SSL/TLS Best Practices Checklist

The SSL/TLS Best Practices Checklist provides a comprehensive framework for implementing and maintaining secure TLS configurations across internet-facing environments such as websites and web apps. Organized into eight logical categories, the checklist offers specific, actionable guidance with clear importance ratings to help prioritize remediation efforts. This structured approach helps organizations systematically establish robust TLS implementations that protect against known vulnerabilities while maintaining operational efficiency.

Find more guides and information at https://geotrust.com/resources/ssl-tls-best-practices/.

Category #1: Private Keys

Use strong private keys to protect all critical accounts and systems

Importance: Critical **Applies To: All Websites**

Use at least 2048-bit RSA or 256-bit elliptic curve cryptography/elliptic curve digital signature algorithm (ECC/ECDSA) keys to provide sufficient security. Consider ECC/ECDSA for better performance and security, if supported by your website or web server.

Protect your private keys using secure storage

Importance: Critical **Applies To: All Websites**

Restrict access to private keys to the fewest employees possible. Ensure your organization's keys can't be accessed publicly (e.g., exclude cryptographic keys from inclusion in code repositories, hard-coding, or public web directories). Use hardware security modules (HSMs) to store cryptographic secrets for critical systems.

Avoid reusing private keys to reduce compromise risks

Importance: High **Applies To: All Websites**

Generate new private keys when renewing SSL/TLS certificates to limit the exposure window if your keys somehow become compromised. To reduce risk, avoid using the same private key and certificate on multiple servers.

Category #2: Certificate Lifecycle Management

Use valid SSL/TLS certificates from reliable certification authorities (CAs)

Importance: Critical **Applies To: All Websites**

When it comes to choosing the most "secure" certificates, there's a secret most companies won't tell you: they all offer the same level of security in terms of encryption. However, where certificates often differ drastically is in their identity vetting of domains and organizations (i.e., domain validation [DV], organization validation [OV], and extended validation [EV]), which is useful for phishing protection and compliance. With this in mind, choose certificates from publicly trusted CAs that are known for having strong security postures, reliable uptime, and dependable technical support.

Employ strong cryptographic signature algorithms

Importance: Critical **Applies To: All Websites**

Ensure SSL/TLS certificates use secure hash algorithms (SHA-256 as a minimum). These ciphers are used to create digital signatures that ensure the integrity of data transmitted by web and email servers, as well as in other uses and applications (e.g., software apps and scripts, PDFs, and other digital signature use cases). Avoid using deprecated algorithms like SHA-1 or MD-5, as they leave your systems and data at risk of exploitation.

Note: All publicly trusted CAs already enforce these rules.

Install complete certificate chains

Importance: Critical **Applies To: All Websites**

Deploy complete SSL/TLS certificate trust chains, including all intermediate certificate authorities (CAs), to avoid browser trust issues. This combination of root CA, intermediate CA(s), and leaf certificates is commonly known as a "CA bundle."

Automate your certificate management to unburden yourself of monotonous tasks

Importance: High **Applies To: All Websites**

Use SSL/TLS automation to handle your certificate issuance, installation, renewal, and monitoring tasks. Doing this reduces risks of human error and expired certificate-related outages and security issues.

Pro Tip: GeoTrust Horizon's AutoInstall SSL automates the entire certificate lifecycle.

Deploy SSL/TLS certificates that include proper hostname coverage

Importance: Critical **Applies To: All Websites**

Include all relevant domain names (including domains that redirect) in certificates (with and without the *www* prefix). Include subdomains individually or as a wildcard.

Implement certificate transparency (CT) log monitoring

Importance: High **Applies To: All Websites**

Monitor CT logs to detect unauthorized certificates issued for your domains. These append-only public logs serve as records of all SSL/TLS certificates, so you know what certificates were issued and when. Continuously monitoring CT logs helps you quickly address any unauthorized certificate issuance.

Configure domain name system (DNS) certificate authority authorization (CAA) records

Importance: Medium **Applies To: All Websites**

Add CAA records to your domain name system to control which CAs can issue certificates for your domain. Furthermore, starting March 15, 2027, publicly trusted CAs will be required to check and enforce CAA record extensions *accounturi* and *validationmethods*, which provide domain owners with more granular control over specifying which accounts can request certificates on behalf of a domain and which validation methods are permitted.

More: Read more about the CAA extensions in the CA/B Forum Ballot SC-098v2: https://cabforum.org/2026/05/13/ballot-sc098v2-process-rfc-8657-caa-parameters/.

Enable the online certificate status protocol (OCSP) or OCSP stapling for real-time revocation checks

Importance: Medium **Applies To: Websites with Self-Managed Server(s)**

Configure OCSP stapling on your server to efficiently deliver certificate revocation information in real time during the initial TLS handshake. (The handshake process takes place when a device or user attempts to connect to your server.)

Pro Tip: Use the TLSRef TLS Configurator tool (https://configurator.tlsref.org/) to simplify this process.

Automate certificate replacements that result from revocations

Importance: High **Applies To: All Websites**

In addition to automating certificate renewal-related tasks, ensure that your certificate automation tool can detect a revocation event and replace the certificate within 24 hours.

Pro tip: GeoTrust Horizon's AutoInstall SSL checks every 12 hours and replaces your certificate if it is scheduled for revocation.

Just say no to HTTP public key pinning (HPKP)

Importance: Low **Applies To: All Websites**

Legacy HPKP headers aren't supported by modern browsers and serve no protective purpose. As such, it's unlikely that you have key pinning in place. However, if you do find a *Public-Key-Pins* header lurking somewhere in your server configurations, it should be removed immediately, as it can otherwise lead to security issues.

Category #3: Protocol Configuration

Use only secure protocol versions (i.e., TLS 1.2 as a minimum)

Importance: Critical **Applies To: Websites with Self-Managed Server(s)**

Only support TLS 1.2 and TLS 1.3. Disable older protocols (i.e., SSL versions 2.0 and 3.0, and TLS versions 1.0 and 1.1), which have known vulnerabilities and exploits that leave your website and the entities connecting to it at risk.

Pro Tip: Use the TLSRef TLS Configurator tool (https://configurator.tlsref.org/). For Windows Server users, see https://learn.microsoft.com/en-us/windows/win32/secauthn/protocols-in-tls-ssl--schannel-ssp-.

Use secure cipher suites across the board

Importance: Critical **Applies To: Websites with Self-Managed Server(s)**

Prioritize authenticated encryption with associated data (AEAD) ciphers (e.g., CHACHA20_POLY1305, GCM, CCM). Eliminate weak, null, and export ciphers from your lists, and ensure the most secure algorithms are prioritized, with the most secure listed up front.

Pro Tip: Use the TLSRef TLS Configurator tool (https://configurator.tlsref.org/). For Windows Server users, use https://nartac.com/Products/IISCrypto instead.

Enable perfect forward secrecy (PFS) to add another layer of security to data transmissions

Importance: High **Applies To: Websites with Self-Managed Server(s)**

Prioritize cryptographic ciphers that enable perfect forward secrecy (i.e., ECDHE, DHE, and all TLS 1.3 ciphers). PFS helps to ensure the security of previously transmitted data, even in the event that a key becomes compromised. This is because PFS is achieved when a unique set of cryptographic keys (i.e., ephemeral keys) is used for every transaction.

Pro Tip: Use the TLSRef TLS Configurator tool (https://configurator.tlsref.org/). Windows Servers can use this tool instead: https://nartac.com/Products/IISCrypto.

Configure secure key exchanges

Importance: Critical **Applies To: Websites with Self-Managed Server(s)**

Use ECDHE with the secp256r1 curve (p-256) or DHE (with at least 2048-bit security) for secure keypair generation. This will help you achieve perfect forward secrecy. For example, you can use standardized Diffie-Hellman groups with sufficient security (e.g., 2048-bit keys) to prevent protocol downgrade attacks.

Pro Tip: Use the TLSRef TLS Configurator tool (https://configurator.tlsref.org/). Windows Servers can use this tool instead: https://nartac.com/Products/IISCrypto.

Disable compression on sensitive responses

Importance: Critical **Applies To: Websites with Self-Managed Server(s)**

Disable TLS compression on sensitive dynamic pages to mitigate CRIME attacks. Additionally, address HTTP response compression-related vulnerabilities (e.g., TIME and BREACH) at the application level to prevent exposure of sensitive data and secrets, such as CSRF tokens and session cookies.

Pro Tip: Use the TLSRef TLS Configurator tool (https://configurator.tlsref.org/) to simplify the configuration process.

Implement support for post-quantum cryptography (PQC)

Importance: High **Applies To: All websites**

Implement PQC algorithms to defend your organization and its sensitive data in transit against harvest now, decrypt later (HNDL) attacks. At present, this means enabling the TLS 1.3 protocol, PQC-ready SSL/TLS libraries, and use of the X25519MLKEM768 hybrid key agreement algorithm. Legacy public key algorithms (i.e., RSA and ECC) will be officially deprecated by the National Institute of Standards and Technology (NIST) for high-risk systems by 2030 and fully disallowed across federal systems by 2035. Read more in the draft report NIST IR 8547: https://csrc.nist.gov/pubs/ir/8547/ipd.

Pro Tip: Use the TLSRef TLS Configurator tool (https://configurator.tlsref.org/) to simplify the configuration process.

Category #4: Monitoring & Maintenance

Implement continuous monitoring of your SSL/TLS certificates and associated settings

Importance: Critical **Applies To: All Websites**

Monitor SSL/TLS certificate status, expiration dates, and configuration settings to prevent outages and security-related issues.

Pro Tip: GeoTrust Horizon's SSL Monitor provides daily automated scans of your website and alerts for 110+ SSL/TLS-related security issues. Check out https://geotrust.com/ssl-monitor to learn more.

Be proactive with SSL/TLS certificate renewals to prevent certificate-related outages

Importance: High **Applies To: All Websites**

Renew certificates at least 30 days before their expiration dates to allow adequate time for testing, troubleshooting, and handling any other issues that may arise.

Build crypto-agility into your infrastructure

Importance: High **Applies To: All Websites**

Design and maintain your systems to enable rapid replacement of SSL/TLS certificates and their cryptographic key pairs as needed without requiring any major overhaul of your infrastructure. This preparation is particularly useful when responding to critical incidents, such as CA compromise issues, or when algorithm-related vulnerabilities arise. It'll also put you on strong footing to prepare for the eventual migration to PQC.

Perform regular security assessments

Importance: High **Applies To: All Websites**

Regularly test your SSL/TLS configurations to identify potential vulnerabilities and misconfigurations that could leave your organization and customers' data at risk.

Apply security patches promptly (but deploy said patches in a testing environment first)

Importance: Critical **Applies To: Websites with Self-Managed Server(s)**

Test any security patches and updates on representative workloads before implementing a full production rollout. This helps to prevent service disruptions. For critical security patches, balance testing rigor with urgency — a staged rollout with monitoring is often a good approach to catch issues early.

Category #5: Application Security

Deploy HTTP strict transport security (HSTS) to force secure HTTPS connections

Importance: High **Applies To: All Websites**

Implement HSTS headers that instruct connecting clients to always use HTTPS connections to your site. This approach, which "forces" HTTPS connections, helps to prevent protocol downgrade attacks that can otherwise compromise sensitive data in transit. Adding a strict-transport-security header to your server configuration typically includes specifying how many seconds the browser should remember to use HTTPS and whether subdomains should also be covered.

*Pro Tip: Don't use the HSTS **includeSubdomains** directive if you have subdomains or applications that use HTTP!*

Harden session cookies using secure attributes to protect against interception and tampering

Importance: Critical **Applies To: All Websites**

Set cookies up with both the secure and HTTP-only flags when appropriate. This approach ensures cookies are sent only via HTTPS connections and keeps Javascript from viewing cookie values, respectively. Consider adding cryptographic validation, such as digital signatures, for sensitive use cases as an additional layer of tamper protection.

More Info: See https://developer.mozilla.org/en-US/docs/Web/Security/Practical_implementation_guides/Cookies.

Load all pages over secure HTTPS connections

Importance: Critical **Applies To: All Websites**

Configure your website/web application to load all pages using the secure protocol HTTPS. Redirect all HTTP URLs to HTTPS, which ensures that data is transmitted securely between your server and a connecting party via an encrypted connection.

Eliminate mixed content errors

Importance: Critical **Applies To: All Websites**

Ensure all resources (e.g., scripts, images, stylesheets, etc.) are loaded over secure HTTPS connections to prevent man-in-the-middle (MitM) attacks.

More Info: GeoTrust Horizon's SSL Monitor alerts you if your website has insecure content.

Deploy content security policies (CSPs)

Importance: High **Applies To: All Websites**

Use content security policy response headers to restrict third-party content, mitigate cross-site scripting (XSS) attacks, and prevent mixed content-related vulnerabilities.

Pro Tip: Check your CSP response headers at https://securityheaders.com.

Category #6: Incident Response

Carry out timely certificate revocation and key replacement

Importance: High **Applies To: All Websites**

Have clear procedures for revoking SSL/TLS certificates in the event of a key compromise. This includes everything from identifying the breach and notifying the issuing CA to replacing the compromised key. (If a private key may have been compromised or exposed, you must revoke all associated certificates immediately to minimize the impact.) This process relies heavily on you having a current inventory of all certificates and keys within your environment and knowing which party (or parties) are responsible for managing them.

Define and document a crypto-incident response plan ahead of time

Importance: High **Applies To: All Websites**

Clearly document your organization's procedures for handling compromises of digital certificates and their corresponding cryptographic keys. This should include everything from documenting who has the authority to declare a key compromise or other incidents, knowing who must be notified (and within what timeframe), and how this information will be communicated. Having these tools in place will help avoid costly delays when things (inevitably) go wrong.

Maintain backup certification authorities

Importance: Medium **Applies To: All Websites**

Use a multi-CA certificate provider or establish relationships with backup CAs to enable a rapid transition in the event of a primary CA compromise.

Pro Tip: GeoTrust Horizon is a multi-CA provider platform! Learn more: https://www.geotrust.com/geotrust-horizon-platform.

Category #7: TLS Traffic Management

Note: Skip this section if you don't inspect outbound/inbound TLS traffic.

Implement TLS traffic inspection judiciously

Importance: Medium **Applies To: Websites with Self-Managed Server(s)**

If TLS traffic inspection is required within your organization, it should be implemented selectively. Not all types of data should be inspected and decrypted. Traffic associated with sensitive categories (e.g., healthcare- and banking-related information) should be excluded, as decrypting these types of data poses significant data privacy and compliance concerns. There are two general approaches to implementing traffic inspection — one involves transporting a key to the inspection device and the other entails generating a key on the inspection device itself.

Preferred method: Generate dedicated digital certificates and keys for inspection devices

Importance: Medium **Applies To: Websites with Self-Managed Server(s)**

Rather than transporting a private key from production, the best approach is to set up an inspection device so that a digital certificate and its corresponding key pair can be generated for it. This can be done using an internal (private) CA (i.e., a re-signing CA that acts as a forward-trust proxy) that is trusted across your IT infrastructure. The idea is to limit the potential blast radius if the inspection device gets compromised.

Fallback method: Secure private keys if transported for inspection

Importance: Medium **Applies To: Websites with Self-Managed Server(s)**

If a re-signing CA isn't feasible and you must transport a key, it's imperative to have secure, documented methods established for transporting your cryptographic private keys. Secure methods include using SCP/SFTP, a secrets vault or manager, etc. However, it's worth noting that this approach only works with static (i.e., non-ephemeral) RSA key exchange methods and isn't compatible with modern cipher suites that support PFS as a result.

Category #8: Performance Optimization

Use session resumption to reduce TLS handshake overhead

Importance: Medium **Applies To: All Websites**

Carefully implement session resumption for returning connections to reduce overhead and improve website performance. This approach allows returning clients to skip the full TLS handshake process, minimizing latency. There are two main approaches to session resumption: using session IDs (i.e., server-cached negotiated keys/parameters) or session tickets (i.e., server-encrypted session state data that the client holds onto) to skip the full TLS handshake. If using the latter (as is required for TLS 1.3), keep ticket lifetimes short and rotate the ticket-encryption key regularly.

Optimize your network and infrastructure to reduce TLS handshake burden

Importance: Medium **Applies To: All Websites**

Use a combination of techniques to reduce your SSL/TLS handshake overhead and latency. Adopt HTTP/2 (multiplex multiple requests using a single TLS connection); use persistent connections (i.e., “keep-alives”) for subsequent client connections; and use a content delivery network (CDN) to reduce handshake roundtrip time (by terminating connections close to users).

Consider hardware acceleration to optimize TLS performance

Importance: Optional **Applies To: Websites with Self-Managed Server(s)**

When provisioning server hardware, ensure that whatever you choose offers hardware-accelerated AES support (i.e., AES New Instructions, or AES-NI) for better TLS performance. Because many modern operating systems (OS) and TLS libraries use AES-NI automatically to offload some of the encryption and decryption processes onto your hardware, it’s good to verify that your new hardware actually supports it.

Note: This point may not apply to SMBs using cloud or managing hosting where the provider already handles AES-NI for you. The scope is really limited to organizations with on-prem and self-managed hardware.