TLS

Upgrade Guideline

February 2020

hkpc 生產力局

HKCERT

Disclaimer

The Hong Kong Computer Emergency Response Team Coordination Centre (HKCERT) and the Hong Kong Productivity Council (HKPC) reserve the right to amend the document from time to time without prior notice.

While we have made every attempt to ensure that the information contained in this document is obtained from reliable sources, HKCERT is not responsible for any errors or omissions, or for the results obtained from the use of this information. All information in this document is provided "as is", with no guarantee of completeness, accuracy, timeliness or of the results obtained from the use of this information, and without warranty of any kind, express or implied, including, but not limited to warranties of performance, merchantability and fitness for a particular purpose.

The information contained in this document is intended to provide general information and for reference only. Reliance or use of this information shall be at the reader's own risk. Nothing herein shall to any extent substitute for the independent investigations and the sound technical and business judgment of the reader. In no event will HKCERT, HKPC or its partners, employees or agents, be liable to you or anyone else for any decision made or action taken in reliance on the information in this document, or for any consequential, special or similar damages, even if advised of the possibility of such damages.

Licence

Table of Contents

1. Executive Summary 4

2. Overview 5

2.1. Objectives 5

2.2. Target Audience 5

2.3. About TLS Upgrade 5

3. TLS and Cipher Suite Baseline 7

3.1. TLS Baseline 7

3.2. Cipher Suite Baseline 7

4. TLS Upgrade Steps 9

4.1. Inventory the Network and Software Assets that require TLS support 9

4.2. TLS Upgrade Pre-requisites Gap Analysis 11

4.3. Prepare Upgrade Plan with Strategy and Priority and prepare Contingency Plan 13

4.4. Implement the Upgrade Plan 15

4.5. Conduct Verification Test on Upgrade 20

5. Appendix: Reference and Tools 22

1. Executive Summary

Communication security protocol Transport Layer Security (TLS) ensures data transmission can stand attack of sniffing and data tampering. The protocol has evolved over time with better security and performance. In March of 2020, insecure versions of the protocol, TLS 1.0 and TLS 1.1, will approach end-of-support. For the sake of security, IT infrastructure supported by TLS needs to be upgraded to the secure versions of TLS 1.2 and TLS 1.3.

The end-of-support will affect many web applications. Website visitors may come across the message "Connection not secure. This page uses weak encryption" if the website has not been upgraded. Please upgrade now to use more secure TLS protocols and algorithms to avoid such embarrassment.

This Guideline aims to provide a handy guide for IT leaders to upgrade TLS protocols and cipher suite used to meet the current standard of security in a systematic manner. It features two simple profiles that can cater different scenarios. One is for maximum security and the other one for balance of security and compatibility. Each profile contains the TLS version(s) and the cipher suites to be used.

Profile	Criteria of Application	TLS Version(s) Used
Modern Security	For maximum security	TLS 1.3 only
Intermediate Security	Balance high compatibility and good security	Both TLS 1.2 and TLS 1.3

Below is the recommended approach on the TLS Upgrade Steps, along with the steps in details and tools to use (see Section 4).

The first step "Inventory" is an important start. Some people only consider the most noticeable asset, such as the public web server, and overlook many other devices. This Guideline has listed a number of services that rely on TLS support which has to be paid more attention to.

The Planning stage offers tips on the strategy and priority of upgrade, and reminds the reader about contingency plan for those assets that cannot be upgraded.

2. Overview

2.1. Objectives

This Guideline aims to provide a handy guide for IT leaders to upgrade TLS protocols and cipher suite used to meet the current standard of security in a systematic manner.

2.2. Target Audience

Technical by nature, this Guideline provides IT leaders with practical guidance in ensuring seamless migration/upgrade to secure TLS versions and ciphers, useful tools to assess the TLS usage on both public and internal services, and references to useful technical resources.

For general users, they can read sections 1 to 3 to raise their awareness of TLS upgrade necessity, and to understand what secure TLS and ciphers are. They can put the necessary requirements of TLS in the procurement specification.

2.3. About TLS Upgrade

What is TLS and applications supported?

Transport Layer Security (TLS) and its deprecated predecessor, Secure Socket Layer (SSL) are critical security protocols designed to provide communication security over insecure network.

TLS is used in many applications such as web browsing, email, instant messaging, virtual private networks, voice over IP and other applications. Most systems use a TLS software library such as OpenSSL, SChannel and NSS. TLS provides confidentiality and integrity for communication between a client and a server.

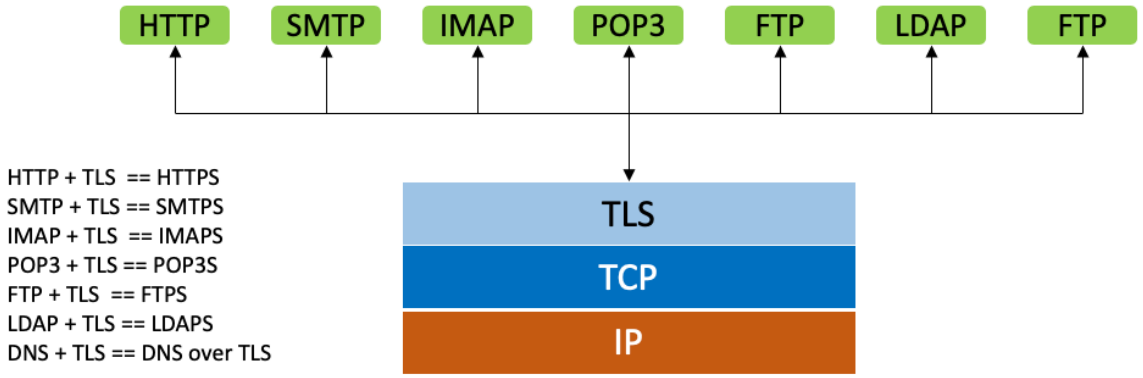


Figure 1: TLS empowers many application protocols by making them secure

Why do we need to upgrade TLS?

TLS has many versions over the years, with the earlier ones such as TLS 1.0 (published in 1999) and TLS 1.1 (published in 2006) now deemed not sufficiently secure according to the current cyber security standard. Attackers are able to exploit their weaknesses to decrypt the transmitted data. These two versions are to be deprecated by technology providers. Major browsers like Chrome and Firefox stop

supporting TLS 1.0 and 1.1 in March 2020. TLS 1.3 which comes with Perfect Forward Secrecy that protects the secrecy of the past sessions has been recommended. With this advanced TLS version, an attacker cannot break the encrypted traffic by recording it and must wait until he has the supercomputer power to do so.

For infrastructure that are still using old TLS versions, they should upgrade to secure versions of TLS 1.2 (published in 2008) and TLS 1.3 (published in 2018). If any systems are not upgraded, it might not be able to serve clients which have been upgraded to newer TLS versions.

What are the issues to consider in upgrading TLS?

IT leaders need to plan the upgrade of TLS systematically. For a start, they must have a visibility on the network assets that provide TLS support, applications and clients that are supported by TLS so that they will not miss any device or client.

Then, they have to take into account compatibility of applications and clients in the planning, as different versions of TLS client and server cannot communicate with each other. The matter is complicated by the fact that full control over the servers and clients, such as the browsers of website visitors, is impossible.

Furthermore, different versions of TLS have different mix of cipher suite and IT leaders must use cipher suite that are secure. Lastly, they have to assess the deployment and benchmark against best practices.

3. TLS and Cipher Suite Baseline

3.1. TLS Baseline

TLS has evolved and improved over time. While TLS 1.2 has provided more secure hash algorithms such as SHA-256 as well as advanced cipher suites Advanced Encryption Standard (AES) and Elliptical Curve Cryptography (ECC), TLS 1.3 has further improved security and performance by removing obsolete and insecure features from TLS 1.2 (MD5, SHA, RC4, DES, 3DEC and AES, etc.) and simplifying the cipher suites.

The TLS version plus the cipher suite combinations might make things very sophisticated. To make things simple, HKCERT provide two profiles: Modern Security and Intermediate Security. For each profile, we provide the following TLS baseline:

Profile	Description	TLS version(s) used
Modern Security	For maximum security	TLS 1.3 only
Intermediate Security	Balance high compatibility and good security	Both TLS 1.2 and TLS 1.3

Table 1: TLS Baseline

3.2. Cipher Suite Baseline

Each TLS version has a list of supported cipher suites. A cipher suite is a combination of algorithms that help secure a network connection. It has the following components:

- **Key Exchange algorithm** – used to exchange keys between client and server
- **Authentication algorithm (signature)** – used to help authenticate the client and server
- **Bulk Encryption algorithm** – used to encrypt data being sent.
- **Message Authentication Code (MAC) algorithm** – used to generate hashes and signatures to ensure data integrity

Each cipher suite has a unique name mentioning the algorithms used. For example, the following name indicates the cipher suite uses ECDHE as key exchange algorithm and ECDSA as authentication algorithm, AES256 as bulk encryption algorithm and SHA384 as MAC.

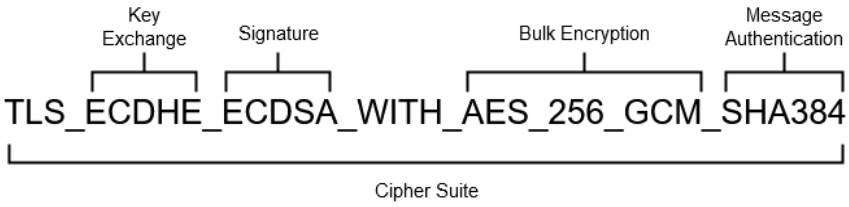


Figure 2: Structure of Cipher Suite Name

The structure of cipher suite name can vary. Sometimes the “TLS_” or even the key exchange and signature algorithms are omitted.

If an Internet user visits a web page protected by TLS, the browser can reveal the details about the cipher suite being used. The following is captured in Firefox.

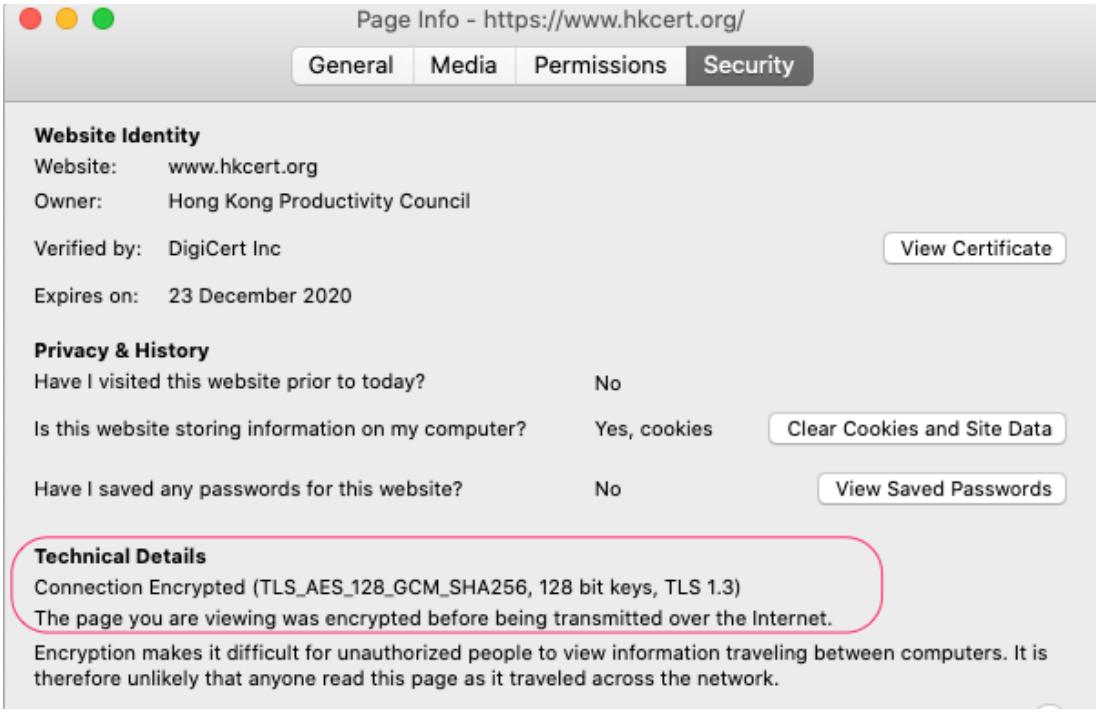


Figure 3: information of an HTTPS page showing the TLS protocol and cipher suite being used.

For the two profiles, the cipher suite baseline is shown below:

Profile	TLS protocol(s) used	Cipher suites used
Modern Security	TLS 1.3 only	• TLS_AES_128_GCM_SHA256 • TLS_AES_128_CCM_8_SHA256 • TLS_AES_128_CCM_SHA256 • TLS_AES_256_GCM_SHA384 • TLS_CHACHA20_POLY1305_SHA256
Intermediate Security	Both TLS 1.2 and TLS 1.3	• TLS_AES_128_GCM_SHA256 • TLS_AES_256_GCM_SHA384 • TLS_CHACHA20_POLY1305_SHA256 • ECDHE-ECDSA-AES128-GCM-SHA256 • ECDHE-RSA-AES128-GCM-SHA256 • ECDHE-ECDSA-AES256-GCM-SHA384 • ECDHE-RSA-AES256-GCM-SHA384 • ECDHE-ECDSA-CHACHA20-POLY1305 • DHE-RSA-AES128-GCM-SHA256 • DHE-RSA-AES256-GCM-SHA384

Table 2: TLS and Cipher Suite Baseline

More details on how the Mozilla SSL Config Generator (https://ssl-config.mozilla.org) can help generate the configuration for the two profiles for different common applications will be available in Section 4.4.

4. TLS Upgrade Steps

Our approach to TLS Upgrade is below:

4.1. Inventory the Network and Software Assets that require TLS support

TLS is widely used in many network assets (systems and services). If only the most obvious asset, like the public web server, is considered, many things will be missed. Hence, let start with an exhaustive inventory of network assets that require TLS support. The following lists some of the key categories of applications that require TLS support. It hopes to provide you with some insight on services to look at.

Service	**Systems that provide this service**	**Protocol(s)**
Web	Public web server – for public access Intranet web server – for internal staff access only	HTTPS
Proxy	Web proxy server - protect internal staff when they access external websites and apply corporate web access policy	HTTPS
WAF	Web application firewall - protect public web servers from attacks via incoming web traffic	HTTPS
VPN	SSL Virtual Private Network – provide secure remote access for out-of-office staff users	HTTPS
Server API	Application programming interface – provide data exchange of backend server with mobile app, IoT devices or other cloud services (machine-to-machine communication)	HTTPS
Email	Email server	SMTPS, IMAPS, POP3S, STARTTLS
FTP	Public file transfer	FTP over SSL (FTPS)
LDAP	Encrypts the authentication session	LDAPS
TLS Tunnel	Some services such as “DNS over TLS” can use TLS tunnel to encrypt connection, to protect against sniffing and man-in-the-middle attack	DNS over TLS, etc.
Internal applications and TLS library	Internal applications might be developed with TLS capability or using TLS software libraries (e.g. OpenSSL, GnuTLS, NSS)	Protocols with TLS support
Browser	User client software for web browsing	HTTPS (client)

Table 3: Key Categories of Applications that require TLS support

Firstly, it is recommended to make an inventory of network assets. Below is a sample table with useful information on the currently supported TLS versions and if they are put in the public network.

		Software used		Existing TLS support (Note 1)				
Systems	**Server name**	**Software Name**	**Version**	**TLS 1.0**	**TLS 1.1**	**TLS 1.2**	**TLS 1.3**	**Public service**
Web	WWW1	Apache	XX	Y	Y	Y		Y
Web	WWW2	Apache	XX	Y	Y	Y		Y
Web Proxy	PROXY	Apache	XX	Y	Y	Y		
Intranet	PORTAL	Apache	XX	Y	Y	Y		
WAF	WAF	Citrix Netscaler	XX	Y	Y	Y		Y
VPN	SSLVPN	FortiVPN	XX	Y	Y			Y
Server API	RESTAPI	Nginx	XX	Y	Y			Y
Email-SMTPS	EMAIL1	Exchange 2013	XX	Y	Y	Y		Y
Email-IMAPS	EMAIL2	Exchange 2013	XX	Y	Y	Y		Y
Email-WWW	WEBMAIL	IIS	XX	Y	Y			Y
FTP-FTPS	FTP1	FileZilla	XX	Y	Y	Y	Y	Y

Table 4A: Sample Inventory Table for systems that require TLS support

Then, it is recommended to make an inventory of software assets that require or provide TLS support and put in a table in the following format:

			Existing TLS support (Note 1)			
Softwre	**Software name**	**Version**	**TLS 1.0**	**TLS 1.1**	**TLS 1.2**	**TLS 1.3**
Browser	Chrome	XX	Y	Y	Y	
	Firefox	XX	Y	Y		
Software library	OpenSSL	XX	Y	Y	Y	
	NSS	XX	Y	Y		

Table 4B: Sample Inventory Table for Software that require or provide TLS support

[Note 1] The “Existing TLS support” information can be found out by several ways.

- Check existing technical documentation
- Use the server software brand and version information to research the information on TLS support. Section 4.2 will have more information on TLS support status of common applications.
- If the system is a web server exposed to the Internet, you can use tools such as Qualys SSL Server Test (https://www.ssllabs.com/ssltest/) to find out the TLS version and cipher suite currently used on the server.

With Tables 4A and 4B, we will not miss any asset and we know which assets needs to be upgraded.

4.2. TLS Upgrade Pre-requisites Gap Analysis

We now have the part of the information we need for the TLS upgrade project in Table 4. Next we must study the gap we need to fill in order to upgrade the assets on the inventory to TLS 1.2 and/or TLS 1.3. You can search the vendor information on the pre-requisite for TLS upgrade for each asset. Below is a table summarizing the TLS upgrade pre-requisites of some common applications. Please note that the list is not exhaustive and you should also do your own research to obtain the most updated information.

		Perquisite for TLS upgrade		
Systems	**Software brand**	**Hardware /Firmware /OS**	**Software version or configuration**	**Remark**
Web Server	Microsoft Internet Information Services (IIS)	TLS 1.2 support by default in Windows Server 2012, 2016 and 2019; TLS 1.2 support by default in Windows 8, Windows 8.1 and Windows 10; Service Pack 1 (SP1) update is required for Windows 7 and Windows Server 2008 R2 to support TLS 1.2, and it is disabled by default	TLS 1.2 support by default starting from IIS version 8.0 or later	TLS 1.3 not yet supported in IIS
	Apache/HTTPD	-	TLS 1.2 support in version 2.2.x or later; TLS 1.3 support in version 2.4.38 or later	-
	Nginx	-	TLS 1.3 support in version 1.13 of later	-
Web Browser	Google Chrome	-	TLS 1.2 support by default in version 38 or later TLS 1.3 support in version 70 or later	-
	Internet Explorer	-	TLS 1.2 support by default in version 11 or later TLS 1.3 not supported in Internet Explorer	-
	Mozilla Firefox	-	TLS 1.2 support by default in version 27 or later TLS 1.3 support in version 63 or later	-
	Opera	-	TLS 1.2 support by default in version 17 or later TLS 1.3 support in version 57 or later	-
	Safari Desktop	TLS 1.3 support in MacOS version 10.14 or later	TLS 1.2 support by default in version 7 or later	-
	Google Android OS Browser	TLS 1.2 support by default from Android 5.0 (Lollipop) and later TLS 1.3 not supported within Google Android OS Browser	-	-
	Mobile Safari	TLS 1.2 support by default from iOS 5 and later	-	-
Email Server	Microsoft Exchange	TLS 1.2 support by default in Windows Server 2012, 2016 and 2019; TLS 1.2 support by default in Windows 8, Windows 8.1 and Windows 10; Service Pack 1 (SP1) update is required for Windows 7 and Windows Server 2008 R2 to support TLS 1.2, and it is disabled by default	Exchange Server 2010 supports TLS 1.2 after SP3 RU19 update has been installed; Update Exchange Server 2010 to SP3 RU20 is required if need to disable TLS 1.0 and 1.1	TLS 1.3 not yet supported in Exchange Server
	EXIM	-	TLS 1.2 support in OpenSSL 1.0.1; TLS 1.3 support in OpenSSL 1.1.1	-

		Perquisite for TLS upgrade		
Systems	**Software brand**	**Hardware /Firmware /OS**	**Software version or configuration**	**Remark**
	Qmail	TLS 1.3 support for Client Certificate Authentication from version 20200107	TLS 1.2 support in OpenSSL 1.0.1; TLS 1.3 support in OpenSSL 1.1.1; TLS 1.3 enabled by default starting from GnuTLS 3.6.5	-
	Ironport	TLS 1.2 support in AsyncOS version 9.5 or later	-	TLS 1.3 not yet supported in AsyncOS
WAF/ Network devices	Citrix Netscaler	TLS 1.3 support in Citrix ADC version 12.1 (build 49.23) or later	-	-
	F5 Advanced WAF	TLS 1.3 support in BIG-IP version 14.1.0.1 or later	-	-
	Fortigate	TLS 1.3 support in IPS engine 4.205 or later and endpoints running FortiClient 6.2.0 or later	-	-
Software Library	OpenSSL	-	TLS 1.2 support in OpenSSL 1.0.1; TLS 1.3 support in OpenSSL 1.1.1	-

Table 5: Sample vendor information on Pre-requisite of TLS 1.2 and TLS 1.3 support

Most web browsers already have TLS 1.3 support. For more detailed and updated information, please refer to this below website for details:

- Browsers and versions supporting TLS 1.3 - https://caniuse.com/tls1-3
- Browsers and versions supporting TLS 1.2 - https://caniuse.com/tls1-2

With the vendor information, we can expand our inventory table to include pre-requisite check for each asset.

		Software used		Existing TLS support (Note 1)					Pre-requisite check
Systems	**Server name**	**Software Name**	**Version**	**TLS 1.0**	**TLS 1.1**	**TLS 1.2**	**TLS 1.3**	**Public service**	**Prequisute (hardware /firmware /software)**
Web	WWW1	Apache	XX	Y	Y	Y		Y	Version XX
Web	WWW2	Apache	XX	Y	Y	Y		Y	Version XX
Web Proxy	PROXY	Apache	XX	Y	Y	Y			Version XX
Intranet	PORTAL	Apache	XX	Y	Y	Y			Version XX
WAF	WAF	Citrix Netscaler	XX	Y	Y	Y		Y	Firmware XX
VPN	SSLVPN	FortiVPN	XX	Y	Y			Y	Firmware XX
Server API	RESTAPI	Nginx	XX	Y	Y			Y	Appl. compatibility issue
Email-SMTPS	EMAIL1	Exchange 2013	XX	Y	Y	Y		Y	CU XX
Email-IMAPS	EMAIL2	Exchange 2013	XX	Y	Y	Y		Y	CU XX
Email-WWW	WEBMAIL	IIS	XX	Y	Y			Y	Version XX
FTP-FTPS	FTP1	FileZilla	XX	Y	Y	Y	Y	Y	-- TLS 1.3 ready --

Table 6: The Inventory Table is extended to include Pre-requisite check

Table 6 provides useful information on which asset needs to be upgraded and what action is required to do so. You may also notice that not all assets need upgrade. Some are already TLS 1.3 ready, and some cannot be upgraded due to application compatibility issue.

4.3. Prepare Upgrade Plan with Strategy and Priority and prepare Contingency Plan

You have to decide the TLS upgrade and priority for different assets.

TLS Upgrade Strategy

You can decide the TLS upgrade strategy according to the control we have on the TLS client and server. Then we apply the TLS Baseline and Cipher Suite Baseline according to the strategy. The following table gives a summary on the strategy.

Criteria	Example	Strategy to adopt
You can control both server and client	- Intranet Server - Server API for corporate mobile app	**Modern Security Profile** - TLS 1.3 for best security and performance
You can control the server only	- WAF and Public Web Server - Public Mail Gateway - SSL VPN - Server API for public	**Intermediate Security Profile** - TLS 1.3 and TLS 1.2 for security and compatibility
You can control the client only	- Browser of staff who needs to access both corporate and public web	**Intermediate Security Profile** - TLS 1.3 and TLS 1.2 for security and compatibility
You have no control at all	- Legacy systems	Keep TLS at **the highest version that can provide compatibility.** Plan for future migration to attain baseline.

Table 7: Determine the Strategy of Upgrade

Note: the TLS version(s) and Cipher Suite Baseline for Modern Security and Intermediate Security Profiles are documented in Table 2.

Priority of Upgrade

In planning the TLS upgrade, we should set priorities in view of limited time. The more important upgrades should be done first. The criteria for consideration are (1) the criticality of the system to the corporation and (2) the exposure of the system. For exposure consideration, the Internet facing services are of higher risk and should be done first. We propose to start from the periphery because when the outermost system is done, the security benefit can immediately be realised.

Criteria (exposure)	Example	Strategy to adopt
Internet gateways	- Web applicattion firewall - Email gateway - SSL VPN	1st priority
Public service servers	- Public web server - Email server	2nd priority
Internal servers	-Intranet server	3rd priority

Table 8: Determine the Priority of Upgrade

Using our example for illustration, we will put the key public service systems at the top of priority (priority = 1). Internal services VPN (which has public network access) follows and Intranet (private network) with the lowest priority. For web and email service, we have two groups of systems A and B. In web service group A, as the WAF is facing the Internet, it will be done first; then move to the two web servers behind it (put at priority 3, after other public gateways).

For most systems whom we cannot control both the server and clients, we will use "Intermediate Security" profile in upgrade strategy. For Intranet server, as we can control the both server and the clients (corporate desktops), we will adopt "Modern Security" profile.

For some systems that cannot be upgraded to either profile due to compatibility issues, we denote them as "Legacy".

		Software used		Existing TLS support (Note 1)					Pre-requisite check			
Systems	**Server name**	**Software Name**	**Version**	**TLS 1.0**	**TLS 1.1**	**TLS 1.2**	**TLS 1.3**	**Public service**	**Prequisute (hardware /firmware /software)**	**Upgrade Strategy**	**Priority**	Group
WAF	WAF	Citrix Netscaler	XX	Y	Y	Y		Y	Firmware XX	Intermediate	1	A
Web	WWW1	Apache	XX	Y	Y	Y		Y	Version XX	Intermediate	3	A
Web	WWW2	Apache	XX	Y	Y	Y		Y	Version XX	Intermediate	3	A
Email-SMTPS	EMAIL1	Exchange 2013	XX	Y	Y	Y		Y	CU XX	Intermediate	1	B
Email-IMAPS	EMAIL2	Exchange 2013	XX	Y	Y	Y		Y	CU XX	Intermediate	1	B
Email-WWW	WEBMAIL	IIS	XX	Y	Y			Y	Version XX	Intermediate	1	B
VPN	SSLVPN	FortiVPN	XX	Y	Y			Y	Firmware XX	Intermediate	3	--
Web Proxy	PROXY	Apache	XX	Y	Y	Y			Version XX	Intermediate	4	--
Intranet	PORTAL	Apache	XX	Y	Y	Y			Version XX	Modern	5	--
Server API	RESTAPI	Nginx	XX	Y	Y			Y	Appl. compatibility issue	Legacy	X	--
FTP-FTPS	FTP1	FileZilla	XX	Y	Y			Y	-- TLS 1.3 ready --	Nil	X	--

Table 9A: The Inventory Table (system) is extended to include Upgrade Strategy and Priority

For software assets that require or provide TLS support, we have a table below:

			Existing TLS support (Note 1)						
Softwre	**Software name**	**Version**	**TLS 1.0**	**TLS 1.1**	**TLS 1.2**	**TLS 1.3**	**Prequisute**	**Upgrade Strategy**	**Priority**
Browser	Chrome	XX	Y	Y	Y		Version XX	Intermediate	5
	Firefox	XX	Y	Y			Version XX	Intermediate	5
Software library	OpenSSL	XX	Y	Y	Y		Version XX	Intermediate	2
	NSS	XX	Y	Y			Version XX	Intermediate	2

Table 9B: The Inventory Table (software) is extended to include Upgrade Strategy and Priority

Contingency Plan for Legacy Systems

If you have legacy systems that do not support TLS 1.2 and TLS 1.3 due to restriction, you have to maintain the current TLS setup. Below are some options to minimise the security risk of these systems:

- Put the system behind application firewall which provides TLS 1.2 or TLS 1.3 support
- Minimise the exposure of these system, e.g. limit the network access from the Internet
- Plan for replacement strategy to circumvent the restrictions if the system is critical; or you have to accept the residue risk if the system is not critical.

4.4. Implement the Upgrade Plan

We have sufficient information in Table 9A and Table 9B to determine the priority and upgrade strategy for each asset in the Inventory Table. Now we proceed to the implementation of TLS Upgrade.

Implementation workflow

Base on the information from the Sections 4.1 – 4.3, repeat the following workflow for each system according to its priority (Tables 9A and 9B).

a. Fulfil the prerequisite: if the system requires certain firmware version, OS version or software as pre-requisite, we have to perform these firmware and system upgrade before proceed to TLS upgrade.
b. Upgrade the support of TLS version and cipher suites according to the upgrade strategy ("Modern security" or "Intermediate security") defined.

a. Fulfil the pre-requisite

Below are the tips and good practices for firmware/ OS /software upgrade.

Preparation and Planning

- ❑ Check the maintenance contract to ensure the entitlement of upgrade.
- ❑ Check for compatibility issue when Firmware /OS / Software is upgraded
- ❑ Read official document for the upgrade, such as release notes and upgrade guide to understand to technical steps
- ❑ Plan the upgrade carefully
- ❑ Prepare the rollback plan – in case the upgrade fails due to any cause, you should be able to fall back
- ❑ Prepare contact list of vendor support, stakeholders and related parties

Checking the Technical Capability

- ❑ Make sure the upgrade project team possess related technical skills to perform the upgrade and maintain the new firmware / OS /software

Implementation and Testing

- ❑ Conduct the upgrade in the testing environment first, before applying to the production system, whenever possible

b. Upgrade the support of TLS version and cipher suites according to the upgrade strategy

Steps for applying secure TLS version and cipher suites and Tips

i. Locate the right configuration files
Different configuration files may supersede each other. To ensure your configuration can be effective, make sure that you find the correct file, especially if you use an application which may affect TLS configuration.

ii. Make a backup for your existing configuration
The backup configuration is always one of your best friends who can save your life when you need to restore from a failure.

iii. Generate TLS configuration by using the Mozilla SSL Configuration Generator
Mozilla SSL Configuration Generator (https://ssl-config.mozilla.org/) makes your life easier. It can help you generate the TLS configuration needed for different applications, e.g. web servers, mail servers, proxy server and database server.

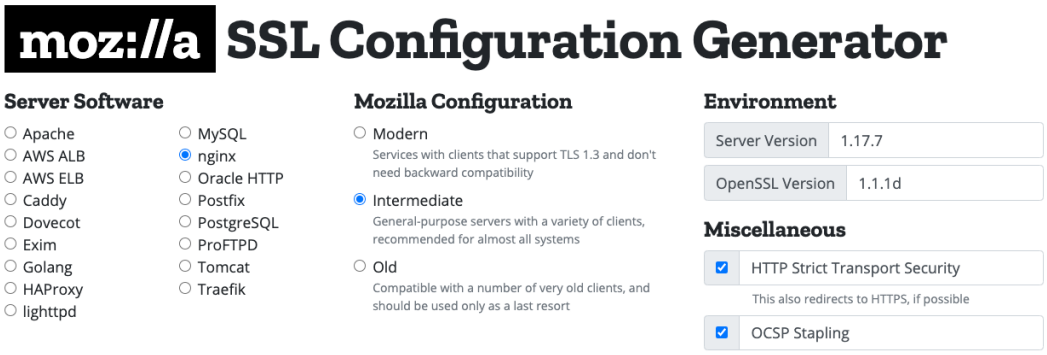


Figure 4: Mozilla SSL Configuration Generator Dashboard

(a) You can just select your platform from the **Server Software**.
(b) From the **Mozilla Configuration,** select your profile ("Modern" or "Intermediate") for the asset based on your upgrade strategy as specified in Table 9A.
(c) The TLS configuration is generated.

iv. Review the existing configuration and apply the new generated TLS configuration

Reasons for reviewing existing configuration
(a) Understand better which TLS versions and cipher suites are currently in use, enabled functionalities and features.
(b) Ensure existing functionalities and features would not be compromised by the new TLS configuration.

Tips for applying new TLS configuration
(a) You are suggested to stop the service before applying new configuration.
(b) Use your favourite tools to edit the configuration e.g. vi, nano etc.
(c) Comment the old configuration (add # in the front of the row) rather than remove it directly, as it can help restore to your original configuration in case encountered any failure.
(d) DO NOT replace the existing configuration file with the new generated TLS configuration by Mozilla's tool directly. It is not for direct replacement and you should have some tailor-made configurations on the existing configuration file.

v. Fine tune the TLS configuration to better fit individual system/business need if necessary.
(a) The suggested TLS protocols and cipher suites are adequate in general, but you can still fine tune them to best fit your requirements, such as policy and compliance, or requirement of specific application.

(b) Remember to remove support legacy protocols (SSLv3 or below). For example, if your application cannot upgrade to latest version and may support legacy protocols by default, you should remove support of these legacy protocols before using that configuration file.
(c) Fine tune the file ownership and permission of the configuration file if necessary.

vi. Check if your service is working well.
Check relevance logs on the system for errors. If you encountered an error, please perform some connectivity tests such as seeking to access the service:

(a) locally;
(b) from another machine on the same internal subnet;
(c) from another machine on the different internal subnet; and
(d) from the Internet.

We will illustrate the above steps with a sample scenario below.

Sample Scenario
Your company website is already providing service via HTTPS (TLS). During annual security assessment, one of the findings is that insecure TLS protocols are in use (TLS1.0 and TLS 1.1). Therefore, you plan to upgrade to more secure TLS protocol versions and cipher suites.

Background
In this example, we assume that the website is hosted by **single web server**. It is running **Apache 2.4.41 on Ubuntu 18.04.4.** If you are running other Linux distribution / OS / application, your commands will be a little bit different.

Steps

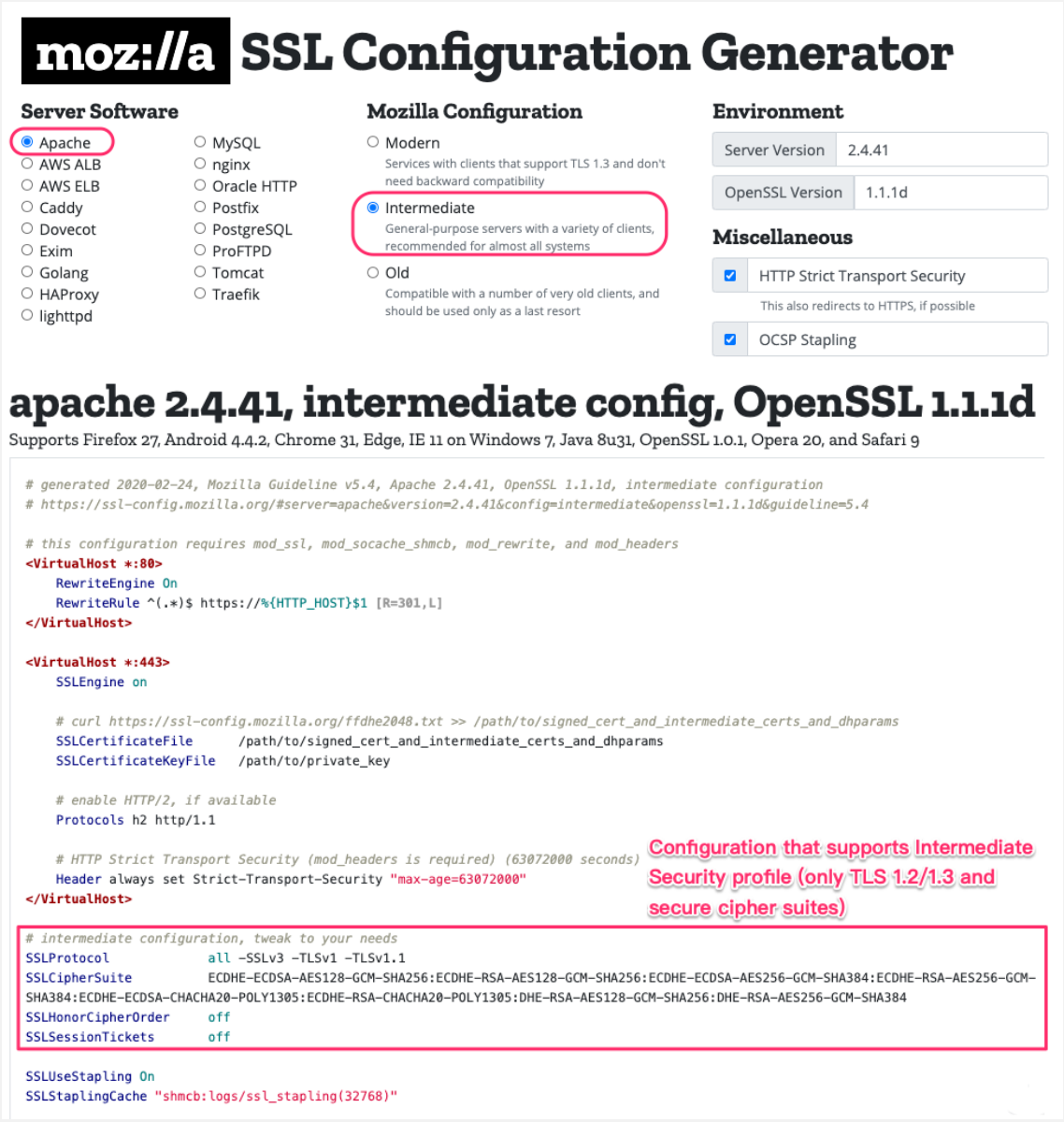
i. Locate the configuration files.
(a) In Apache, there are 2 keywords control the application of TLS protocols and cipher suites: "SSLProtocol" and "SSLCipherSuite".
(b) Search the path of your configuration file:
`Syntax: $ grep -i -r "searching keyword" searching path`
`$ grep -i -r "SSLProtocol" /etc/apache2`
(c) Record the path(s) you found. It is possible that you can find more than one path, for example:
`/etc/apache2/mods-available/ssl.conf`
Note: If you are running other application which may supersede traditional TLS configuration, please check its configuration, for example: for Let's Encrypt user, please check /etc/letsencrypt/options-ssl-apache.conf.

ii. Make a backup for your existing configuration
(a) Change directory to the path you found the configuration file
`$ cd /etc/apache2/mods-available`
(b) Make a backup of existing configuration file, the backup configuration file can save your life if you found error after the change.
`$ sudo cp ssl.conf ssl.conf.backup`
You may use `"ls -la"` to verify if the backup file has been copied under same directory.
(c) You may download the configuration file to desktop for easier editing, e.g. through Winscp/PSCP.

iii. Generate TLS configuration by using the Mozilla SSL Configuration Generator

(a) Generate an “Intermediate” configuration for Apache, as shown below.

(b) Click **Copy** button at the lower right-hand corner, and paste it to your favourite editor, save it as “TLS.conf”.



iv. Review the existing configuration and apply the new generated TLS configuration (“TLS.conf”).

(a) Assume that you have done review the existing configuration.

(b) You may need to enable some Apache modules if they are not yet enabled currently, as reminded from the generated configuration. Here is the example command to enable the "mod_headers" module.

```
$ sudo a2enmod headers
```

(c) Stop the service

```
$ sudo /etc/init.d/apache2 stop
```

(d) Find the keyword “SSLProtocol” from the existing configuration file located before (in part i), comment the original row and paste the same part from “TLS.conf”.

Example:
Disable current configuration (turn into comment lines) by adding “#” at the beginning of existing SSLProtocol config line:

```
# SSLProtocol all -SSLv3
```

Add the SSLProtocol config line according to the Mozilla SSL Configuration Generator:

```
SSLProtocol all -SSLv3 -TLSv1 -TLSv1.1
```

(e) Find the keyword “SSLCipherSuite” from the existing configuration file located before (in part i), comment the original row and paste the same part from “TLS.conf”.

Note: in Apache, when you use “modern security”, you may comment the original row “SSLCipherSuite” and save the configuration directly, those cipher suites for TLS1.3 will be supported by default.

Configuration Before Upgrade:

```
SSLProtocol  all -SSLv3
SSLCipherSuite         ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-
ECDSA-CHACHA20-POLY1305:
ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-
SHA384:DHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-
AES128-SHA256:
ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-
SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:
DHE-RSA-AES256-SHA256:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-
SHA256:AES256-SHA256:AES128-SHA:AES256-SHA:DES-CBC3-SHA
```

Configuration After Upgrade:

```
SSLProtocol  all -SSLv3 -TLSv1 -TLSv1.1
SSLCipherSuite  ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-
ECDSA-CHACHA20-POLY1305:
ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-
SHA384
# SSLProtocol  all -SSLv3
# SSLCipherSuite  ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-
ECDSA-CHACHA20-POLY1305:
ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-
SHA384:DHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-
AES128-SHA256:
ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-
SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:
DHE-RSA-AES256-SHA256:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-
SHA256:AES256-SHA256:AES128-SHA:AES256-SHA:DES-CBC3-SHA
```

v. Fine tune the TLS configuration to better fit individual system/business need if necessary.
   (a) You may need to upload the fine-tuned configuration file to the server if you have edited it on desktop. Once uploaded, fine tune file ownership and permission if necessary.
   (b) You may test if the syntax of configuration is correct by command below, it should be fine if you see "Syntax OK".
   `$ sudo apache2ctl configtest`
   (c) Start the apache service to make the new configuration effective.
   `$ sudo /etc/init.d/apache2 start`

vi. Check if your service is working well.
   Assume that no error was found from the logs and the service is running properly.

4.5. Conduct Verification Test on Upgrade

(1) Conduct Local Test

This test aims to check if the upgrade performed in previous steps are working properly, before releasing the service to the public.

Below table summarises the test cases, openssl commands and the expected result for "Modern Security" and "Intermediate Security".

Note: The test cases use the command line tool "openssl" latest stable version 1.1.1 as an example. Please change the IP address `127.0.0.1` and port number `443` in the command based on the target server IP address and the service port number.

Test case		Openssl Command	Expected Result of Configuration Profiles	
			Modern Security	Intermediate Security
1	1. Test TLS connection using TLS 1.3	`openssl s_client -connect 127.0.0.1:443 -tls1_3 2>/dev/null | grep 'New, TLS'`	TLS 1.3 connection **success.** The returned message should look like below: `New, TLSv1.3, Cipher is TLS_AES_256_GCM_SHA384`	TLS 1.3 connection **success**. The returned message should look like below: `New, TLSv1.3, Cipher is TLS_AES_256_GCM_SHA384`
2	Test TLS connection using TLS 1.2	`openssl s_client -connect 127.0.0.1:443 -tls1_2 2>/dev/null | grep 'New, TLS'`	TLS 1.2 connection **failed**. No TLS message returned indicates connection failed.	TLS 1.2 connection **success**. The returned message should look like below: `New, TLSv1.2, Cipher is ECDHE-RSA-AES256-GCM-SHA384`

3	Test TLS connection using TLS 1.1	`openssl s_client -connect 127.0.0.1:443 -tls1_1 2>/dev/null | grep 'New, TLS'`	TLS 1.1 connection **failed**. No TLS message returned indicates connection failed.	TLS 1.1 connection **failed**. No TLS message returned indicates connection failed.
4	1. Test TLS connection using TLS 1.0	`openssl s_client -connect 127.0.0.1:443 -tls1 2>/dev/null | grep 'New, TLS'`	TLS 1.0 connection **failed**. No TLS message returned indicates connection failed.	TLS 1.0 connection **failed**. No TLS message returned indicates connection failed.

Below is the sample test result for Modern Security. Note the result returned is in **bold**.

```
$ openssl s_client -connect 127.0.0.1:443 -tls1_3 2>/dev/null | grep 'New, TLS'
New, TLSv1.3, Cipher is TLS_AES_256_GCM_SHA384
$ openssl s_client -connect 127.0.0.1:443 -tls1_2 2>/dev/null | grep 'New, TLS'
$ openssl s_client -connect 127.0.0.1:443 -tls1_1 2>/dev/null | grep 'New, TLS'
$ openssl s_client -connect 127.0.0.1:443 -tls1 2>/dev/null | grep 'New, TLS'
```

Below is the sample test result for Intermediate Security:

```
$ openssl s_client -connect 127.0.0.1:443 -tls1_3 2>/dev/null | grep 'New, TLS'
New, TLSv1.3, Cipher is TLS_AES_256_GCM_SHA384
$ openssl s_client -connect 127.0.0.1:443 -tls1_2 2>/dev/null | grep 'New, TLS'
New, TLSv1.2, Cipher is ECDHE-RSA-AES256-GCM-SHA384
$ openssl s_client -connect 127.0.0.1:443 -tls1_1 2>/dev/null | grep 'New, TLS'
$ openssl s_client -connect 127.0.0.1:443 -tls1 2>/dev/null | grep 'New, TLS'
```

There are more tools and information for testing internally (see Appendix 5.5 TLS Checking Tools)

- testssl.sh
- sslscan
- SSLyze

(2) Conduct External Test

If your system is accessible from the Internet, you can use some good free Internet testing tools to verify your TLS configuration. You should do this as it provides the exact user experience of your visitors.

- Website Test Tool: Qualys SSL Server Test
  https://www.ssllabs.com/ssltest/
- Email Test Tool: checktls.com
  https://www.checktls.com/

5. Appendix: Reference and Tools

Disclaimer:
HKCERT does not endorse specific vendor products/tools. Inclusion of products/tools in this reference list does not indicate endorsement by HKCERT. Tools are listed with no quality rating. The tools in this list are owned by tool developers or vendors and they can be modified any time. HKCERT does not verify the accuracy of these tools. If you have any question about these tools, please direct contact tool developers or vendors.

5.1 TLS Protocols

- IETF : The Transport Layer Security (TLS) Protocol Version 1.3
  https://tools.ietf.org/html/rfc8446
- OpenSSL : TLS1.3
  https://wiki.openssl.org/index.php/TLS1.3
- Responsibly Intercepting TLS and the Impact of TLS 1.3
  https://www.symantec.com/content/dam/symantec/docs/other-resources/responsibly-intercepting-tls-and-the-impact-of-tls-1.3-en.pdf
- The SSL Store: TLS 1.3 Update: Everything you possibly needed to know. TLS1.3: A Complete Overview
  https://www.thesslstore.com/blog/tls-1-3-everything-possibly-needed-know
- These truly are the end times for TLS 1.0, 1.1: Firefox hopes to 'eradicate' weak HTTPS standard by blocking it
  https://www.theregister.co.uk/2020/02/10/tls_10_11_firefox_complete_eradication

5.2 Guidelines

- ACSC : Implementing Certificates, TLS and HTTPS
  https://www.cyber.gov.au/publications/implementing-certificates-tls-and-https
- NCSC UK: Using TLS to protect data
  https://www.ncsc.gov.uk/guidance/tls-external-facing-services
- NIST: Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations: NIST SP 800-52 Rev. 2
  https://csrc.nist.gov/News/2019/nist-publishes-sp-800-52-revision-2
- NCSC NL: Future-proof TLS configuration using the updated TLS guidelines from NCSC
  https://english.ncsc.nl/latest/news/2019/juli/01/future-proof-tls-configuration
- Mozilla : Security/Server Side TLS
  https://wiki.mozilla.org/Security/Server_Side_TLS
- SSL and TLS Deployment Best Practices
  https://github.com/ssllabs/research/wiki/SSL-and-TLS-Deployment-Best-Practices
- Apache: SSL/TLS Strong Encryption: How-To
  https://httpd.apache.org/docs/trunk/ssl/ssl_howto.html

5.3 TLS Migration Tools

- Mozilla SSL Configuration Generator
  https://ssl-config.mozilla.org
- Can I use (Check browser compatibility)
  https://caniuse.com/tls1-3
  https://caniuse.com/tls1-2

5.4 Information for checking and verification

- Verifying SSL/TLS configuration (part 1)
  https://isc.sans.edu/diary/Verifying+SSLTLS+configuration+%28part+1%29/25162
- Verifying SSL/TLS configuration (part 2)
  https://isc.sans.edu/forums/diary/Verifying+SSLTLS+configuration+part+2/25214
- Testing TLSv1.3 and supported ciphers
  https://isc.sans.edu/forums/diary/Testing+TLSv13+and+supported+ciphers/25442
- OpenSSL test TLSv1.3 connection and ciphersuites with s_client
  https://raymii.org/s/tutorials/OpenSSL_test_TLSv1.3_connection_with_s_client.html
- Testing server for TLS 1.2 in Linux
  https://devanswers.co/test-server-tls-1-2-ubuntu

5.5 TLS Checking Tools

- testssl.sh
  https://testssl.sh
- sslscan
  https://github.com/rbsec/sslscan
- SSLyze
  https://github.com/nabla-c0d3/sslyze
- Website: Qualys SSL Server Test
  https://www.ssllabs.com/ssltest
- Email: checktls.com
  https://www.checktls.com