367a4a5255 cae956611f71b426bf494f2a68f41ca4aa122c2a07b570880e81fd
SHA3-512	3a82d58e17f3991413c5f4e9811930b69513bba02a860eed82070f892ab381f9fd926a88cf68745 565f51a93b97a1317ae8b84e2dfb798e4a2aa331187dc9e34

5.2.2 Hash Algorithms [2]

Hashing Message Authentication Code (HMAC)

- There is a possibility for a message to be altered intentionally during transmission, deletes the original hash, change the message and recomputes a new one. A simple hashing algorithm cannot account for this scenario.
- Using Message Authentication Code (MAC) is one way of detecting intentional alterations in a message. A MAC is often called a keyed cryptographic hash function [?]

Hashing Message Authentication Code (HMAC)

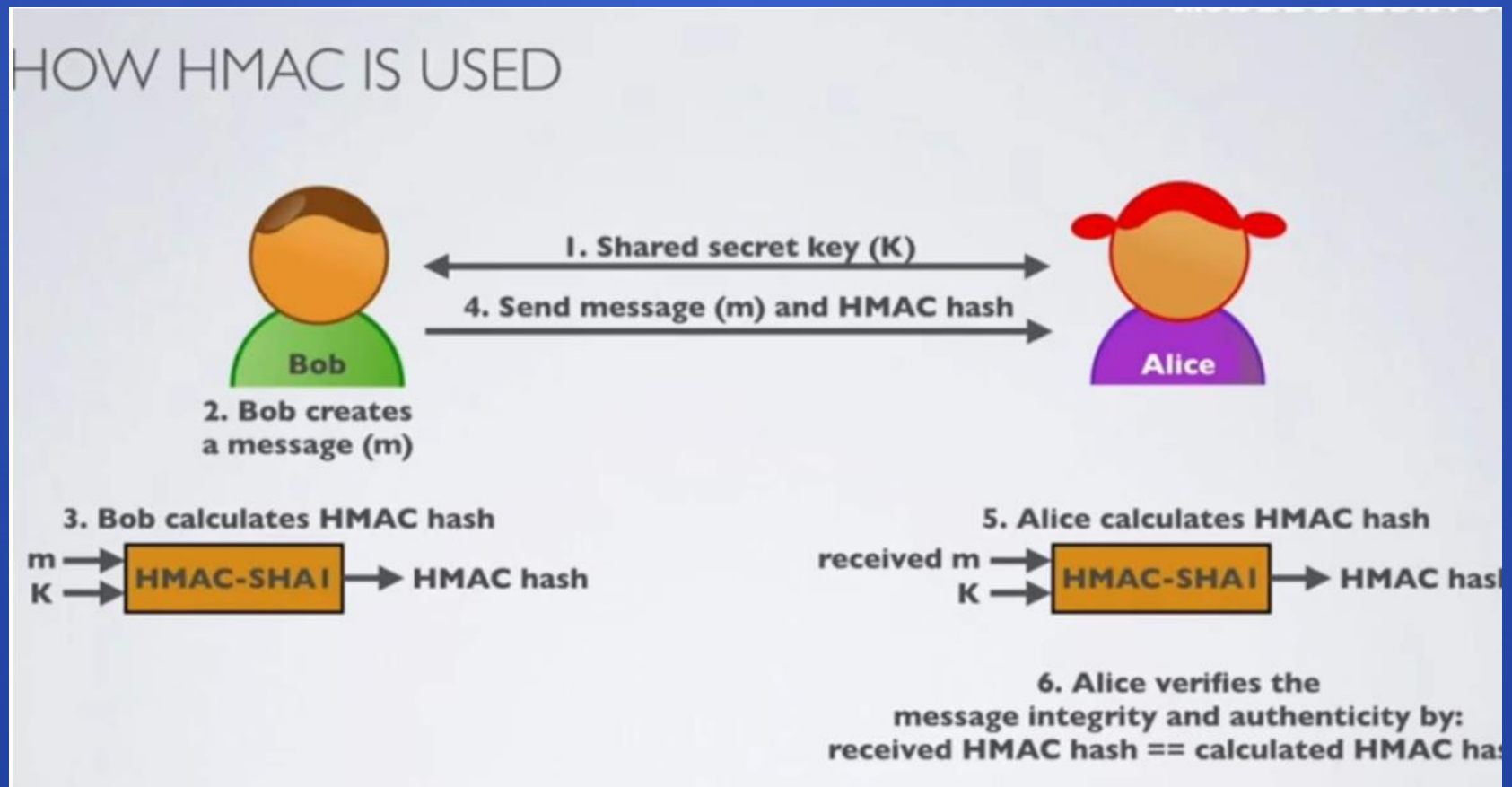
5.2.2 Hash Algorithms [2]

Hashing Message Authentication Code (HMAC)

- Assuming SHA-1 is used to verify message integrity. Both sender and recipient pre-share or exchange a key of the same size (in this case, 128 bits). In other words, 128-bit key.
- The sender will hash the message and then XOR that hash with the 128-bit key.
- The recipient will hash what she receives and XOR that computed hash with the 128-bit key.
- Any intercepting party would not have the 128-bit key to XOR that with, thus the hash the interceptor creates will not match with the hash that the recipient computes, and discrepancies will be detected.

5.2.2 Hash Algorithms [2]

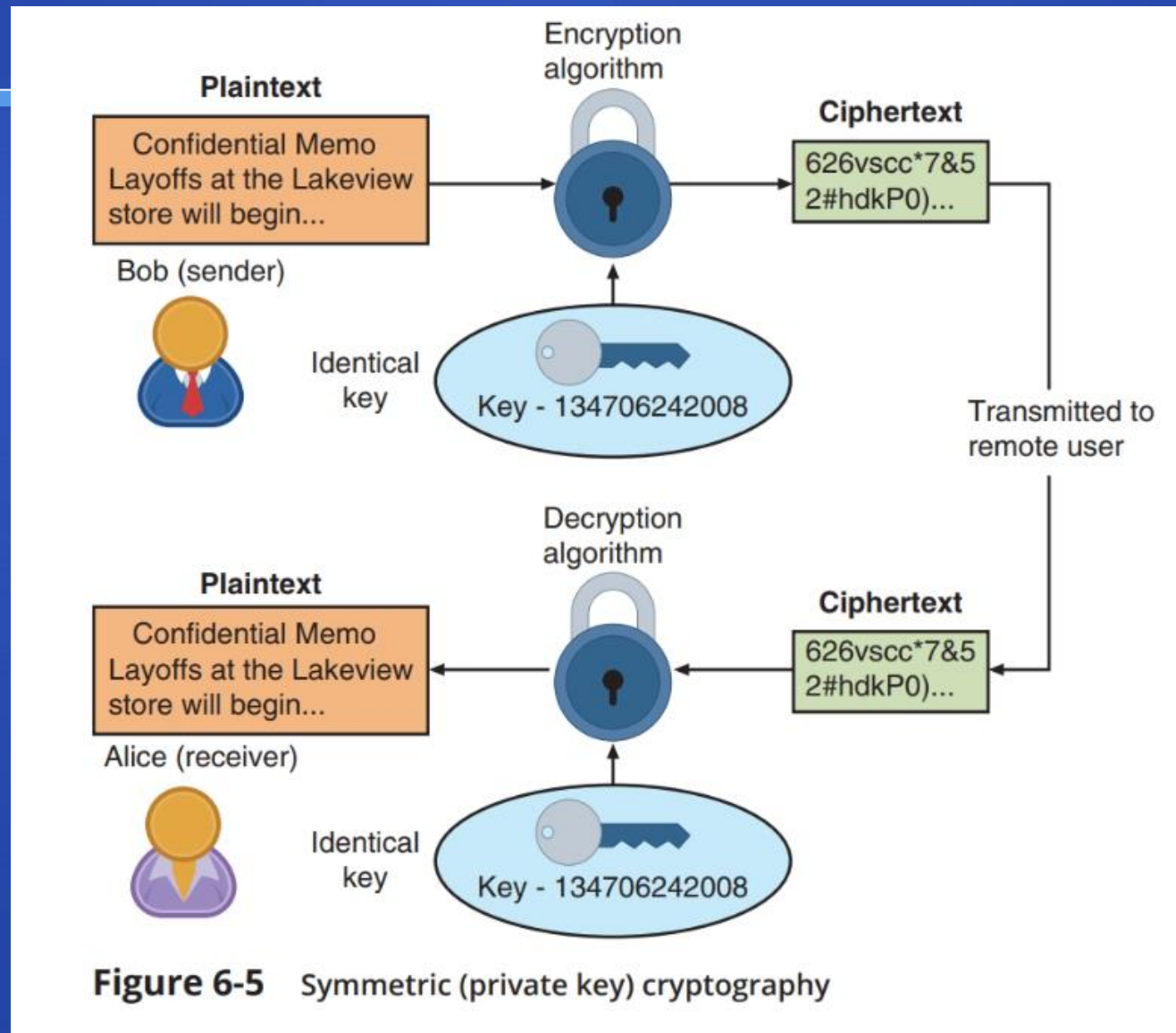
Hashing Message Authentication Code (HMAC)



5.3 Symmetric Encryption Principles [1] [4]

- Symmetric cryptographic algorithms use the **same key** to encrypt and decrypt the data.
- Example, data encrypted by Bob with a key can only be decrypted by Alice using that same key. Because the **key** must be kept **private** (confidential), symmetric encryption is also called **private key cryptography**.
- The **strength of the scheme** is largely dependent on the **size of the key** and on keeping it **secret**. Generally, the larger the key, the more secure the scheme.
- In addition, symmetric key encryption is relatively **fast**.

5.3 Symmetric Encryption Principles [1]



5.3 Symmetric Encryption Principles [4]

- The main **weakness** of the system is that:
 - the key or algorithm has to be shared. Sharing the key information over an unsecured network without compromising the key is not feasible. As a result, private key cryptosystems **are not well suited for spontaneous communication** over open and unsecured networks.
 - symmetric key provides **no process for authentication or non-repudiation**.

5.3 Symmetric Encryption Principles [4]

Table 6.1 The Advantages and Disadvantages of Symmetric Key Cryptography.

Advantages	Disadvantages
Fast Relatively secure Widely understood	Requires secret sharing Complex administration No authentication No non-repudiation

5.3.1 Data Encryption Standard (DES) [1] [2]

- One of the first widely used symmetric cryptography algorithms was the Data Encryption Standard (DES). Developed by IBM in the early 1970s.
- The U.S. government officially adopted DES as the de-facto international standard for encrypting information in 1977.
- DES is a block cipher, which divides the plain text into 64-bit blocks and encrypts each block.
- Although DES was once widely implemented, it is no longer considered suitable for use.
- The small key size, 56 bits (plus 8 parity bits), is not good enough to defend against brute-force attacks with modern computers.

5.3.1 Data Encryption Standard (DES) [1] [2]

- Thus Triple-DES (3DES) was introduced in 1998 to replace DES.
- 3DES uses three rounds of encryption instead of just one.
- The ciphertext of one round becomes the entire input for the second iteration.

Source: <https://www.cryptomathic.com/news-events/blog/symmetric-encryption-algorithms-their-strengths-and-weaknesses-and-the-need-for-crypto-agility>

5.3.1 Data Encryption Standard (DES) [1]

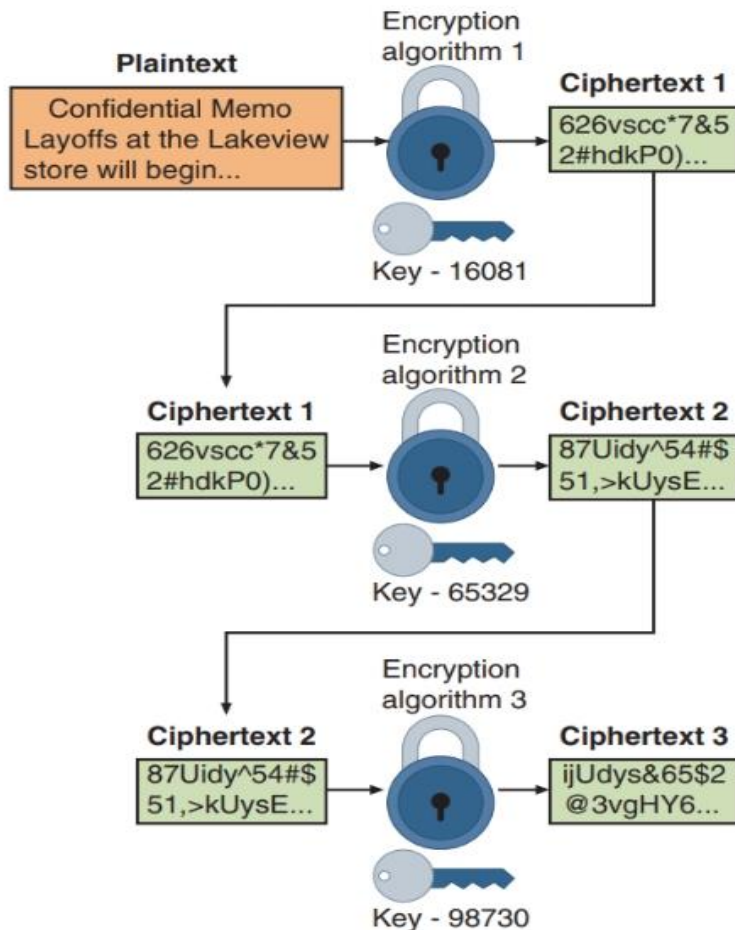


Figure 6-6 3DES

The most secure versions of 3DES use different keys for each round, however problem is slow performance.

Although 3DES addresses several of the key weaknesses of DES, it is no longer considered the most secure symmetric cryptographic algorithm.

5.3.2 Advanced Encryption Standard (AES) [2]

- The Advanced Encryption Standard (AES) is a symmetric algorithm that eventually chosen to replace DES.
- The U.S. government selected AES to be the replacement of DES, and it is now the most widely used symmetric key algorithm.
- It is a block cipher that works on 128-bit blocks. The block size can also be 192 or 256 bit. However, U.S. government uses 128-bit block.
- It can have one of the three key sizes: 128 (AES 128), 192 (AES 192), or 256 (AES 256) bits.

5.3.2 Advanced Encryption Standard (AES)

- AES is the symmetric algorithm-of-choice for most applications today and is very widely used, mostly with 128 or 256-bit keys, with the latter key length even considered strong enough to protect military TOP SECRET data.
- Note that, assuming there are no known weaknesses in an algorithm, a single 128-bit key will take billions of years to brute force using any classical computing technology today in or even possibly with quantum computing.

Source: <https://www.cryptomathic.com/news-events/blog/symmetric-encryption-algorithms-their-strengths-and-weaknesses-and-the-need-for-crypto-agility>

5.3.3 Rivest Cipher (RC) [1]

- Rivest Cipher (RC) is a family of six algorithms.
- RC4, the most common RC cipher and widely used (e.g. in the SSL/TLS protocol and early Wi-Fi security standards), is a stream cipher that accepts keys up to 128 bits in length.
- RC4 are considered not secure today.
- RC5 is a block cipher with a variable block size (32, 64 or 128 bits), variable key length (up to 2,040 bits) and variable number of rounds (up to 255).

5.3.3 Rivest Cipher (RC) [1]

- This enables a trade-off between performance and security, and it is still considered secure when used with suitable parameters. It was later modified to produce RC6 with a fixed block size of 128 bits
- However, RC5 and RC6 are not widely used as they are patented.

Source: <https://www.cryptomathic.com/news-events/blog/symmetric-encryption-algorithms-their-strengths-and-weaknesses-and-the-need-for-crypto-agility>

5.3.4 Location of Symmetric Encryption Devices [3]

- In using encryption, we need to decide what to encrypt and where the encryption gear should be located.
- There are two fundamental alternatives:
 - link encryption
 - end-to-end encryption

5.3.4 Location of Symmetric Encryption Devices [3]

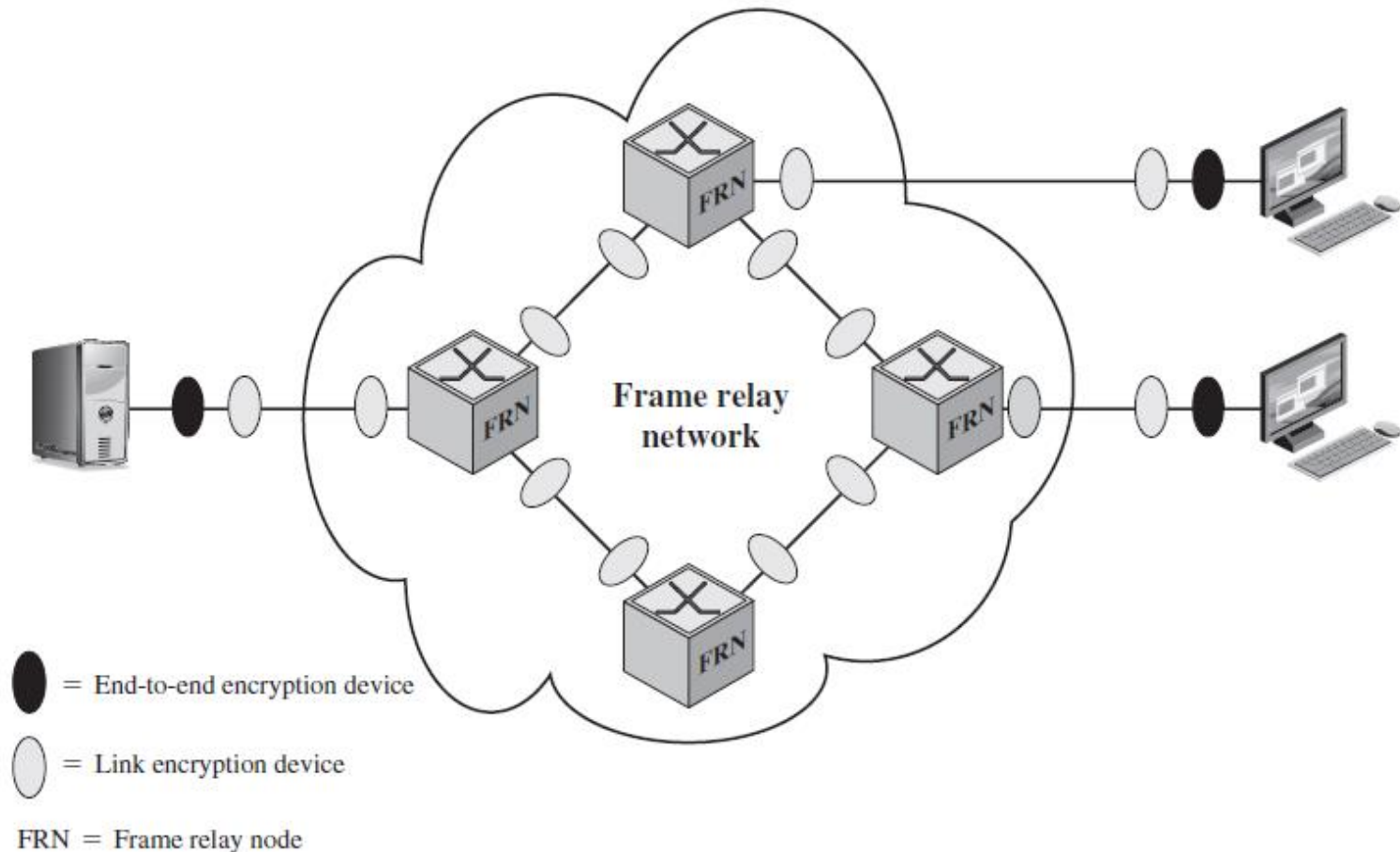


Figure 20.9 Encryption Across a Frame Relay Network

5.3.4 Location of Symmetric Encryption Devices [3]

Link Encryption

- Each vulnerable communications link is equipped on both ends with an encryption device.
- Thus, all traffic over all communications links is secured.
- Although this requires a lot of encryption devices in a large network, it provides a high level of security.
- One disadvantage of this approach is that the message must be decrypted each time it enters a frame switch; this is necessary because the switch must read the address (connection identifier) in the frame header to route the frame. Thus, the message is vulnerable at each switch. If this is a public frame-relay network, the user has no control over the security of the nodes.

5.3.4 Location of Symmetric Encryption Devices [3]

End-to-End Encryption

- With end-to-end encryption, the encryption process is carried out at the two end systems.
- The source host or terminal encrypts the data. The data, in encrypted form, are then transmitted unaltered across the network to the destination terminal or host.
- The destination shares a key with the source and so is able to decrypt the data.

5.3.4 Location of Symmetric Encryption Devices [3]

- To achieve greater security, both link and end-to-end encryption are needed. The host encrypts the user data portion of a frame using an end-to-end encryption key. The entire frame is then encrypted using a link encryption key.
- As the frame traverses the network, each switch decrypts the frame using a link encryption key to read the header and then encrypts the entire frame again for sending it out on the next link.
- Now the entire frame is secure except for the time that the frame is actually in the memory of a frame switch, at which time the frame header is in the clear.

5.3.5 Key Distribution [3]

- For symmetric encryption to work, the two parties to an exchange must share the same key, and that key must be protected from access by others.
- Key distribution technique: a means of delivering
- a key to two parties that wish to exchange data, without allowing others to see the key.

5.3.5 Key Distribution [3]

1. Host sends packet requesting connection.
2. Security service buffers packet; asks KDC for session key.
3. KDC distributes session key to both hosts.
4. Buffered packet transmitted.

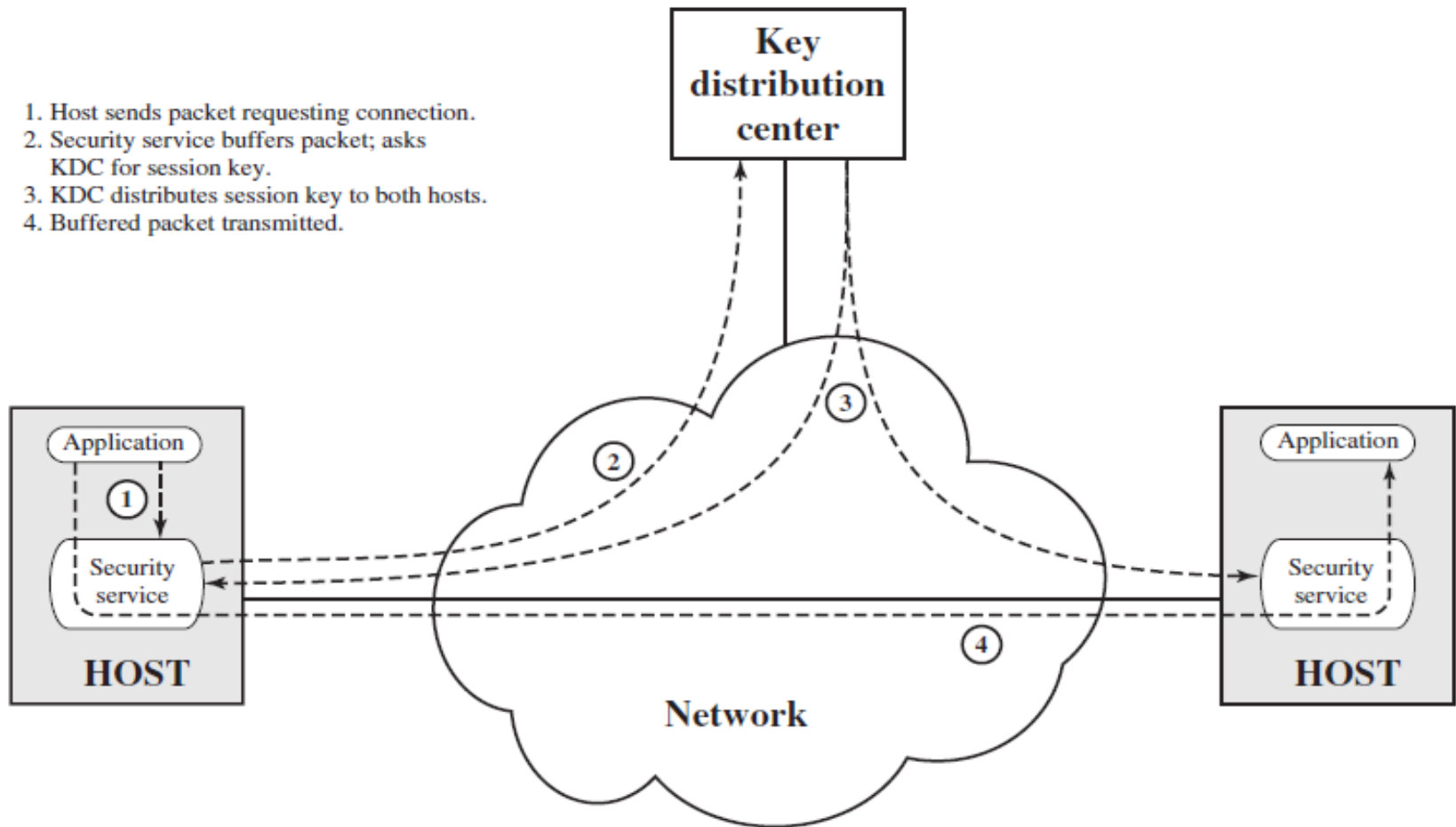


Figure 20.10 Automatic Key Distribution for Connection-Oriented Protocol

5.3.5 Key Distribution [3]

Two types of keys used:

- **Session key:** When two end systems (hosts, terminals, etc.) wish to communicate, they establish a logical connection (e.g., virtual circuit). For the duration of that logical connection, all user data are encrypted with a one-time session key. At the conclusion of the session, or connection, the session key is destroyed.
- **Permanent key:** A permanent key is a key used between entities for the purpose of distributing session keys.

5.3.5 Key Distribution [3]

The configuration consists of the following elements:

- **Key distribution center:** The key distribution center (KDC) determines which systems are allowed to communicate with each other. When permission is granted for two systems to establish a connection, the KDC provides a one-time session key for that connection.
- **Security service module (SSM):** This module, which may consist of functionality at one protocol layer, performs end-to-end encryption and obtains session keys on behalf of users.

5.3.5 Key Distribution [3]

Step 1: When one host wishes to set up a connection to another host, it transmits a connection request packet

Step 2: The SSM saves that packet and applies to the KDC for permission to establish the connection

Step 3: The communication between the SSM and the KDC is encrypted using a master key shared only by this SSM and the KDC. If the KDC approves the connection request, it generates the session key and delivers it to the two appropriate SSMs, using a unique permanent key for each SSM.

Step 4: The requesting SSM can now release the connection request packet, and a connection is set up between the two end systems

All user data exchanged between the two end systems are encrypted by their respective SSMs using the one-time session key.

5.4 Asymmetric Encryption or Public Key Cryptography [1]

- Asymmetric encryption uses **two keys** instead of only one.
- The keys are mathematically related and are known as the **public key** and the **private key**.
- The public key is known to everyone and can be freely distributed, while the private key is known only to the individual to whom it belongs.
- When Bob wants to send a secure message to Alice, he uses Alice's public key to encrypt the message. Alice then uses her private key to decrypt it.

5.4 Asymmetric Encryption or Public Key Cryptography [1]

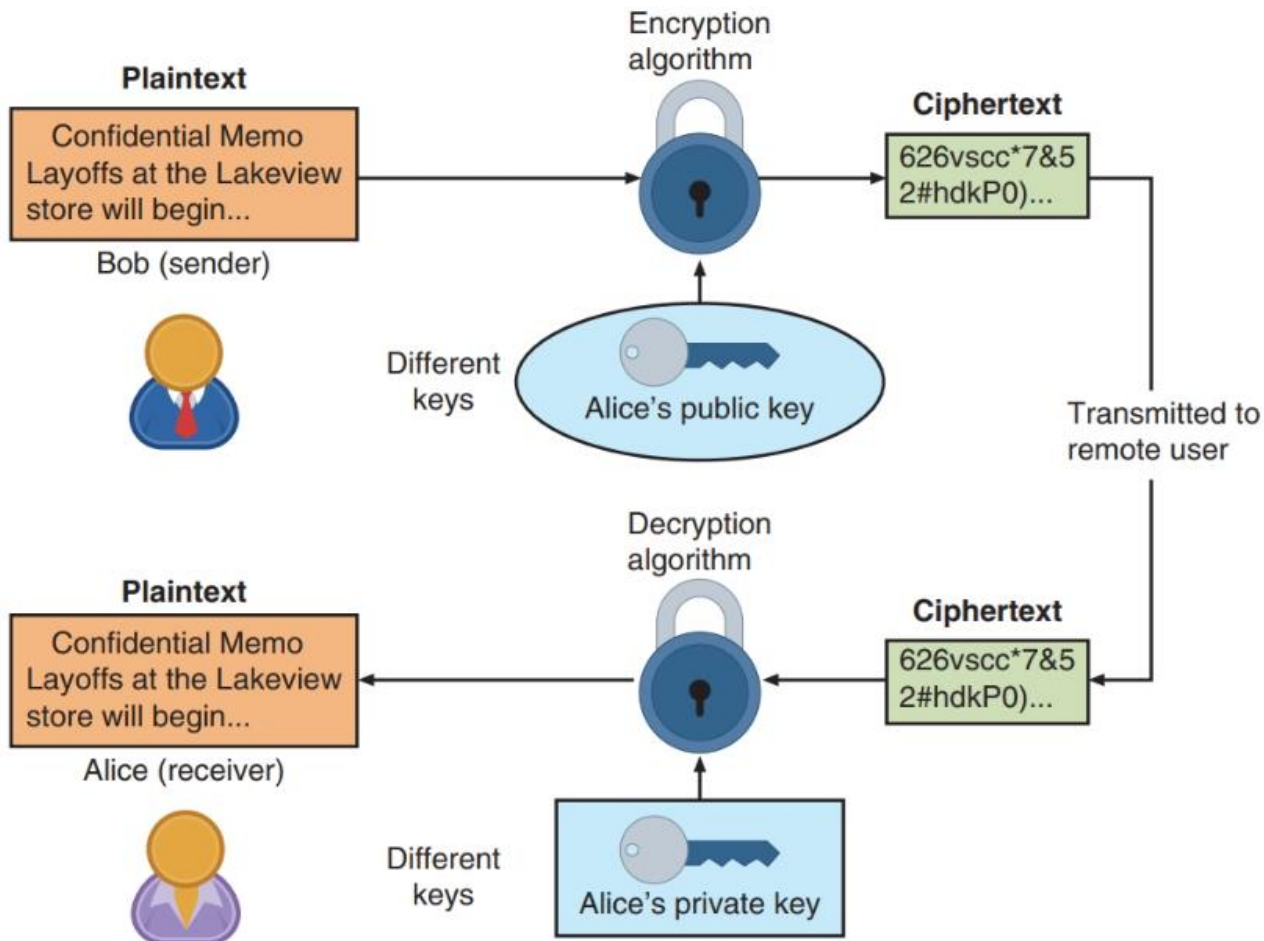


Figure 6-7 Asymmetric (public key) cryptography

5.4 Asymmetric Encryption or Public Key Cryptography [1]

Principles regarding asymmetric cryptography are as follows:

- **Key pairs.** Unlike symmetric cryptography that uses only one key, asymmetric cryptography requires a **pair of keys**.
- **Public key.** Public keys are designed to be public and do not need to be protected. They can be freely given to anyone or even posted on the Internet.
- **Private key.** The private key must be kept confidential and never shared. No user other than the owner must ever have the private key.
- **Both directions.** Asymmetric cryptography keys can work in both directions. A document encrypted with a public key can be decrypted with the corresponding private key. In the same way, a document encrypted with a private key can be decrypted with its public key

5.4 Asymmetric Encryption or Public Key Cryptography [1]

There are several asymmetric encryption algorithms:

5.4.1 RSA Public-Key Encryption

5.4.2 Elliptic Curve Cryptography (ECC)

5.4.3 Digital Signature Algorithm (DSA)

5.4.4 Diffie-Hellman Key Exchange

5.4.1 RSA Public-Key Encryption [1] [4]

- The asymmetric algorithm RSA (Rivest-Shamir-Adleman) was published in 1977 and became the basis for several products.
- The RSA algorithm is a block cipher. The typical block size for RSA is 1024 bits
- The RSA algorithm multiplies two large prime numbers (a prime number is a number divisible only by itself and 1), p and q , to compute their product ($n = pq$).
- Next, a number e is chosen that is less than n and not sharing a prime factor to $(p - 1)(q - 1)$ which is denoted as m .
- Another number d is determined so that $(ed - 1)$ is divisible by m .
- The values of e and d are the public and private exponents.
- The **public key** is the pair (n, e) while the **private key** is (n, d) .
- The numbers p and q can be discarded.

5.4.1 RSA Public-Key Encryption [1]

An illustration of the RSA algorithm using very small numbers is as follows:

1. Select two prime numbers, p and q (in this example, $p = 7$ and $q = 19$).
2. Multiply p and q together to create n ($7 \times 19 = 133$).
3. Calculate m as $(p - 1)(q - 1)$ ($[7 - 1] \times [19 - 1]$ or $6 \times 18 = 108$).
4. Find a number e so that it and m have no common positive divisor other than 1 ($e = 5$).
5. Find a number d so that $(ed-1)$ is divisible by m .
 $5d-1=108 \rightarrow 5d = 109$ (not divisible by 5)
 $5d-1=108 \times 2 \rightarrow 5d = 217$ (not divisible by 5)
 $5d-1=108 \times 3 \rightarrow 5d = 325 \rightarrow d = 65$

For this example, the public key n is 133 and e is 5, while for the private key n is 133 and d is 65.

5.4.1 RSA Public-Key Encryption

- RSA is used in web browsers, email, VPNs, chat and other communication channels.
- RSA is often used to make secure connections between VPN clients and VPN servers.
- Under protocols like OpenVPN, Transport Layer Security (TLS) handshakes can use the RSA Algorithm to exchange keys and establish a secure channel.

5.4.2 Elliptic Curve Cryptography (ECC) [1]

- The basis of RSA asymmetric encryption security is factoring, or the big prime numbers that make up a value. As computers become faster and more powerful, the ability to “crack” RSA asymmetric encryption by computing the factoring has grown.
- Instead of using factoring as the basis, other research looked at an obscure branch of mathematics called elliptic curves. In short, an elliptic curve is a set of points that satisfy a specific mathematical equation.
- Elliptic curves paved the way for a different form of asymmetric encryption not based on factoring.

5.4.2 Elliptic Curve Cryptography (ECC) [1]

- Instead of using large prime numbers as with RSA, elliptic curve cryptography (ECC) uses sloping curves.
- An elliptic curve is a function drawn on an X-Y axis as a gently curved line. By adding the values of two points on the curve, a third point on the curve can be derived, of which the inverse is used.

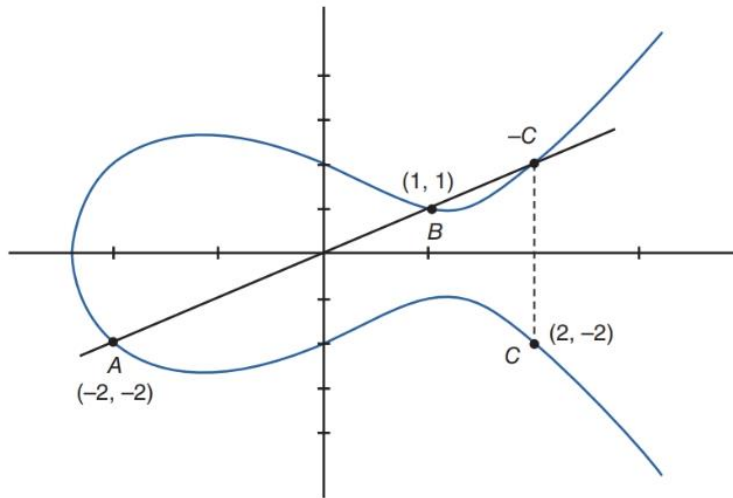


Figure 6-8 Elliptic curve cryptography (ECC)

Users share one elliptic curve and one point on the curve. One user chooses a secret random number and computes a public key based on a point on the curve; the other user does the same. They can now exchange messages because the shared public keys can generate a private key on an elliptic curve.

5.4.2 Elliptic Curve Cryptography (ECC) [1]

- The difference in size necessary to produce the same level of security between RSA and ECC keys is significant.
- According to one study to break a 228-bit RSA key, the amount of energy needed would take less energy than what is required to boil a teaspoon of water. However, breaking a 228-bit ECC key would require more energy than it would take to boil all the water on Earth.

Table 6-5 RSA vs. ECC key length for same security level

RSA key length	ECC key length
1,024	160
2,048	224
3,072	256
7,680	384
15,360	521

5.4.2 Elliptic Curve Cryptography (ECC) [1]

- Despite a slow start, ECC has gained wide popularity.
- It is used by the U.S. government to protect internal communications, by the Tor project to help assure anonymity, and as the mechanism to prove ownership of bitcoins.
- All modern OSs and web browsers rely on ECC.
- Because mobile devices are limited in terms of computing power due to their smaller size, ECC offers security that is comparable to other asymmetric cryptography but with smaller key sizes, resulting in faster computations and lower power consumption.

5.4.3 Digital Signature Algorithm (DSA) [1]

- Asymmetric cryptography also can be used to provide proofs.
- Suppose Alice receives an encrypted document that says it came from Bob. Although Alice can be sure that the encrypted message was not viewed or altered by someone else while being transmitted, how can she know for certain that Bob was the sender? Because Alice's public key is widely available, anyone could use it to encrypt the document. Another individual could have created a fictitious document, encrypted it with Alice's public key, and then sent it to Alice while pretending to be Bob. Alice's key can verify that no one read or changed the document in transport, but it cannot verify the sender.

5.4.3 Digital Signature Algorithm (DSA) [1]

- Proof can be provided with asymmetric cryptography, however, by creating a **digital signature**, which is an electronic verification of the sender.
- A digital signature can:
 - **Verify the sender.** A digital signature serves to confirm the identity of the person from whom the electronic message originated.
 - **Prevent the sender from disowning the message.** The signer cannot later attempt to disown it by claiming the signature was forged (nonrepudiation).
 - **Prove the integrity of the message.** A digital signature can prove that the message has not been altered since it was signed.

5.4.3 Digital Signature Algorithm (DSA) [1]

- The basis for a digital signature rests on the ability of asymmetric keys to work in both directions (a public key can encrypt a document that can be decrypted with a private key, and the private key can encrypt a document that can be decrypted by the public key).

5.4.3 Digital Signature Algorithm (DSA) [1]

The steps for Bob to send a digitally signed message to Alice are as follows:

1. After creating a memo, Bob generates a digest on it.
2. Bob encrypts the digest with his private key. The encrypted digest is the digital signature for the memo.
3. Bob sends both the memo and the digital signature to Alice.

5.4.3 Digital Signature Algorithm (DSA) [1]

The steps for Bob to send a digitally signed message to Alice are as follows (cont.):

4. When Alice receives them, she decrypts the digital signature using Bob's public key, revealing the digest. If she cannot decrypt the digital signature, then she knows that it did not come from Bob (because only Bob's public key can decrypt the digest generated with his private key).

5.4.3 Digital Signature Algorithm (DSA) [1]

The steps for Bob to send a digitally signed message to Alice are as follows (cont.):

5. Alice then hashes the memo with the same hash algorithm Bob used and compares the result to the digest she received from Bob. If they are equal, Alice can be confident that the message has not changed since he signed it. If the digests are not equal, Alice will know the message has changed since it was signed.

Note: Using a digital signature does not encrypt the message itself. In the example, if Bob wanted to ensure the privacy of the message, he also would have to encrypt it using Alice's public key.

5.4.3 Digital Signature Algorithm (DSA) [1]

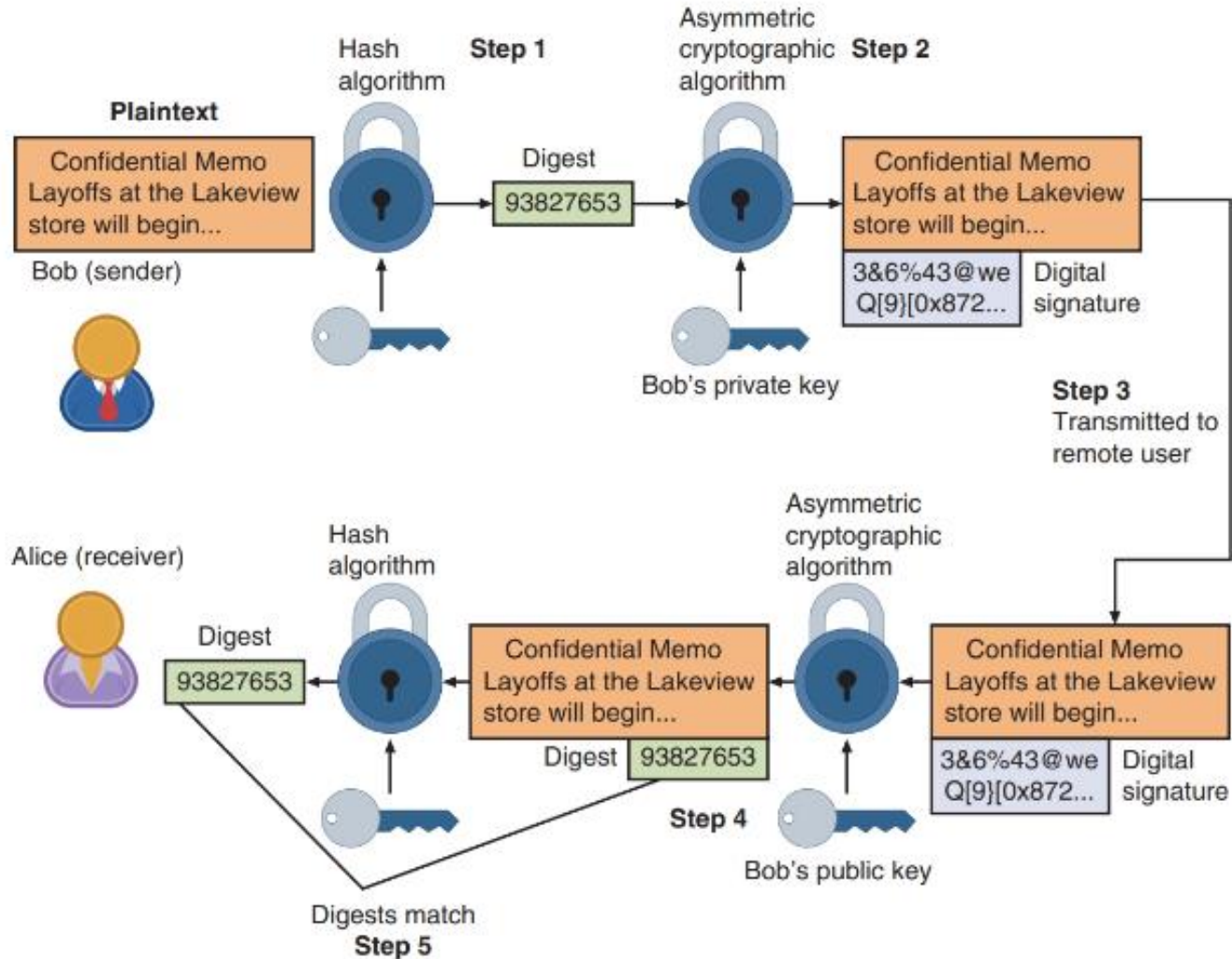


Figure 6-9 Digital signature

5.4.3 Digital Signature Algorithm (DSA) [1]

- The Digital Signature Algorithm (DSA) is a U.S. federal government standard for digital signatures.
- DSA was proposed by NIST in 1991 for use in their Digital Signature Standard (DSS). Although patented, NIST has made the patent available worldwide royalty-free. The standard continues to be revised and updated periodically by NIST.

5.4.4 Diffie-Hellman Key Exchange[1]

- Public and private keys may result in confusion regarding whose key to use and which key should be used.

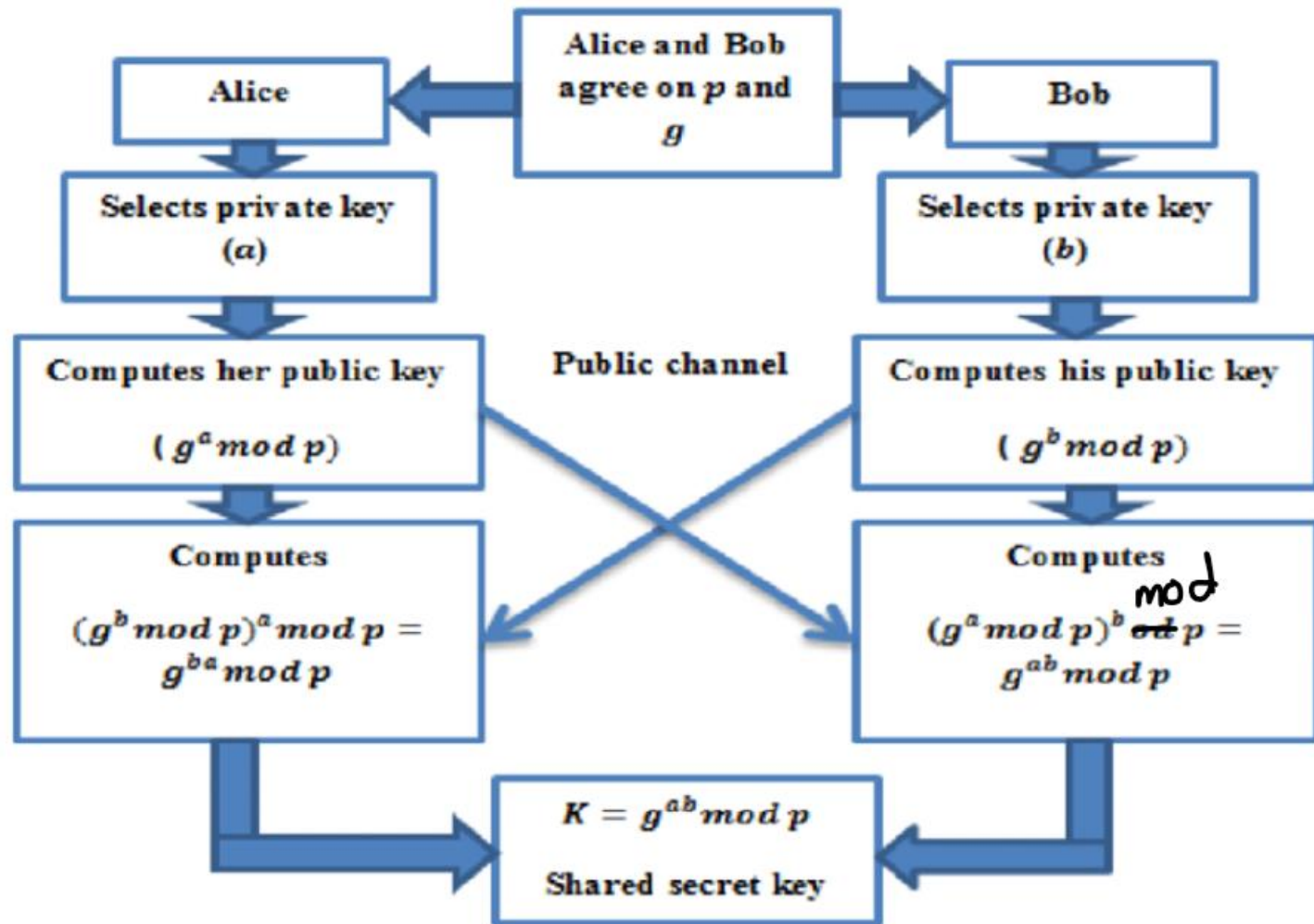
Table 6-6 Asymmetric cryptography practices

Action	Whose key to use	Which key to use	Explanation
Bob wants to send Alice an encrypted message.	Alice's key	Public key	When an encrypted message is to be sent, the recipient's, and not the sender's, key is used.
Alice wants to read an encrypted message sent by Bob.	Alice's key	Private key	An encrypted message can be read only by using the recipient's private key.
Action	Whose key to use	Which key to use	Explanation
Bob wants to send a copy to himself of the encrypted message that he sent to Alice.	Bob's key	Public key to encrypt Private key to decrypt	An encrypted message can be read only by the recipient's private key. Bob would need to encrypt it with his public key and then use his private key to decrypt it.
Bob receives an encrypted reply message from Alice.	Bob's key	Private key	The recipient's private key is used to decrypt received messages.
Bob wants Susan to read Alice's reply message that he received.	Susan's key	Public key	The message should be encrypted with Susan's key for her to decrypt and read with her private key.
Bob wants to send Alice a message with a digital signature.	Bob's key	Private key	Bob's private key is used to encrypt the hash.
Alice wants to see Bob's digital signature.	Bob's key	Public key	Because Bob's public and private keys work in both directions, Alice can use his public key to decrypt the hash.

5.4.4 Diffie-Hellman Key Exchange[1]

- In addition to confusion regarding which key to use, there are also issues with sending and receiving keys (key exchange) such as exchanging a symmetric private key.
- One of the solutions for a key exchange that occurs within the normal communications channel of cryptography is to use Diffie-Hellman (DH) key exchange.
- The Diffie-Hellman (DH) key exchange requires Alice and Bob to each agree upon a large prime number and related integer. Those two numbers can be made public, yet Alice and Bob, through mathematical computations and exchanges of intermediate values, can separately create the same key (shared secret key) to encrypt and exchange information.

5.4.4 Diffie-Hellman Key Exchange[1]



5.4.4 Diffie-Hellman Key Exchange[1]

- For illustration using small numbers, for example, both Alice and Bob have agreed on a large prime number (**p = 23**) and related integer (**g = 5**).
- Alice picks a secret number (**a = 4**), Bob picks a secret number (**b=3**)
- Alice calculates her public key (A):

$$\begin{aligned} A &= g^a \bmod p \\ &= 5^4 \bmod 23 \\ &= 4 \end{aligned}$$

- Bob calculates his public key (B):

$$\begin{aligned} B &= g^b \bmod p \\ &= 5^3 \bmod 23 \\ &= 10 \end{aligned}$$

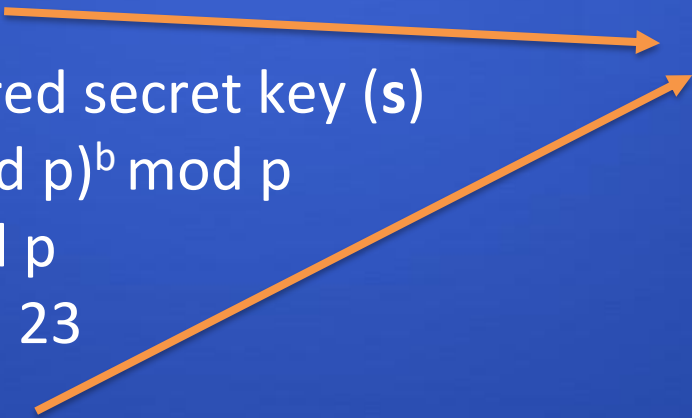
5.4.4 Diffie-Hellman Key Exchange[1]

- Both Alice and Bob exchange their public keys to each other via the public channel.
- Alice calculates the shared secret key (s)

$$\begin{aligned}s &= (g^b \bmod p)^a \bmod p \\ &= B^a \bmod p \\ &= 10^4 \bmod 23 \\ &= 18\end{aligned}$$

- Bob calculates the shared secret key (s)

$$\begin{aligned}s &= (g^a \bmod p)^b \bmod p \\ &= A^b \bmod p \\ &= 4^3 \bmod 23 \\ &= 18\end{aligned}$$



Same shared
secret key

5.5 New Cryptographic Standard

- When large-scale **quantum computing** becomes available, possibly in about **10 years from now**, it will have a major impact on cryptography. In particular, the **asymmetric cryptographic algorithms** predominantly used today will be **effectively broken**.

Source: <https://www.cryptomathic.com/news-events/blog/symmetric-encryption-algorithms-their-strengths-and-weaknesses-and-the-need-for-crypto-agility>

5.5 New Cryptographic Standard

- Organizations are working towards preparing a new cryptographic standard to replace the public key cryptography to protect against future quantum-based threats



CYBERSECURITY
& INFRASTRUCTURE
SECURITY AGENCY



Alerts and Tips

Resources

National Cyber Awareness System > Current Activity > Prepare for a New Cryptographic Standard to Protect Against Future Q

Prepare for a New Cryptographic Standard to Protect Against Future Quantum-Based Threats

Original release date: July 05, 2022



The National Institute of Standards and Technology (NIST) has announced that a new post-quantum cryptographic standard will replace current public-key cryptography, which is vulnerable to quantum-based attacks. **Note:** the term “post-quantum cryptography” is often referred to as “quantum-resistant cryptography” and includes, “cryptographic algorithms or methods that are assessed not to be specifically vulnerable to attack by either a CRQC [cryptanalytically relevant quantum computer] or classical computer.” (See the [National Security Memorandum on Promoting United States Leadership in Quantum Computing While Mitigating Risks to Vulnerable Cryptographic Systems](#) for more information).

Although NIST will not publish the new post-quantum cryptographic standard for use by commercial products until 2024, CISA and NIST strongly recommend organizations start preparing for the transition now by following the [Post-Quantum Cryptography Roadmap](#), which includes:

Source:

<https://www.cisa.gov/uscert/ncas/current-activity/2022/07/05/prepare-new-cryptographic-standard-protect-against-future-quantum>

Main References

- [1] Mark Ciampa. 2022. CompTIA Security+ Guide To Network Security Fundamentals. Cengage Learning.
- [2] Easttom, Chuck. 2020. Computer Security Fundamentals. 4th ed. Pearson
- [3] William Stallings and Lawrie Brown. 2018. Computer Security: Principles and Practice. Pearson.
- [4] Chauhan, S. R., and Jangra S., 2020, Computer Security and Encryption: An Introduction, Mercury Learning & Information.
<https://tarc.idm.oclc.org/login?url=https://ebookcentral.proquest.com/lib/tarc-ebooks/detail.action?docID=6404902>