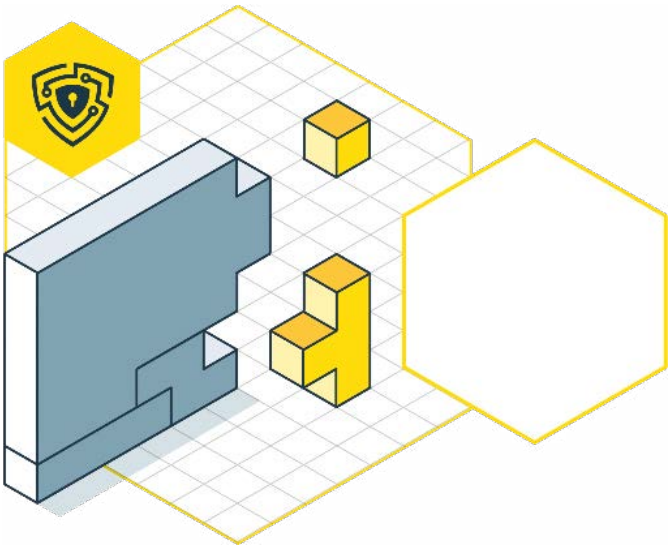
CRASHTEST **SECURITY**

GUIDE FOR PREVENTING SSL/TLS VULNE-RABILITIES

WHAT ARE THE STEPS TO KEEP YOUR WEB APP OR API SAFE FROM SUCH VULNERABILITY

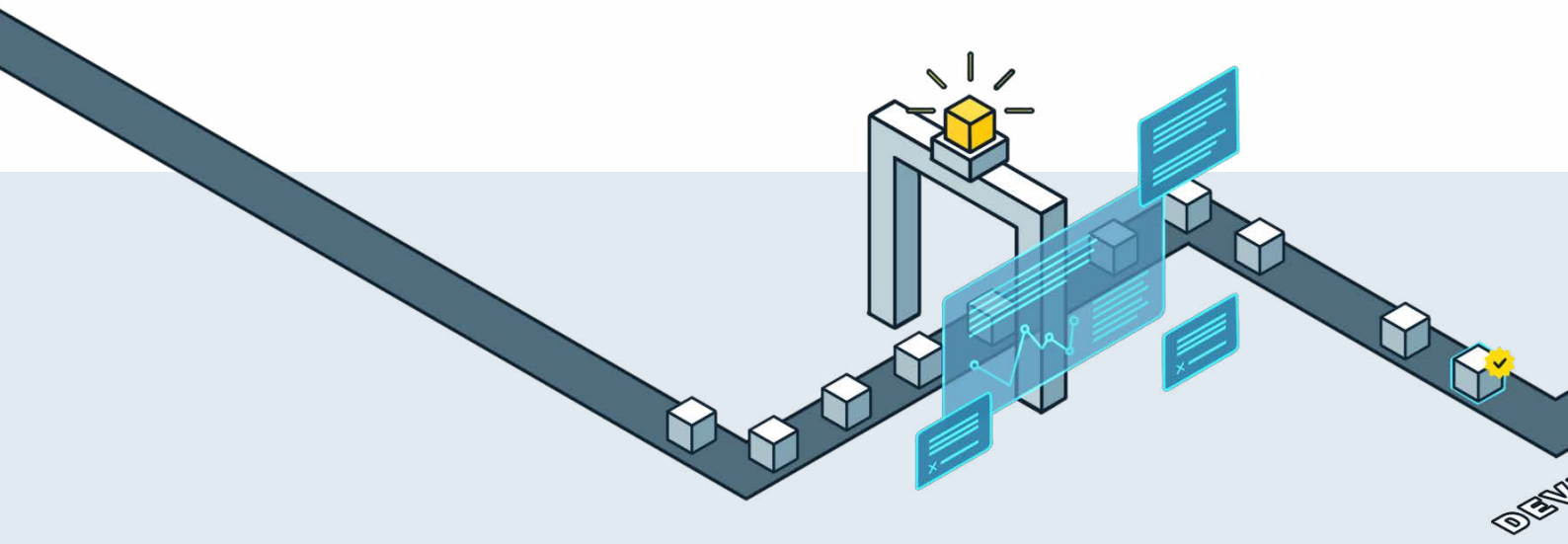
GUIDE FOR THE SSL/TLS VULNERABILITY PREVENTION

Table of Contents



INTRODUCTION TO THIS GUIDE

Transport Layer Security (TLS) and Secure Sockets Layer (SSL) are cryptographic protocols that offer secure authentication and transport of data for modern web systems and major browsers. While such protocols are critical to support modern computing, these also come with challenges in maintaining robust communications security. This comprehensive guide teaches what SSL/TLS vulnerabilities are, various forms of attacks that can be orchestrated over such vulnerabilities, prevention techniques, and how Crashtest Security can help mitigate such vulnerabilities.



WHAT ARE SSL/TLS VULNERABILITIES?

Given the heavy reliance on TLS/SSL protocols in modern web and network traffic, attacks targetting vulnerabilities within such protocols are prevalent. Although such protocol suites are constantly evolving to address advanced threats, configurations of a deprecated version or those with an original protocol continue to be prime targets for hackers.

Significant vulnerabilities in TLS/SSL include:

Attacker encrypted communication

If the attacker can get a hold of data in transit, they may encrypt, misuse and lock sensitive user and system data. This also potentially builds the foundation of a ransomware attack that leads to attackers demanding ransom from the affected user before restoring their access to the data.

Wildcard certificates

Some enterprises use a single, self-signed certificate to secure all their subdomains of a single domain name. While this allows for the quick deployment of secure transport protocols across all subdomains, it also introduces many security risks. If a public-facing web server using a wildcard is compromised, attackers can impersonate any subdomain protected by the wildcard key certificate. Attackers can also use a webserver's identity to host malicious content for phishing attacks and other social engineering campaigns.

Untrusted certificate authorities

Certificate Authorities (CAs) are responsible for issuing SSL/TLS certificates. Attackers exploit inherent security misconfigurations to act as a CA and issue forged self-signed certificates. Cryptographic configurations may fail to detect malicious activity in such instances where the CA is compromised or untrusted. Since the browser trusts the CA, users continue to transact sensitive data being under the impression that they are connected to an authentic, secure webserver.

TLS/SSL stripping vulnerabilities

TLS/SSL stripping is a popular vector for modern man-in-the-middle attacks built on the SSL strip vulnerability. The vulnerability allows hackers to downgrade the security potency of web systems by depriving them of the encryption layer that offers HTTPS security. After a successful attack, attackers intercept users' HTTP requests and proxy them to a malicious server during HTTPS redirection. Since users miss noticing the redirect, they continue interacting with the attacker's server over HTTP.

RSA key transport weaknesses

Devices that use RSA encryption keys and certificates are susceptible to the Trusted Platform Module firmware vulnerability that mishandles key generation. This allows hackers to access private keys and generate their certificates. The keys generated by this encryption standard are not genuinely random and allow for a practical factorization or birthday attack.

CBC-mode cipher vulnerabilities

Ciphers that use the CBC mode of operation use padding to ensure a uniform block size. CBC-cipher vulnerabilities facilitate Padding Oracle attacks - exploits in which the attacker tampers with the ciphertext to check whether it causes an error in padding format. If the server returns an implementation error while decrypting invalid security content, the hacker can deceptively create malicious ciphertext and extract enough information to reconstruct the plaintext data.

TYPES OF SSL/TLS ATTACKS

With the changing threat landscape, hackers have developed different attack patterns to exploit vulnerabilities in TLS/SSL. Some most advanced forms of TLS/SSL attack include:

POODLE ATTACK

In October 2014, the **Padding Oracle On Downgraded Legacy Encryption (POODLE)** attack exploited an SSL 3.0 protocol vulnerability. In this type of attack, hackers leverage protocol negotiation capabilities to enforce the use of SSL 3.0. This starts with the attacker eavesdropping on all communication between the server and the client to manipulate the network traffic to impersonate entities. Following this, the attacker tricks the server into thinking that the client can not communicate with newer protocols by dropping connections. This convinces the server to use the SSL 3.0 protocol while the attacker tricks the client browser into running JavaScript that helps decrypt sections of encrypted data.

POODLE attacks are carried out on servers that use 128 and 64-bit block ciphers and a symmetric encryption algorithm. The cipher suite selection used on these systems relies on the CBC mode of operation and uses the Message Authentication Code (MAC) to encrypt input plain text and padding values. Hackers can reconstruct several bytes of the original plain text by crafting similar malicious requests.

BEAST ATTACK

The **Browser Exploit Against TLS/SSL (BEAST)** attack allows hackers to capture encrypted sessions and obtain plaintext data. The BEAST vulnerability can be found on any webserver/website that supports application protocol SSL, TLS 1.0, or older cryptographic protocols with weak ciphers.

Given the number of input-key combinations, breaking cryptographic algorithms typically requires considerable time and effort. However, the BEAST vulnerability simplifies the deciphering of cryptographic algorithms by allowing the attacker to guess a single byte at a time.

CRIME ATTACK

Compression Ratio Info-leak Made Easy (CRIME) attack targets a client-side implementation error on connections that use a stream cipher, HTTPS/SPDY, TLS data compression, or web cookies. A man-in-the-middle attack can hijack session cookies for web servers with this implementation error while the user is authenticated to the web application. The primary objective of a CRIME attack is to infer the value of the session cookie by observing the change in compression method or ciphertext length.

When orchestrating a CRIME attack, a hacker typically abuses vulnerabilities in the compression mechanism, allowing them to control the path taken by new requests and inject malicious data into the stream. A successful attack allows the hacker to obtain data on the size of security content sent by the browser, the compression method, and the payload compression ratios.

BREACH ATTACK

The Browser Reconnaissance and Exfiltration via Adaptive Compression (BREACH) attack targets HTTPS connections using HTTP-level compression. BREACH attacks are typically targeted at applications that include at least one of the following flaws:

- Reflect user input in the body of HTTP responses
- Reflect security content within the response bodies
- Connect to a server that uses compression at the HTTP level

Attackers can exploit these vulnerabilities to extract cookies, client certificates, authentication tokens, and other sensitive security content. A BREACH attack involves sending several requests to the server while observing data returned in the server responses.

FREAK ATTACK

A FREAK attack is a commonly found SSL/TLS vulnerability that forces target clients and servers to use export-grade cryptography by intercepting HTTPS connections. In a FREAK attack scenario, threat actors trick cipher suite selection mechanisms to establish a connection and downgrade the application protocol to a vulnerable version.

A FREAK attack is commonly orchestrated on servers that meet at least one of the following conditions:

- The server supports RSA export cipher suites
- Clients offer the RSA cipher suite/use vulnerable OpenSSL version
- Clients use Secure Channel or Apple Secure Transport for enterprise transport security

DROWN CROSS-PROTOCOL ATTACKS

Decrypting RSA with Obsolete and Weakened eNcryption (DROWN) is a collection of downgrade attacks aimed at servers that support SSL v2 connections. To do so, attackers leverage cross-protocol vulnerabilities that allow modern, TLS-enabled websites to use the insecure SSL 2.0 protocol. SSLv2 being vulnerable to a **Bleichenbacher RSA padding oracle** attack, attackers exploit its vulnerability to decrypt RSA security content without possessing the private key. Successful DROWN attacks result in the exposure of sensitive, encrypted data, such as initialization vectors, passwords, trade secrets, financial data, and credit card information.

RENEGOTIATION ATTACKS

A renegotiation procedure is essential for TLS/SSL communications since it allows for creating a new transport layer security handshake inside an existing successful connection. This helps re-establish secure connection after a terminated negotiation phase while enabling entities to change cipher suites and renew client certificates if they need to. In instances where a client is allowed to initiate renegotiation handshakes, attackers can inject malicious content into the protocol stream and perform denial-of-service attacks to lead the webserver into a downtime.

SSL/TLS VULNERABILITY SEVERITY LEVEL

Transport Layer Security plays a crucial role in modern web communications as it encompasses various encryption protocols and techniques designed to provide integrity and confidentiality. Vulnerabilities in the middlebox security protocol are severe since they allow attackers to access hidden information. Unsurprisingly, two of OWASP's top 10 vulnerabilities of 2021 -- **Broken Access Control (A01:2021)** and **Cryptographic failures (A02:2021)** -- are directly associated with misconfigurations in transport layer security.

These vulnerabilities allow hackers to intercept encrypted communications and snoop on traffic between web servers and clients. Successful TLS/SSL attacks lead to the disclosure of sensitive information, thereby allowing attackers to unleash various attacks based on enterprise transport security flaws, such as:

- Account takeover
- Data breaches
- Denial-of-service
- Exposure of encryption algorithms in proprietary networks with weak ciphers
- Ransomware attacks
- Theft of client certificates using FREAK attack
- Phishing attacks
- Deployment of advanced persistent malware
- Privilege escalation attacks
- Loss of business reputation and customer trust

HOW TO IDENTIFY SSL/TLS VULNERABILITIES WITH CRASHTEST SECURITY?

The Crashtest Security Suite includes a new generation TLS/SSL scanner to identify and report application vulnerabilities in a single click. The scanner automatically connects to target ports and cross-validates the cipher suite list and protocol versions to discover common vulnerabilities that lead to transport layer security attacks. The scanner also checks chain certificates to ensure they are from a valid CA and point to a legitimate server. Crashtest Security provides actionable vulnerability reports with comprehensive details such as supported TLS/SSL versions, the server's handshake preference, certificate details, and encryption cipher details to ensure complete hardening of the transport layer.

Crashtest Security also offers a heartbleed tester out of the box that checks the configuration of every webserver to prevent heartbleed vulnerabilities in SSL client certificates. As heartbleed attacks are aimed at OpenSSL, the scanner helps avoid vectors that enable hackers to obtain a private key and impersonate the webserver.

The security suite additionally comes with an extensive list of scanners to detect, identify and remediate various attack vectors associated with transport layer security. These include **CSRF scanner, port scanner, FREAK attack vector detection, CRIME vulnerability scanner, BEAST vulnerability scanner, HTTP Header scanner,** and **API vulnerability scanner**.

SSL/TLS VULNERABILITIES PREVENTION TECHNIQUES

Some ways to prevent transport layer security vulnerabilities include:

USING THE LATEST TLS VERSION (TLS 1.3)

Use the most stable, up-to-date version of the TLS/SSL cryptographic protocol as it includes the most recent enhancements to avoid known vulnerabilities of previous versions. The current TLS version has dropped support for legacy insecure protocol and encryption algorithms, such as export cipher suites, making it less susceptible to obsolete cryptographic algorithm attacks. TLS 1.3 features fewer and faster handshakes, with TLS handshakes requiring one round trip instead of two in the negotiation phase. TLS 1.3 also features a more extensive cipher suite list that replaces weak ciphers with GCM ciphers for a successful connection.

CHOOSING AN APPROPRIATE CERTIFICATE AUTHORITY

Certification Authority Authorization (CAA) records specify which CAs can issue digital certificates for a particular domain, eliminating the need for self-signed certificates. Any CA who is missing from the CAA records should be avoided for issuing a valid certificate. The CA should be well known and trusted automatically by browsers and operating systems for public-facing applications. Organizations can also use internal CAs for proprietary networks, although these can only generate key certificates for internal users and servers within the organization‘s network.

USING HTTP STRICT TRANSPORT SECURITY (HSTS)

Using a special response header, modern web servers specify HSTS configuration as an extended enterprise transport security protection layer. Although most browsers support HSTS, it is recommended to explicitly include it in the server configuration to ensure it is active for all internet-facing web applications. Once this header reaches the client, the browser ensures all communications sent to the specified domain are in HTTPS and not HTTP. HSTS also prevents browser prompts for HTTPS click-throughs. By automatically redirecting HTTP requests to HTTPS, HSTS forms the first line of defense against man-in-the-middle attacks.

ENFORCING CSRF PROTECTION

For applications susceptible to cross-site request forgery, attackers can force authenticated users to inject malicious payloads into the data stream by submitting requests to malicious websites through phishing attacks. These malicious payloads can be used to modify ciphertext as in POODLE attacks or abuse the application‘s compression mechanism. CSRF protection techniques keep attackers from successfully instigating man-in-the-middle attacks, hence preemptively avoiding enterprise transport security threats.

BEST PRACTICES IN PREVENTING SSL/TLS ATTACKS

Best practices to avoid compromising TLS/SSL security include:

USE CRYPTOGRAPHICALLY STRONG CIPHERS

TLS supports an extensive cipher suite list that includes ciphers with varying encryption strength and security levels. As a recommended practice, an organization's cipher suite selection should involve a cipher suite list that covers the most appropriate encryption algorithms. Whenever possible, it is also recommended to enforce the use of Galois/Counter Mode ciphers that offer enhanced security by leveraging universal hashes over binary Galois fields for authenticated encryption. Weak ciphers such as anonymous, null, and EXPORT cipher suites should also be avoided as these are considered vulnerable versions that expand the attack surface.

AVOID TLS COMPRESSION

TLS compression is known for its vulnerability to leaking plaintext information, such as session cookies and other security-related HTTP headers, subsequently allowing for CRIME attacks. Hackers can steal cookie data and hijack user sessions using a CRIME attack. Although modern upgraded browsers do not support TLS compression, it is recommended that browser versions are upgraded regularly while TLS compression is explicitly disabled in browsers.

PERFORM REGULAR SERVER CONFIGURATION TESTS

The OWASP security bulletin offers guidance on SSL/TLS testing to help identify weaknesses in transport layer security. This allows teams to establish whether they have implemented adequate security controls to protect the TLS/SSL layer. The tests include **server cipher suite selection validation, configuration validation, certificate strength, validity scans**, and **appropriate implementation of TLS**.

Crashtest Security Suite's TLS/SSL vulnerability scanner allows you to scan your web application directly through an efficient webhook integration to help test multiple attack vectors, including **portscan, common misconfiguration**, and **breach attacks**.

USE WILDCARD CERTIFICATE WHEN ABSOLUTELY NECESSARY

Since wildcards make a single certificate valid for all associated subdomains, they violate the principles of **same-origin** and **least privilege**. As multiple server systems commonly share a wildcard certificate, the certificate's private key must exist on numerous systems, increasing the likelihood of compromise. Since attackers value the single private key highly, it becomes a prime target for breaches. It is recommended that instead of using wildcard certificates on production systems, developers should leverage **short-lived, sub-domain-specific certificates** that are regularly rotated to prevent malicious attacks.

AVOID MIXING TLS AND NON-TLS CONTENT

A page that uses TLS/SSL protocols should not include resources loaded over unsecured HTTP connections. Attackers can use these resources (CSS and Javascript) to inject malicious code or sniff session cookies from a target webpage. It is also recommended to use updated versions of modern browsers that are configured to deny a successful connection when dynamic loading content served over HTTP to HTTP-secured pages.

PREVENT TLS/SSL ATTACKS WITH CRASHTEST SECURITY

Crashtest Security's new generation TLS/SSL scanner emulates the updated TLS version to scan for secure cryptographic protocols in your existing tech stack. Crashtest Security initiates a periodic, comprehensive scan of your web app and APIs to prevent enterprise transport security misconfigurations with a simple click of a button.

Try Crashtest Security's automated scanning today for TLS/SSL vulnerability identification, classification, and remediation advice.

Start 2-Week Trial for Free

CRASHTEST **SECURITY**

WWW.CRASHTEST-SECURITY.COM