

NASA TECHNICAL SPECIFICATION

National Aeronautics and Space Administration

NASA-SPEC-2650 v4.0

Approved: 2022-02-07

Superseding: NASA-SPEC-2650 v3.0

**TRANSPORT LAYER SECURITY (TLS)
SECURITY CONFIGURATION SPECIFICATION**

TABLE OF CONTENTS

- Document History Log 1
- Foreword 2
- 1. Scope 3
 - 1.1. Purpose 3
 - 1.2. Applicability 3
 - 1.3. Tailoring 3
 - 1.4. Authority 3
- 2. Applicable Documents 4
 - 2.1. Government Documents 4
 - 2.2. Non-Government Documents 5
- 3. Acronyms, Abbreviations, and Definitions 6
- 4. Version and Lifecycle 8
- 5. Use Case Matrix 9
- 6. Servers 10
 - 6.1. Protocols 10
 - 6.1.1. Required Versions 10
 - 6.1.2. Prohibited Versions 10
 - 6.2. Extensions 10
 - 6.3. Certificates 11
- 7. Profile 14
 - 7.1. TLS 14
 - 7.1.1. Protocol Support 14
 - 7.1.2. Cipher Suite Support 14
- Appendix A: Implementation Examples 16
 - A.1. Apache 16
 - A.1.1. TLS Profile 16
 - A.2. Nginx 17
 - A.2.1. TLS Profile 17

DOCUMENT HISTORY LOG

Document Version	Approval Date	Description
4.0	2022-02-07	Windows 11, Windows Server 2022 added. Windows 10, Windows Server 2016/19 modified. Win7, Windows Server 2008 removed.
3.0	2020-02-20	Win10 settings modified
2.0	2018-08-24	Initial public release

FOREWORD

This NASA Technical Specification is published by the National Aeronautics and Space Administration (NASA) to describe technical requirements for purchased or in-house items, services, functions, or processes for NASA programs and projects.

This NASA Technical Specification is approved for use by NASA Headquarters and NASA Centers and Facilities, and applicable technical requirements may be cited in contract, program, and other Agency documents. This Specification also applies to the Jet Propulsion Laboratory (JPL) (a Federally Funded Research and Development Center (FFRDC)), other contractors, recipients of grants and cooperative agreements, and parties to other agreements only to the extent specified or referenced in applicable contracts, grants, or agreements.

Adherence to this NASA Technical Specification ensures compliance with NASA-STD-2601, *Minimum Cybersecurity Requirements for Computing Systems*, which defines operating system and application security requirements that must be implemented on NASA information systems. This Specification provides the enforceable, measurable details of NASA-STD-2601.

Requests for information, corrections, or additions to this Specification can be made via the "Contact ASCS" form, found here: <https://cset.nasa.gov/ascs/requests/>.

[Refer to NASA-SPEC-2600 - Enumeration of ASCS Cybersecurity Requirements \(Signed PDF\)](#)

Michael Witt

Senior Agency Information Security Officer

1. SCOPE

1.1. Purpose

The purpose of this NASA Technical Specification is to define version and configuration requirements for Transport Layer Security (TLS) deployment and operation. TLS is commonly used for web services, electronic mail, instant messaging, and other protocols in order to provide integrity and confidentiality of information conveyed by the protocol.

1.2. Applicability

This NASA Technical Specification is applicable to all computing systems.

This NASA Technical Specification is approved for use by NASA Headquarters and NASA Centers and Facilities, and applicable technical requirements may be cited in contract, program, and other Agency documents. This Specification also applies to JPL (an FFRDC), other contractors, recipients of grants and cooperative agreements, and parties to other agreements only to the extent specified or referenced in applicable contracts, grants, or agreements.

1.3. Tailoring

In accordance with NASA-STD-2601, *Minimum Cybersecurity Requirements for Computing Systems*, any and all risk-based decisions (RBDs) to tailor this NASA Technical Specification in order to meet the needs of a specific program, project, or system **SHALL** be approved by the responsible Information System Owner (ISO) and Authorizing Official (AO) and formally documented by the ISO or system administrators in the System Security Plan (SSP) under program or project requirements.

NASA-STD-2601 mandates that the AO **SHALL** ensure that only systems posing an acceptable level of risk to Agency assets, data, and personnel are approved for production operation and that the ISO **SHALL** ensure all necessary documentation is produced and maintained.

Note that some NASA Technical Specifications include "critical" configuration settings. "Critical" configuration settings **SHALL NOT** be eligible for any modifications through the application of an RBD.

1.4. Authority

The Agency Chief Information Officer (CIO) and Senior Agency Information Security Officer (SAISO) have authorized the Cybersecurity Standards and Engineering Team (CSET) via the Agency Security Configuration Standards (ASCS) initiative to create binding Technical Standards related to Agency cybersecurity topics.

The NASA Technical Standards Program (NTSP), sponsored by the Office of the NASA Chief Engineer, recognizes CSET as a standards-developing organization within the Agency. NTSP provides access to all technical standards at: <https://standards.nasa.gov/>.

2. APPLICABLE DOCUMENTS

The documents listed in this section contain provisions that constitute requirements of this NASA Technical Specification. These documents can serve as additional support in meeting the requirements defined in this Specification.



A suffix of x on a document identifier indicates that the latest issuance of the document applies.

2.1. Government Documents

Table 1. White House Releases

Document Identifier	Document Title
Executive Order	<i>Executive Order on Improving the Nation's Cybersecurity</i>

Table 2. NASA Documents

Document Identifier	Document Title
ICAM Certificate Information	<i>ICAM-PKI Server Certificate Information and Frequently Asked Questions (FAQ)</i>
NASA-STD-2601	<i>Minimum Cybersecurity Requirements for Computing Systems</i>
NPR 2810.1x	<i>Security of Information Technology</i>

Table 3. NIST Documents

Document Identifier	Document Title
SP 800-52r2	<i>Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations</i>

Table 4. Other Government Documents

Document Identifier	Document Title
BOD 18-01	<i>Binding Operational Directive - Department of Homeland Security: Enhance Email and Web Security</i>
HTTPS-Only	<i>The HTTPS-Only Standard — Certificates</i>
NSA Guidance	<i>Avoid Dangers of Wildcard TLS Certificates and the ALPACA Technique</i>

2.2. Non-Government Documents

Document Identifier	Document Title
Apple Security	<i>Apple Platform TLS Security</i>
Microsoft Security	<i>Cipher Suites in TLS/SSL (Schannel SSP)</i>

3. ACRONYMS, ABBREVIATIONS, AND DEFINITIONS

Table 5. Acronyms and Abbreviations

AO	Authorizing Official
ASCS	Agency Security Configuration Standards
BOD	Binding Operational Directive
CA	Certificate Authority
CAB/F BR	Certificate Authority Browser Forum Baseline Requirements
CIO	Chief Information Officer
CN	Common Name
CRL	Certificate Revocation List
CSET	Cybersecurity Standards and Engineering Team
DANE	DNS-Based Authentication of Named Entities
DNS	Domain Name System
ECDSA	Elliptic Curve Digital Signature Algorithm
FCPCA	Federal Common Policy Certificate Authority
FFRDC	Federally Funded Research and Development Center
FQDN	Fully Qualified Domain Name
HTTP	Hypertext Transfer Protocol
HTTPS	HTTP over TLS
ICAM	Identity, Credential, and Access Management
ISO	Information System Owner
JPL	Jet Propulsion Laboratory
NASA	National Aeronautics and Space Administration
NIST	National Institute of Standards and Technology
NOCA	NASA Operational Certificate Authority
NPR	NASA Procedural Requirement
NTSP	NASA Technical Standards Program
OCSP	Online Certificate Status Protocol
PKI	Public Key Infrastructure
POA&M	Plan of Action and Milestone
RBD	Risk-Based Decision
RFC	Request for Comment
RSA	Rivest–Shamir–Adleman (Cryptographic System)

SAISO	Senior Agency Information Security Officer
SAN	Subject Alternative Name
SHA-2	Secure Hash Algorithm 2
SNI	Server Name Indication
SP	Special Publication
SSL	Secure Sockets Layer
SSP	System Security Plan
TLS	Transport Layer Security

4. VERSION AND LIFECYCLE

Function	Encryption Protocol
Application	TLS

5. USE CASE MATRIX

This section contains all *required* security configuration settings as well as a description for each setting.

Required settings **SHALL** be adhered to, as they are monitored, scored, and reported by the Agency.

Required settings **MAY**, in limited and justified instances, be amended through the proper application of an RBD. To pursue an RBD, see [Section 1.3, "Tailoring"](#) in this Specification.

Identified below are the minimum TLS configurations required based on use case.

Table 6. Use Case Profile Matrix

System Type	Requirements
Server (NASA-STD-2824) <ul style="list-style-type: none"> • Windows Server • Linux Server 	<ul style="list-style-type: none"> • Server Requirements • TLS Profile
Client (NASA-STD-2804) <ul style="list-style-type: none"> • Windows 10/11 • macOS • Linux Workstation 	<ul style="list-style-type: none"> • TLS Profile

6. SERVERS

6.1. Protocols

6.1.1. Required Versions

The following TLS versions **SHALL** be implemented:

Table 7. Required Versions

TLS Version
1.3 (if supported)
1.2



Systems which cannot operate in TLS 1.2 or higher **SHALL** have a Plan of Action and Milestones (POA&M) to remedy this deficiency.

Per the National Institute of Standards and Technology (NIST) Special Publication (SP) 800-52 Rev 2, TLS 1.3 is approved for use on NASA systems. Section 3.1 states the following:

"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. 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."

6.1.2. Prohibited Versions

The following TLS and Secure Sockets Layer (SSL) versions **SHALL NOT** be implemented:



Table 8. Prohibited Versions

TLS Version	SSL Version
1.0	2.0
1.1	3.0

6.2. Extensions

The TLS service **SHALL** provide the following extensions:

Table 9. Required Extensions

Extension	Requirement
OCSP Stapling	<ul style="list-style-type: none"> The TLS service SHALL provide a stapled Online Certificate Status Protocol (OCSP) response upon TLS client request (i.e, at TLS session initialization). <div style="border-left: 1px solid #ccc; padding-left: 10px; margin-top: 10px;">  <p>With OCSP stapling, the server obtains and caches the certificate authority's (CA) OCSP response for the server certificate and bundles it into the TLS session initiation. This improves performance and privacy when using OCSP. It removes the need for the client to perform its own OCSP checking, and most clients do not use Certificate Revocation Lists (CRLs) for end-entity certificate validation.</p> </div>
Server Name Indication	<ul style="list-style-type: none"> The TLS service SHALL support Server Name Indication (SNI). <div style="border-left: 1px solid #ccc; padding-left: 10px; margin-top: 10px;">  <p>Apache and Nginx built with openssl version 0.9.8j and later support SNI.</p> </div>
Renegotiation Indication	<ul style="list-style-type: none"> The TLS service SHALL support Renegotiation Indication.
Trusted CA Indication	<ul style="list-style-type: none"> The TLS service SHALL support Trusted CA Indication.






6.3. Certificates







Identity, Credential, Access Management (ICAM) has provided [guidance](#) for selecting a CA and choosing certificate attributes. The same document also addresses the use of wildcard certificates stating the practice is prohibited. Wildcard certificates should not be used due to a vulnerability known as Application Layer Protocols Allowing Cross-Protocol Attack ([ALPACA](#)).

"Wildcard certificates are not permitted by Common Policy X.509 Certificate Policy or Treasury X.509 Certificate Policy. The two certificate policies can be found at <https://pki.treas.gov>. Those needing support for multiple Fully Qualified Domain Name (FQDN) sites on the same server, they can have multiple Subject Alternative Name (SAN) entries in the certificate instead of a wildcard."

Table 10. Certificate Attributes

Attribute	Requirement
RSA key encipherment certificate mandatory	<ul style="list-style-type: none"> The server SHALL be configured with an Rivest–Shamir–Adleman (RSA) key encipherment certificate.
Server Signature Certificate	<ul style="list-style-type: none"> The server SHOULD be configured with an Elliptic Curve Digital Signature Algorithm (ECDSA) signature certificate or RSA signature certificate.
ECDSA Suite B curves	<ul style="list-style-type: none"> If the server is not configured with an RSA signature certificate, an ECDSA signature certificate using a Suite B named curve for the signature and public key in the ECDSA certificate SHOULD be used. These curves have the names P-256 (aka secp256r1) and P-384 (aka secp384r1).

Attribute	Requirement
CA-issued certificate mandatory	<ul style="list-style-type: none"> The server SHALL be configured with certificates issued by a CA, rather than self-signed certificates. <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="margin-right: 10px;"></div> <div> <p>The Federal Common Policy Certificate Authority (FCPCA) root certificate, a trust anchor which is required for NASA-issued certificates to be trusted, does not appear in all commonly-encountered trust anchor compilations. Noteworthy absences of the FCPCA root are the Mozilla Trusted Root Program, used by Mozilla Firefox, Android, Linux, Java, and other trust anchor compilations.</p> </div> </div> <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="margin-right: 10px;"></div> <div> <p>Do not use the guidance found in Federal Public Key Infrastructure (PKI) Trust Store Management Guide. While there is some informative information, including the location of the FCPCA root, the instructions related to intermediate CAs are inappropriate.</p> </div> </div>
CA OCSP Required	<ul style="list-style-type: none"> Server certificates SHALL be issued by a CA that publishes revocation information in OCSP responses via a reliable low-latency OCSP responder.
CA Issuer CA/Browser Forum Baseline Requirements	<ul style="list-style-type: none"> The server SHOULD be configured with certificates issued by a CA which conforms to the Certificate Authority Browser Forum Baseline Requirements (CAB/F BR). <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="margin-right: 10px;"></div> <div> <p>NASA Operational Certificate Authority (NOCA) issued certificates do not conform to the CA/Browser Forum Baseline Requirements.</p> </div> </div>
RSA Key Length	<ul style="list-style-type: none"> RSA keys SHALL be 2048, 3072, or 4096 bits in length. <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="margin-right: 10px;"></div> <div> <p>Choose key length in proportion to perceived risk. 2048 bits is currently adequate. Larger RSA key lengths adversely affect TLS session initiation duration.</p> </div> </div>
ECDSA Key Length	<ul style="list-style-type: none"> ECDSA keys SHALL be ≥ 224 bits in length. ECDSA keys SHOULD be ≥ 256 bits in length. ECDSA keys SHOULD use the named curves P-256 or P-384. The named curve P-256 is RECOMMENDED. <div style="display: flex; align-items: flex-start; margin-top: 10px;"> <div style="margin-right: 10px;"></div> <div> <p>Curves P-256 and P-384 are widely supported.</p> </div> </div>

Attribute	Requirement
TLS service certificates must have a SHA-2 signature	<ul style="list-style-type: none"> • TLS service certificates SHALL be signed using a Secure Hash Algorithm 2 (SHA-2) hashing algorithm (SHA-256 or larger). <div style="display: flex; align-items: center; margin-top: 10px;">  <p>Contemporary browsers continue to implement strategies to deprecate, or even refuse, certificates issued which do not conform to this practice.</p> </div>
FQDN SAN entries	<ul style="list-style-type: none"> • Certificates SHALL be requested with one or more FQDN to appear in the certificate Subject Alternative Name (SAN) extension. • The names SHALL be constrained in accordance with the CAB/F BR §7.1.4.2.1. • The guidance in SP 800-52r1 §3.2.1 Table 3-1 regarding domain name in certificate Subject Common Name (CN) SHOULD be ignored. • The guidance in SP 800-52r1 §3.2.1 Table 3-1 regarding host IP address in certificate Subject CN and SAN SHALL be ignored. <div style="display: flex; align-items: center; margin-top: 10px;">  <p>The guidance in SP 800-52r1 §3.2.1 Table 3-1 regarding domain name in Subject CN is deprecated by the CAB/F BRs.</p> </div> <div style="display: flex; align-items: center; margin-top: 10px;">  <p>The guidance in SP 800-52r1 §3.2.1 Table 3-1 regarding host IP address in Subject CN and SAN is prohibited by the CAB/F BRs.</p> </div>
Certificate chain	<ul style="list-style-type: none"> • The server SHALL provide a correct certificate chain to the client. • The root CA certificate SHOULD NOT appear at the end of the chain. The only known exception is when Domain Name System (DNS) Based Authentication of Named Entities (DANE) is used, in which case the root CA certificate MUST appear at the end of the chain. <div style="display: flex; align-items: center; margin-top: 10px;">  <p>When DANE is not used, inclusion of the root CA certificate in the certificate chain will not cause clients to trust a server when they lack that root as a trusted root CA — mistrust will regardless prevail. The root CA certificate is not a talismanic amulet, nor will its absence in the chain result in persecution. All it does is bulk up the TLS initiation sequence and cause the server administrator to be regarded as superstitious or worse.</p> </div> <div style="display: flex; align-items: center; margin-top: 10px;">  <p>TLS clients will not trust a TLS server that provides an incorrect chain.</p> </div>
TLS service intermediate CA certificates should have a SHA-2 signature	<ul style="list-style-type: none"> • TLS service intermediate CA certificates SHALL be signed using a SHA-2 hashing algorithm (SHA-256 or larger). <div style="display: flex; align-items: center; margin-top: 10px;">  <p>Contemporary browsers continue to implement strategies to deprecate, or even refuse, certificates issued which do not conform to this practice.</p> </div>

7. PROFILE

7.1. TLS

This profile applies to all TLS services at NASA.

Implementation examples may be found in [Appendix A](#) of this document.

7.1.1. Protocol Support

The following TLS protocols **SHALL** be implemented:

Table 11. Required Versions

TLS Version
1.3 (if supported)
1.2

The following TLS and Secure Sockets Layer (SSL) protocol versions **SHALL NOT** be implemented:

Table 12. Forbidden Versions

TLS Version	SSL Version
1.0	2.0
1.1	3.0

7.1.2. Cipher Suite Support

The server **SHALL** be configured to offer cipher suites from the following list. Configuring the server to require all listed cipher suites is not mandatory.

INFO: On modern Linux distributions, the `openssl ciphers | sed 's:/\n/g'` command will display a list of current ciphers available.

Table 13. Permitted Cipher Suites

NASA Use	Microsoft Naming	Linux Naming
SHOULD	TLS_AES_256_GCM_SHA384 *	TLS_AES_256_GCM_SHA384 *
SHOULD	TLS_AES_256_GCM_SHA256 *	TLS_AES_128_GCM_SHA256 *
SHOULD	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	ECDHE-ECDSA-AES128-GCM-SHA256
SHOULD	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDHE-RSA-AES128-GCM-SHA256

NASA Use	Microsoft Naming	Linux Naming
SHOULD	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	ECDHE-ECDSA-AES256-GCM-SHA384
SHOULD	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	ECDHE-RSA-AES256-GCM-SHA384
MAY	TLS_DHE_RSA_WITH_AES_128_GCM_SHA256	DHE-RSA-AES128-GCM-SHA256
MAY	TLS_DHE_RSA_WITH_AES_256_GCM_SHA384	DHE-RSA-AES256-GCM-SHA384
MAY	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	ECDHE-ECDSA-AES128-SHA256
MAY	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	ECDHE-RSA-AES128-SHA256
MAY	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	ECDHE-ECDSA-AES256-SHA384
MAY	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	ECDHE-RSA-AES256-SHA384

When server cipher suite order is preferred, the cipher suites used SHOULD be ordered in the order in which they appear in the table. If server cipher suite order is not preferred, clients may choose inferior cipher suites.

* TLS 1.3 cipher(s)



For Microsoft operating systems, TLS 1.3 ciphers are only available in Windows 11 and Windows Server 2022.

APPENDIX A: IMPLEMENTATION EXAMPLES

The following are implementation examples for Apache and Nginx:

A.1. Apache



In Linux, finding the openssl build can be done with the following command:

```
openssl version
```

A.1.1. TLS Profile

Below is an example of a [TLS profile](#) configuration implementation in Apache:

Built with openssl 1.0.1+, but prior to openssl 1.1.1

```
SSLProtocol TLSv1.2

SSLHonorCipherOrder On

SSLCipherSuite ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-
AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
ECDSA-AES128-SHA:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA

SSLUseStapling on
SSLStaplingCache "shmcb:/run/httpd/ssl_stapling(32768)"

SSLInsecureRenegotiation off
```

Built with openssl 1.1.1+

```
SSLProtocol TLSv1.3

SSLHonorCipherOrder On

SSLCipherSuite ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-
AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
ECDSA-AES128-SHA:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA

SSLUseStapling on
SSLStaplingCache "shmcb:/run/httpd/ssl_stapling(32768)"

SSLInsecureRenegotiation off
```

A.2. Nginx



In Linux, finding the openssl build can be done with the following command:

```
openssl version
```

A.2.1. TLS Profile

Below is an example of a [TLS profile](#) configuration implementation in Nginx:

Build with openssl 1.0.1+, but prior to openssl 1.1.1

```
ssl_protocols TLSv1.2;
ssl_prefer_server_ciphers on;

ssl_ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-
AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384;

ssl_stapling on;
```

Build with openssl 1.1.1+

```
ssl_protocols TLSv1.3;
ssl_prefer_server_ciphers on;

ssl_ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-
AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384;

ssl_stapling on;
```