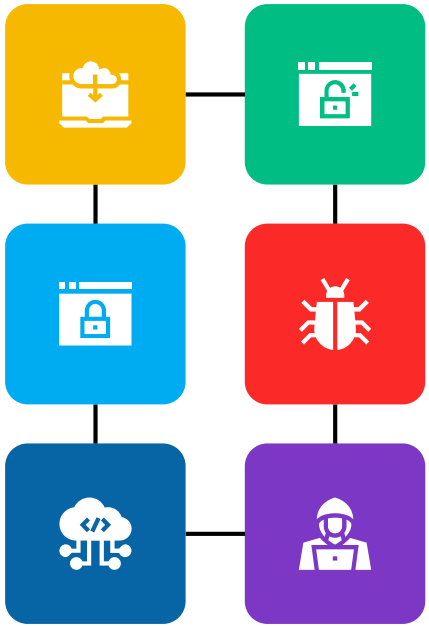
Cryptographic Services and PKI

MNU-2025

Dr. Ahmed Samy

Lecture 06



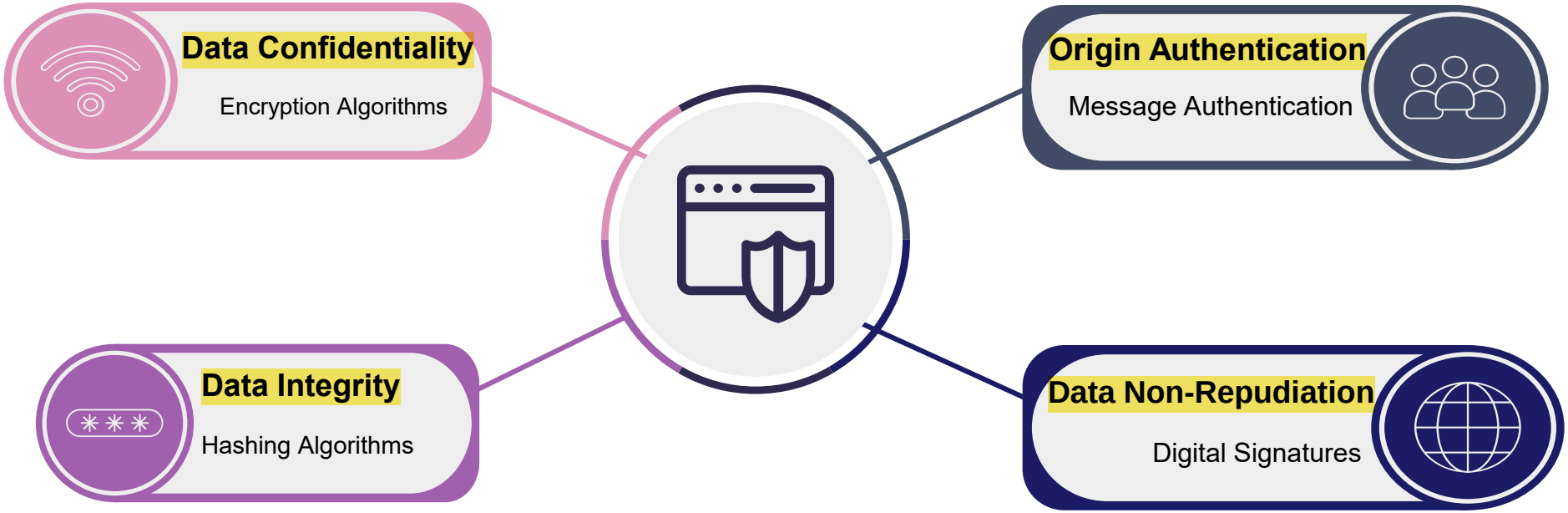
Module Contents

In this module, we will cover the below topics:

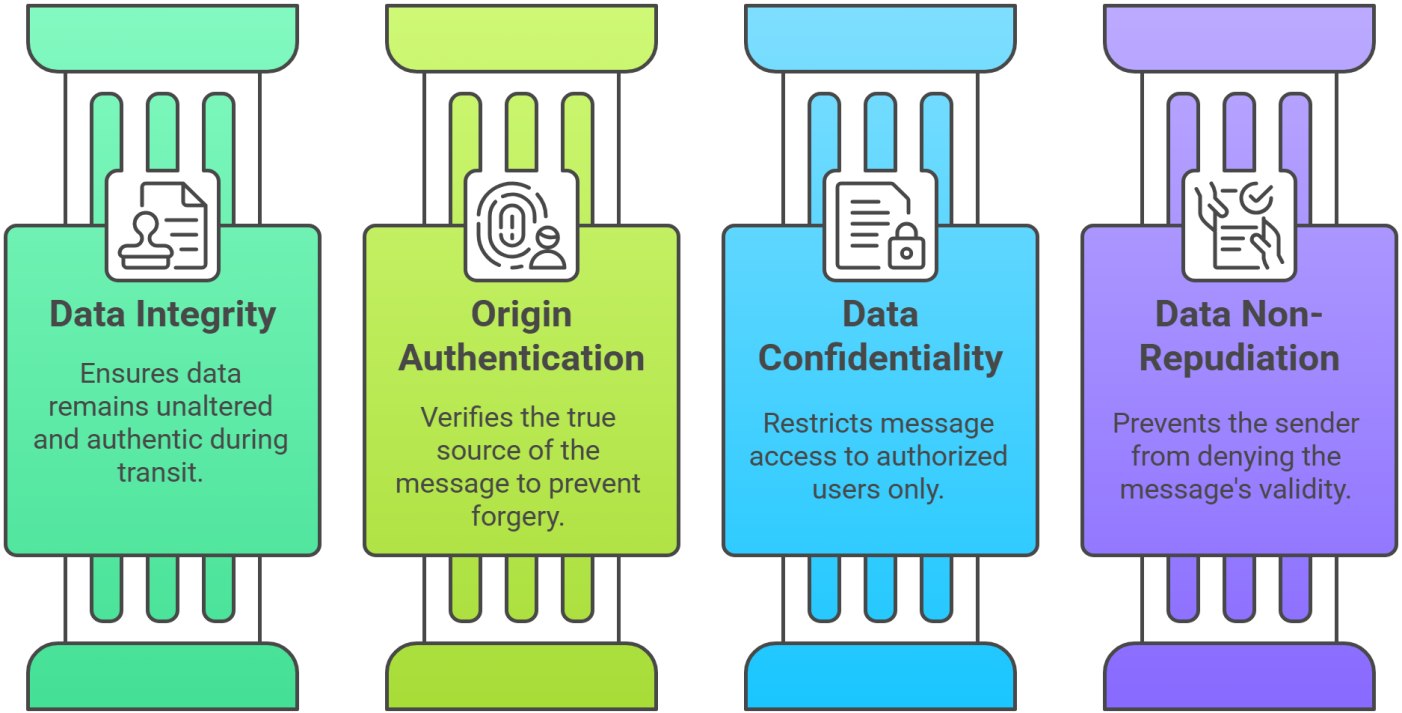
- Secure Communications
- Digital Signatures
- Cryptographic Hash Functions
- Public Key Infrastructure (PKI)
- Symmetric and Asymmetric Encryption
- SSL/TLS for secure web browsing

Secure Communications

Secure Communications



Ensuring Secure and Trustworthy Data Communication

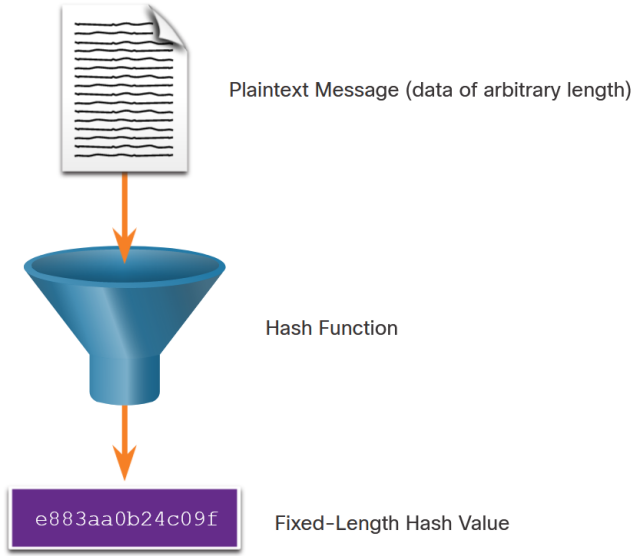


Cryptographic Hash Functions

(Data Integrity, Origin Authentication)

Cryptographic Hash Functions

- Hashes are used to verify and ensure data integrity. Hashing is based on a one-way mathematical function that is relatively easy to compute, but significantly harder to reverse.
- As shown in the figure, a hash function takes a variable block of binary data, called the message, and produces a fixed-length, condensed representation, called the hash.
- The resulting hash is also sometimes called the message digest, digest, or digital fingerprint.
- With hash functions, it is computationally infeasible for two different sets of data to come up with the same hash output.

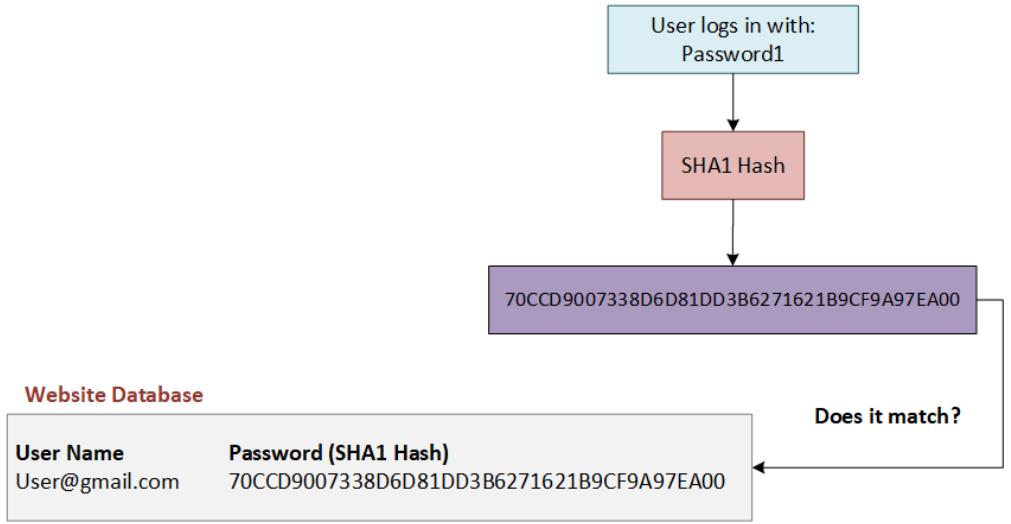


MD5 and SHA1

- Message Digest Algorithm 5 (MD5) is a widely used cryptographic hash function that produces a 128-bit hash value, typically expressed as a 32-character hexadecimal number.
- Secure Hash Algorithm 1 (SHA-1) produces a 160-bit hash, While initially considered stronger than MD5.
- Drawbacks of MD5 and SHA1 Algorithms:
  - They may generate the same hash function for different inputs (**hash collision**). This weakness allows attackers to manipulate data without detection.
  - Hashing is vulnerable to man-in-the-middle attacks and does not provide security to transmitted data.
- However, due to their vulnerabilities, they should no longer be used for cryptographic purposes.

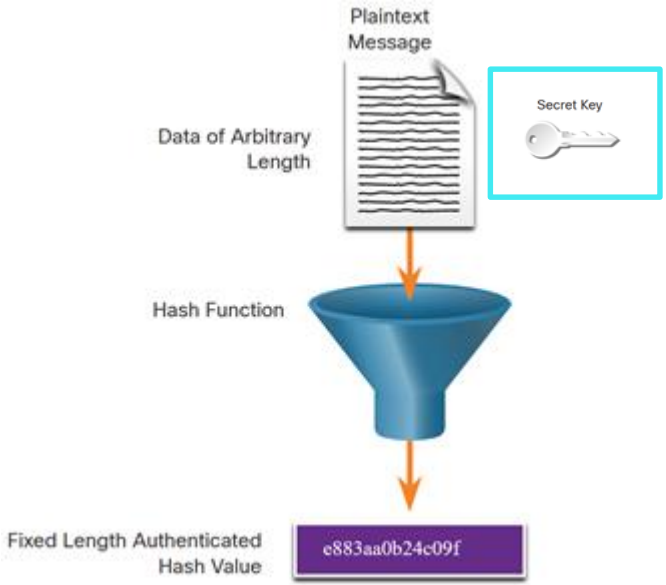
MD5 and SHA1

- **Example**: Websites will commonly store passwords in hashed format as a measure of increased security. When a user logs in, their password is 'hashed' and compared to the hash in the database. If the hashes match, the user is allowed to login.
- Two different passwords may generate the same hashed value, which leads to unauthorized access to the data.
- If the attacker has a copy of hashed password database or perform a MITM attack, he can do brute-force attack to know the plain text passwords.
- To provide integrity against these attacks, origin authentication is also required.



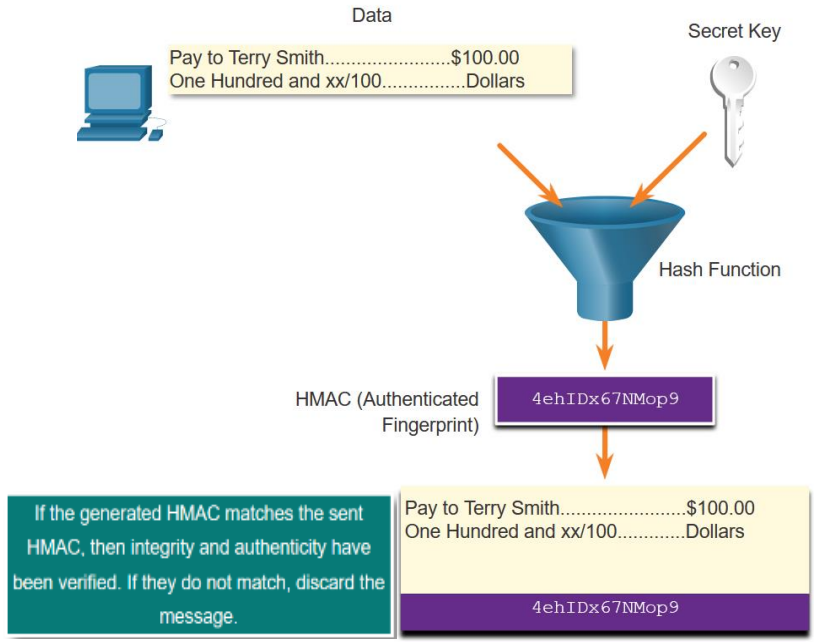
Origin Authentication

- To add origin authentication and integrity assurance, use a keyed-hash message authentication code (HMAC). HMAC uses an additional secret key as input to the hash function.
- As shown in the figure, an HMAC is calculated using any cryptographic algorithm that combines a cryptographic hash function with a secret key. Hash functions are the basis of the protection mechanism of HMACs.
- Only the sender and the receiver know the secret key, and the output of the hash function now depends on the input data and the secret key.
- Only parties who have access to that secret key can compute the digest of an HMAC function. This defeats man-in-the-middle attacks and provides authentication of the data origin.



Origin Authentication

- As shown in the figure, the sending device inputs data into the hashing algorithm and calculates the fixed-length HMAC digest. This authenticated digest is then attached to the message and sent to the receiver.
- The receiving device removes the digest from the message and uses the plaintext message with its secret key as input into the same hashing function.
- If the digest that is calculated by the receiving device is equal to the digest that was sent, the message has not been altered.
- Additionally, the origin of the message is authenticated because only the sender possesses a copy of the shared secret key. The HMAC function has ensured the authenticity of the message.

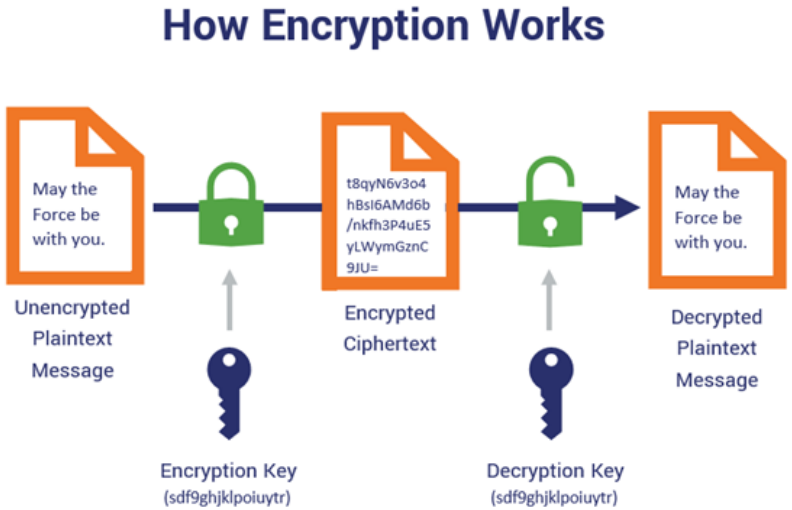


Encryption and Decryption

(Data Confidentiality)

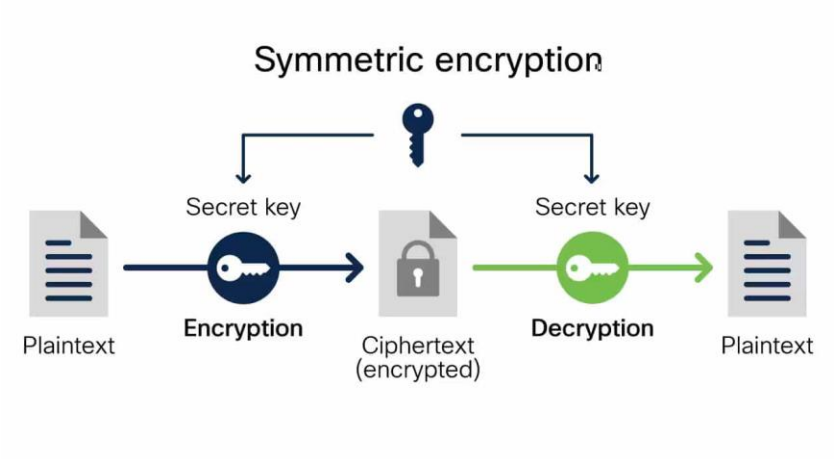
Cryptology

- **Cryptology** is essentially the practice and study of techniques for secure communication.
- **Encryption** is the process of converting readable date (plaintext) into unreadable format (cipher text) using algorithms (ciphers) and keys to protect data confidentiality.
- **Decryption** is the process of converting cipher text back to plaintext using a key or algorithm.
- In decryption, the correct key and algorithm must match the encryption method.
- Only authorized parties with the correct key can decrypt and access the original data.



Symmetric Encryption

- **Symmetric algorithms** use the same pre-shared key to encrypt and decrypt data.
- A pre-shared key, also called a secret key, is known by the sender and receiver before any encrypted communications can take place.
- Symmetric encryption can be created using a block algorithm or a stream algorithm.
- The benefits of symmetric encryption are that it is a very fast form of encryption and is good to use for bulk encryption needs.



Symmetric Encryption

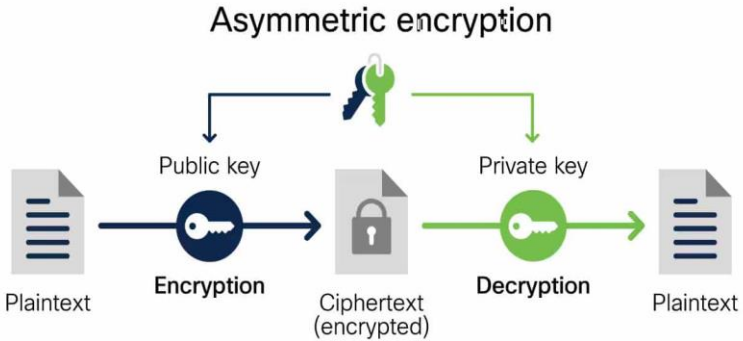
Symmetric Encryption Algorithms	Description
Data Encryption Standard (DES)	This is a legacy symmetric encryption algorithm. It uses a short key length that makes it insecure for most current uses.
3DES (Triple DES)	The is the replacement for DES and repeats the DES algorithm process three times. It should be avoided if possible as it is scheduled to be retired in 2023. If implemented, use very short key lifetimes.
Advanced Encryption Standard (AES)	AES is a popular and recommended symmetric encryption algorithm. It offers combinations of 128-, 192-, or 256-bit keys to encrypt 128, 192, or 256 bit-long data blocks.
Software-Optimized Encryption Algorithm (SEAL)	SEAL is a faster alternative symmetric encryption algorithm to AES. SEAL is a stream cypher that uses a 160-bit encryption key and has a lower impact on the CPU compared to other software-based algorithms.
Rivest ciphers (RC) series algorithms	This algorithm was developed by Ron Rivest. Several variations have been developed, but RC4 was the most prevalent in use. RC4 is a stream cipher that was used to secure web traffic. It has been found to have multiple vulnerabilities which have made it insecure. RC4 should not be used.

Asymmetric Encryption

- **Asymmetric algorithms** are designed so that the key that is used for encryption is different from the key that is used for decryption.

- The decryption key cannot, in any reasonable amount of time, be calculated from the encryption key and vice versa.

- Examples of protocols that use asymmetric key algorithms include:
- Internet Key Exchange (IKE) - This is a fundamental component of IPsec VPNs.
- Secure Socket Layer (SSL) - This is now implemented as IETF standard Transport Layer Security (TLS).
- Secure Shell (SSH) - This protocol provides a secure remote access connection to network devices.



Asymmetric Encryption

Asymmetric Encryption Algorithm	Key Length	Description
Diffie-Hellman (DH)	512, 1024, 2048, 3072, 4096	The Diffie-Hellman algorithm allows two parties to agree on a key that they can use to encrypt messages they want to send to each other. The security of this algorithm 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.
Rivest, Shamir, and Adleman encryption algorithms (RSA)	512 to 2048	RSA is for public-key cryptography that is based on the current difficulty of factoring very large numbers. It is the first algorithm known to be suitable for signing, as well as encryption. It is widely used in electronic commerce protocols and is believed to be secure given sufficiently long keys and the use of up-to-date implementations.
ElGamal	512 - 1024	An asymmetric key encryption algorithm for public-key cryptography which is based on the Diffie-Hellman key agreement. A disadvantage of the ElGamal system is that the encrypted message becomes very big, about twice the size of the original message and for this reason it is only used for small messages such as secret keys.
Elliptic curve techniques	224 or higher	Elliptic curve cryptography can be used to adapt many cryptographic algorithms, such as Diffie-Hellman or ElGamal. The main advantage of elliptic curve cryptography is that the keys can be much smaller.

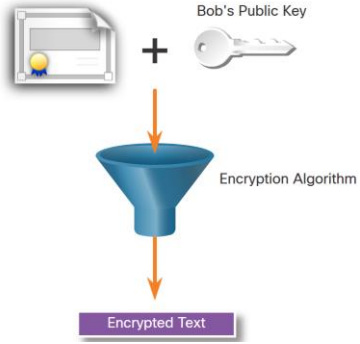
Asymmetric Encryption - Confidentiality

- The process can be summarized using the formula:
- **Public Key (Encrypt) + Private Key (Decrypt) = Confidentiality**

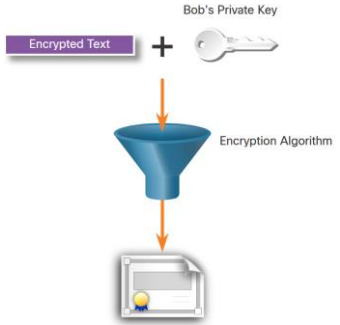
- When the public key is used to encrypt the data, the private key must be used to decrypt the data. Only one host has the private key; therefore, confidentiality is achieved.

Alice requests and obtains Bob's public key.

Alice uses Bob's public key to encrypt a message using an agreed-upon algorithm. Alice sends the encrypted message to Bob.



Bob then uses his private key to decrypt the message. Since Bob is the only one with the private key, Alice's message can only be decrypted by Bob and thus confidentiality is achieved.

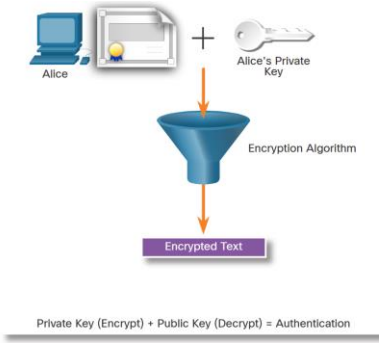


Asymmetric Encryption - Authentication

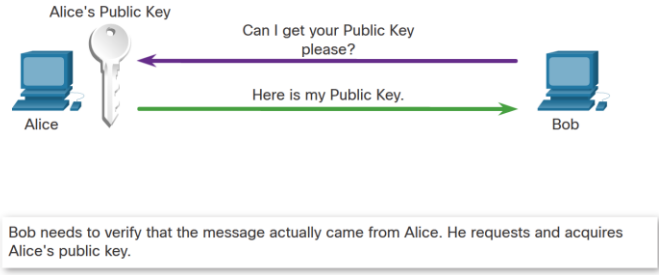
- The authentication objective of asymmetric algorithms is initiated when the encryption process is started with the private key.

- The process can be summarized using the formula:
- **Private Key (Encrypt) + Public Key (Decrypt) = Authentication.**

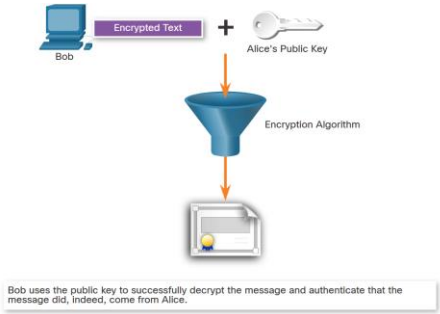
Alice encrypts a message using her private key. Alice sends the encrypted message to Bob. Bob needs to authenticate that the message did indeed come from Alice.



In order to authenticate the message, Bob requests Alice's public key.



Bob uses Alice's public key to decrypt the message.

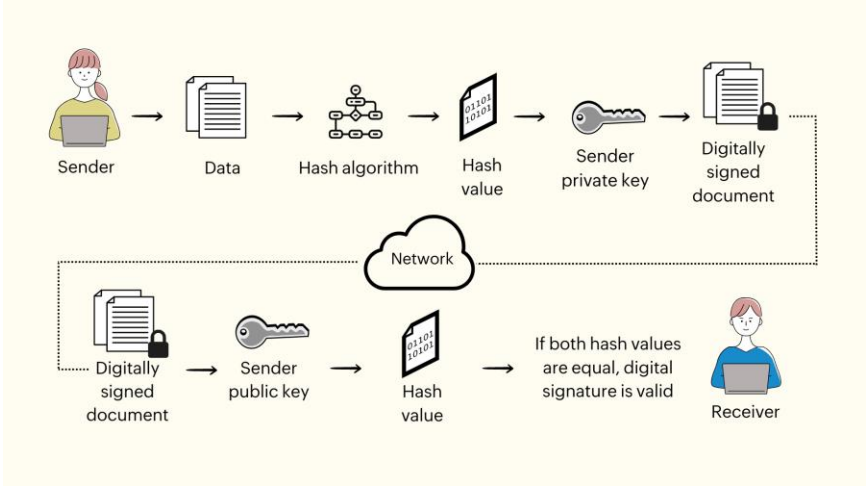


Digital Signatures

(Authenticity, integrity, and nonrepudiation)

Digital Signature Overview

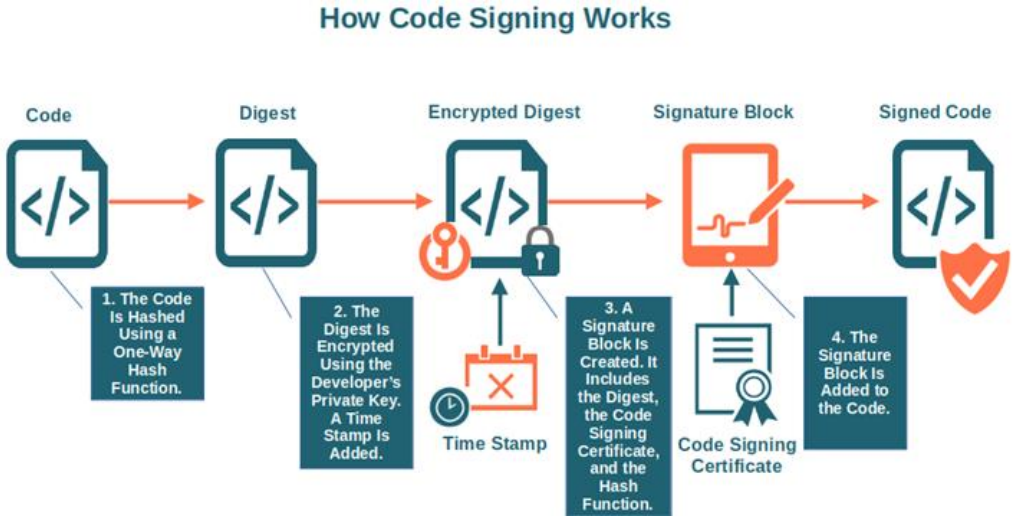
- Digital signatures are a mathematical technique used to provide authenticity, integrity, and nonrepudiation.
- The following are characteristics of digital signatures:
- Authentic - The signature cannot be forged and provides proof that the signer, and no one else, signed the document.
- Unalterable - After a document is signed, it cannot be altered.
- Not reusable - The document signature cannot be transferred to another document.
- Non-repudiated - The signed document is considered to be the same as a physical document.



Digital signatures are commonly used in the following two situations: **code signing** and **digital certificates**.

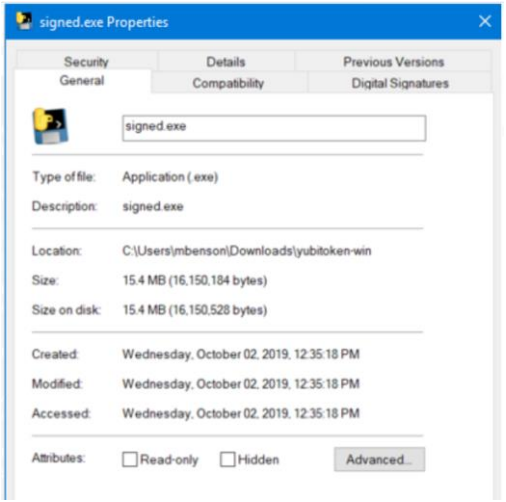
Digital Signature for Code Signing

- Executable files are wrapped in a digitally signed envelope, which allows the end user to verify the signature before installing the software.
- Digitally signing code provides several assurances about the code.
    1. The code is authentic and is actually sourced by the publisher.
    2. The code has not been modified since it left the software publisher.
    3. The publisher undeniably published the code. This provides nonrepudiation of the act of publishing

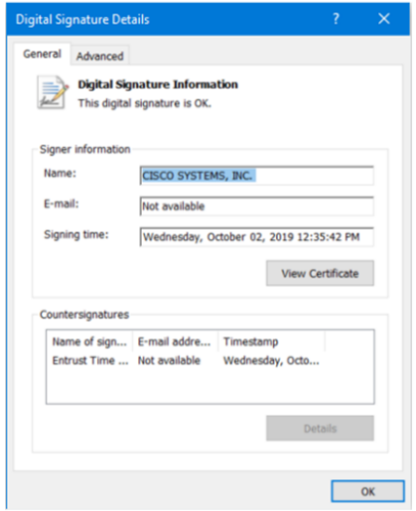


Digital Signature for Code Signing

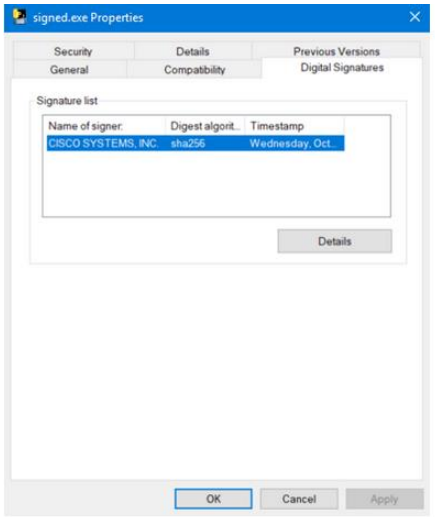
This executable file was downloaded from the internet. The file contains a software tool from Cisco Systems.



Clicking the **Digital Signatures** tab reveals that the file is from a trusted organization, Cisco Systems Inc.

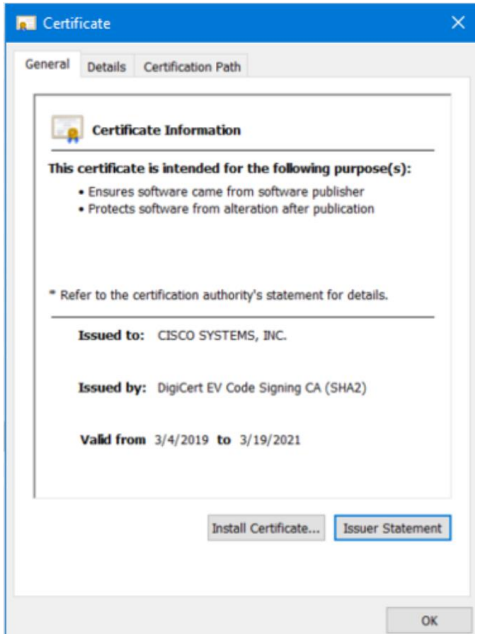


The Digital Signature Details window reveals that the file was signed by Cisco Systems, Inc in October of 2019.

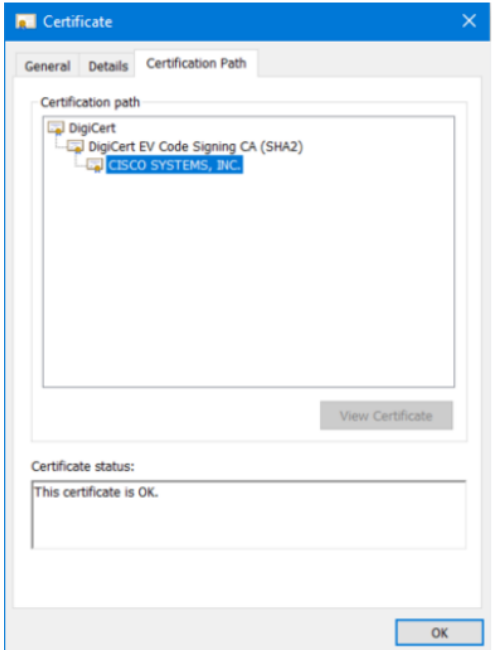


Digital Signature for Code Signing

The **Certificate Information** tab provides the purposes of the certificate, who the certificate was issued to, and who issued the certificate. It also displays the period for which the certificate is valid.

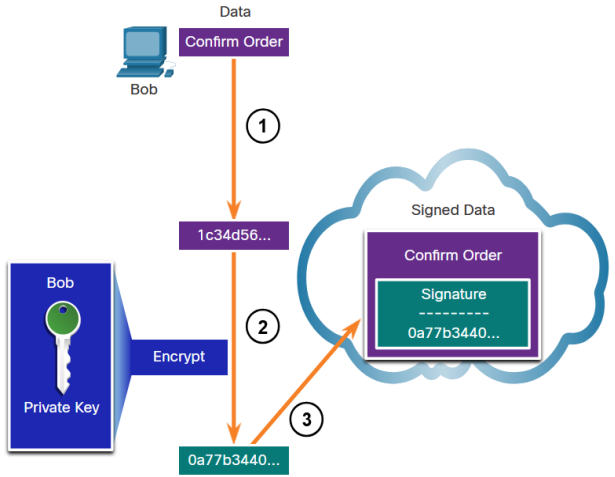


The **Certification Path** tab to see the file was signed by Cisco Systems, as verified to DigiCert.



Digital Signature for Digital Certificates

- **A digital certificate** is used to authenticate and verify that a user who is sending a message is who they claim to be. Digital certificates can also be used to provide confidentiality for the receiver with the means to encrypt a reply.
- This scenario will help you understand how a digital signature is used.
- Bob is confirming an order with Alice. Alice is ordering from Bob's website. Alice has connected with Bob's website, and after the certificate has been verified, the Bob's certificate is stored on Alice's website.
- The certificate contains Bob's public key. The public key is used to verify the Bob's digital signature.



Bob confirms the order and his computer creates a hash of the confirmation. The computer encrypts the hash with Bob's private key. The encrypted hash, which is the digital signature, is appended to the document. The order confirmation is then sent to Alice over the internet.

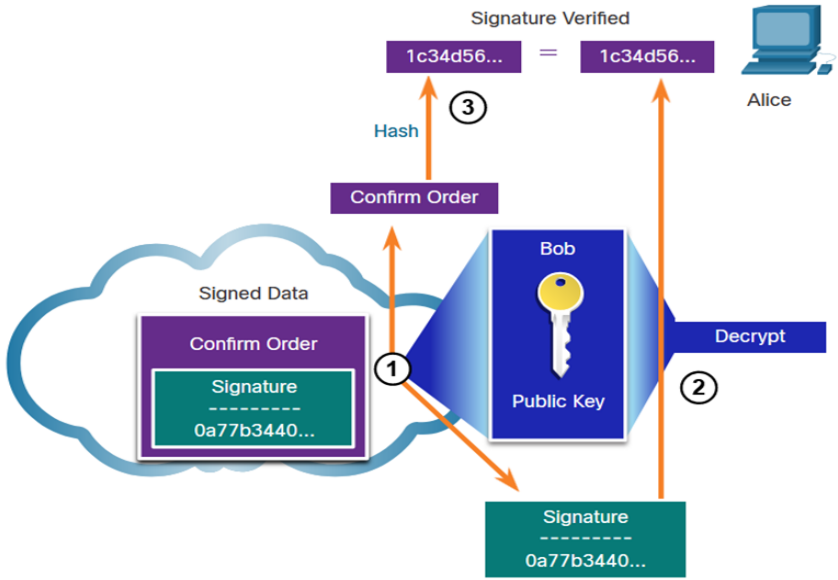
Digital Signature for Digital Certificates

- When Alice receives the digital signature, the following process occurs:

1. Alice's receiving device accepts the order confirmation with the digital signature and obtains Bob's public key.

2. Alice's computer then decrypts the signature using Bob's public key. This step reveals the assumed hash value of the sending device.

3. Alice's computer creates a hash of the received document, without its signature, and compares this hash to the decrypted signature hash. If the hashes match, the document is authentic.

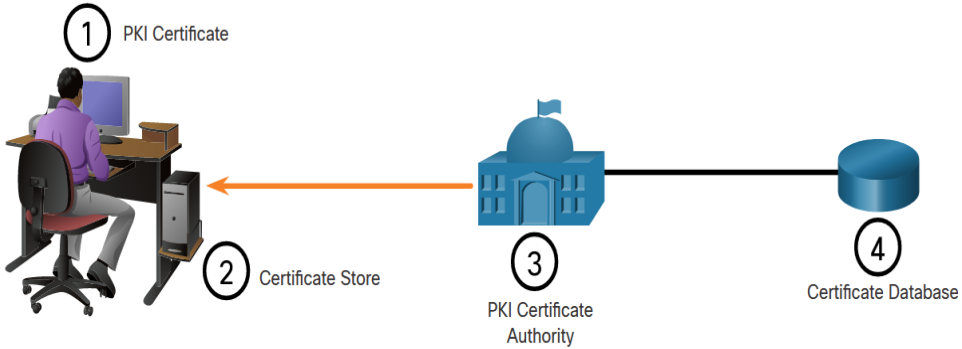


Public Key Infrastructure (PKI)

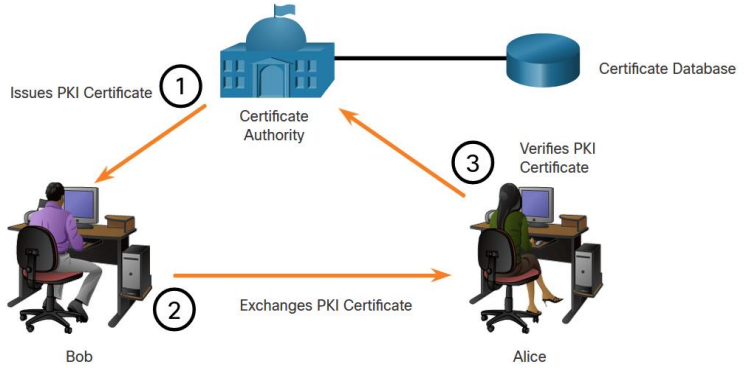
What is PKI?

- Public key infrastructure (PKI) is the set of technology and processes that make up a framework of encryption and manage public keys to protect and authenticate digital communications.
- PKI is built into all web browsers used today, and it helps secure public internet traffic.
- PKI uses cryptographic public keys that are connected to a digital certificate, which authenticates the device or user sending the digital communication.
- Digital certificates are issued by a trusted source, a certificate authority (CA), and act as a type of digital passport to ensure that the sender is who they say they are.
- PKI is used commonly in HTTPS, SSH, and email signing.

Public Key Management



1. PKI certificates contain an entity's or individual's public key, its purpose, the certificate authority (CA) that validated and issued the certificate, the date range during which the certificate is valid, and the algorithm used to create the signature.
2. The certificate store resides on a local computer and stores issued certificates and private keys.
3. The PKI Certificate of Authority (CA) is a trusted third party that issues PKI certificates to entities and individuals after verifying their identity. It signs these certificates using its private key.
4. The certificate database stores all certificates approved by the CA.



1. **Issues PKI Certificate**. Bob initially requests a certificate from the CA. The CA authenticates Bob and stores Bob's PKI certificate in the certificate database.
2. **Exchanges PKI Certificate**. Bob communicates with Alice using his PKI certificate.
3. **Verifies PKI Certificate**. Alice communicates with the trusted CA using the CA's public key. The CA refers to the certificate database to validate Bob's PKI certificate.

The PKI Authorities System

- CAs, especially those that are outsourced, issue certificates based on classes which determine how trusted a certificate is.
- The table provides a description of the classes.

Class	Description
0	Used for testing in situations in which no checks have been performed.
1	Used by individuals who require verification of email.
2	Used by organizations for which proof of identity is required.
3	Used for servers and software signing. Independent verification and checking of identity and authority is done by the certificate authority.
4	Used for online business transactions between companies.
5	Used for private organizations or government security.

PKI Work Flow

1. Key Pair Generation:
   - A pair of cryptographic keys (public and private) is created using algorithms like RSA or ECC.
   - The public key is shared openly, while the private key remains confidential.

2. Certificate Signing Request (CSR):
   - The entity (user, device, or organization) requesting a digital certificate generates a CSR.
   - The CSR includes the public key and identity information (e.g., domain name, organization).

3. Certificate Authority (CA):
   - The CSR is sent to a trusted Certificate Authority (CA).
   - The CA verifies the requester's identity and issues a digital certificate, which binds the public key to the entity.

PKI Work Flow

4. Digital Certificate:
   - The digital certificate contains: Public key, Identity information, Certificate Authority details, Expiration date, A digital signature from the CA.

5. Trust Establishment:
   - A root certificate (issued by the CA) is trusted by all parties.
   - Devices and applications trust certificates issued by the CA if they recognize the CA's root certificate.

6. Secure Communication:
   - The public key is used to encrypt data sent to the entity.
   - The entity decrypts it using its private key, ensuring secure communication.

7. Certificate Revocation and Renewal:
   - If a private key is compromised or the certificate expires, it can be revoked by the CA and added to a Certificate Revocation List (CRL).

SSL/TLS for Web Browsing

What is Transport Layer Security (TLS)?

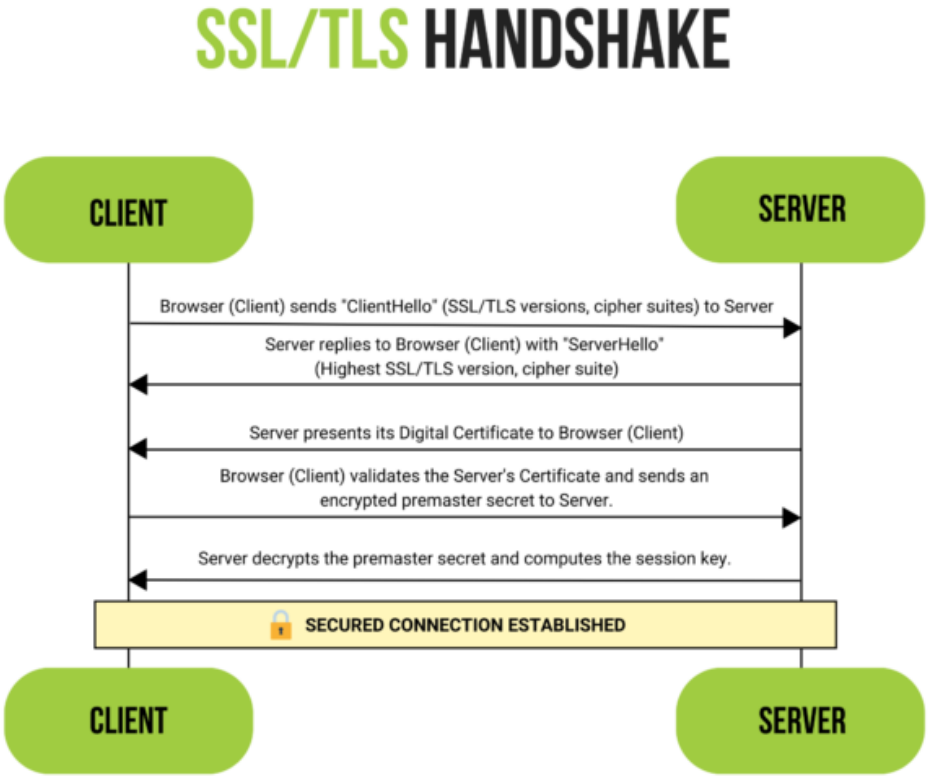
- HTTPS is an implementation of TLS encryption on top of the HTTP protocol, which is used by all websites as well as some other web services. Any website that uses HTTPS is therefore employing TLS encryption.
- SSL and TLS are cryptographic protocols that provide secure communication over a network to encrypting data sent between a website and a browser. All versions of SSL are now **deprecated.**
- Transport Layer Security, or TLS, is a widely adopted security protocol designed to facilitate privacy and data security for communications over the Internet.
- A primary use case of TLS is encrypting the communication between web applications and servers, such as web browsers loading a website. It also ensures integrity and authentication.
- The first version of the TLS protocol was published in 1999. The most recent version is TLS 1.3, which was published in 2018.

How does TLS Work?

A TLS connection is initiated using a sequence known as the TLS handshake. When a user navigates to a website that uses TLS, the TLS handshake begins between the user's device and the web server.

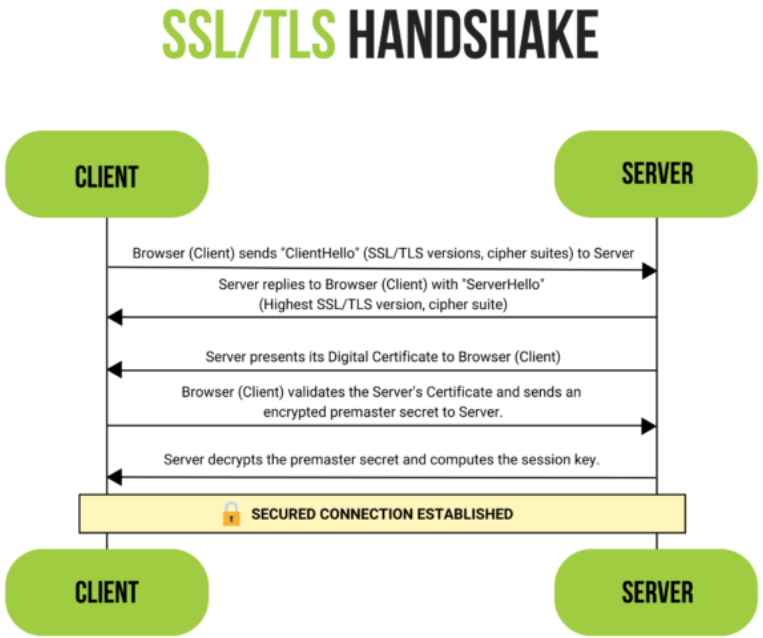
During the TLS handshake, the user's device and the web server:

- Specify which version of TLS (TLS 1.0, 1.2, 1.3, etc.) they will use.
- Decide on which cipher suites they will use.
- Authenticate the identity of the server using the server's TLS certificate.
- Generate session keys for encrypting messages between them after the handshake is complete.



How does TLS Work?

- The TLS handshake establishes a cipher suite for each communication session.
- The cipher suite is a set of algorithms that specifies details such as which shared encryption keys, or session keys, will be used for that particular session.
- The handshake also handles authentication, which usually consists of the server proving its identity to the client. This is done using public keys. Anyone with the public key can unscramble the data encrypted with the server's private key to ensure its authenticity
- Once data is encrypted and authenticated, it is then signed with a message authentication code (MAC). The recipient can then verify the MAC to ensure the integrity of the data.



Thanks!

Do you have any questions?