Practical Encryption

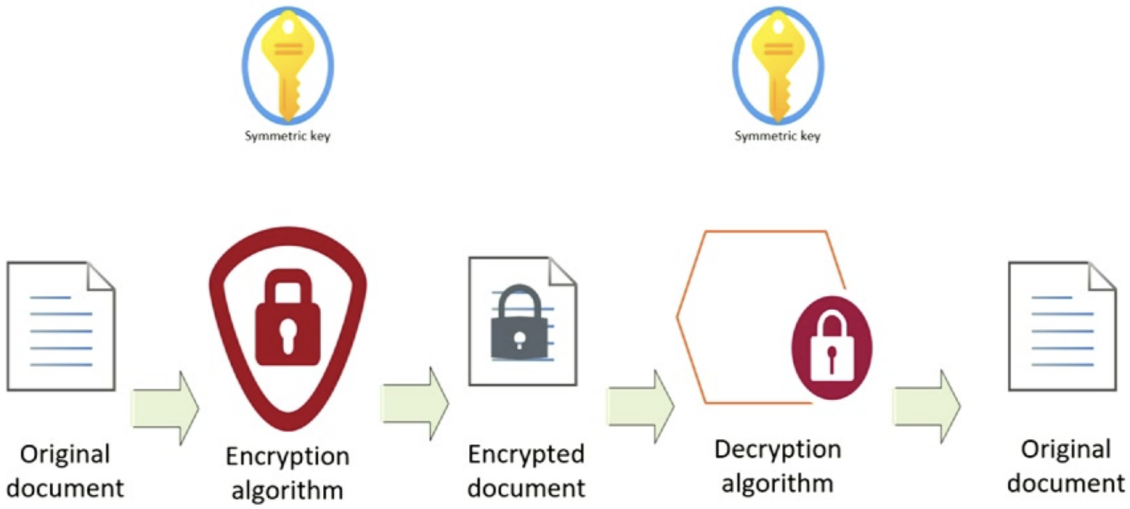
Non-Repudiation Definition:

Nonrepudiation ensures that no party can deny that it sent or received a message via encryption and/or digital signatures or approved some information. It also cannot deny the authenticity of its signature on a document.

Types of Cryptographic Algorithms

Symmetric Key Encryption

- imagine you have to send a document over a link
- you then generate a key and use it to encrypt that document - creating a **cipher text (**encrypted plain text)
- you then send this document to the other party/other side and then by using the **same symmetric key** they can decrypt the document and see the text contained within:



Advantages & Disadvantages of Symmetric keys:

Advantages

- this algorithm is **efficient** and **fast**: suitable for encrypting and decrypting large amounts of data (e.g. streaming data)
- it is fairly **simple,** and uses less computational steps and effort and power
- More suitable for embedded systems, Internet of Things, and industrial devices.
    - in some cases this type of encryption is used for resource constrained environments

Disadvantages

- **Limited Authentication:** Only the person with the key can decrypt the message.
- **Key Management:** Harder to revoke the keys in large environments
    - E.g. if these are installed on many systems in an organisation and you need to revoke the key, you would have to go to each file system and erase the key and then issue another one
- It has a **single point of failure:** you have to keep you symmetric key **safe and secure**
    - one key used for encrypts and decrypts
    - if compromised, an attacker can use the key to decipher your documents

Asymmetric Key Encryption AKA Public Key Encryption

- you have two parties (Alice and Bob) who wish to communicate with each other

- Alice will create a **key pair:** a **public key** and a **private key** mathematically linked (dont need to understand how)
- Alice wants to send a message to Bob so she uses Bob's public key to encrypt the document
- she then sends the document over a link to Bob
- Bob can then use his **private key** to decrypt the document and see the original content

Advantages & Disadvantages of Asymmetric keys/Public Keys:

Advantages

- **Distrubution:** only uses public keys, relyijng on trust in the authenticity of public keys

Disadvantages

- **Slow Encryption Algorithm:** For larger messages it is typically

- **Non-Repudiation:** using certifications means a user cannot deny having performed a transaction, non-repudiation authenticates the identity of a user who is sending messages
- **Authentication:** Third parties can validate the certificates sent with public keys

slower and more computationally expenstive

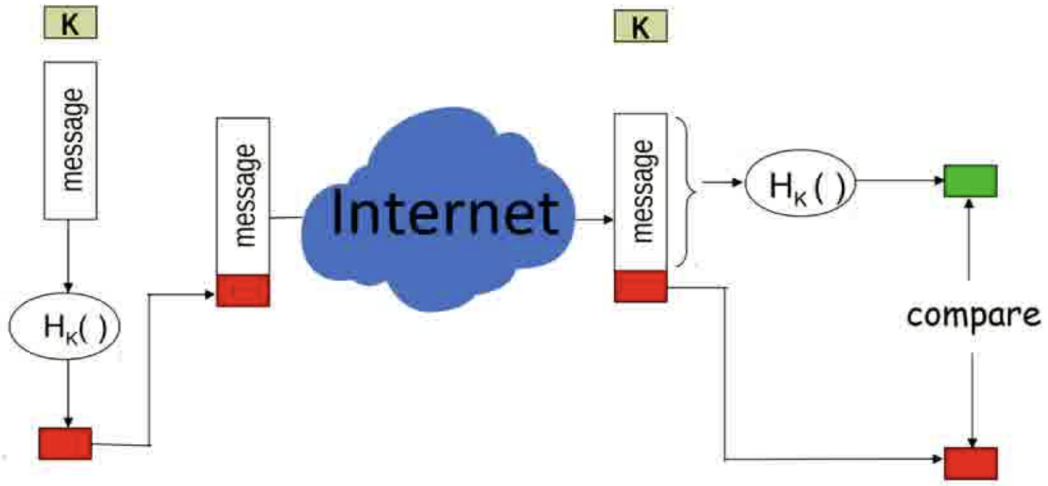
- **Not useful for large data:** used for key exchange protocols
- **Large Key Size:** Produced keys will be significantly larger than *symmetric* keys (increased bandwidth and storage requirements)

Cryptographic Hash Functions

- these are called **checksum algorithms**
  - for example MD5 (128 bits) and SHA1 (160 bits)
    - MD5 is not recommended as it is believed to be nearly broken and therefor unsafe to use
- hash functions convert data into a *fixed-size* checksum which is known as **Message Digest**
- Any change to the data itself will create a different output
  - if the attacker changes data then the output will be able to reflect this tampering (if the two checksums dont match)
- The output reveals **no information** about the data
- It is **impossible** to find two inputs that produce the same checksum
- It is practically **impossible** to algorithmically reconstruct the input (it is a **one-way algorithm)**
- The output is **twice as large** as the *symmetric key* algorithm

Message Authentication Codes (MACs)

- this algorithm provides a means to send a message over the internet
- We have a private key (the K number)
- with K we encrypt the message which also has a **hash value**
- we can then send this encrypted message over the internet and the other party can check the hash value of the message, if it is not the same as is expected, then the message is compromised
    - They also require the key to decrypt the message
    - Key = decrypt
    - Hash function = validates the message hasnt been altered



K = secret key, MAC is supported in SSL and in OpenSSL as only HMAC
Ensure integrity for the "***message digest***"

Encryption Algorithms: Advantages & Disadvantages:

Data Encryption Standard (DES)

- one of the first widely used but is no longer considered secure because of its **small key size**

Triple DES (3DES)

- more effective to DES as it applies the DES algorithm **three times** (creates a larger 168-bit key) but is much more **time consuming**

Advanced Encryption Standard (AES)

- strong security, efficient in terms of computational resources and memory usaged
- flexible as it supports many key lengths
- However, **vulnerable** to side-channel attacks

RSA (Rivest-Shamir-Aldleman)

- **most widely used currently** to secure key exchange, digital signatures and is the base of public key encryption
- it has built-in mechanisms for non-repudiation through digital signatures
- To resist attacks use it to create large RSA keys (smaller keys is also possible), it is however more time for encryption

Elliptic Curve Cryptography (ECC)

- strong security with **small key sizes**
- efficient in terms of bandwidth and computational resources
    - it is used in many IOT devices because of this and its small key size
- it has implementation complexity (**difficult to implement** correctly) and if dont wrong, will compromise the device

How to Select Key Lengths

If you are a developer and you need to choose an encryption algorithm, how do you choose a specific one?

First consider which algortithm you need

- length of keys in public keys are large numbers comparable to symmetric algorithms
    - a 512-bit key is too weak, larger 2,048 bit keys may be too slow

Consider your security requirements

- the sensitivity of the data being encrypted
- consider the **lifespan** of your data (how long you need to protect it for)
    - e.g. a couple of days may only requirer smaller keys

Compliance with Regulation is important

- e.g. in organisation there are audits which can enforce specific key lengths (e.g. ISO 27001)

Balanacing security with computational power and slower performance required for long keys

---

Overview of SSL/TLS

SSL (Secure Sockets Layer) and TLS (Transport Layer Security) are cryptographic protocols designed to provide secure communication over a computer network. They are commonly used on the internet to secure data exchanges between a client (such as a web browser) and a server.

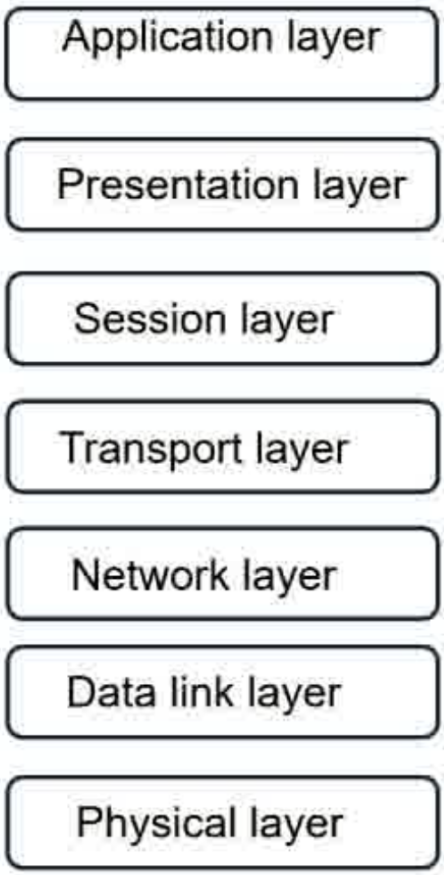
Here’s how they work and what they do:

1. **Encryption:** SSL/TLS encrypts the data exchanged, preventing unauthorized parties from reading it while it is transmitted. This is crucial for protecting sensitive information like credit card numbers, login credentials, and personal information.
2. **Authentication:** The protocol uses certificates to authenticate the identity of parties involved in the communication. This helps ensure that data is sent to and received from the intended server and not an imposter.
3. **Integrity:** SSL/TLS provides mechanisms to verify that the data has not been altered during transmission. This integrity check prevents data tampering.

Historically, SSL was the earlier protocol developed by Netscape in the 1990s. TLS is its successor, standardized by the Internet Engineering Task Force (IETF) and considered more secure. **SSL has been phased out due to vulnerabilities**, and **TLS is now the protocol widely adopted and recommended for secure communications.**

In everyday use, the terms SSL and TLS are often used interchangeably, though technically, TLS is the correct term for the protocols in use today. For example, when you visit a website with HTTPS in the URL, that's using TLS, even though people might still refer to it as using SSL.

Network Layers - Open Systems Interconnection (OSI) Model



The Open Systems Interconnection (OSI) model is a conceptual framework used to understand network interactions in seven distinct layers. Each layer serves a specific function and communicates with the layers directly above and below it. Here's a breakdown of each layer:

1. **Physical Layer (Layer 1)**
    - **Function**: Transmits raw bits over a communication channel.
    - **Components**: Hardware (cables, switches, routers, modems, etc.), physical media (optical fibers, copper wires), and electrical signals.
    - **Purpose**: To establish, maintain, and deactivate physical connections.
2. **Data Link Layer (Layer 2)**
    - **Function**: Handles node-to-node data transfer, error correction from the physical layer, flow control, and frame synchronization.
    - **Components**: Network interface cards (NICs), bridges, switches.

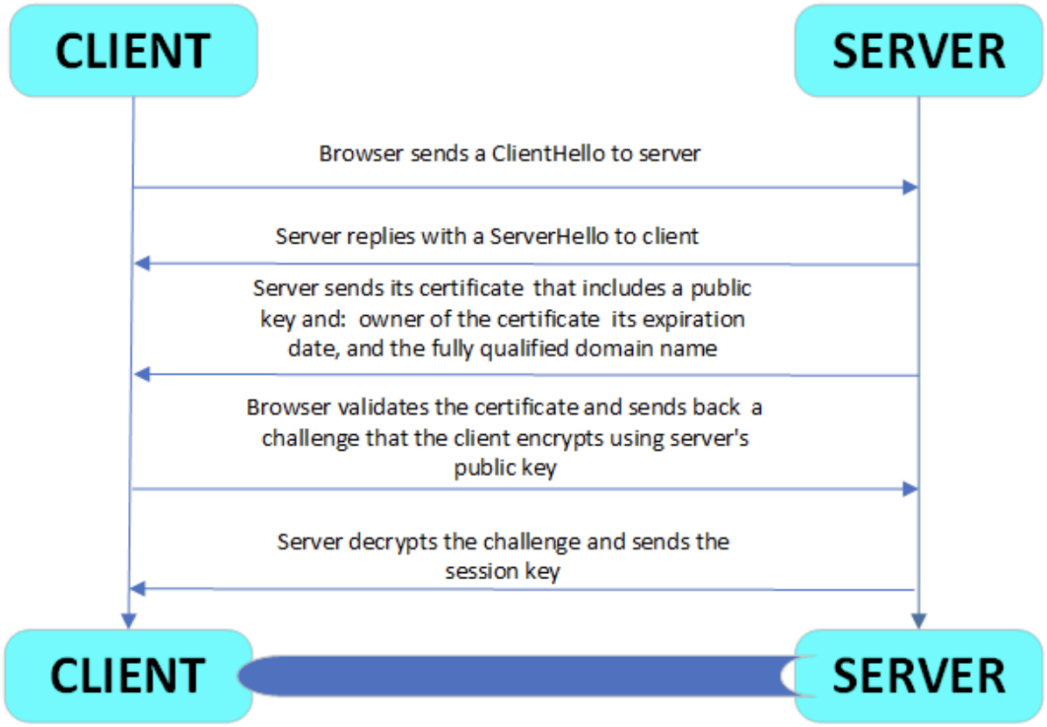
- **Purpose**: To organize data into frames, handle MAC addressing, and manage error checking and frame control.

3. **Network Layer (Layer 3)**
    - **Function**: Determines how data is sent to the receiver from the sender.
    - **Components**: Routers, layer 3 switches.
    - **Purpose**: To perform routing of packets by logical addressing (IP addresses) and manage traffic.
4. **Transport Layer (Layer 4)**
    - **Function**: Provides transparent transfer of data between end systems.
    - **Components**: Transmission Control Protocol (TCP), User Datagram Protocol (UDP).
    - **Purpose**: To handle data transfer, error recovery, and flow control between host-to-host.
5. **Session Layer (Layer 5)**
    - **Function**: Manages sessions between applications.
    - **Components**: APIs, sockets that keep different applications' data separate.
    - **Purpose**: To establish, manage, and terminate connections between applications.
6. **Presentation Layer (Layer 6)**
    - **Function**: Ensures that the data is in a usable format and is where data encryption occurs.
    - **Components**: Encryption, compression, data conversion, and file format translation.
    - **Purpose**: To translate data between the application layer and the network.
7. **Application Layer (Layer 7)**
    - **Function**: Closest to the end user, both the application layer and the end user interact directly with the software application.

- **Components**: Web browsers, email clients, and other applications that require network access.
- **Purpose**: To provide network services to the applications of the user.

Each layer in the OSI model only communicates with its adjacent layers, and each has specific protocols that standardize communications. The model helps in understanding and troubleshooting network issues by segmenting different functions that occur at each level.

SSL/TLS Handshakes

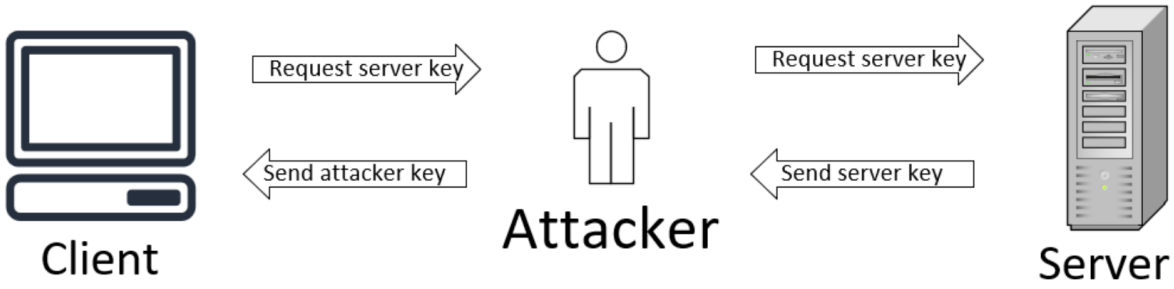


- once the client has the **session key**, the **communication is established**

This communication is vulnerable to attacks however...

Man in the Middle Attack for SSL/TLS

- The attacker needs a copy of the certificate and a private key of the server to masquerade as a known server
- attacker can sniff the server messages and present the attackers certificate
- the forged certificate can look legitimate
- MitM attack allows the attacker to eavesdrop of all communication



Limitations of SSL/TLS

- SSL is slower than a HTTP connection due to the overhead of handshakes and public key exchanges
- overhead in encrypting and decrypting data
- doesnt work with other transport layer protocols which are not connection oriented, such as UDP
- SSL has **no** support for non-repudiation (e.g. say of one party attaches a messaged with invalid signature)
- SSL doesnt protect against flaws in the application itself, e.g. if the engineer of the server has creeated a flaw in its security ssl wont protect against this

---

OpenSSL Library

- cryptographic library able to implement many encryption algorithms such as DES, AES, and RSE
- supports most popular algorithms for symmetric and public keys and hash algorithms
- it has a feature of psuedorandom number generators to increase the uniqueness (entropy) of a key

OpenSSL File Types

`.KEY`

- a file containing the private key

`.CSR` **(**Certificate Signing Request)

- a file that is sent to the certificate authority with information and needs the private key

`.CRT` (Certificate)

- the is the secutiry certificate file, created to establish secure connections

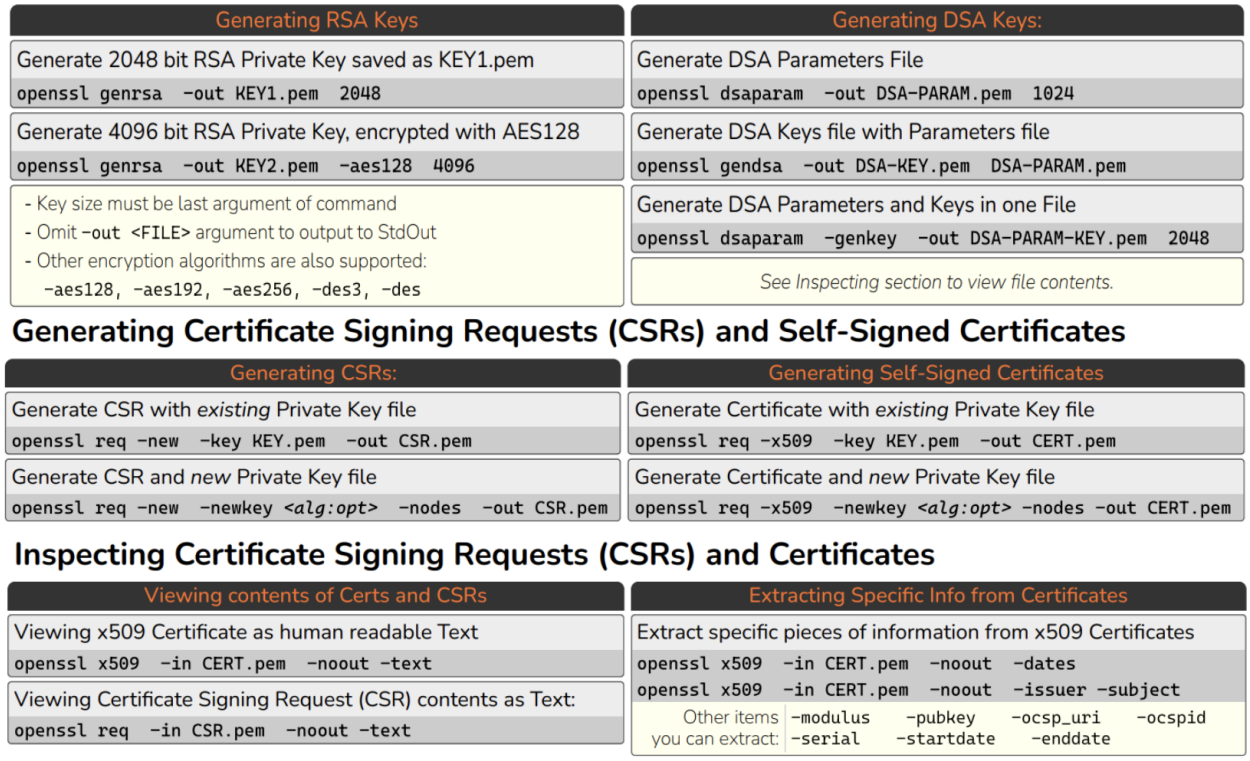
`.PEM` (Privacy Enhanced Mail)

- may include the publci certificate or an entuire certificate chain (public key → private key → certificate)

`.CRL` (Certificate Revocation List)

- a file used to de-authorise certificates (retract publci keys) before expiration

Generating Public and Private Keys



Known SSL/TLS Attacks (CVEs?)

Downgrade Attack

- the attacker tries to make the system use an older, more insecure, version protocol e.g. a cryptographic algorithm with known vulnerabilities

CRIME Attack (Compression ratio Info-leak Made Easy)

- the attacker uses a vulnerability that exploits the use of data compression in HTTPS connections
- CRIME attacks leverage the way data is compressed before being encrypted and sent over the network. The core idea is that if an attacker can inject a known piece of data into the communication (like a cookie in a header of an

HTTP request), they can use the changes in the size of the compressed output to infer critical information about the content of the data being sent.

BREACH attack (Browser Reconnaissance and Exfiltration via Adaptive Compression of Hypertext)

- is a security exploit against HTTP that, like CRIME, uses compression to enable unauthorized reading of secure data. Discovered by researchers Angelo Prado, Neal Harris, and Yoel Gluck in 2013, BREACH exploits the way HTTP responses are compressed, thereby enabling an attacker to read encrypted data (such as session cookies, CSRF tokens, OAuth tokens, etc.) without actually decrypting them.
- the attacker uses a vulnerability to view encrypted traffic and force the victim to send HTTP requests to a vulnerable server

How BREACH Attacks Work

BREACH, unlike CRIME which targets requests, attacks the responses in an HTTP communication using SSL/TLS. Here's a step-by-step explanation of how a BREACH attack is conducted:

1. **Prerequisites**:
    - The target website must use HTTP compression.
    - The attacker must be able to: (a) trigger requests to the server from the victim (for example, via an XSS vulnerability or by inducing the victim to visit a malicious site), and (b) observe the size of the responses.
2. **Injection and Observation**:
    - The attacker forces the victim's browser to make multiple specially crafted HTTP requests to the target site. These requests are designed to include user-controlled input that affects the response.
3. **Response Compression**:
    - If part of the response includes attacker-controlled input along with secret data (like a CSRF token or a session cookie), the HTTP compression algorithm (e.g., GZIP) used on the server side will compress similar data

more efficiently. This leads to smaller response sizes when the guess about the secret content is correct.

4. **Analysis of Response Sizes**:
    - By analyzing the length of each compressed response, the attacker can infer which parts of their input were similar to the secret data in the response. For instance, if adding a guessed character to the input reduces the size of the response, it suggests that guessed character matches a character in the secret.
5. **Iterative Refinement**:
    - The attacker refines their guesses character by character. With each request, they adjust the input based on previous response sizes, gradually homing in on the exact secret.

TCP Sequence Predcation Attack

TCP sequence attacks typically involve an attacker predicting or guessing the sequence numbers used in a TCP connection. If an attacker can accurately predict the sequence numbers, they may be able to:

1. **Hijack Connections**: By predicting the sequence numbers, an attacker can inject their own packets into the TCP stream, effectively hijacking the connection. This can allow them to intercept sensitive information or manipulate the data being transmitted.
2. **Session Hijacking**: With accurate sequence number prediction, an attacker can impersonate one of the communicating parties and take over the TCP session. This can lead to unauthorized access to data or services.
3. **Denial of Service (DoS)**: In some cases, attackers may disrupt TCP connections by sending spoofed packets with predicted sequence numbers, causing confusion and potentially leading to connection resets or other disruptions.
4. **Data Injection**: Attackers may inject malicious data into the TCP stream by predicting sequence numbers, potentially leading to the execution of arbitrary

commands or other security breaches.

Public Key Infrastructure (PKI)

- PKI provides a means to establish **trust** between public keys and identities with certificates
- With PKI we are **sure that data is decrypted** with a corresponding private key
- Combing this with Hash Functions creates a **signature** we can then be sure than encrypted data also hasnt been **tampered with**
  - strengthens the encryption
- Certificates can be signed with the **issuer** private key with all info to validate the identity

---

Certification Authorities

- A private certificate authority issues certificates **locally**
  - e.g. for an organisation trusted by its members
- Public CAs issue certificates publically for members and must be trusted by the public (third party CA certificate)
- Certificate Authorities **must be trusted**, to extend trust the certificate includes the public key
  - their public keys are freely distributed to everyone

One such organisation is: Let's Encrypt

Let's Encrypt (CA organisation)

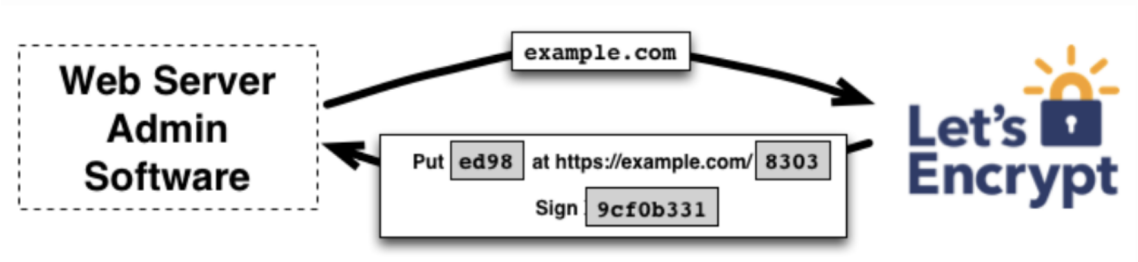
- Lets Encrypt is a public CA that can automatically grant a browser-trusted certificate for an HTTPS server **for free**
  - they give you an agent which you install on your server called: **“Certbot”**
  - this agent lets you generate certificates
- in order to use this method iyou must have **a valid registered domain name** and then install a certificate management agent on the web server (Certbot)
- This service is free and is an open certificate authority provided by the Internet Security Research Group (ISRG)
- It provides security with TLS, which is the best practise for admins to secure their websites
- The service offers **transparency,** the certificates that are issued are publically available for anyone to inspect

How does Let’s Encrypt work?

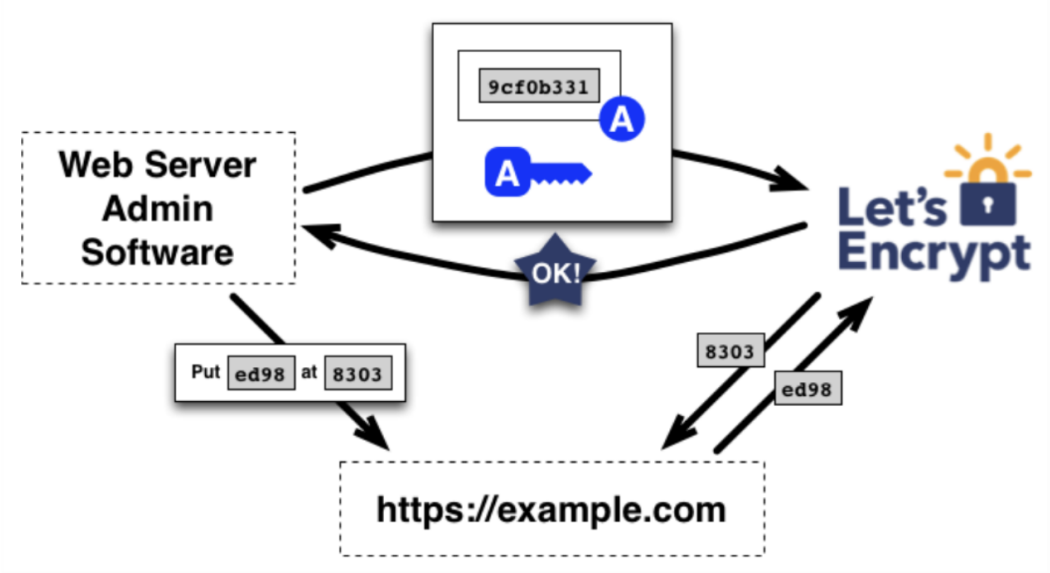
1. Lets Encrypt identifies the srver admin with the publci key. The installed agent (Certbot) generates a **new key pair** (public and private key) and informs Lets Encrypt that the server controls a valid registered domain
2. Let’s Encrypt (the CA) will issue a set of challenges to the server:

   for example:

   - Provide the **DNS Record:** (Domain Name System, which is essentially the internet's phonebook. DNS translates human-friendly domain names (like `www.example.com`) into machine-friendly IP addresses (like `192.0.2.1`).
   - **Provide a HTTP resource**
   - sign an arbitrary random number (called Nonce) with private key



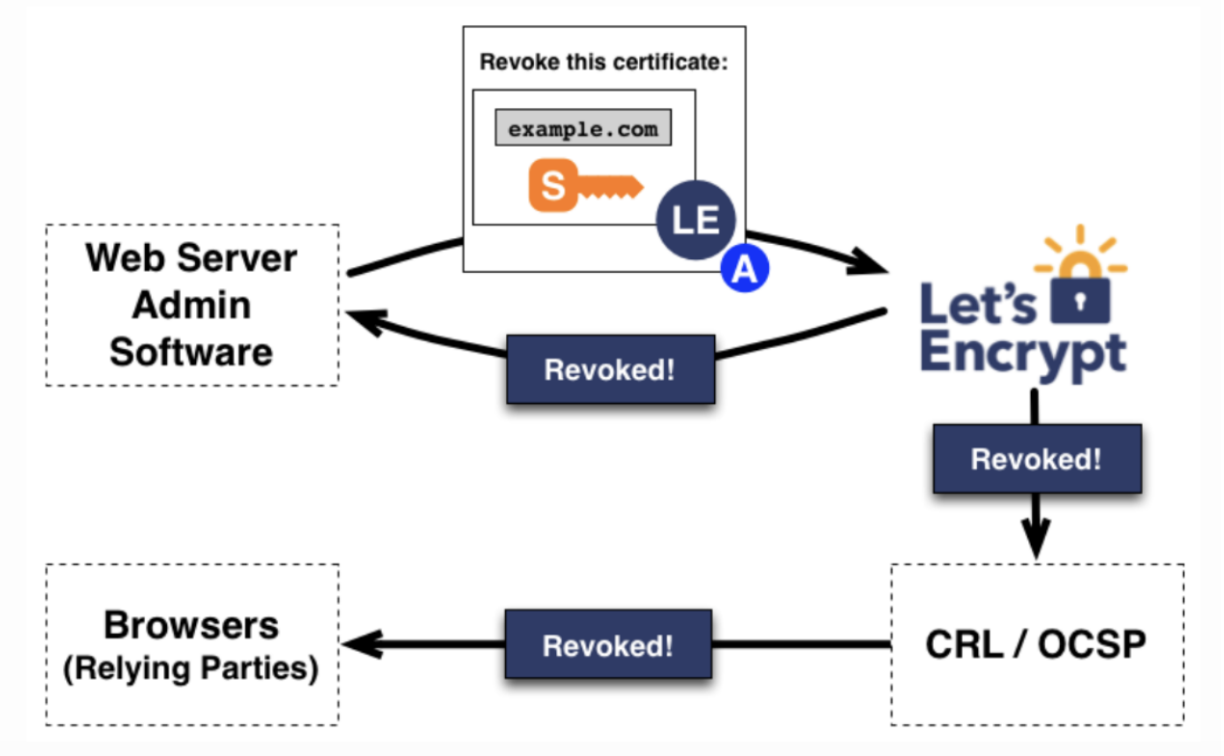
3. The agent completes the tasks, and the CA validates the signature of the nonce and the tasks, grants the agent the ability to request, renew, and revoke certificates



4. The agent then constructs a **Certificate Signing Request (CSR)** with a signature (public key) and asks Let's Encrypt to issue a certificate for the domain with its public key (whole CSR is signed with the private key)

Revocation works similarly

- the agent signes a revocation request and then Let's Encrypt CA verfies the request and authorises it, browers then stop accepting the invalid cerficate



Intro to PGP

Pretty Good Privacy

- defacto standard for secure exchange of information
- Today, PGP has become open standard known as OpenPGP
- PGP can encrypt messages online: email, plain text files etc
- Close to *military-grade* symmetric and asymmetric encryption

- Relies on a private key, integrity checks, message authentication, and signed certificates
- PGP is **slow** therefore is not considered for use in application

DOWNLOAD THE READING AT END OF SLIDES