IT Security Procedural Guide: SSL/TLS Implementation CIO-IT Security-14-69

Revision 4

May 26, 2020

Office of the Chief Information Security Officer

VERSION HISTORY/CHANGE RECORD

Change Number	Person Posting Change	Change	Reason for Change	Page Number of Change
		Initial Version – December 24, 2014		
N/A	ISE	New guide created		
		Revision 1 – March 15, 2016		
1	Salamon	Administrative updates to align/reference to the current version of the GSA IT Security Policy and to CIO-IT Security-09-43, *IT Security Procedural Guide: Key Management*	Clarify relationship between this guide and CIO-IT Security-09-43	2-4
2	Berlas / Salamon	Updated recommendation for obtaining and using certificates	Clarification of requirements	7
3	Salamon	Integrated with OMB M-15-13 and related TLS implementation guidance	New OMB Policy	9
4	Berlas / Salamon	Updates to clarify TLS protocol recommendations	Clarification of guidance	11-12
5	Berlas / Salamon	Updated based on stakeholder review / input	Stakeholder review / input	Throughout
6	Klemens/ Cozart-Ramos	Formatting, editing, review revisions	Update to current format and style	Throughout
		Revision 2 – October 11, 2016		
1	Berlas / Salamon	Allow use of TLS 1.0 for certain server through June 2018	Clarification of guidance	Throughout
		Revision 3 – April 30, 2018		
1	Berlas / Salamon	Remove RSA ciphers from approved cipher stack	ROBOT vulnerability affected these ciphers	4-6
2	Berlas / Salamon	Requirement for valid Subject Alternative Names (SAN)	Chrome 58 requirement	7
3	Berlas / Salamon	Remove 3DES from approved cipher stack and reinforced other BOD 18-01 mandates	BOD 18-01 mandated removal	Throughout
		Revision 4 – May 26, 2020		
1	Richards	Updated references and minor language clarifications	Scheduled update	Throughout
2	Richards	Updated throughout for NIST SP 800-52 revision	Scheduled update	Throughout

Approval

IT Security Procedural Guide: SSL/TLS Implementation Guide CIO-IT Security-14-69, Revision 4 is hereby approved for distribution.

X DocuSigned by: Bo Berlas FD717926161544F...

Bo Berlas
Chief Information Security Officer

Contact: GSA Office of the Chief Information Security Officer (OCISO), Security Engineering Division (ISE) at SecEng@gsa.gov

Table of Contents

1 Introduction

The Transport Layer Security (TLS) protocol is used to secure communications in a wide variety of online transactions, including but not limited to financial (e.g., banking, trading stocks, e-commerce), healthcare (e.g., viewing medical records or scheduling medical appointments), and social (e.g., email or social media). All network services, whether or not they handle Personally Identifiable Information (PII), financial data, and/or login information need to protect the confidentiality and integrity of the transmitted information. TLS provides a protected channel for sending data between a server and the client. The client is often, but not always, a web browser. TLS is based on an older protocol called Secure Sockets Layer (SSL), and is considered to be an improvement over its predecessor. While SSL 3.0 is the most secure of the SSL protocol versions, it is not approved by the National Institute of Standards and Technologies (NIST) for use in the protection of federal information because it relies in part on the use of cryptographic algorithms that are not approved. TLS versions 1.1 and 1.2 are approved for the protection of federal information, when properly configured. TLS version 1.0 is approved only when it is required for interoperability with non-government systems and is configured according to these guidelines.

TLS is a security protocol that runs on top of a reliable transport layer protocol – typically the Transmission Control Protocol (TCP). Application layer protocols such as the Hypertext Transfer Protocol (HTTP) and the Internet Message Access Protocol (IMAP) can leverage TLS. TLS is application independent and is used to provide security for any two communicating applications that transmit data over a network via an application layer protocol. A virtual private network (VPN) can use TLS to securely connect an external system to an internal network, allowing that system to access a multitude of internal services and resources as if it were an internal system.

NIST Special Publication (SP) 800-52, Rev. 2, provides guidance for the selection and configuration of TLS protocol implementations while making effective use of Federal Information Processing Standards (FIPS) and NIST-recommended cryptographic algorithms. It requires that TLS 1.2 configured with FIPS-based cipher suites be supported by all government TLS servers and clients. This Special Publication also provides guidance on certificates and TLS extensions that impact security. Support for TLS 1.3[1] is strongly recommended.

[1] Agencies shall support TLS 1.3 by January 1, 2024. After this date, servers shall support TLS 1.3 for both government-only and citizen or business-facing applications.

1.1 Purpose

The recommendations in this guide aim to facilitate more consistent and secure implementations of SSL/TLS throughout GSA applications and systems, including use of approved protocols, FIPS 140-2[2] validated cryptographic modules, FIPS-approved ciphers, and related configuration best practices. This guide is not platform specific but instead provides a framework for testing web servers using SSL Labs to ensure secure SSL/TLS implementations.

1.2 Scope

This implementation guide addresses both NIST and commercial best practice methodologies for securely configuring TLS, including the SSL Labs SSL and TLS Deployment Best Practices. Another industry best practice resource is the Open Web Application Security Project (OWASP) Transport Layer Protection Cheat Sheet. Additional TLS best practices are identified at the CIO.gov HTTPS-Only Standard site. Specific configuration information with related command-line switches for varying platforms is not provided as it is beyond the scope of this document. Please refer to Appendix A for vendor guidance in securing TLS implementations.

This document is a GSA procedural guide that should be followed. Deviations from the SSL/TLS configuration herein shall be coordinated with the GSA Office of the Chief Information Security Officer (OCISO), Security Engineering Division.

2 Policy

The following are applicable policy references from General Services Administration (GSA) Order CIO 2100.1, "*GSA Information Technology (IT) Security Policy.*"

Chapter 4: Policy for Protect Function

1. Identity management, authentication and access control.

yy. Systems with a NIST SP 800-63-3 AAL of 2 or above used by Federal employees or contractors must accept Federal PIV cards and verify them IAW NIST SP 800-63-3 series requirements.

2. Awareness and training.

s. Users must avoid prohibited Internet usages including:

(7) Sending email messages including sensitive information, such as PII, as deemed by the Data Owner, without GSA provided encryption. Certified encryption modules must be used IAW FIPS 140-2, Security requirements for Cryptographic Modules.

[2] Please note that while FIPS 140-3 has been released, implementing guidance is still in progress and FIPS 140-2 certificates will continue to be issued.

3. Data security.

f. Web sites (internal and public) with logon functions, must implement TLS encryption with a FIPS 140-2 validated encryption module. SSL/TLS implementation must be IAW GSA CIO-IT Security-14-69: SSL/TLS Implementation Guide.

g. All sensitive information, such as PII, as deemed by the data owner, which is transmitted outside the GSA firewall, must be encrypted. Certified encryption modules must be used IAW FIPS 140-2, Security requirements for Cryptographic Modules.

4. Information protection processes and procedures.

b. GSA information systems, including vendor owned/operated systems on behalf of GSA, must configure their systems in agreement with GSA technical guidelines, NIST guidelines, Center for Internet Security guidelines (Level 1), or industry best practice guidelines, as deemed appropriate. Where a GSA benchmark exists, it must be used. GSA benchmarks may be exceeded but not lowered.

3 Achieving FIPS 140-2 Compliant Encryption

FIPS 140-2 compliant encryption is achieved when the following conditions are met:

1. Implement FIPS 140-2 Encryption Modules AND enable the FIPS 140-2 Object Module
2. Implement Secure Protocols
3. Implement FIPS-approved Ciphers
4. One or both sides of the communication session (client and/or server) must be set up in FIPS mode

The related set of NIST SP 800-53, Revision 4, "*Security and Privacy Controls for Federal Information Systems and Organizations*" controls include but are not limited to:

- IA-2(8) Identification and Authentication (Organizational Users) | Network Access To Privileged Accounts – Replay Resistant
- IA-2(9) Identification and Authentication (Organizational Users) | Network Access To Non-Privileged Accounts – Replay Resistant
- IA-7 Cryptographic Module Authentication
- SC-8 Transmission Confidentiality and Integrity
- SC-8(1) Transmission Confidentiality and Integrity | Cryptographic or Alternate Physical Protection
- SC-13 Cryptographic Protection

Additional information related to implementation of FIPS 140-2 compliant encryption can be found in CIO-IT Security-09-43, "*Key Management*."

3.1 Implement FIPS 140-2 Encryption Modules AND enable the FIPS 140-2 Object Module

TLS implementation must use FIPS 140-2 validated cryptographic modules in order to achieve FIPS compliance. NIST maintains a list of FIPS 140-2 Cryptographic Modules. A cryptographic module may either be an embedded component of a product or application, or an individual product in-and-of-itself. If the validated cryptographic module is a component of a larger product or application, one should contact the product or application vendor to determine how the product utilizes the embedded cryptographic module.

It is not sufficient to simply have a FIPS 140-2 validated cryptographic module; it must also be enabled. Appendix A includes references to vendor guidance to enable FIPS modules for several platforms. Additional information related to implementation of FIPS 140-2 compliant encryption can be found in CIO-IT Security-09-43, "*Key Management*."

3.2 Implement Secure Protocols

FIPS 140-2 compliant encryption requires the use of TLS 1.0 or higher. Government-only applications should use TLS 1.2 or higher.

There are six (6) protocols in the SSL/TLS family: SSL 2.0, SSL 3.0, TLS 1.0, TLS 1.1, TLS 1.2 and TLS 1.3. Of these:

- SSL 2.0 is insecure, must not be used, and is prohibited per DHS Binding Operational Directive (BOD) 18-01.
- SSL 3.0 is obsolete and is also prohibited per DHS Binding Operational Directive (BOD) 18-01. The algorithms used by SSL 3.0 are not approved by NIST and therefore GSA implementations must not support this protocol. According to analytics.usa.gov, older versions of Internet Explorer (e.g., IE6) which do not support TLS are rarely being used anymore. Additionally, SSL 3.0 is vulnerable to exploits (e.g., POODLE attack) which can break the cryptographic security of SSL 3.0 for clients and servers supporting the protocol.
- TLS 1.0 and TLS 1.1 are more secure than their SSL predecessors but are still vulnerable to exploits (e.g., BEAST and Klima attacks). While the use of TLS 1.0/1.1 is not recommended, support for these versions may be necessary to enable interaction with the private sector. Government-only applications are generally accessed by devices that support modern web browsers and therefore do not need to support TLS 1.0/1.1. The decision to support TLS 1.0/1.1 must be technically evaluated on a case-by-case basis.
- TLS 1.2 and TLS 1.3 are more secure protocols which include several cryptographic enhancements aimed to mitigate threats that have been discovered over time. TLS 1.2/1.3 protocols are recommended for GSA implementations.

3.3 Implement FIPS-approved ciphers

TLS implementations should use FIPS-approved ciphers. Implemented ciphers should be stacked with the most secure ciphers presented first, and least secure ciphers presented last. See below for cipher stacks and order of preference for TLS 1.0/1.1, TLS 1.2, and TLS 1.3, respectively. Forward secrecy is a property of a secure communication protocol where the compromise of long-term keys does not compromise past session keys. All FIPS-approved ciphers are from the ECDHE suites that provide forward secrecy. See SSL Labs SSL and TLS Deployment Best Practices and Appendix B for further details.

For TLS 1.0/1.1 implementations, NIST SP 800-52 and SSL Labs jointly recommend the use of the following FIPS-approved ciphers in this order of preference (from highest to lowest), with the exception of GSA's removal of 3DES[3] in accordance with DHS Binding Operational Directive (BOD) 18-01 and removal of the specific RSA ciphers (i.e., TLS_RSA) exploited by the ROBOT attack:

1. TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
2. TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
3. TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
4. TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA

For TLS 1.2 implementations, NIST SP 800-52 and SSL Labs jointly recommend the use of the following FIPS-approved ciphers in this order of preference (from highest to lowest), with the exception of GSA's removal of 3DES in accordance with DHS Binding Operational Directive (BOD) 18-01 and removal of the specific RSA ciphers (i.e., TLS_RSA) exploited by the ROBOT attack:

1. TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
2. TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
3. TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
4. TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
5. TLS_ECDHE_ECDSA_WITH_AES_256_CCM
6. TLS_ECDHE_ECDSA_WITH_AES_128_CCM
7. TLS_ECDHE_ECDSA_WITH_AES_256_CCM_8
8. TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8
9. TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
10. TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
11. TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA

[3] While RC4 is also prohibited by BOD 18-01, it is not FIPS-approved and was never part of the NIST SP 800-52 cipher stacks.

12. TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
13. TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
14. TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
15. TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
16. TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA

For TLS 1.3[4] implementations, NIST SP 800-52 and SSL Labs jointly recommend the use of the following FIPS-approved ciphers in this order of preference (from highest to lowest):

1. TLS_AES_256_GCM_SHA384
2. TLS_AES_128_GCM_SHA256
3. TLS_AES_128_CCM_SHA256
4. TLS_AES_128_CCM_8_SHA256

Conditions which explicitly allow for deviations from the afore-listed cipher stacks are identified as follows:

1. If your web server no longer supports a particular cipher, it may be omitted from the cipher stack.
2. If a new cipher is available with a larger key-length (but no other changes), it may be used and should be inserted directly above the smaller key-length cipher in the order of preference. Using the TLS 1.3 stack as an example, if a TLS_AES_256_CCM_SHA384 cipher was released, it could be inserted in the third position in the cipher stack, making TLS_AES_128_CCM_SHA256 fourth out of a total of (now) five (5) ciphers.
3. A cipher stack may become outdated if an attack emerges on one of the preferred cipher components. In the event that the integrity of a cipher is compromised, NIST will address the issue in an announcement on the NIST Computer Security Resource Center website.

3.4 Enable FIPS Mode

FIPS mode needs to be enabled on one or both sides of the communication session (i.e., client and/or server). Enabling FIPS mode on the client ensures that the workstation will use strong cryptographic algorithms that are FIPS 140-2 compliant for encryption, hashing, and signing. If FIPS mode is not enabled on client workstations, then the requirement will need to be implemented on the server-side to allow for FIPS-compliant connections to be made.

[4] These cipher suites may be used with either RSA or ECDSA server certificates; DSA and DH certificates are not supported by TLS 1.3.

4 SSL/TLS Best Practices

The following section describes additional SSL/TLS best practices, many of which are adapted from SSL Labs SSL and TLS Deployment Best Practices. Another industry best practice reference is the OWASP Transport Layer Protection Cheat Sheet.

4.1 Disable Client-Initiated Renegotiation

In TLS, renegotiation allows parties to stop exchanging data in order to renegotiate how the communication is secured. There are some cases in which renegotiation needs to be initiated by the server, but there is no known need for clients to do so. Furthermore, allowing client-initiated renegotiation may make your servers more vulnerable to Denial of Service (DoS) attacks.

4.2 Disable TLS Compression

In 2012, the Compression Ratio Info-leak Made Easy (CRIME) attack demonstrated how information leakage introduced by TLS compression can be used by attackers to uncover parts of sensitive data (e.g., session cookies). Very few clients supported TLS compression back then and even fewer support it now, which means that it is unlikely that you will experience any performance issues by disabling TLS compression on your servers.

4.3 Ensure that the Server Certificate Is Valid, Secure, and from a Trusted Source

The server certificate is often the weakest component of TLS implementation. A certificate that is not trusted (i.e., is not ultimately signed by a well-known Certificate Authority (CA)) will fail to prevent Man-In-The-Middle (MITM) attacks and renders TLS effectively useless. A certificate that is invalid in some other way (e.g., a certificate that has expired) erodes trust and, in the long term, jeopardizes the security of the Internet as a whole. These concerns apply to internal server certificates as well. Certificates obtained should be low cost, automatable, short-lived, and published to Certificate Transparency logs. Domain Validation (DV) certificates are usually less expensive and more suitable to automation than "*Extended Validation*" (EV) certificates. Use of EV certificates will generally make the domain owner's name visible (e.g., in the URL) to website visitors. **Ordinary DV certificates are completely acceptable for government use.** Internal sites (i.e., not publicly accessible) may utilize certificates signed by GSA's Root CA.

While there is no authoritative list of trusted certificate providers, the Federal Public Key Infrastructure (FPKI) maintains a list of Certified Shared Service Providers. Although FPKI certificates **shall not** be used for publicly accessible web services, since they are not widely accepted, their trusted vendors list can be a starting point in identifying where to obtain certificates for publicly accessible web services.

According to NIST, the certificate validity period should be (3) years or less. An exception to this rule would be for root certificates where best practice is to ensure that key lifetimes are long enough to avoid renewal problems for issued certificates. However, for most certificates,

shorter lifetimes are always preferred. There is no minimum validity period for a certificate, and administrators should choose the shortest validity lifetime that they can responsibly manage. For example, some commercial certificate authorities are beginning to issue certificates that are valid for only (90) days, or even less. Use of such short-lived certificates may only be viable with automation, but any program office that can accomplish such automation and maintain valid short-lived certificates is encouraged to do so.

Additionally, NIST SP 800-57, Part 1, Revision 4, page 54, states: "*SHA-1 has been demonstrated to provide less than 80 bits of security for digital signatures, which require collision resistance; at the publication of this Recommendation* [January 2016]*, the security strength against digital signature collisions remains the subject of speculation."* The use of SHA-1 is not recommended for the generation of digital signatures in new systems; new systems should use one of the larger hash functions. **Furthermore, existing certificates using SHA-1 should be replaced as soon as possible (including any GSA root certificates).** System owners may consider the implications of the removal of SHA-1 support which would include the inability of older clients (prior to Windows XP SP3 and Android 2.3) to connect to the server. However, currently, most (if not all) GSA web services are unlikely to have a significant user base using such older clients.

Due to updates made starting in Chrome 58, all certificates must include the domain in the *Subject Alternative Name (SAN)* or it will be identified as invalid. Based on this, **all certificates (including both internal and external) must include the domain in the *Subject Alternative Name*.**

In summary, all TLS certificates should be free of the following:

- Certificate expired
- Certificate is self-signed
- Certificate key is insecure
- Certificate not yet valid
- Certificate revoked
- Certificate signature is insecure (MD2/MD5 or SHA-1 [for new certificates])
- Certificate that is not trusted (unknown CA or some other validation error)
- Certificate valid for more than (3) years
- Domain name mismatch (including Subject Alternative Name (SAN) errors)

4.4 Certificate Transparency

Web server certificates should be submitted to public Certificate Transparency (CT) logs wherever feasible. Certificate Transparency is a mechanism to allow domain owners to detect misissuance of certificates, and its use by browser vendors and commercial certificate authorities is growing rapidly.

Some commercial certificate authorities automatically submit many or all certificates they issue to CT logs, but many do not. Web server owners are encouraged to choose commercial certificate authorities which automatically publish all certificates to CT logs. When using a certificate authority which does not do this, web server owners should alternatively submit all certificates they have been provisioned to CT logs, whenever possible.

4.5 Ensure Sufficient Hostname Coverage

Ensure that your server certificates cover all the names you wish to use for a website. For example, your primary domain name is *www.examplegsasite.gov*, but you may also have *support.examplegsasite.gov* configured. Your goal is to avoid invalid certificate warnings which will confuse your users and weaken their trust. Even when there is only one domain name configured on your web servers, remember that you cannot control how visitors arrive at the site or how others may link to the site. In most cases, you should ensure that the server certificate works with and without the www prefix (e.g., for both *examplesite.gov* and *www.examplesite.gov*). The rule of thumb is this: a secure web server should have a certificate that is valid for every DNS name configured to point to it. Wildcard certificates may be suitable, but their usage should be avoided if it results in exposing the underlying keys to a larger group of people, and especially if crossing organizational boundaries. In other words, the fewer people who have access to the private keys, the better.

4.6 Protect Private Keys

Treat your private keys as an important asset, restricting access to the smallest possible group of users while still keeping the arrangements practical.

Recommended policies include the following:

- Generate private keys and Certificate Signing Requests (CSRs) on a trusted computer. Some CAs may offer to generate keys and CSRs for you, but that is ill-advised.
- Never use email to transmit an unencrypted private key or the decryption passphrase to an encrypted private key. In general, do not use email for anything related to private key management.
- Password-protect keys to prevent compromise when they are stored in backup systems.
- After compromise, revoke old certificates and generate new keys to use with new certificates.
- Renew certificates every year and always do so with new private keys.

Note: .CER and .CRT file extensions, commonly used for certificates, are blocked from being sent as email attachments. In almost all modern operating systems, double-clicking or otherwise opening files with these extensions will prompt the user to install them into the OS's trusted certificate store. This is a potential vector of compromise as a malicious certificate would enable a MITM attack to go undetected. For this reason, these file extensions are blocked as email attachments. System owners/developers attempting to import certificates into the network should use encrypted email or some other delivery mechanism.

4.7 Encrypt 100% of Site and Avoid Mixed Content

The only way to reliably protect web site communication is to enforce encryption throughout without exception, to include all files transmitted by third parties. Web sites often use third-party services activated via JavaScript code downloaded from another server. The fact that encryption is optional is probably one of the biggest security problems today. The following problems are common:

- No TLS on sites that need it;
- Sites that have TLS but do not enforce it;
- Sites that mix TLS and non-TLS content (sometimes even within the same page); and
- Sites with programming errors that subvert TLS

Mixed-content pages are those that are transmitted over TLS but include some resources (e.g., JavaScript files, images, CSS files) that are not transmitted over TLS. Such pages are not secure. An active MITM attacker can piggyback on a single unprotected JavaScript resource, for example, and hijack the entire user session. Even if you encrypt your entire web site, you should ensure that you are not retrieving resources unencrypted from third-party web sites.

According to OMB M-15-13, "*Policy to Require Secure Connections across Federal Websites and Web Services*", agencies were required to make all existing websites and services accessible through a secure connection (HTTPS-only, with HSTS) by December 31, 2016. All newly developed websites and services must use HTTPS-only upon launch. HTTPS-only requirements were reinforced in DHS Binding Operational Directive (BOD) 18-01 in October 2017.

4.8 Disable Insecure HTTP Compression (if possible)

HTTP compression should be disabled, if possible. Two variations of the CRIME attack were disclosed in 2013. Rather than exploit TLS compression (which is what the CRIME attack did), TIME and BREACH attacks target secrets in the HTTP response bodies compressed using HTTP compression. Given that HTTP compression is very important to many organizations, alternative configurations may be somewhat difficult to accept. Mitigation might actually require changes to application code. TIME and BREACH attacks both require significant resources to carry out. But, if an adversary is motivated enough to perpetrate these types of attacks, the payoff would be equivalent to that achieved by cross-site request forgery (CSRF).

Please note, for web services using HTTP/2, the default compression algorithm, HPACK, is permitted. If subsequent versions of HTTP also implement trusted compression algorithms not vulnerable to the attacks as described above, then these algorithms will be permitted as well.

4.9 Implement TLS_FALLBACK_SCSV

TLS_FALLBACK_SCSV mitigates protocol downgrade attacks for the TLS protocol. TLS_FALLBACK_SCSV addresses the broader issue of downgrade attacks across protocol versions (including those currently unknown).

4.10 Implement HTTP Strict Transport Security (HSTS)

Websites and services available over HTTPS must enable HTTP Strict Transport Security (HSTS) to instruct compliant browsers to assume HTTPS going forward. This reduces the number of insecure redirects and protects users against attacks that attempt to downgrade connections to plain HTTP.

Once HSTS is in place, domains can be submitted to a "*preload list*" used by all major browsers to ensure that the HSTS policy is in effect at all times. Preloading provides the strongest guarantee of security for an entire domain. However, preloading generally requires that the HSTS policy contains an "*includeSubDomains*" component, which requires all subdomains (even those used as intranet sites) of a given parent domain to, at a minimum, support valid HTTPS.

The CIO.gov HTTPS-Only Standard Compliance Guide recommends preloading, and notes that HSTS policies are not required for each individual subdomain when a parent domain has been successfully preloaded.

According to OMB M-15-13, agencies were required to configure all existing websites with HSTS by December 31, 2016. All newly deployed websites and services must implement HSTS upon launch. DHS Binding Operational Directive (BOD) 18-01, issued in October 2017, reinforced this requirement.

5 Additional NIST SP 800-52 TLS Server Recommendations

Section 3 of NIST Special Publication (SP) 800-52, Rev. 2, identifies its recommendations for TLS server configurations. We have highlighted several of these recommendations below for particular consideration in this section. A checklist with all of the relevant configurations is available in Appendix A of this guide.

Please note that where there is a conflict between GSA guidance and NIST SP 800-52 guidance, the GSA guidance should take precedence.

5.1 All TLS Server Certificates Shall Be X.509 Version 3 Certificates

NIST requires that all TLS server certificates shall be X.509 version 3 certificates. This requirement is reinforced by the FPKIPA X.509 Version 1.32, "*Certificate Policy For The U.S. Federal PKI Common Policy Framework*" which states in Section 7.1.1 that the CA shall issue X.509 v3 certificates (populate version field with integer "*2*").

5.2 The TLS Server Should Not Support Client Certificate URL Extension

The Client Certificate URL extension allows a client to send a URL pointing to a certificate rather than sending a certificate to the server during mutual authentication. This can be very useful for mutual authentication with constrained clients. However, this extension can be used for malicious purposes. The URL could belong to an innocent server on which the client would like

to perform a denial of service attack, turning the TLS server into an attacker. A server that supports this extension also acts as a client while retrieving a certificate, and therefore, becomes subject to additional security concerns. For these reasons, the Client Certificate URL extension, in most cases, should not be supported.

If an Authorizing Official (AO) determines that the risks are minimal and this extension is needed for environments where clients are in constrained devices, the extension may be supported. If the client certificate URL extension is supported, the server shall be configured to mitigate the security concerns described above and in Section 11.3 of RFC6066, "*Transport Layer Security (TLS) Extensions: Extension Definitions.*"

5.3 TLS Servers Supporting Client Authentication Shall Support Certificate-Based Client Authentication

Per OMB M-11-11, "*Continued Implementation of Homeland Security Presidential Directive (HSPD) 12 – Policy for a Common Identification Standard for Federal Employees and Contractors*" (effective February 2011), all new systems under development must be enabled to use PIV credentials, in accordance with NIST guidelines, prior to being made operational. This effectively means that client authentication should utilize PIV cards. If this is not possible, client authentication should use multi-factor authentication (MFA) under the TLS protocol rather than rely on username/password alone.

NIST SP 800-53, Revision 4, control IA-2 (12), *Identification and Authentication | Acceptance of PIV Credentials*, requires all information systems (FIPS 199 – Low, Moderate, and High) to accept and electronically verify Personal Identity Verification (PIV) credentials. The requirement applies to organizations implementing logical access control systems (LACS) and physical access control systems (PACS). Personal Identity Verification (PIV) credentials are those credentials issued by federal agencies that conform to FIPS Publication 201 and supporting guidance documents. OMB Memorandum 11-11 requires federal agencies to continue implementing the requirements specified in HSPD-12 to enable agency-wide use of PIV credentials.

Multi-factor authentication (MFA) with PIV credentials can be implemented for web applications through integration with GSA's Single Sign-on (SSO) solution (SecureAuth) over SAML 2.0, or integration with MAX.gov Authentication.

5.4 NIST SP 800-52 Guidelines Shall Be Used to Identify an Appropriate Source for Server Certificates

In addition to Sections 4.3 and 4.5 of this guide, NIST SP 800-52 provides guidelines on recommended attributes for certificates. These guidelines are summarized in Appendix B, Section B.1, 2.

5.5 Support for TLS 1.3

Per SP 800-52, NIST recommends the transition away from earlier versions of TLS protocols and standardizing around TLS 1.3 for web transmission. This protocol version is superior because it offers important features that are unavailable in earlier protocol versions. If your server platform and applications (or any intermediary device) do not yet support TLS 1.3, NIST recommends development of plans to facilitate the transition to support this protocol. NIST requires support for TLS 1.3 by January 1, 2024.

5.6 Only Support TLS 1.0/1.1 if Required for Non-Government Users

Per NIST SP 800-52, when interoperability with non-government systems is required, TLS 1.1 and TLS 1.0 may be supported. To be sure, NIST acknowledges that servers which support citizens or business-facing applications may be configured to support TLS 1.0/1.1 to enable interaction with the private sector. In addition, if TLS 1.0/1.1 is supported, the use of TLS 1.2 and 1.3 shall be preferred over TLS 1.0/1.1.

Note: *GSA prohibited support for TLS 1.0, effective June 2018.*

6 Assessment Resources for Web Servers

The configuration of all web servers that implement TLS should be reviewed to ensure that their settings are consistent with this guide. For publicly facing web servers, automated tests for many of the recommended settings are available from SSL Labs. A detailed explanation of the scoring methodology used for those automated tests is also available from the SSL Labs Server Rating Guide. For internally facing web servers, various command-line tools are available to perform assessments, many of which do not require local access to the server. Python-based SSLyze is one such tool that provides assessment results for many of the settings identified in this guide.

Appendix A – Vendor References for TLS Settings

The following vendor references will assist in identifying specific command-line syntax used in common platforms, including OpenSSL, Apache, Nginx, and IIS.

A.1 OpenSSL TLS Settings

- The OpenSSL Cookbook covers the most frequently used OpenSSL features and commands.
- Configuring SSL and TLS Protocols in OpenSSL.
- Configuring OpenSSL in FIPS Mode.
- Configuring the OpenSSL FIPS 140-2 Validated Module – Search for Certificate #1747.
- Configuring OpenSSL for Forward Secrecy.

A.2 Apache TLS Settings

- The Center for Internet Security (CIS) provides the Apache HTTP Server 2.4 Benchmark which includes, in Section 7, recommended TLS settings for Apache web servers.
- Documentation for various Apache versions.
- Configuring Apache for Forward Secrecy.

A.3 Nginx TLS Settings

- Configuring ngx_http_ssl_module.
- Configuring Nginx for Forward Secrecy.

A.4 IIS TLS Settings

- Securing Communications with Secure Socket Layer (SSL).
- Settings to enable use of FIPS Compliant Algorithms.

Appendix B – Checklist of NIST SP 800-52 Recommendations

This section contains a checklist of select NIST SP 800-52 recommendations related to TLS server installation, configuration, and administration. **Please note that where there is a conflict between GSA guidance and NIST SP 800-52 guidance, the GSA guidance should take precedence**. Also note that a subset of the following recommendations make references to document sections (*e.g., Section 3.2.2*) that are contained within NIST SP 800-52 and not within this guide.

B.1 Recommendations for TLS Server Installation and Configuration

The following summary of recommendations is meant to assist system administrators tasked with the installation and initial configuration of a TLS server implementation. Recommendations for TLS server configuration are as follows:

1. **Protocol Version Support**
    a. Servers that support government-only applications **shall** be configured to use TLS 1.2 and **should** be configured to use TLS 1.3 as well.
    b. Servers that support government-only applications **should not** be configured to use TLS 1.1 and **shall not** use TLS 1.0, SSL 3.0, or SSL 2.0.
    c. Servers that support citizen or business-facing applications (i.e., the client may not be part of a government IT system) **shall** be configured to negotiate TLS 1.2 and **should** be configured to negotiate TLS 1.3.
    d. Servers that support citizen or business-facing applications may be configured to use TLS 1.1 and TLS 1.0 when necessary to enable interaction with citizens and businesses, **but this is generally discouraged**.
    e. Servers that support citizen or business-facing applications **shall not** allow the use of SSL 2.0 or SSL 3.0.
    f. Agencies **shall** support TLS 1.3 by **January 1, 2024**. After this date, servers **shall** support TLS 1.3 for both government-only and citizen or business-facing applications.
    g. In general, servers that support TLS 1.3 **should** be configured to use TLS 1.2 as well. However, TLS 1.2 may be disabled on servers that support TLS 1.3 if it has been determined that TLS 1.2 is not needed for interoperability.
    h. Servers that incorrectly implement TLS version negotiation **shall not** be used.
2. **Server Keys and Certificates**
    a. The TLS server **shall** be configured with one or more public-key certificates and the associated private keys.
    b. TLS server implementations **should** support the use of multiple server certificates with their associated private keys to support algorithm and key size agility.

c. TLS servers **shall** be configured with an RSA signature certificate or an ECDSA signature certificate.
d. If the server is configured with an ECDSA signature certificate, either curve P-256 or curve P-384 **should** be used for the public key in the certificate.
e. TLS servers **shall** be configured with certificates issues by a CA that publishes revocation information in Online Certificate Status Protocol (OCSP) responses. The source(s) for the revocation information **shall** be included in the CA-issued certificate in the appropriate extension to promote interoperability.

3. Server Certificate Profile

a. TLS server certificate **shall** be an X.509 version 3 certificate; both the public key contained in the certificate and the signature **shall** provide at least 112 bits of security.
b. If the server supports TLS versions prior to TLS 1.2, the certificate **should** be signed with an algorithm consistent with the public key (*ref. Section 3.2.1*).
c. The extended key usage extension limits how the keys in a certificate are used. There is a key purpose specifically for server authentication, and the server **should** be configured to allow its use.
d. The recommended server certificate profile is listed in Table 3-1 (*ref. Section 3.2.1*). In the absence of agency-specific certificate profile requirements, this certificate profile **should** be used for the server certificate.

4. Obtaining Revocation Status Information for the Client Certificate

a. The server **shall** perform revocation checking of the client certificate when client authentication is used.
b. Revocation information **shall** be obtained by the server from one or more locations (*ref. Section 3.2.2*).
c. When the local store does not have the current or a cogent CRL or OCSP response and the OCSP responder and the CRL distribution point are unavailable or inaccessible at the time of the TLS session establishment, the server **should** either accept or reject a potentially revoked or compromised certificate according to agency policy.

5. Server Public-Key Certificate Assurance

a. It is generally necessary for TLS server certificates to be issued by CAs that only issue certificates in accordance with a certificate policy that specifies adequate security controls.
b. When an agency is obtaining a certificate for a TLS server for which all the clients are under the agency's control, the agency may issue the certificate from its own CA if it can configure the clients to trust that CA.
c. In other cases, the agency **should** obtain a certificate from a publicly-trusted CA (a CA that clients that will be connecting to the server have already been configured to trust).

6. Cipher Suites

a. Cipher suites specify the cryptographic algorithms that will be used for a session.

b. The server **shall** be configured to only use cipher suites that are composed entirely of NIST-approved cryptographic algorithms. A complete list of acceptable cipher suites for general use is provided (*ref. Section 3.3.1*), grouped by certificate type and TLS protocol version.

c. NIST is deprecating the use of RSA key transport as used in TLS. Some applications or environments may require the use of RSA key transport during a transition period.

d. Cipher suites that use the Triple Data Encryption Algorithm (TDEA, also written as 3DES) are no longer allowed due to the limited amounts of data that can be processed under a single key.

e. The server **shall** be configured to only use cipher suites for which it has a valid certificate containing a signature providing at least 112 bits of security.

f. Prefer cipher suites using ephemeral keys over static keys (i.e., prefer DHE over DH, and prefer ECDHE over ECDH). Ephemeral keys provide perfect forward secrecy. While the use of static keys is technically acceptable, the use of ephemeral key cipher suites is encouraged and preferred.

g. Prefer cipher suites using GCM or CCM modes over CBC mode. The use of an authenticated encryption mode prevents several attacks (*ref. Section 3.3.2*). Note that these are not available in versions prior to TLS 1.2.

h. Prefer cipher suites using CCM over CCM_8. The latter contains a shorter authentication tag, which provides lower authentication strength.

i. TLS 1.2 includes authenticated encryption modes and support for the SHA-256 and SHA-384 hash algorithms, which are not supported in prior versions of TLS.

j. Cipher suites are defined differently in TLS 1.3; they do not specify the key exchange algorithm.

k. TLS 1.3 cipher suites cannot be negotiated for TLS 1.2 connections, and TLS 1.2 cipher suites cannot be negotiated with TLS 1.3.

7. Implementation Considerations

a. TLS implementations **shall** use the bad_record_mac error to indicate a padding error when communications are secured using a CBC cipher suite.

b. Implementations **shall** compute the MAC regardless of whether padding errors exist.

c. TLS implementations **should** support constant-time decryption or near constant-time decryption to prevent timing attacks (e.g., Lucky 13 attack).

d. TLS implementations **shall** correctly decode the CBC padding bytes by not reusing SSL decoder code to process TLS data (mitigates POODLE attack vulnerability).
e. CBC-based attacks can be prevented by using AEAD cipher suites (e.g., GCM, CCM), which are supported in TLS 1.2.

8. Algorithm Support

a. It is important that the server is configured to only use NIST-recommended cipher suites.
b. For some server implementations, the only way to ensure that a server uses NIST-approved algorithms is to disable cipher suites that use other algorithms.

9. Validated Cryptography

a. The cryptographic module used by the server **shall** be a FIPS 140-validated cryptographic module.
b. All cryptographic algorithms that are included in the configured cipher suites and the random number generator **shall** be within the scope of the validation.
c. All server and client certificates **shall** contain public keys that offer at least 112 bits of security.
d. All server and client certificates and certificates in their certification paths **shall** be signed using key pairs that offer at least 112 bits of security and SHA-224 or a stronger hashing algorithm.
e. All ephemeral keys used by the client and server **shall** offer at least 112 bits of security.
f. All symmetric algorithms used to protect the TLS data **shall** use keys that offer at least 112 bits of security.
g. The FIPS 140 validation certificate for the cryptographic module used by the server **shall** indicate that the random bit generator (RBG) has been validated in accordance with the SP 800-90 series.
h. The validated random number generator **shall** be used to generate the random bytes of the server random value.
i. The validated random number generator **should** be used to generate the 4-byte timestamp of the server random value.

10. Mandatory TLS Extensions

a. Servers **should** only be configured to support extensions that are required by the application or that enhance security. Extensions that are not needed **should not** be enabled.
b. Enabling extensions increases the potential for implementation flaws and could leave a system vulnerable (e.g., Heartbleed bug).

c. The server **shall** support the use of the following TLS extensions: **1) Renegotiation Indication; 2) Server Name Indication; 3) Extended Master Secret; 4) Signature Algorithms; and 5) Certificate Status Request**

11. Conditional TLS Extensions

a. The Fallback Signaling Cipher Suite Value (SCSV) **shall** be supported if the server supports versions of TLS prior to TLS 1.2 and does not support TLS 1.3.
b. The Supported Groups extension **shall** be supported if the server supports ephemeral ECDH cipher suites or if the server supports TLS 1.3.
c. The Key Share extension **shall** be supported if the server supports TLS 1.3.
d. The EC Point Format extension **shall** be supported if the server supports EC cipher suites.
e. The Multiple Certificate Status extension **should** be supported if status information for the server's certificate is available via OCSP and the extension is supported by the server implementation.
f. The Trusted CA Indication extension **shall** be supported if the server communicates with memory-constrained clients (e.g., low-memory client devices in the Internet of Things) and the server has been issued certificates by multiple CAs.
g. The Encrypt-then-MAC extension **shall** be supported if the server is configured to negotiate CBC cipher suites.
h. The Truncated HMAC extension may be supported if the server communicates with constrained-device clients, cipher suites that use CBC mode are supported, and the server implementation does not support variable-length padding.
i. The Pre-Shared Key extension may be supported if the server supports TLS 1.3.
j. The Pre-Shared Key Exchange Modes extension **shall** be supported if the server supports TLS 1.3 and the Pre-Shared Key extension.
k. The Supported Versions extension **shall** be supported if the server supports TLS 1.3.
l. The Cookie extension **shall** be supported if the server supports TLS 1.3.
m. The Certificate Signature Algorithms Extension **shall** be supported if the server supports TLS 1.3 and **should** be supported for TLS 1.2.
n. The Post-handshake Client Authentication extension may be supported if the server supports TLS 1.3.
o. The Signed Certificate Timestamps extension **should** be supported if the server's certificate was issued by a publicly trusted CA and the certificate does not include a Signed Certificate Timestamps List extension.

12. Discouraged TLS Extensions

a. The following extensions **should not** be used: **1) Client Certificate URL; and 2) Early Data Indication**
b. The Raw Public Keys extension **shall not** be supported

13. Client Authentication

a. All TLS servers that perform client authentication **shall** implement certificate-based client authentication.
b. To ensure that cryptographic authentication actually results in strong authentication, client keys **shall** be capable of providing at least 112 bits of security.
c. The TLS server **shall** be configurable to terminate the connection with a fatal "handshake failure" alert when a client certificate is requested and the client does not have a suitable certificate.

14. Path Validation

a. The revocation status of each certificate in the certification path **shall** be validated using the Online Certificate Status Protocol (OCSP) or a certificate revocation list (CRL). OCSP checking **shall** be in compliance with RFC6960.
b. The server **shall** be able to determine the certificate policies that the client certificate is trusted for by using the certification path validation rules specified (*ref. Section 6 of RFC5280*).
c. The server **shall** be able to provide the client certificate and the certificate policies for which the client certification path is valid to consuming applications in order to support access control decisions.

15. Trust Anchor Store

a. Having an excessive number of trust anchors installed in the TLS application can expose the application to all the PKIs emanating from those trust anchors.
b. The server **shall** be configured only with trust anchors that the system owner trusts, and of those, only the ones that are required to authenticate the clients in the case where the server supports client authentication in TLS.
c. System administrators of a TLS server that supports certificate-bases client authentication **shall** perform an analysis of the client certificate issuers and use that information to determine the minimum set of trust anchors required for the server.
d. The default set of trust anchors **shall** be examined to determine if any of them are required for client authentication.
e. In the U.S. Federal Government environment, in most situations, the Federal Common Policy Root or the agency root (if cross-certified with the Federal Bridge Certification Authority or the Federal Common Policy Root) should be sufficient to build a certification path to the client certificates

16. Checking the Client Key Size

a. The server **shall** check the client key length if client authentication is performed, and the server implementation provides a mechanism to do so.
b. Federal agencies **shall** use the key size guidelines provided in SP 800-131A to check the client key size.

17. Server Hints List

a. The server **shall** either: 1) maintain the trust anchors of the various PKIs whose subscribers are the potential clients for the server and include them in the hints list; or 2) be configured to send an empty hints list so that the client can always provide a certificate it possesses.
b. The hints list **shall** be distinct from the server's trust anchor store.
c. The server **shall** continue to only populate its trust anchor store with the trust anchor of the server's PKI domain and the domains it needs to trust directly for client authentication.
d. Note that the distinction between the server hints list and the server's own trust store is as follows: 1) the hints list is the list of trust anchors that a potential client might trust, and 2) the server's trust store is the list of trust anchors that the server explicitly trusts.

18. Session Resumption and Early Data

a. Previous TLS sessions can be resumed, allowing for a connection to be established using an abbreviated handshake. All versions of TLS offer session resumption, although the mechanism for performing resumption differs.
b. If resumption is allowed, frequent key replacement and short lifetimes for resumption information are recommended, as applicable.
c. In general, 0-RTT data **should not** be accepted by the server.
d. If the server does allow 0-RTT data, then the server **should** use the single-use ticket mechanism in accordance with RFC8446.

19. Compression Methods

a. The use of compression may enable attackers to perform attacks using compression-based side channels.
b. To defend against these attacks, the null compression method **shall** be enabled, and all other compression methods **shall** be disabled.

20. Operational Considerations

a. Federal agencies **shall** ensure that TLS servers include appropriate network security protections as specified in other NIST guidelines, such as SP 800-53.
b. The server **shall** operate on a secure operating system.

c. Where the server relies on a FIPS 140 Level 1 cryptographic module, the software and private key **shall** be protected using the operating system identification, authentication, and access control mechanisms.
d. The server and associated platform **shall** be kept up-to-date in terms of security patches. This is critical to various aspects of security.