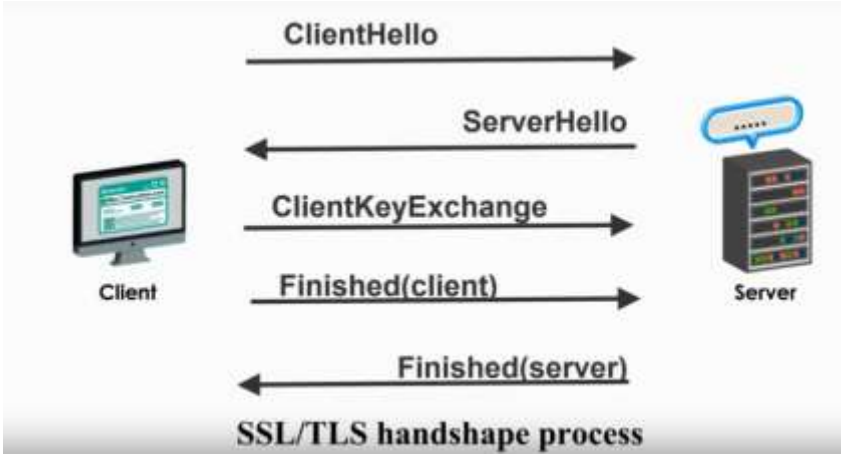
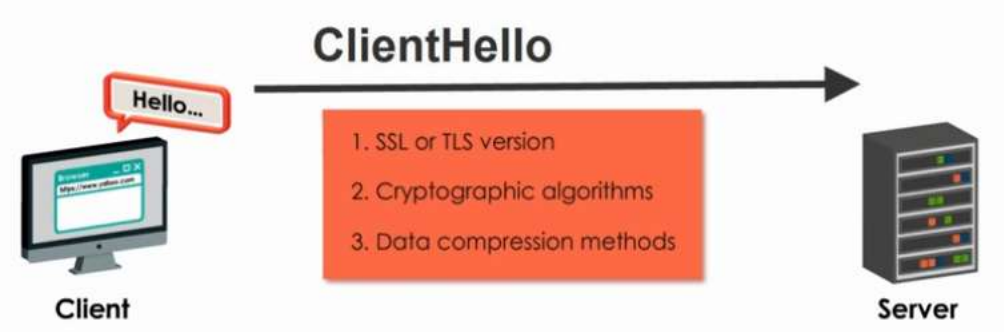
1. **How SSL/TLS Handshake works?**

SSL (Secure Socket Layer)& TLS (Transport Layer Security) are cryptographic protocols that provide security on the internet.



The handshake protocol is used between a Web client and Web Server to establish trust & then negotiate what a secret key should be used to encrypt and decrypt the conversation

Step 1



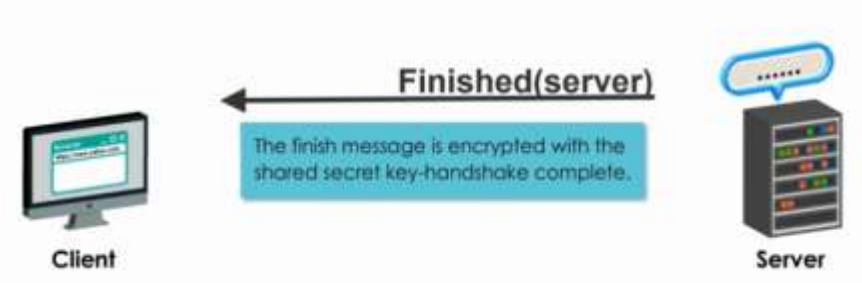
Step 2

Step 3

Step 4

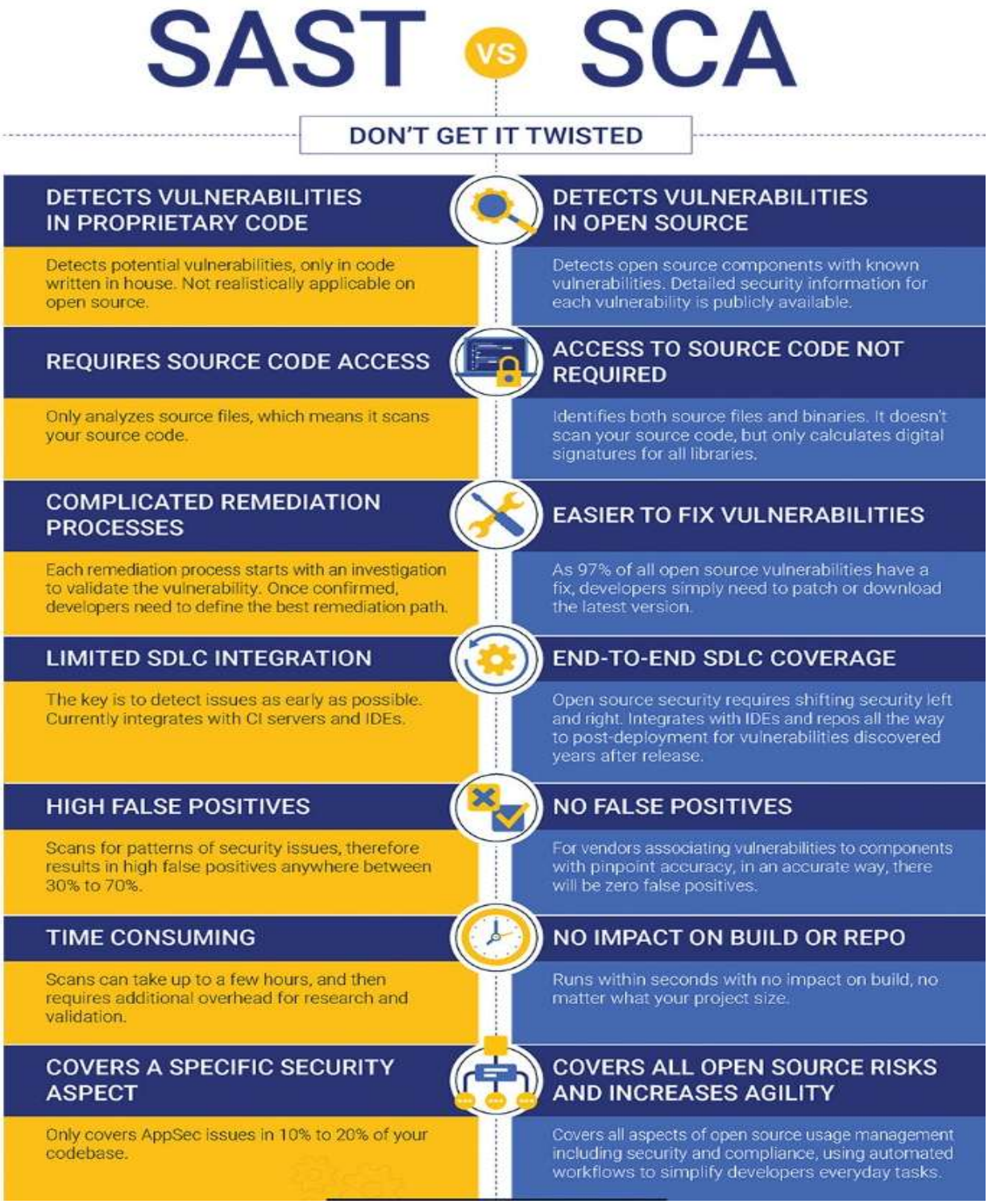
Step 5

Step 6



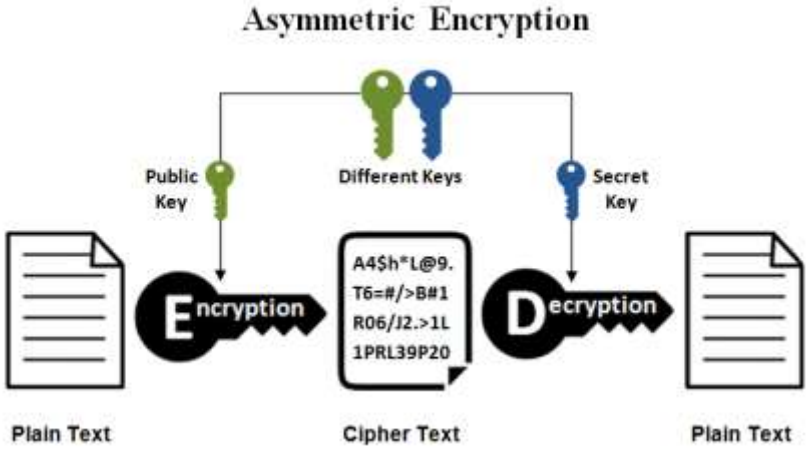
Once the handshake is done Client and Server can now exchange messages that symmetrically encrypted.

2. What is SCA(Software Composition Analysis)



3. **Explain Buffer Overflow**
   A **buffer** is a temporary area for data storage. When more data (than was originally allocated to be stored) gets placed by a program or system process, the extra data overflows. It causes some of that data to leak out into other buffers, which can corrupt or overwrite whatever data they were holding.
   Attacker would use a buffer-overflow exploit to take advantage of a program that is waiting on a user's input. There are two types of buffer overflows: stack-based and heap-based. Heap-based, which are difficult to execute and the least common of the two, attack an application by flooding the memory space reserved for a program. Stack-based buffer overflows, which are more common among attackers, exploit applications and programs by using what is known as a stack: memory space used to store user input.

4. **Asymmetric key Encryption**



Asymmetrical encryption is also known as public key cryptography, which is a relatively new method, compared to symmetric encryption. Asymmetric encryption uses two keys to encrypt a plain text. Secret keys are exchanged over the Internet or a large network. It ensures that malicious persons do not misuse the keys. It is important to note that anyone with a secret key can decrypt the message and this is why asymmetrical encryption uses two related keys to boosting security. A public key is made freely available to anyone who might want to send you a message. The second private key is kept a secret so that you can only know.

A message that is encrypted using a public key can only be decrypted using a private key, while also, a message encrypted using a private key can be decrypted using a public key. Security of the public key is not required because it is publicly available and can be passed over the internet. Asymmetric key has a far better power in ensuring the security of information transmitted during communication.

Asymmetric encryption is mostly used in day-to-day communication channels, especially over the Internet. Popular asymmetric key encryption algorithm includes EIGamal, RSA, DSA, Elliptic curve techniques, PKCS.

5. **How to detect SQL Injection**

Types of SQL Injection Attack

There are several types of SQL Injections, depending on the method of attack, the information to which the hacker can get access, and on the available 'surface area' of attack, which is increased by insecure use of tools, such as extended stored procedures.

In-Band Injection

This is the classic attack where the attacker can both launch the attack and obtain results through the same communication channel. This is done with two in-band techniques:

- **Error-based SQL Injection** gets information about the database from the error messages that are displayed
- **Union-based SQL Injection** relies on the attacker being able to concatenate (UNIONALL) the results of the information being stolen with the legitimate results.

Both techniques rely on the attacker modifying the SQL being sent by the application and on the errors and returned information being displayed in the browser. It succeeds where either the application developer or the database developer fails to properly parameterize the values that they use in their queries. Both are trial and error techniques, and the errors can be detected.

Blind Injection

Blind SQL injection is used where a result or message can't be seen by the attacker. Instead, the technique relies on detecting either a delay or a change in the HTTP response to distinguish between a query resolving to TRUE or FALSE. It's rather like communicating with the spirit world via tapping.

The errors will be like those for in-band injection, but the process is slower with more errors. Blind SQL Injection will produce several syntax errors and object-not-found errors since the only way of telling that something has worked is the length of time between the call being made and the error is returned.

Out of Band Injection

In out-of-band SQL Injection, the attacker uses SQL Server extensions, such as xp_dirtree, xp_cmdshell, sp_makewebtask (now removed), and xp_sendmail to provide 'exfiltration,' and send results to the attacker via HTTP or DNS. Here, the attackers need to find out whether they have permission to use these tools, so there will be errors generated if access to them is denied.

What Sort of Errors Do We Need to Detect?

- **Error 18456** — this is for failed logins, in case someone tries to gain a password by 'brute-force'
- **Errors 102 and 105** — I'd want to see all SQL that fails to execute because the syntax is incorrect. These errors are expected on the development and test servers, but should they ever happen in a production database? You'd certainly find a sudden jump in their frequency during SQL Injection.
- **Errors 208 and 2812 —** these attempt to access an invalid object or a stored procedure that cannot be found, illustrating the very characteristic of an injection attack.
- **Error 245** — this is used by hackers to get values, such as the name of the database.
- **Error 205** - This will happen if an attacker's Union-based injection triggers an error when using a UNION ALL SELECT phrase to find out more about the number of columns in a table.
- **All errors involving permissions** — I want to know about attempts to access objects to which the intruder is denied.

Assuming that application logins are denied access to system stored procedures, I'd also want to know about all errors involving the use of xp_dirtree, xp_cmdshell, sp_makewebtask, and xp_sendmail, which will tell you that a hacker is attempting to extract data into a file for sending. I'd also want errors from using xp_regread, xp_regwrite from attempts to view or write to the registry. It may be worth providing a way of looking at all SQL being executed for the use of these procedures.

6. Explain Threat Model

Threat modeling is a structured approach to identifying, quantifying, and addressing threats. It allows system security staff to communicate the potential damage of security flaws and prioritize remediation efforts.

In threat modeling, we cover the three main elements:

1. **Assets:** What valuable data and equipment should be secured?
2. **Threats:** What may an attacker do to the system?
3. **Vulnerabilities:** What flaws in the system allow an attacker to realize a threat?

In an organization, there are different threats that are addressed to different layers of an organization framework and environment. The three main layers of a threat target are:

- **Network: T**he threat includes spoofed, malicious packets, etc.
- **Host:** The threat includes Buffer overflow, malicious file, etc.
- **Application:** The threat includes SQL injection, XSS, input tampering, etc.

Who Do Threat Models and When

Ideally, threat models are created during system design before any deployment. In practice, threat models are often created for existing systems, making it part of maintenance. System designers with security experience are best equipped to identify the threats.

7. Steps to Threat Modeling

1. **Identify the Assets**
2. **Describe the Architecture**
3. **Break down the Applications**
4. **Identify the Threats**
5. **Document and Classify the Threats**
6. **Rate the Threats**

Identifying the Assets:

In this step, we identify the assets that what are the potential assets that are valuable to the organization:

- Entry and exit points
- System assets and resources System assets and resources
- Trust levels (access categories)

Describe the architecture:

In this process, we describe the architecture on which the valuable asset is being processed. It may include the software framework, version, and other architectural details.

Break down the application:

In this step, we break down the application regarding its process. All the sub-processes that are running the application.

Identifying the threats:

In this step, we list down the threat in a descriptive way, so that it can be reviewed to process further.

Categorizing and classifying the threats:

In this step we categorize the threat in predefined classes that are:

- Spoofing Identity
- Tampering with Data

- Repudiation
- Information Disclosure
- Denial of Service
- Elevation of Privilege

Rate the threats:

In this step we rate the severity of the threat based on a scale developed by Microsoft:

- **Damage Potential**: How bad can an exploit hurt?
- **Reproducibility**: How reliably can the flaw be exploited? How reliably can the flaw be exploited?
- **Exploitability:** How easy is the flaw to exploit?
- **Affected Users**: How many users can be impacted by an exploit?
- **Discoverability**: How "visible" is the vulnerability?

Example:

A Corporation has a data collection web application that allows users to login in and enter or change personal data.

The following information was collected on the application:

Architecture:

- Web Application – ASP.Net
- Database – SQL Server 2000

Assets:

- User Login Credentials
- User Personal Information
- Administrative Resources
- System Hardware

Microsoft Threat Reporting Template:

- **ID –** Unique ID # of the threat
- **Name –** Brief name of the asset threat
- **Description –** Detailed description of threat and its importance.
- **STRIDE –** How can we classify this threat?
- **Mitigated–** Is the application safe from this threat? Is the application safe from this threat?
- **Known Mitigation –** How can we protect against the threat?
- **Investigation Notes –** What do we know about this threat so far?

- **Entry Points–** What possible means does an adversary have?
- **Assets –** What assets could be damaged?
- **Threat Tree –** How can we visualize the threat? (Optional)

Threat Description:

ID: 1

- Name: Login Subversion
- Description: An adversary tries to inject SQL command through a request into the application to circumvent the login process.
- STRIDE Classification: Tampering with data, Elevation of privilege
- Mitigated: No
- Known Mitigation: Stored Procedures, Parameterized Queries
- Investigation Notes: The database calls to in the application were reviewed, and string concatenation was used on the login query.
- Entry Point: Login Page
- Assets: Access to backed database
- Threat Tree: None

Categorizing Threats with STRIDE:

A standardize short form created by Microsoft to help categorize threats.

- **S**poofing Identity
- **T**ampering with Data
- **R**epudiation
- **I**nformation Disclosure
- **D**enial of Service
- **E**levation of Privilege

Rating Threats with DREAD:

A standardize short form created by Microsoft to rate the severity of a threat. Each quality is rated based on a scale developed for each project. For most projects, a scale of 1– 3 is sufficient 3 is sufficient.

- **D**amage Potential – How bad can an exploit hurt?
- **R**eproducibility – How reliably can the flaw be exploited? How reliably can the flaw be exploited?
- **E**xploitability – How easy is the flaw to exploit?
- **A**ffected Users – How many users can be impacted by an exploit?
- **D**iscoverability – How "visible" is the vulnerability?

    a. Damage Potential

1. Attacker can retrieve extremely sensitive data and corrupt or destroy data
2. Attacker can retrieve sensitive data but do little harm
3. Attacker can only retrieve data that has little or no potential for harm

b. Reproducibility

1. Work every time; does not require a timing window or specific extreme cases
2. Timing-dependent; works only within a time window
3. Rarely works

c. Exploitability

1. Just about anyone could do it
2. Attacker must be somewhat knowledgeable and skilled
3. Attacker must be very knowledgeable and skilled

d. Affected Users

1. Most or all users
2. Some users
3. Few if any users

e. Discoverability

1. Attacker can easily discover the vulnerability
2. Attacker might discover the vulnerability
3. Attacker will have to dig to discover the vulnerability

Threat modeling helps organizations to prioritize their processes with respect to threats and effective response. It is carried out through the complete life cycle of the process from initialization to the deployment and also remains under consideration in the maintenance process.