Cryptographic Systems

Managing Administrative Access

- A network LAN can be secured through:
  - Device hardening
  - AAA access control
  - Firewall features
  - IPS implementations
- How is network traffic protected when traversing the public Internet?
  - Using cryptographic methods

Secure Communications Requires …

Authentication

Integrity

Confidentiality

Authentication

- Authentication guarantees that the message:
  - Is not a forgery.
  - Does actually come from who it states it comes from.
- Authentication is similar to a secure PIN for banking at an ATM.
  - The PIN should only be known to the user and the financial institution.
  - The PIN is a shared secret that helps protect against forgeries.

Authentication

- Data nonrepudiation is a similar service that allows the sender of a message to be uniquely identified.
- This means that a sender / device cannot deny having been the source of that message.
  - It cannot repudiate, or refute, the validity of a message sent.

Integrity

- Data integrity ensures that messages are not altered in transit.
  - The receiver can verify that the received message is identical to the sent message and that no manipulation occurred.
- European nobility ensured the data integrity by creating a wax seal to close an envelope.
  - The seal was often created using a signet ring.
  - An unbroken seal on an envelope guaranteed the integrity of its contents.
  - It also guaranteed authenticity based on the unique signet ring impression.

Confidentiality

- Data confidentiality ensures privacy so that only the receiver can read the message.
- Encryption is the process of scrambling data so that it cannot be read by unauthorized parties.
  - Readable data is called plaintext, or cleartext.
  - Encrypted data is called ciphertext.
- A key is required to encrypt and decrypt a message.
  - The key is the link between the plaintext and ciphertext.

Managing Administrative Access

- Authentication, integrity, and confidentiality are components of cryptography.
- Cryptography is both the practice and the study of hiding information.
- It has been used for centuries to protect secret documents.
  - Today, modern day cryptographic methods are used in multiple ways to ensure secure communications.

Cryptographic Hashes

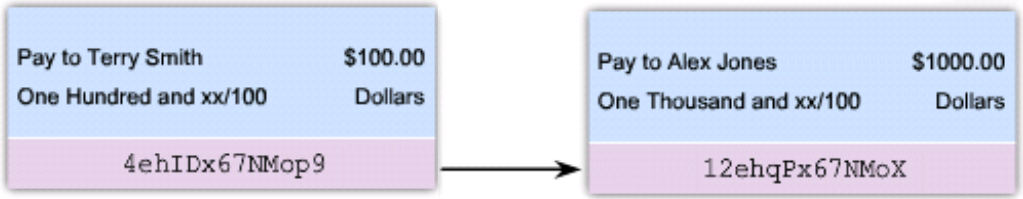
- A hash function takes binary data (message), and produces a condensed representation, called a hash.
  - The hash is also commonly called a Hash value, Message digest, or Digital fingerprint.
- Hashing is based on a one-way mathematical function that is relatively easy to compute, but significantly harder to reverse.
- Hashing is designed to verify and ensure:
  - Data integrity
  - Authentication

Hashes are used …

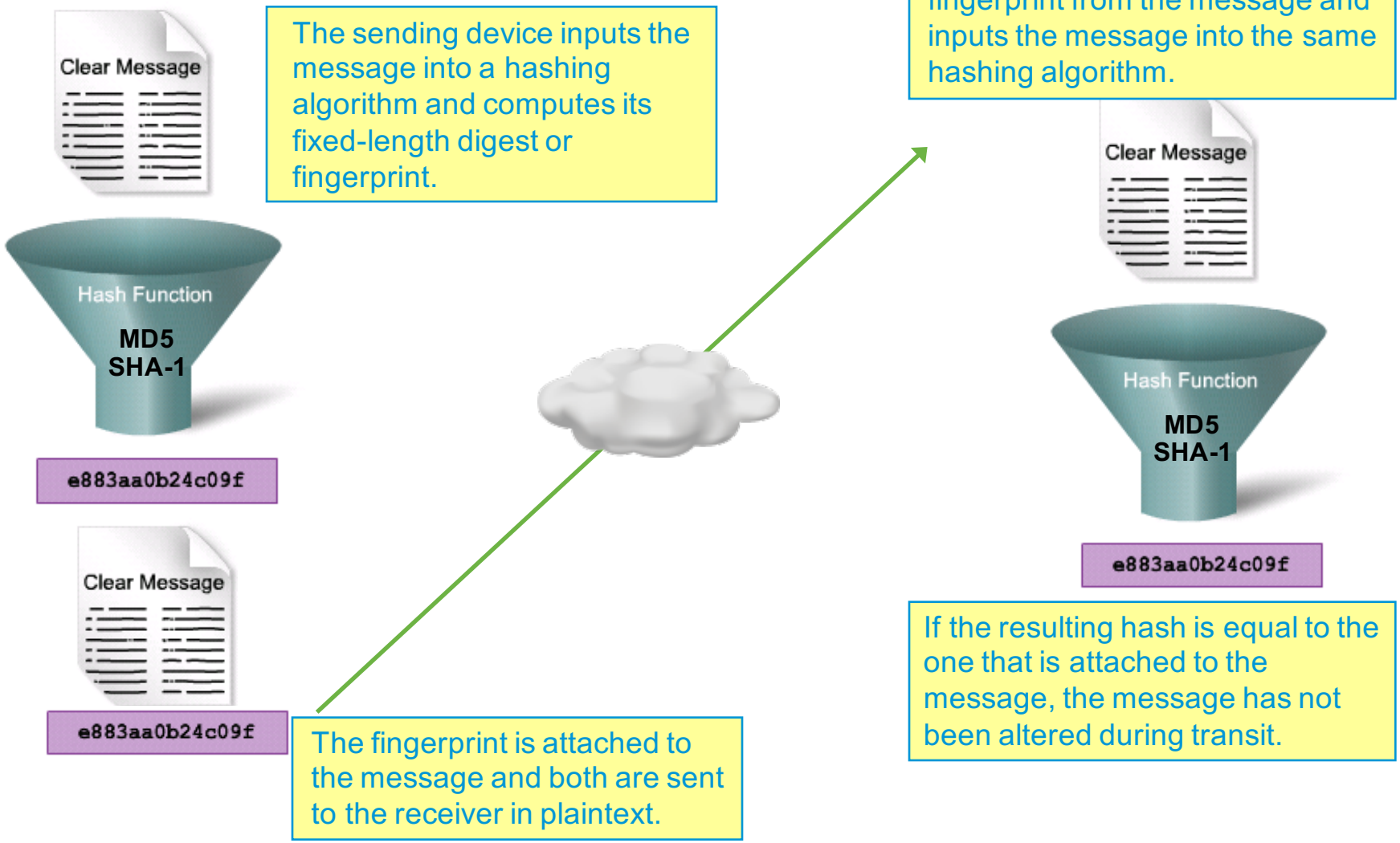
- To provide proof of authenticity when it is used with a symmetric secret authentication key, such as IP Security (IPsec) or routing protocol authentication.
- To provide authentication by generating one-time and one-way responses to challenges in authentication protocols such as the PPP CHAP.
- To provide a message integrity check proof such as those accepted when accessing a secure site using a browser.
- To confirm that a downloaded file (e.g., Cisco IOS images) has not been altered.

Hash for Integrity

- Hash functions (MD5 and SHA-1) can ensure message integrity but not confidentiality.
  - For instance, the sender wants to ensure that the message is not altered on its way to the receiver.



Hash for Integrity



Hash for Integrity

- Hashing only prevents the message from being changed accidentally, such as by a communication error.
- It's still susceptible to man-in-the-middle attacks.
  - A potential attacker could intercept the message, change it, recalculate the hash, and append it to the message.
  - There is nothing unique to the sender in the hashing procedure, so anyone can compute a hash for any data, as long as they have the correct hash function.
- These are two well-known hash functions:
  - Message Digest 5 (MD5) with 128-bit digests
  - Secure Hash Algorithm 1 (SHA-1) with 160-bit digests

Secure Hash Algorithm (SHA)

- The U.S. National Institute of Standards and Technology (NIST) developed the Secure Hash Algorithm (SHA).
  - SHA-1, published in 1994, corrected an unpublished flaw in SHA.
  - It's very similar to the MD4 and MD5 hash functions.
- The SHA-1 algorithm takes a message of less than 2^{64} bits in length and produces a 160-bit message digest.
- This makes SHA-1 slightly slower than MD5, but the larger message digest makes it more secure against brute-force collision and inversion attacks.

MD5 versus SHA-1

MD5	SHA-1
Based on MD4	Based on MD4
Computation involves 64 steps	Computation involves 80 steps
Algorithm must process a 128-bit buffer	Algorithm must process a 160-bit buffer
Faster	Slower
Less Secure	More secure

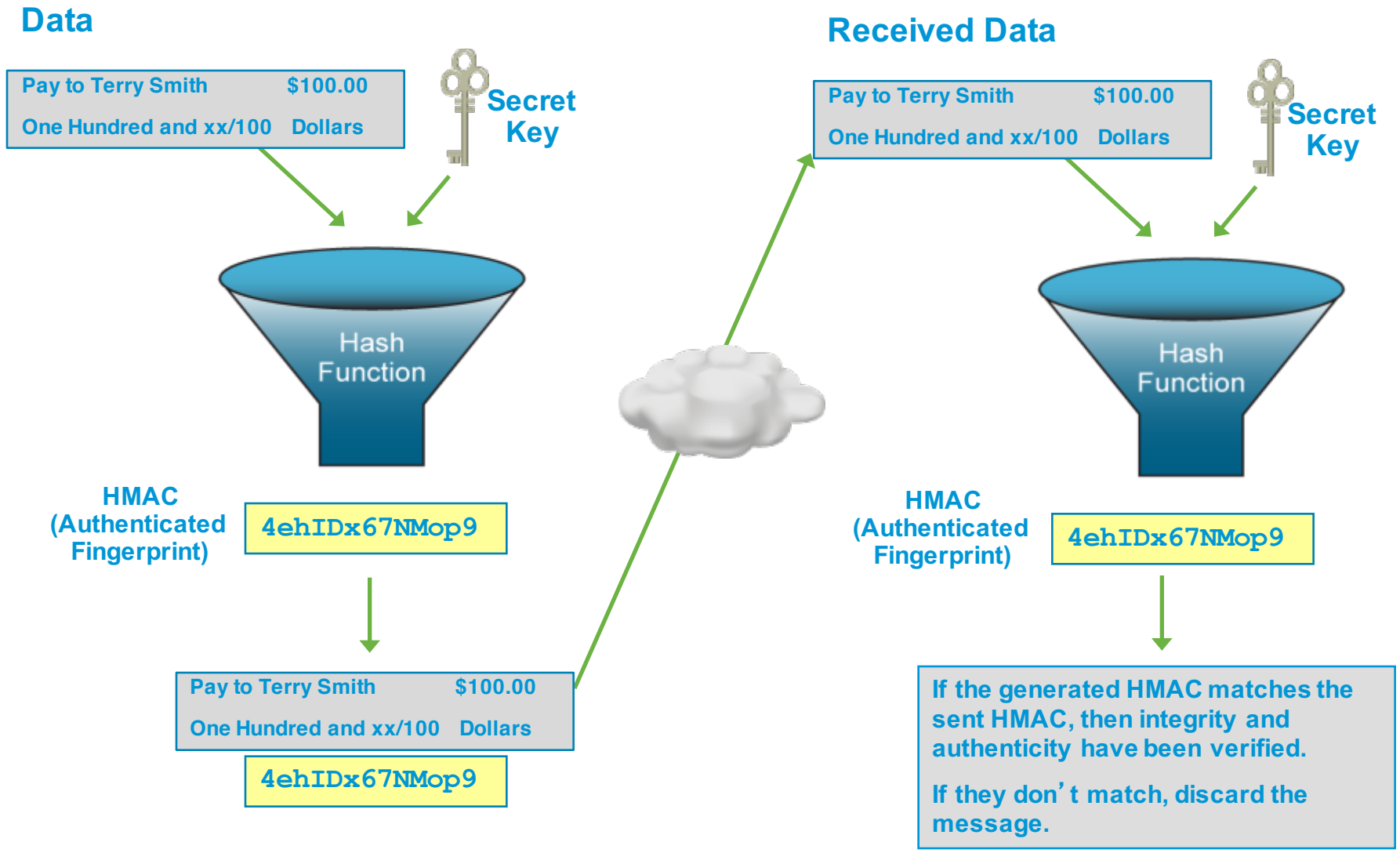
Keyed-Hash Message Authentication Code

- HMAC (or KHMAC) is a message authentication code (MAC) that is calculated using a hash function and a secret key.
  - Hash functions are the basis of the protection mechanism of HMACs.
  - The output of the hash function now depends on the input data and the secret key.
- Authenticity is guaranteed because only the sender and the receiver know the secret key.
  - Only they can compute the digest of an HMAC function.
  - This characteristic defeats man-in-the-middle attacks and provides authentication of the data origin.

Keyed-Hash Message Authentication Code

- The cryptographic strength of the HMAC depends on the:
  - Cryptographic strength of the underlying hash function.
  - Size and quality of the key.
  - Size of the hash output length in bits.
- Cisco technologies use two well-known HMAC functions:
  - Keyed MD5 or HMAC-MD5 is based on the MD5 hashing algorithm.
  - Keyed SHA-1 or HMAC-SHA-1 is based on the SHA-1 hashing algorithm.

HMAC in Action



Key Management

- Often considered the most difficult part of designing a cryptosystem.
- There are several essential characteristics of key management to consider:
  - Key Generation
  - Key Verification
  - Key Storage
  - Key Exchange
  - Key Revocation and destruction

Key Management

- Key Generation:

Key generation is usually automated and not left to the end user. The use of good random number generators is needed to ensure that all keys are likely to be equally generated so that the attacker cannot predict which keys are more likely to be used.

- Key Verification:

Almost all cryptographic algorithms have some weak keys that should not be used, and with the help of key verification procedures, you can regenerate these keys if they occur.

- Key Storage:
  - Modern cryptographic system store keys in memory. More secure to store the key on a USB stick and require a password to unlock that key.

Key Management

- Key Exchange:
  - Key management procedures should provide a secure key exchange mechanism over an untrusted medium.
- Key Revocation and Destruction:
  - Revocation notifies all interested parties that a certain key has been compromised and should no longer be used.
  - Destruction erases old keys in a manner that prevents malicious attackers from recovering them.

Key Length and Keyspace

- The key length is the measure in bits and the keyspace is the number of possibilities that can be generated by a specific key length.
- As key lengths increase, keyspace increases exponentially:
  - 22 key = a keyspace of 4
  - 23 key = a keyspace of 8
  - 24 key = a keyspace of 16
  - 240 key = a keyspace of **1,099,511,627,776**

Keyspace

- Adding one bit to a key doubles the keyspace.
- For each bit added to the DES key, the attacker would require twice the amount of time to search the keyspace.
- Longer keys are more secure but are also more resource intensive and can affect throughput.

DES Key Length	Keyspace	# of Possible Keys
56 bit	2^{56}	72,000,000,000,000,000
57 bit	2^{57}	144,000,000,000,000,000
58 bit	2^{58}	288,000,000,000,000,000
59 bit	2^{59}	576,000,000,000,000,000

Types of Cryptographic Keys

- Symmetric keys which can be exchanged between two routers supporting a VPN.
- Asymmetric keys which are used in secure HTTPS applications.
- Digital signatures which are used when connecting to a secure website.
- Hash keys which are used in symmetric and asymmetric key generation, digital signatures, and other types of applications.

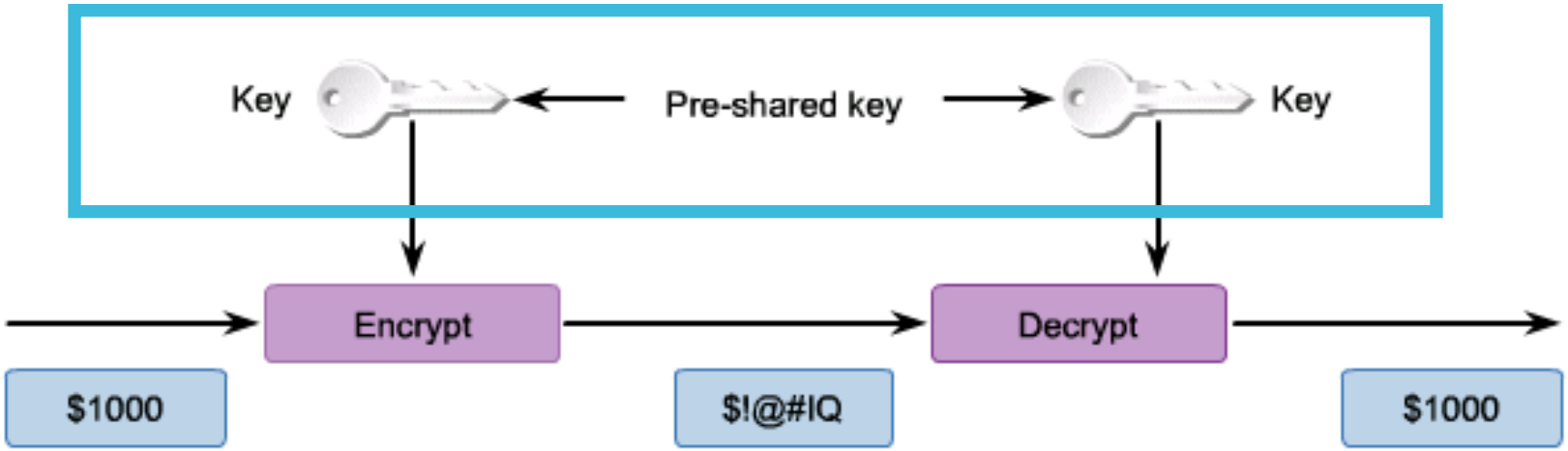
Protection Provided by Key Type

	Symmetric Key	Asymmetric Key	Digital Signature	Hash
Protection up to 3 years	80	1248	160	160
Protection up to 10 years	96	1776	192	192
Protection up to 20 years	112	2432	224	224
Protection up to 30 years	128	3248	256	256
Protection against quantum computers	256	15424	512	512

Symmetric Encryption

- Symmetric encryption algorithms, also called shared secret-key algorithms, use the same pre-shared secret key to encrypt and decrypt data.
  - The pre-shared key is known by the sender and receiver before any encrypted communications begins.
- Because both parties are guarding a shared secret, the encryption algorithms used can have shorter key lengths.
  - Shorter key lengths mean faster execution.
- For this reason symmetric algorithms are generally much less computationally intensive than asymmetric algorithms.

Symmetric Encryption

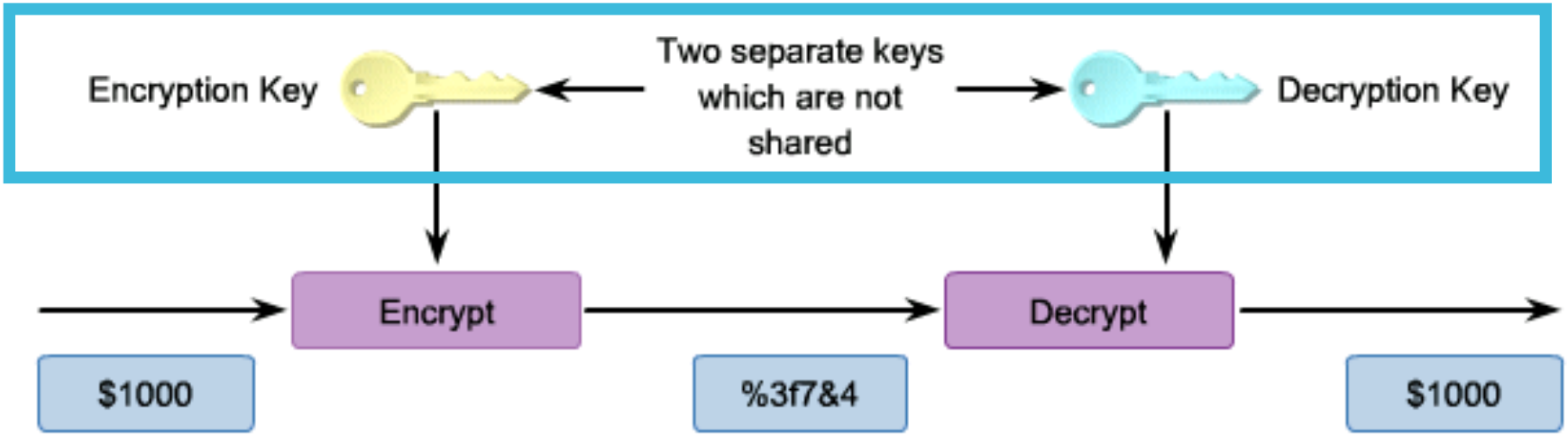


- Symmetric encryption algorithms are best known as shared-secret key algorithms.
- The usual key length is 80 - 256 bits.
- A sender and receiver must share a secret key.
- They are usually quite fast (wire speed) because these algorithms are based on simple mathematical operations.
- Examples of symmetric encryption algorithms are DES, 3DES, AES, IDEA, RC2/4/5/6, and Blowfish.

Asymmetric Encryption

- Asymmetric encryption algorithms, also called public key algorithms, use different keys to encrypt and decrypt data.
- Secure messages can be exchanged without having to have a pre-shared key.
- Because both parties do not have a shared secret, very long key lengths must be used to thwart attackers.
  - These algorithms are resource intensive and slower to execute.
- In practice, asymmetric algorithms are typically 100 to 1,000 times slower than symmetric algorithms.

Asymmetric Encryption



- Asymmetric encryption algorithms are best known as public key algorithms.
- The usual key length is 512–4096 bits.
- A sender and receiver do not share a secret key.
- These algorithms are relatively slow because they are based on difficult computational algorithms.
- Examples of asymmetric encryption algorithms are RSA, ElGamal, elliptic curves, and DH.

Symmetric Encryption

- Symmetric, or secret key, encryption is the most commonly used form of cryptography, because the shorter key length increases the speed of execution.
  - Symmetric key algorithms are based on simple mathematical operations that can easily be accelerated by hardware.
  - Symmetric encryption is often used for wire-speed encryption in data networks and to provide bulk encryption when data privacy is required, such as to protect a VPN.

Symmetric Key Management

- Key management can be a challenge since the encryption and decryption keys are the same.
- The security of a symmetric algorithm rests in the secrecy of the symmetric key.
  - By obtaining the key, anyone can encrypt and decrypt messages.
  - Sender and receiver must exchange the secret key using a secure channel before any encryption can occur.

Symmetric Key Management

- Well-known encryption algorithms that use symmetric keys including:
  - DES
  - 3DES
  - AES

- Other symmetric encryption algorithms include Blowfish, Twofish, Threefish, and Serpent.
  - However, these protocols are either not supported on Cisco platforms or have yet to gain wide acceptance.

Symmetric Encryption Algorithms

Symmetric Encryption Algorithm	Key length (in bits)	Description
DES	56	Designed at IBM during the 1970s and adopted as the NIST standard until 1997. Although considered outdated, DES remains widely in use. DES was designed to be implemented only in hardware, and is therefore extremely slow in software.
3DES	112 and 168	Based on using DES three times which means that the input data is encrypted three times and therefore considered much stronger than DES. However, it is rather slow compared to some new block ciphers such as AES.
AES	128, 192, and 256	AES is fast in both software and hardware, is relatively easy to implement, and requires little memory. As a new encryption standard, it is currently being deployed on a large scale.

Data Encryption Standard (DES)

- The most popular symmetric encryption standards.
  - Developed by IBM
  - Thought to be unbreakable in the 1970s
  - Shared keys enable the encryption and decryption
- DES converts blocks of 64-bits of clear text into ciphertext by using an encryption algorithm.
  - The decryption algorithm on the remote end restores ciphertext to clear text.

DES Scorecard

Description	Data Encryption Standard
Timeline	Standardized 1976
Type of Algorithm	Symmetric
Key size (in bits)	56 bits
Speed	Medium
Time to crack (Assuming a computer could try 255 keys per second)	Days (6.4 days by the COPACABANA machine, a specialized cracking device)
Resource Consumption	Medium

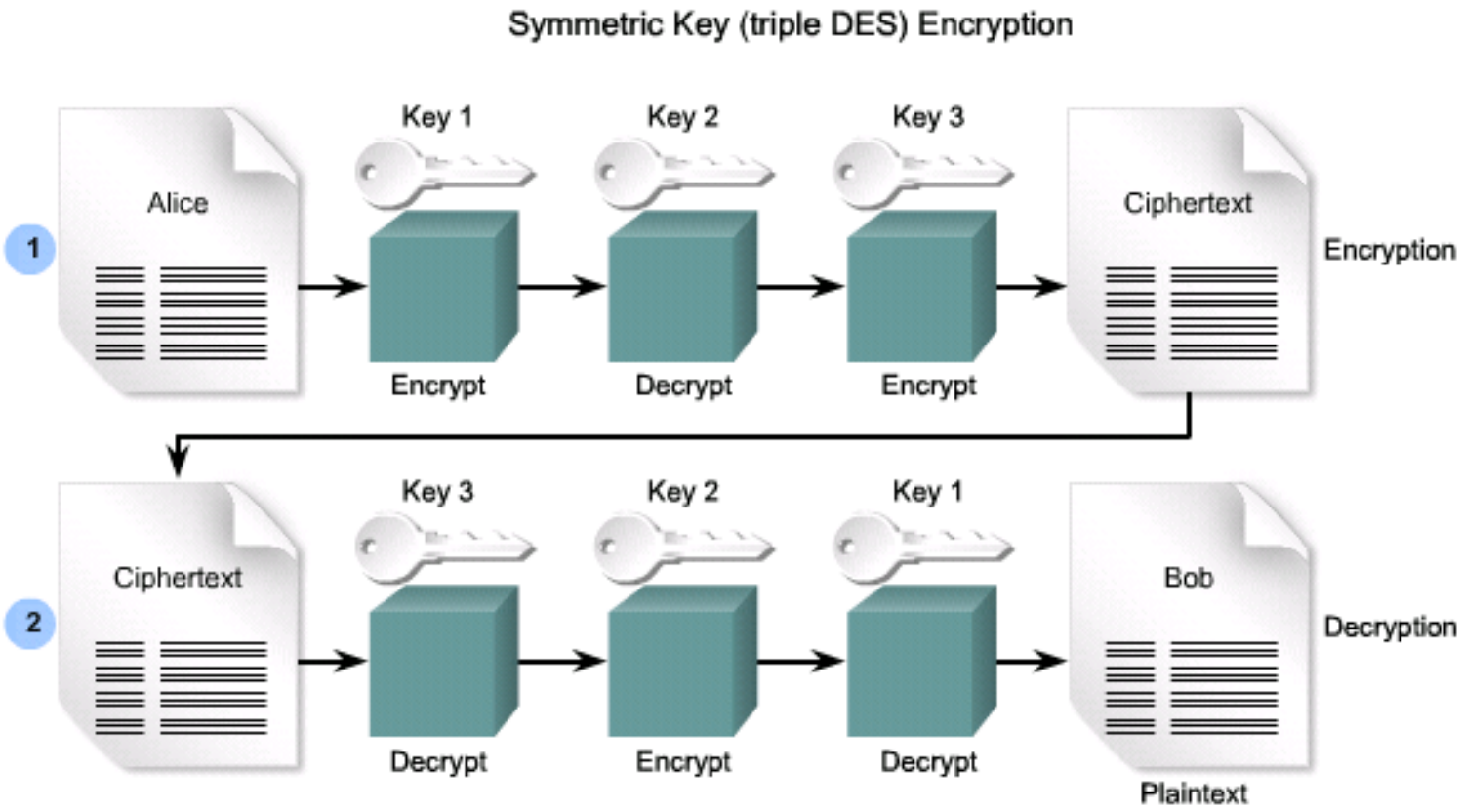
Triple DES (3DES or TDES)

- 3DES is 256 times stronger than DES.
- It takes a 64-bit block of data and performs three DES operations in sequence:
  - Encrypts, decrypts, and encrypts.
  - Requires additional processing time.
  - Can use 1, 2, or 3 different keys (when used with only one key, it is the same as DES).
- 3DES software is subject to US export laws.

3DES Scorecard

Description	Triple Data Encryption Standard
Timeline	Standardized 1977
Type of Algorithm	Symmetric
Key size (in bits)	112 and 168 bits
Speed	Low
Time to crack (Assuming a computer could try 255 keys per second)	4.6 Billion years with current technology
Resource Consumption	Medium

3DES



1. The clear text from Alice is encrypted using Key 1. That ciphertext is decrypted using a different key, Key 2. Finally that ciphertext is encrypted using another key, Key 3.

2. When the 3DES ciphered text is received, the process is reversed. That is, the ciphered text must first be decrypted using Key 3, encrypted using Key 2, and finally decrypted using Key 1.

Advanced Encryption Standard (AES)

- AES is an extremely secure Federal Information Processing Standard (FIPS)-approved cryptographic algorithm.
  - Based on the Rijndael (“Rhine dahl”) algorithm.
  - It use keys with a length of 128, 192, or 256 bits to encrypt blocks with a length of 128, 192, or 256 bits.
    - All 9 combinations of key length and block length are possible.
- AES is now available in the latest Cisco router images that have IPsec DES/3DES functionality.

AES Scorecard

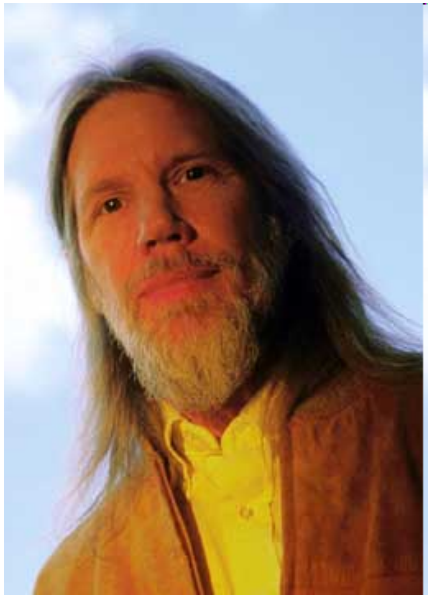
Description	Advanced Encryption Standard
Timeline	Official Standard since 2001
Type of Algorithm	Symmetric
Key size (in bits)	128, 192, and 256
Speed	High
Time to crack (Assuming a computer could try 255 keys per second)	149 Trillion years
Resource Consumption	Low

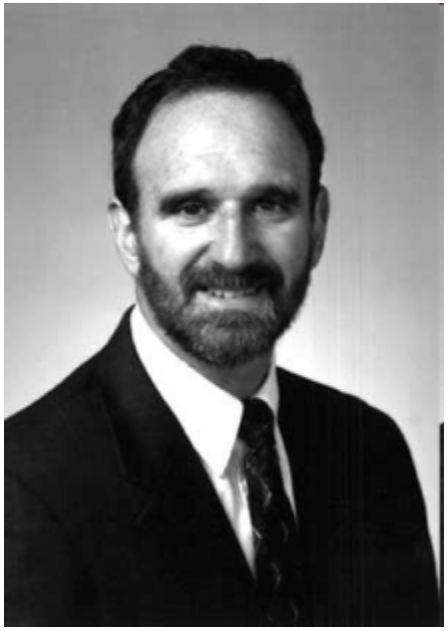
AES

- AES was chosen to replace DES for a number of reasons:
  - The key length of AES makes the key much stronger than DES.
  - AES runs faster than 3DES on comparable hardware.
  - AES is more efficient than DES and 3DES on comparable hardware, usually by a factor of five when it is compared with DES.
  - AES is more suitable for high-throughput, low-latency environments, especially if pure software encryption is used.
- However, AES is a relatively young algorithm and the golden rule of cryptography states that a mature algorithm is always more trusted.
- 3DES is therefore a more trusted choice in terms of strength, because it has been tested and analyzed for 35 years.

Diffie-Hellman (DH)

- DH is an asymmetric cryptographic protocol that allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel.
  - This key can then be used to encrypt subsequent communications using a symmetric key cipher.
- Published by Whitfield Diffie and Martin Hellman in 1976.





DH

- DH is commonly used when data is exchanged using an IPsec VPN, data is encrypted on the Internet using either SSL or TLS, or when SSH data is exchanged.
- It is not an encryption mechanism and is not typically used to encrypt data because it is extremely slow for any sort of bulk encryption.
- This is why it is common to encrypt the bulk of the traffic using a symmetric algorithm and use the DH algorithm to create keys that will be used by the encryption algorithm.

DH Scorecard

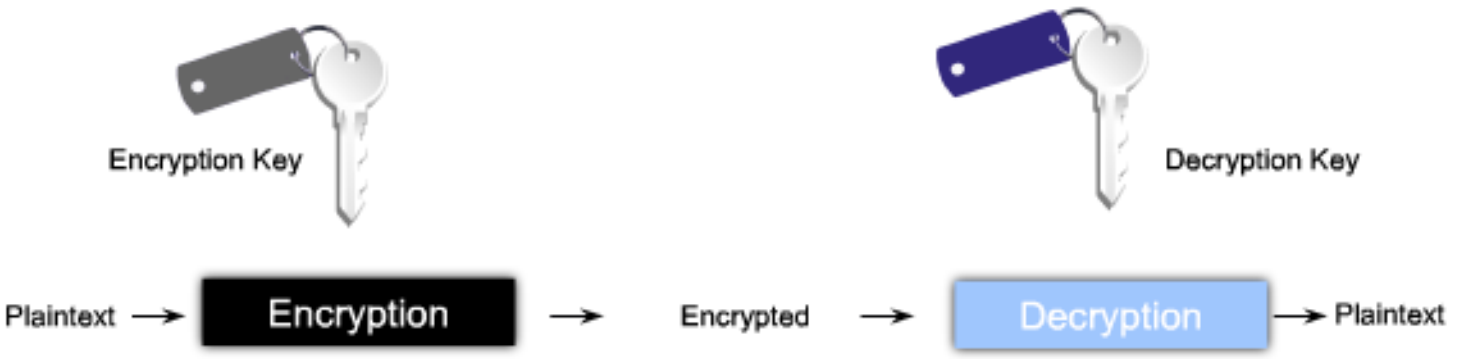
Description	Diffie-Hellman Algorithm
Timeline	1976
Type of Algorithm	Asymmetric
Key size (in bits)	512, 1024, 2048
Speed	Slow
Time to crack (Assuming a computer could try 255 keys per second)	Unknown but considered very safe
Resource Consumption	Medium

Public-key Algorithms

- Public-key algorithms are asymmetric algorithms based on the use of two different keys instead of one.
  - **Private key**: This key must be know *only* by its owner.
  - **Public key**: This key is known to everyone (it is *public*).
- The key that is used for encryption is different from the key that is used for decryption.
  - However, the decryption key cannot, in any reasonable amount of time, be calculated from the encryption key and vice versa.
- Public-key systems have a clear advantage over symmetric algorithms:
  - There is no need to agree on a common key for both the sender and the receiver.

Fundamental Concept

- Either key can be used for encryption but the complementary matched key is required for decryption.
  - If a public key encrypts data, the matching private key decrypts data.
  - If a private key encrypts data, the matching public key decrypts data.

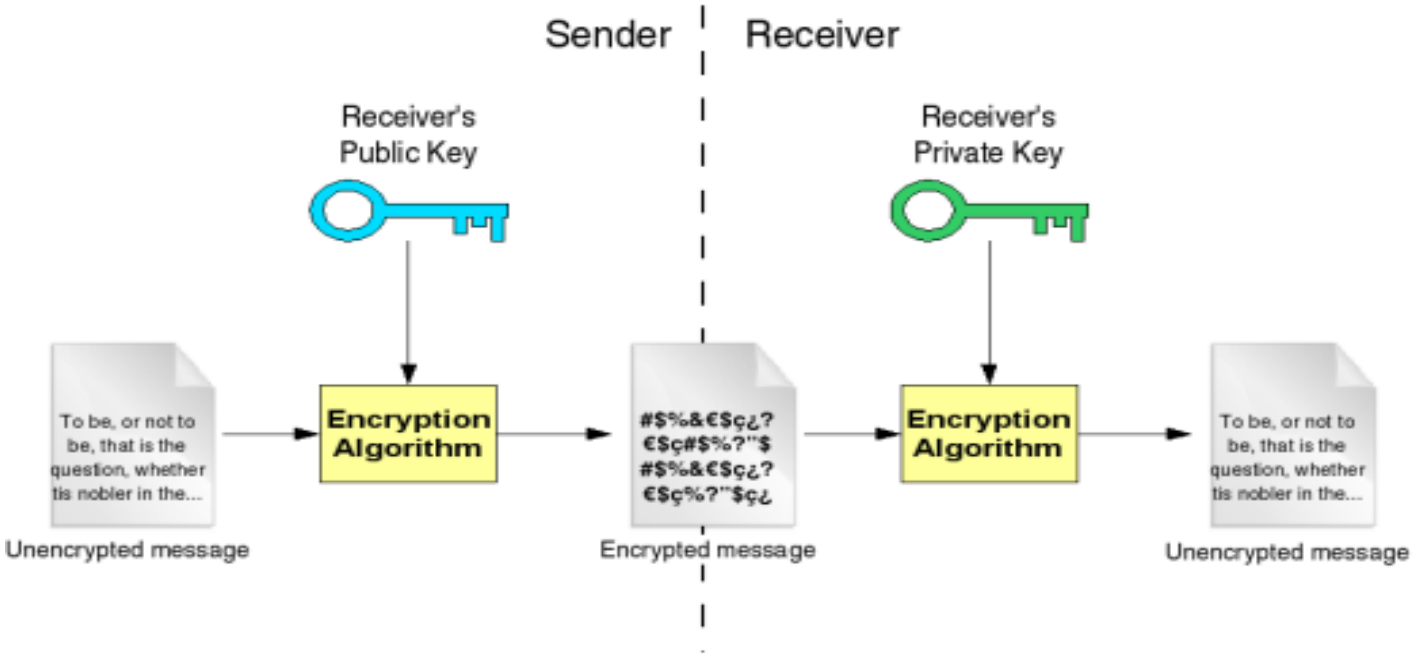


Asymmetric Key Characteristics

- The typical key length is 512–4096 bits.
- Key lengths greater than or equal to 1024 bits can be trusted.
- Key lengths that are shorter than 1024 bits are considered unreliable for most algorithms.

Process

- Sender encrypts the message using the receiver's **public** key.
  - Remember that this key is known to everyone.
- The encrypted message is sent to the receiving end, who will decrypt the message with his **private** key.
  - Only the receiver can decrypt the message because no one else has the private key.



CIA

- This process enables asymmetric algorithms to achieve:
  - Confidentiality
  - Integrity
  - Authentication

Authentication = Private Key (Encrypt) + Public Key (Decrypt)

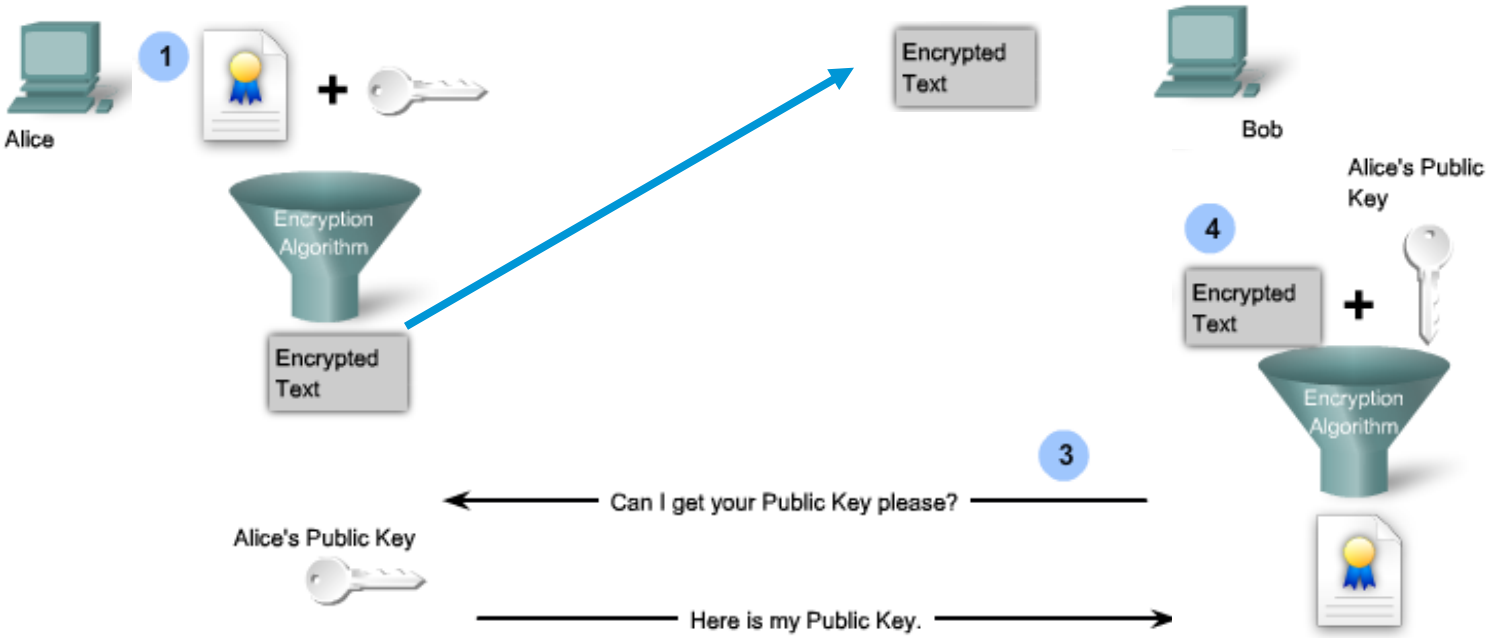
Confidentiality = Public Key (Decrypt) + Private Key (Encrypt)

Authentication

- Authentication is achieved when the encryption process is started with the private key.
  - The corresponding public key must be used to decrypt the data.
- Since only one host has the private key, only that host could have encrypted the message, providing authentication of the sender.

Asymmetric Algorithms for Authentication

Private Key (Encrypt) + Public Key (Decrypt) = Authentication



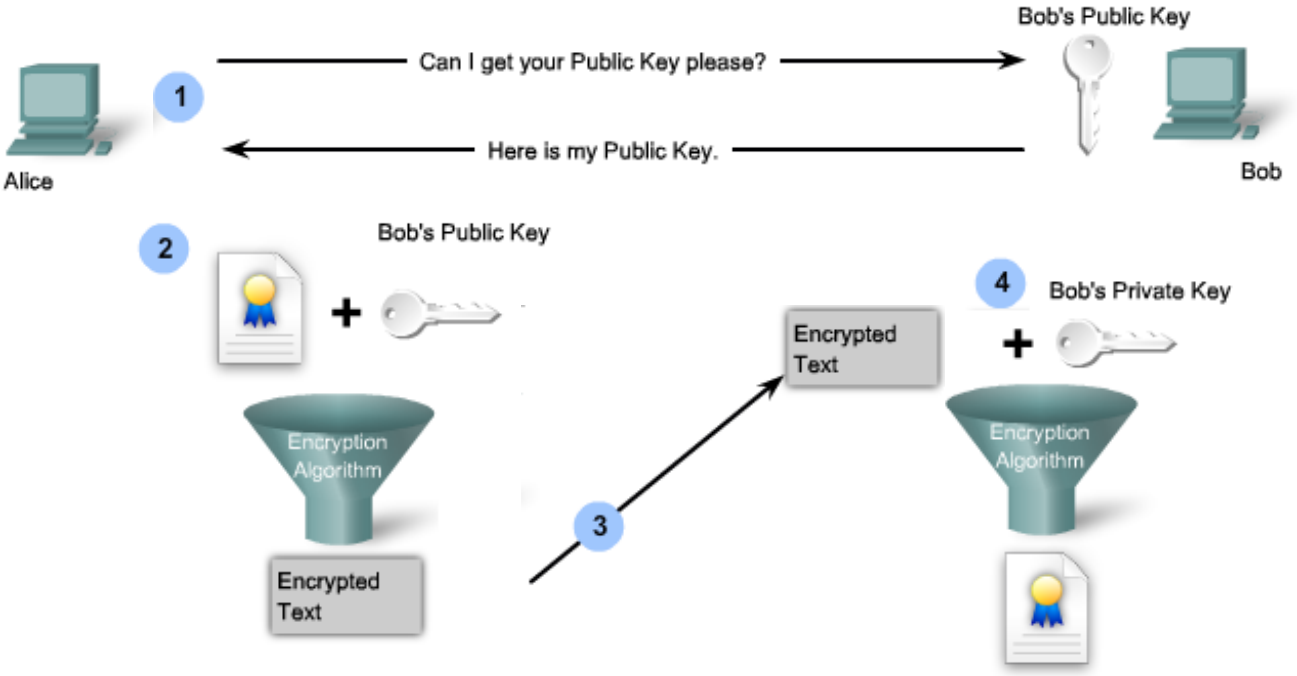
1. Alice encrypts a message with her private key.
2. Alice transmits the encrypted message to Bob.
3. To verify that the message actually came from Alice, Bob requests and acquires Alice's public key.
4. Bob uses the public key to successfully decrypt the message and authenticate that the message did, indeed, come from Alice.

Confidentiality

- Confidentiality is achieved when the encryption process is started with the public key.
- When the public key is used to encrypt the data, the private key must be used to decrypt the data.
  - Only one host has the private key guaranteeing confidentiality.

Asymmetric Algorithms for Confidentiality

Public Key (Encrypt) + Private Key (Decrypt) = Confidentiality

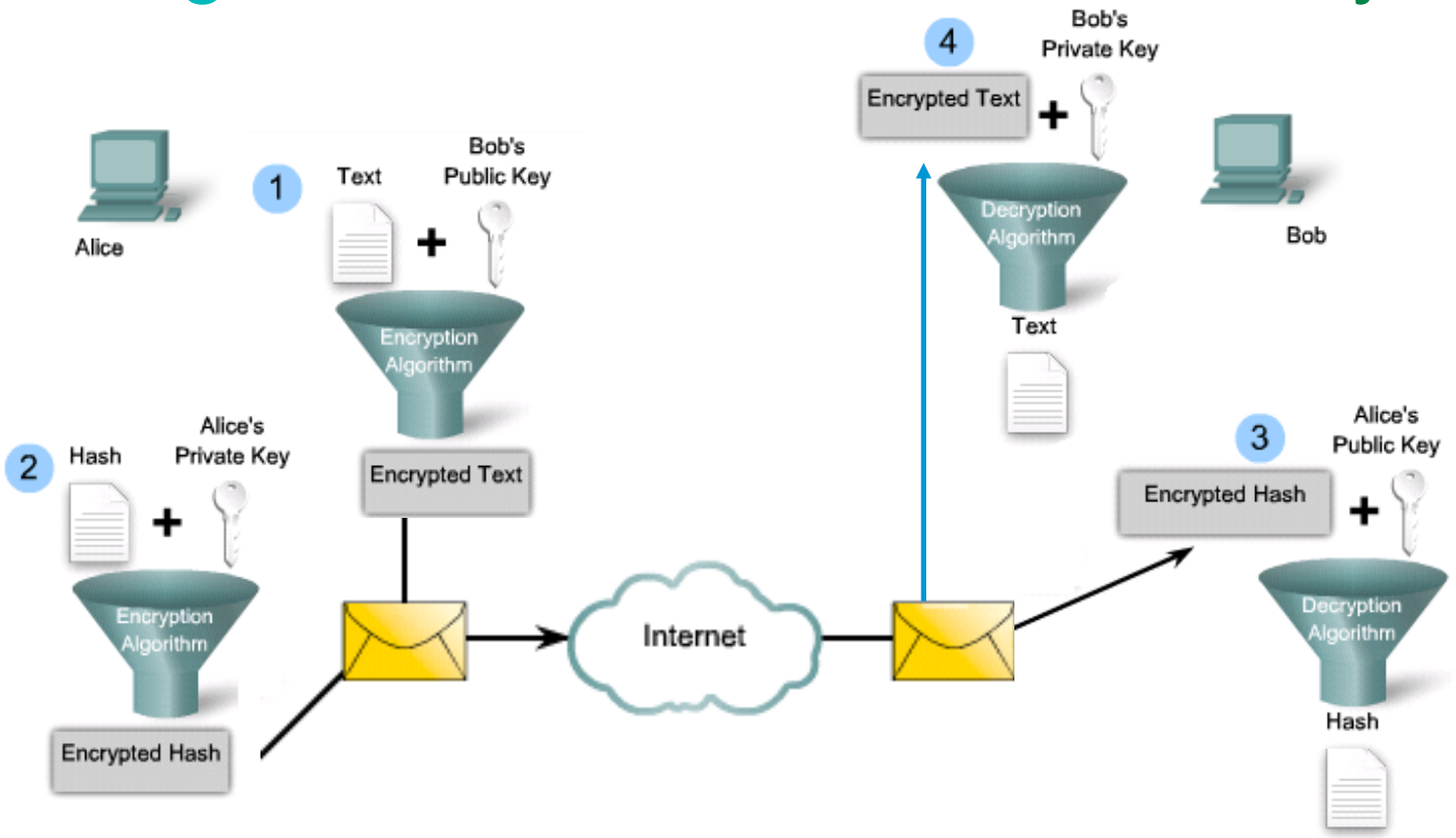


1. Alice asks Bob for his public key and Bob sends it to her.
2. Alice uses Bob's public key to encrypt a message using an agreed-upon algorithm.
3. Alice sends the encrypted message to Bob.
4. Bob uses his private key to decrypt and reveal the message.

Combining Authentication and Confidentiality

- To provide confidentiality, authentication and integrity, the combination of two phases is necessary.
  - Phase 1 - Confidentiality
  - Phase 2 - Authentication

Combining Authentication and Confidentiality



1. Alice encrypts a message using Bob's public key.
2. Alice encrypts a hash of the message using her private key.
3. Bob uses Alice's public key to decrypt and reveal the hash.
4. Bob uses his private key to decrypt and reveal the message.

Asymmetric Key Algorithms

- Well-known asymmetric key algorithms:
  - Diffie-Hellman
  - Digital Signature Standard (DSS), which incorporates the Digital Signature Algorithm
  - RSA encryption algorithms

Asymmetric Encryption Algorithms

Algorithm	Key length (in bits)	Description
Diffie-Hellman (DH)	512, 1024, 2048	Public key algorithm invented in 1976 by Whitfield Diffie and Martin Hellman that allows two parties to agree on a key that they can use to encrypt messages. Security depends on the assumption that it is easy to raise a number to a certain power, but difficult to compute which power was used given the number and the outcome.
Digital Signature Standard (DSS) and Digital Signature Algorithm (DSA)	512 - 1024	Created by NIST and specifies DSA as the algorithm for digital signatures. DSA is a public key algorithm based on the ElGamal signature scheme. Signature creation speed is similar with RSA, but is 10 to 40 times as slow for verification.
RSA encryption algorithms	512 to 2048	Developed by Ron Rivest, Adi Shamir, and Leonard Adleman at MIT in 1977. It is an algorithm for public-key cryptography based on the difficulty of factoring very large numbers. It is the first algorithm known to be suitable for signing as well as encryption, and one of the first great advances in public key cryptography. Widely used in electronic commerce protocols, and is believed to be secure given sufficiently long keys and the use of up-to-date implementations.

Asymmetric Key Algorithms

- Although the mathematics differ with each algorithm, they all share one trait in that the calculations required are complicated.
- Design is based on factoring extremely large numbers or computing discrete logarithms of extremely large numbers.
  - As a result, computation takes more time for asymmetric algorithms.
  - Can be up to 1,000 times slower than symmetric algorithms.
- Because they lack speed, they are typically used in low-volume cryptographic mechanisms.

Key Lengths

- Typical key lengths for asymmetric algorithms range from 512 to 4096 bits.
  - Key lengths >= 1024 bits — Considered to be trustworthy
  - Key lengths < 1024 bits — Considered unreliable
- Do not compare asymmetric and symmetric algorithms because they're underlying designs differ greatly.
  - For example:
    - 2048-bit encryption key of RSA is roughly equivalent to a 128-bit key of RC4 in terms of resistance against brute-force attacks.

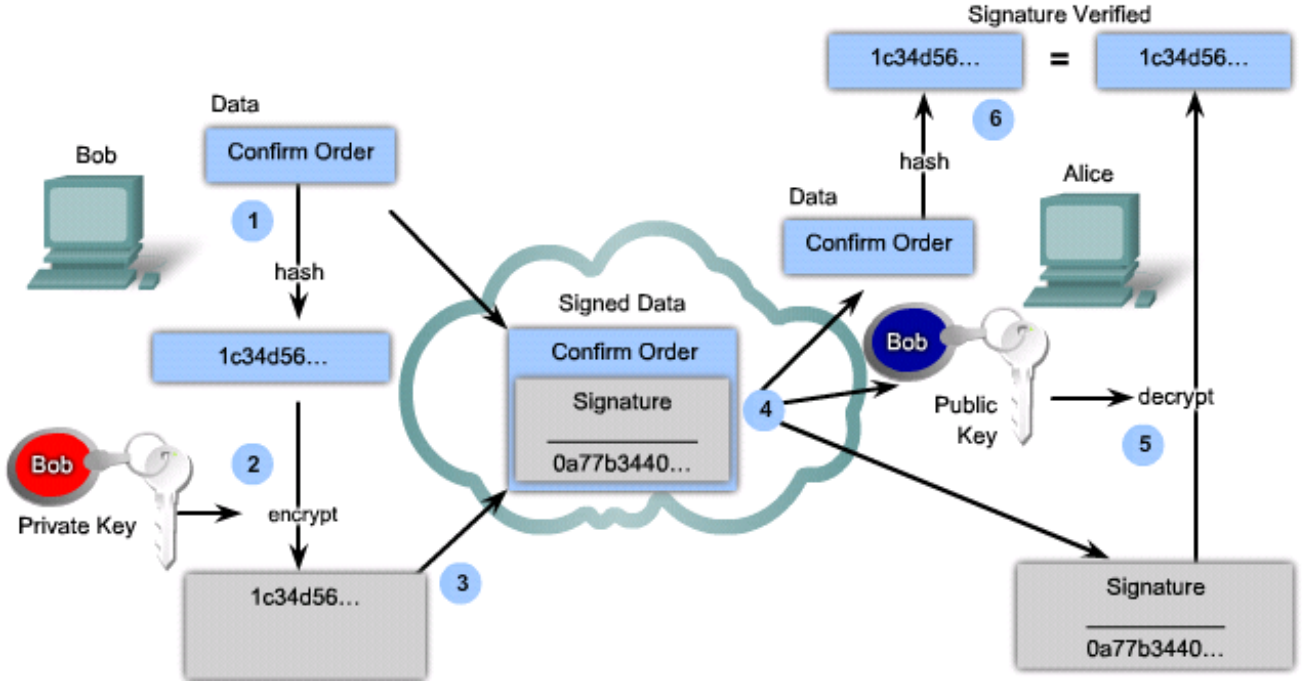
Digital Signatures Security Services

- Authenticity of digitally signed data:
  - Digital signatures authenticate a source, proving that a certain party has seen and signed the data in question.
- Integrity of digitally signed data:
  - Digital signatures guarantee that the data has not changed from the time it was signed.
- Nonrepudiation of the transaction:
  - The recipient can take the data to a third party, and the third party accepts the digital signature as a proof that this data exchange did take place.
  - The signing party cannot repudiate that it has signed the data.

Digital Signatures

- Digital signatures are often used in the following situations:
  - To provide a unique proof of data source, which can only be generated by a single party, such as contract signing in e-commerce environments.
  - To authenticate a user by using the private key of that user and the signature it generates.
  - To prove the authenticity and integrity of PKI certificates.
  - To provide nonrepudiation using a secure timestamp and a trusted time source.
    - Each party has a unique, secret signature key, which is not shared with any other party, making nonrepudiation possible.

Digital Signatures



1. Bob creates a hash of the document.
2. Bob encrypts the hash with the private key.
3. The encrypted hash, known as the signature, is appended to the document.
4. Alice accepts the document with the digital signature and obtains Bob's public key.
5. Alice decrypts the signature using Bob's public key to unveil the assumed hash value.
6. Alice calculates the hash of the received document, without its signature, and compares this hash to the decrypted signature hash and if the hashes match = document is authentic.

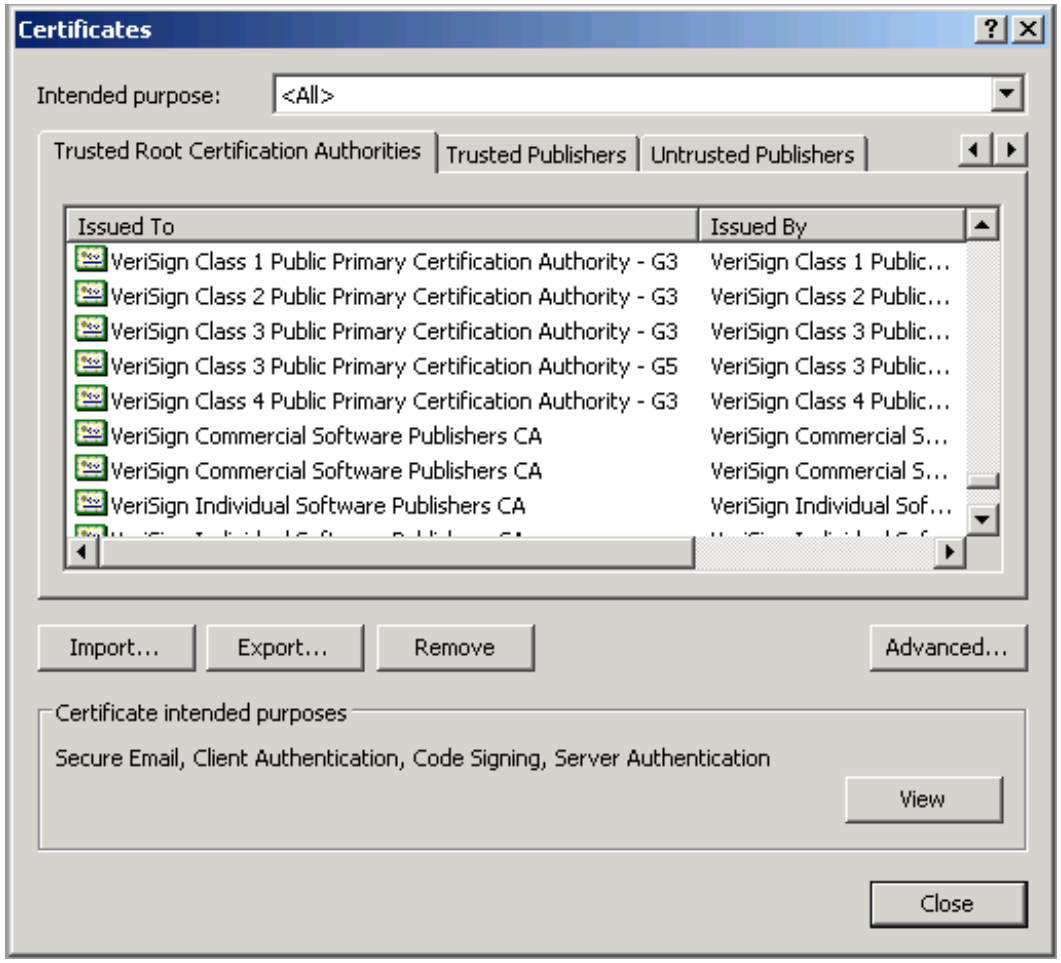
PKI

- PKI is the service framework needed to support large-scale public key-based technologies.
  - Very scalable solutions which is an extremely important authentication solution for VPNs.
- PKI is a set of technical, organizational, and legal components that are needed to establish a system that enables large-scale use of public key cryptography to provide authenticity, confidentiality, integrity, and nonrepudiation services.
  - The PKI framework consists of the hardware, software, people, policies, and procedures needed to create, manage, store, distribute, and revoke digital certificates.

PKI Terms

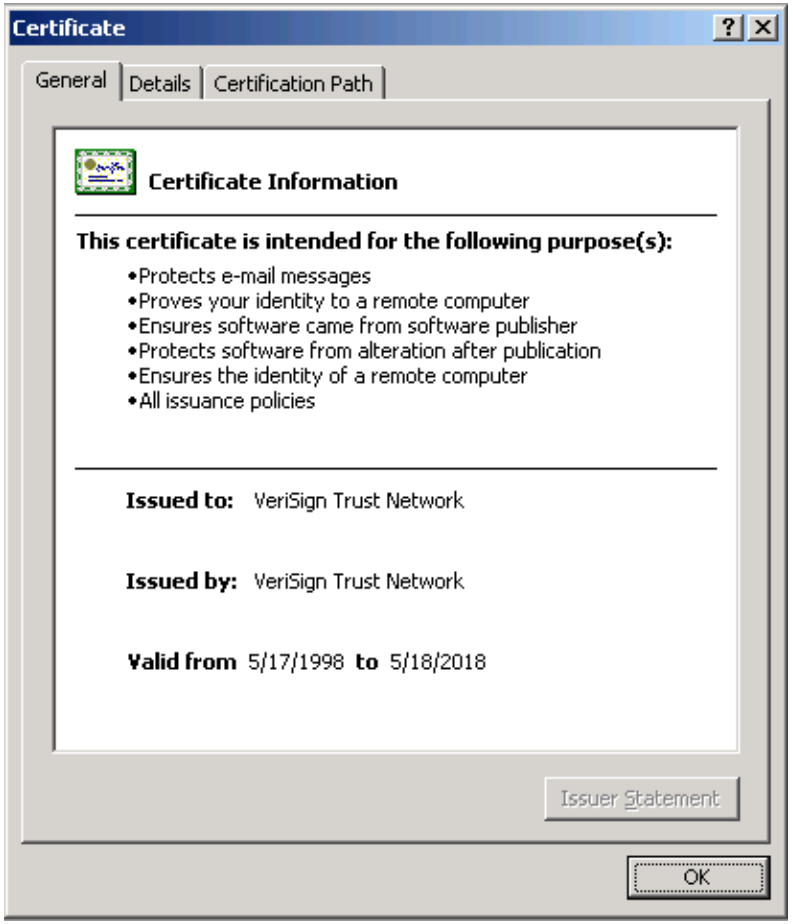
- Certificates:
  - Published public information containing the binding between the names and public keys of entities.
- Certificate authority:
  - A trusted third-party entity that issues certificates.
  - The certificate of a user is always signed by a CA.
  - Every CA also has a certificate containing its public key, signed by itself.
  - This is called a CA certificate or, more properly, a self-signed CA certificate.

PKI Example



X.509v3

- Defines basic PKI formats such as the certificate and certificate revocation list (CRL) format to enable basic interoperability.
- Widely used for years:
  - Secure web servers: SSL and TLS
  - Web browsers: SSL and TLS
  - Email programs: S/MIME
  - IPsec VPN: IKE



PKCS

- The Public-Key Cryptography Standards (PKCS) refers to a group of Public Key Cryptography Standards devised and published by RSA Laboratories.
  - PKCS provides basic interoperability of applications that use public-key cryptography.
  - PKCS defines the low-level formats for the secure exchange of arbitrary data, such as an encrypted piece of data or a signed piece of data.

PKI Standards

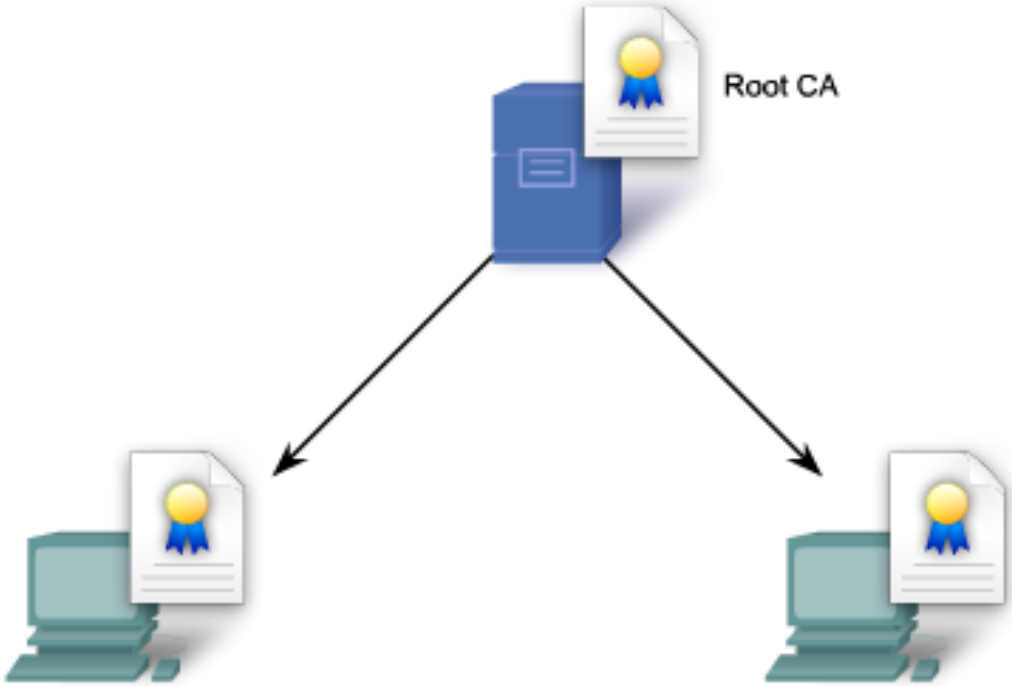
- PKCS #1: RSA Cryptography Standard
- PKCS #3: DH Key Agreement Standard
- PKCS #5: Password-Based Cryptography Standard
- PKCS #6: Extended-Certificate Syntax Standard
- PKCS #7: Cryptographic Message Syntax Standard
- PKCS #8: Private-Key Information Syntax Standard
- PKCS #10: Certification Request Syntax Standard
- PKCS #12: Personal Information Exchange Syntax Standard
- PKCS #13: Elliptic Curve Cryptography Standard
- PKCS #15: Cryptographic Token Information Format Standard

Certificate Authorities

Level of Trust

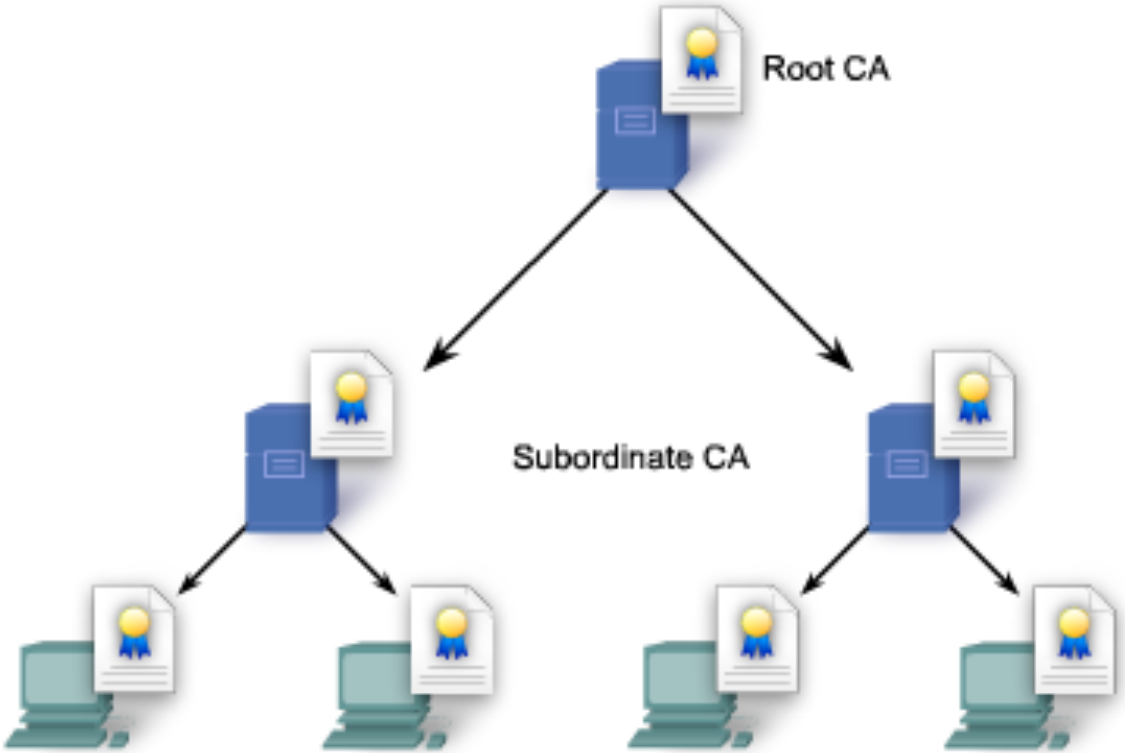
- PKIs can form different topologies of trust, including:
  - Single-root PKI topologies
  - Hierarchical CA topologies
  - Cross-certified CA topologies

Single-Root PKI Topology (Root CA)



- Certificates issued by one CA
- Centralized trust decisions
- Single point of failure

Hierarchical CA Topology

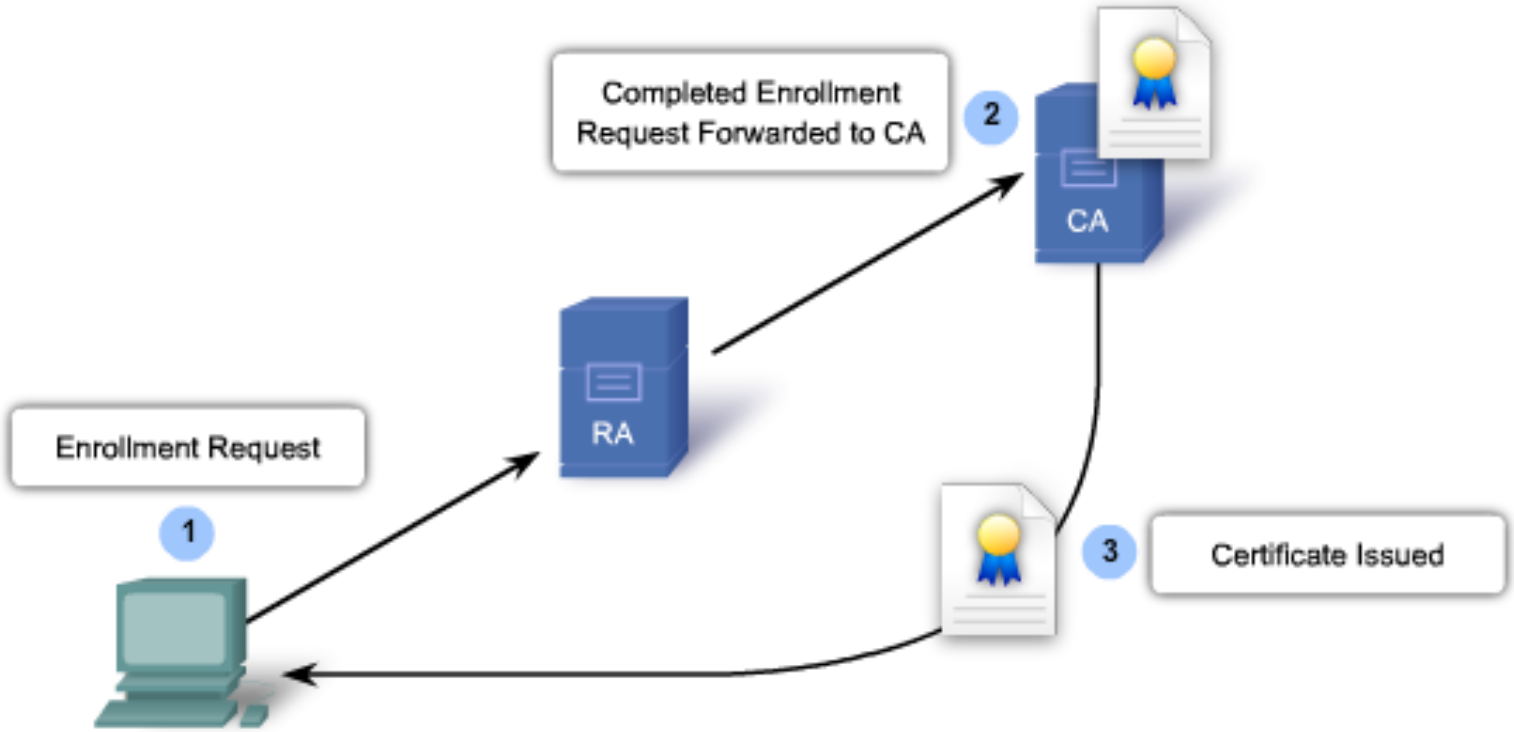


- Delegation and distribution of trust
- Certification paths

PKI Enrollment Process

- The issuing CA may be a:
  - Root CA (the top-level CA in the hierarchy)
  - Subordinate CA
- The PKI might employ registration authorities (RAs) to accept requests for enrollment in the PKI.
  - This reduces the burden on CAs in an environment that supports a large number of certificate transactions or where the CA is offline.

PKI Enrollment Process



1. Hosts will submit certificate requests to the RA.
2. After the Registration Authority adds specific information to the certificate request and the request is approved under the organization's policy, it is forwarded on to the Certification Authority.
3. The CA will sign the certificate request and send it back to the host.

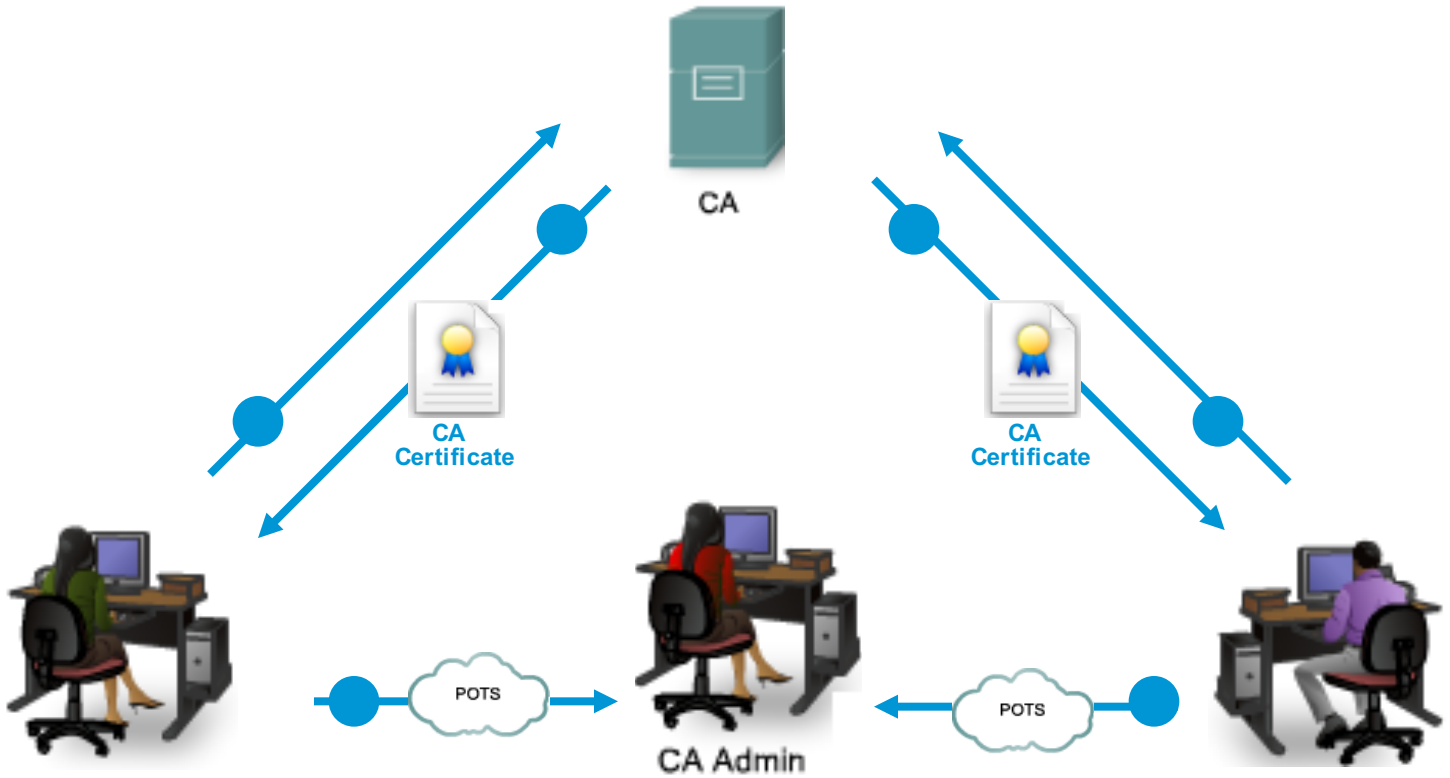
PKI Enrollment Process

- Usually tasks offloaded to an RA:
  - Authentication of users when they enroll with the PKI.
  - Key generation for users that cannot generate their own keys.
  - Distribution of certificates after enrollment.
- Additional tasks include:
  - Verifying user identity.
  - Establishing passwords for certificate management transactions.
  - Submitting enrollment requests to the CA.
  - Handling certificate revocation and re-enrollment.

CA Authentication Procedure

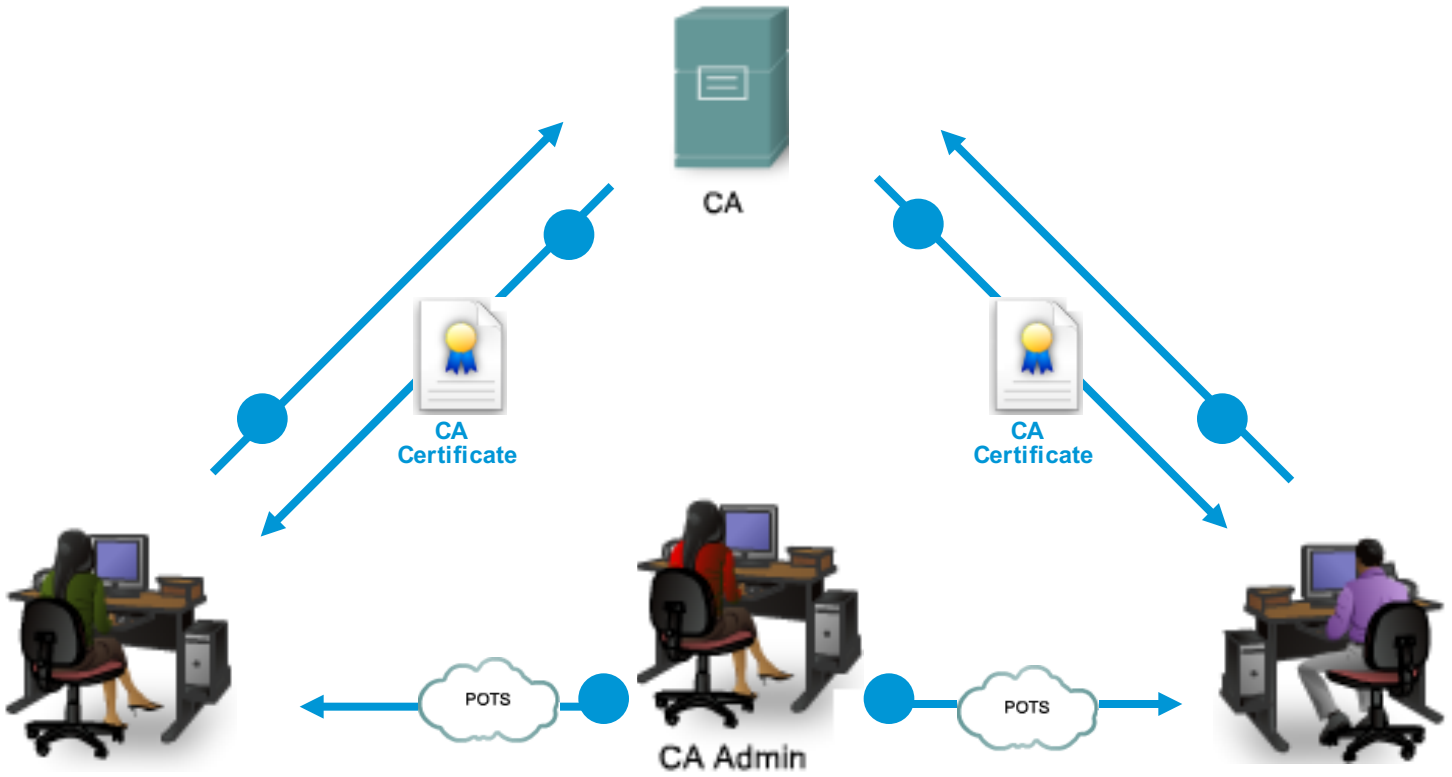
- The first step of the user is to securely obtain a copy of the public key of the CA.
  - The public key verifies all the certificates issued by the CA and is vital for the proper operation of the PKI.
- The public key, called the self-signed certificate, is also distributed in the form of a certificate issued by the CA itself.
- Only a root CA issues self-signed certificates.

CA Authentication Procedure



1. Alice and Bob request the CA certificate that contains the CA public key.
2. Upon receipt of the CA certificate, each system (of Alice and Bob) verifies the validity of the certificate using public key cryptography.
3. Alice and Bob follow up the technical verification done by their system by telephoning the CA administrator and verifying the public key and serial number of the certificate.

CA Authentication Retrieval



1. Alice and Bob forward a certificate request which includes their public key along and information that is encrypted using the public key of the CA.
2. Upon receipt of the certificate requests, the CA administrator telephones Alice and Bob to confirm their submittal and the public key and issues the certificate by adding some additional data to the certificate request, and digitally signing it all.
3. Either the end user manually retrieves the certificate or SCEP automatically retrieves the certificate, and the certificate is installed onto the system.