CIS Benchmarks

CIS NGINX Benchmark

v1.0.0 - 12-10-2018

Terms of Use

Please see the below link for our current terms of use:

https://www.cisecurity.org/cis-securesuite/cis-securesuite-membership-terms-of-use/

Table of Contents

Terms of Use 1
Overview 6
Intended Audience 6
Consensus Guidance 6
Typographical Conventions 7
Scoring Information 7
Profile Definitions 8
Acknowledgements 10
Recommendations 11
1 Initial Setup 11
1.1 Installation 12
1.1.1 Installing NGINX (Scored) 12
1.2 Configure Software Updates 14
1.2.1 Ensure Package Manager Repositories are Configured (Scored) 14
1.2.2 Ensure the Latest Software Package is Installed (Scored) 16
2 Basic Configuration 18
2.1 Minimize NGINX Modules 18
2.1.1 Ensure only Required Modules are Installed (Scored) 18
2.1.2 Ensure HTTP DAV Module is not Installed (Scored) 20
2.1.3 Ensure Modules With GZIP Functionality are Disabled (Scored) 22
2.1.4 Ensure the Autoindex Module is Disabled (Scored) 24
2.2 Account Security 26
2.2.1 Ensure that NGINX is Run Using a Non-Privileged, Dedicated Service Account (Scored) 26
2.2.2 Ensure the NGINX Service Account is Locked (Scored) 29
2.2.3 Ensure the NGINX Service Account has an Invalid Shell (Scored) 31
2.3 Permissions and Ownership 33
2.3.1 Ensure NGINX Directories and Files are Owned by Root (Scored) 33
2.3.2 Ensure Access on NGINX Directories and Files is Restricted (Scored) 35

2.3.3 Ensure NGINX Process ID (PID) File is Secured (Scored) 37
2.3.4 Ensure Core Dump Directory Is Secured (Not Scored) 39
2.4 Network Configuration 41
2.4.1 Ensure NGINX only Listens for Network Connections on Authorized Ports (Not Scored) 41
2.4.2 Ensure that Requests for Unknown Host Names are Rejected (Not Scored) 43
2.4.3 Ensure keepalive_timeout is Configured (Scored) 45
2.4.4 Ensure send_timeout is configured (Scored) 47
2.5 Information Disclosure 49
2.5.1 Ensure server_tokens directive is set to Off (Scored) 49
2.5.2 Ensure Default Error Pages and index.html Pages don't reference NGINX (Scored) 51
2.5.3 Ensure Hidden File Serving Is Disabled (Scored) 53
2.5.4 Ensure Proxy does not Enable Information Disclosure (Scored) 55
3 Logging 57
3.1 Ensure Detailed Access and Request Logging Is Enabled (Not Scored) 57
3.2 Ensure that Access Logging is Enabled (Scored) 60
3.3 Ensure that Error Logging is Enabled and Set to the Info Logging Level (Scored) 62
3.4 Ensure Log Files are Rotated (Scored) 64
3.5 Ensure NGINX Error Logs are Sent to a Remote Syslog Server (Not Scored) .. 67
3.6 Ensure NGINX Access Logs are Sent to a Remote Syslog Server (Not Scored) 69
3.7 Ensure Proxies Pass Source IP Information (Scored) 71
4 Encryption 73
4.1 TLS / SSL Configuration 73
4.1.1 Ensure HTTP is Redirected to HTTPS (Scored) 73
4.1.2 Ensure a Trusted Certificate and Trust Chain is Installed (Scored) 75
4.1.3 Ensure Private Key Permissions are Restricted (Scored) 78
4.1.4 Ensure Only Modern TLS Protocols are Used (Scored) 80
4.1.5 Restrict Weak Ciphers From Being Used (Scored) 83
4.1.6 Ensure Custom Diffie-Hellman Parameters are Used (Scored) 88

4.1.7 Ensure Online Certificate Status Protocol (OCSP) Stapling is Enabled (Scored) 90
4.1.8 Ensure HTTP Strict Transport Security (HSTS) is Enabled (Scored) 92
4.1.9 Ensure HTTP Public Key Pinning Extension is Enabled (Not Scored) 94
4.1.10 Ensure Upstream Server Traffic is Authenticated with a Client Certificate (Scored) 96
4.1.11 Ensure Upstream Traffic Server Certificate is Trusted (Not Scored) 98
4.1.12 Ensure your Domain is Preloaded (Not Scored) 100
4.1.13 Ensure Session Resumption Is Disabled To Enable Perfect Forward Security (Not Scored) 102
4.1.14 Ensure HTTP/2.0 is Used (Not Scored) 104
5 Request Filtering and Restrictions 106
5.1 Access Control 106
5.1.1 Ensure Allow and Deny Filters are Used to Limit Access to Specific IP Addresses (Not Scored) 106
5.1.2 Ensure only Whitelisted HTTP Methods are Allowed (Scored) 109
5.2 Request Limits 111
5.2.1 Ensure Timeout Values for Reading Client Request Header and Client Request Body are Set (Scored) 111
5.2.2 Ensure a Limit for Reading Client Request Body is Set (Scored) 113
5.2.3 Ensure Limits for Reading Large Client Request Headers are Set (Scored) 115
5.2.4 Ensure Number of Connections are Limited by IP Address (Not Scored) ... 117
5.2.5 Ensure Rate Limits by IP Addresses are Set (Not Scored) 119
5.3 Browser Security 121
5.3.1 Ensure X-Frame-Options Header is Configured and Enabled (Scored) 121
5.3.2 Ensure X-Content-Type-Options Header is Configured and Enabled (Scored) 123
5.3.3 Ensure that X-XSS-Protection Header is Configured and Enabled (Scored) 125
5.3.4 Ensure that Content Security Policy (CSP) is Configured and Enabled (Not Scored) 127
5.3.5 Ensure Referrer Policy is Configured and Enabled (Not Scored) 129

6 Linux Kernel Security 131
6.1 SELinux 131
6.1.1 Ensure SELINUX is Set to Enforcing (Scored) 131
6.1.2 Ensure the httpd_t Type is Not in Permissive Mode (Scored) 133
Appendix: Summary Table 135
Appendix: Change History 138

Overview

This document, CIS NGINX Benchmark, provides prescriptive guidance for establishing a secure configuration posture for NGINX version 1.14.0 running on Linux.
This guide was tested against NGINX version 1.14.0 using the packages installed using yum from nginx.org. This Benchmark was written using commands for, and tested on Centos 7.6. For other version of Linux, please substitute the Centos specific commands for the equivalent commands on the Linix distribution you are using.
To obtain the latest version of this guide, please visit http://benchmarks.cisecurity.org. If you have questions, comments, or have identified ways to improve this guide, please write us at feedback@cisecurity.org.

Intended Audience

This document is intended for system and application administrators, security specialists, auditors, help desk, and platform deployment personnel who plan to develop, deploy, assess, or secure solutions that incorporate NGINX.

Consensus Guidance

This benchmark was created using a consensus review process comprised of subject matter experts. Consensus participants provide perspective from a diverse set of backgrounds including consulting, software development, audit and compliance, security research, operations, government, and legal.

Each CIS benchmark undergoes two phases of consensus review. The first phase occurs during initial benchmark development. During this phase, subject matter experts convene to discuss, create, and test working drafts of the benchmark. This discussion occurs until consensus has been reached on benchmark recommendations. The second phase begins after the benchmark has been published. During this phase, all feedback provided by the Internet community is reviewed by the consensus team for incorporation in the benchmark. If you are interested in participating in the consensus process, please visit https://workbench.cisecurity.org/.

Typographical Conventions

The following typographical conventions are used throughout this guide:

Convention	Meaning
`Stylized Monospace font`	Used for blocks of code, command, and script examples. Text should be interpreted exactly as presented.
Monospace font	Used for inline code, commands, or examples. Text should be interpreted exactly as presented.
<italic font in brackets>	Italic texts set in angle brackets denote a variable requiring substitution for a real value.
Italic font	Used to denote the title of a book, article, or other publication.
Note	Additional information or caveats

Scoring Information

A scoring status indicates whether compliance with the given recommendation impacts the assessed target's benchmark score. The following scoring statuses are used in this benchmark:

Scored

Failure to comply with "Scored" recommendations will decrease the final benchmark score. Compliance with "Scored" recommendations will increase the final benchmark score.

Not Scored

Failure to comply with "Not Scored" recommendations will not decrease the final benchmark score. Compliance with "Not Scored" recommendations will not increase the final benchmark score.

Profile Definitions

The following configuration profiles are defined by this Benchmark:

- **Level 1 - Webserver**

  Items in this profile intend to:

  - be practical and prudent;
  - provide a clear security benefit; and
  - not inhibit the utility of the technology beyond acceptable means.

- **Level 1 - Proxy**

  Items in this profile intend to:

  - be practical and prudent;
  - provide a clear security benefit; and
  - not inhibit the utility of the technology beyond acceptable means.

- **Level 1 - Loadbalancer**

  Items in this profile intend to:

  - be practical and prudent;
  - provide a clear security benefit; and
  - not inhibit the utility of the technology beyond acceptable means.

- **Level 2 - Webserver**

  This profile extends the "Level 1" profile. Items in this profile exhibit one or more of the following characteristics:

  - are intended for environments or use cases where security is paramount
  - acts as defense in depth measure
  - may negatively inhibit the utility or performance of the technology

- **Level 2 - Proxy**

  This profile extends the "Level 1" profile. Items in this profile exhibit one or more of the following characteristics:

  - are intended for environments or use cases where security is paramount

- acts as defense in depth measure
- may negatively inhibit the utility or performance of the technology

- **Level 2 - Loadbalancer**

  This profile extends the "Level 1" profile. Items in this profile exhibit one or more of the following characteristics:

  - are intended for environments or use cases where security is paramount
  - acts as defense in depth measure
  - may negatively inhibit the utility or performance of the technology

Acknowledgements

This benchmark exemplifies the great things a community of users, vendors, and subject matter experts can accomplish through consensus collaboration. The CIS community thanks the entire consensus team with special recognition to the following individuals who contributed greatly to the creation of this guide:

Editor
Alexander Sennhauser
James Scott

Recommendations

1 Initial Setup

This section contains recommendations for the installation and maintenance of a NGINX server.

1.1 Installation

1.1.1 Installing NGINX (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The CIS NGINX Benchmark recommends using the NGINX binary provided by your vendor for most situations in order to reduce the effort and increase the effectiveness of maintenance and security patches.

As an alternative, packages from nginx.org are available for a variety of platforms including Linux and FreeBSD.

Rationale:

The main benefits of using NGINX packages from your vendor are:

- Ease of installation
- Dependency resolution
- Increased effectiveness of maintenance and security patches
- Q&A procedures carried out by your vendor

Audit:

To check if nginx is installed on your server run the following command:

```
nginx -v
```

The command output should return the version of nginx that is installed on the server. If there is no output then nginx is not installed.

Remediation:

Installation depends on the operating system platform. For a source build, consult the NGINX documentation "Building nginx from Sources".
Configure repo:
Example:

```
#Configure your repo
cat << EOF > /etc/yum.repos.d/nginx.repo
[nginx]
name=nginx repo
baseurl=http://nginx.org/packages/centos/7/\$basearch/
gpgcheck=1
enabled=1
EOF
```

Download signing key:
Example:

```
#Download Signing Key From The Internet
curl -O https://nginx.org/keys/nginx_signing.key
#import signing key so you do not get an error installing nginx
rpm --import nginx_signing.key
```

Install NGNIX:
Example:

```
yum install nginx -y
```

Default Value:

NGINX is not installed by default.

References:

1. http://nginx.org/en/docs/install.html
2. http://nginx.org/en/linux_packages.html

CIS Controls:

Version 7

2 Inventory and Control of Software Assets
Inventory and Control of Software Assets

1.2 Configure Software Updates

1.2.1 Ensure Package Manager Repositories are Configured (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

Systems need to have package manager repositories configured to ensure they receive the latest patches and updates.

Rationale:

If a system's package repositories are misconfigured important patches may not be identified or a rogue repository could introduce compromised software.

Audit:

Verify package manager repositories are configured correctly.
Run the following command:

```
yum repolist -v nginx
```

Remediation:

Configure your package manager repositories according to your vendor.
As an alternative, package repositories from nginx.org are available for a variety of Linux platforms.

References:

1. http://nginx.org/en/linux_packages.html

Notes:

Package update and installation commands are based on centOS 7. If using a different Linux Distribution, please substitute with the appropriate command(s).

CIS Controls:

Version 7

3.4 Deploy Automated Operating System Patch Management Tools

Deploy automated software update tools in order to ensure that the operating systems are running the most recent security updates provided by the software vendor.

3.5 Deploy Automated Software Patch Management Tools

Deploy automated software update tools in order to ensure that third-party software on all systems is running the most recent security updates provided by the software vendor.

1.2.2 Ensure the Latest Software Package is Installed (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

As new security vulnerabilities are discovered the corresponding fixes are implemented by your NGINX software package provider. Installing the latest software version ensures these fixes are available on your system.

Rationale:

Up-to-date software provides the best possible protection against security vulnerabilities such as the execution of malicious code.

Audit:

Verify your NGINX package is up to date.
Run the following command:

```
yum info nginx
```

Remediation:

Install the latest NGINX package.
Run the following command

```
yum update nginx -y
```

References:

1. http://nginx.org/en/linux_packages.html

Notes:

Package update and installation commands are based on centOS 7. If using a different Linux Distribution, please substitute with the appropriate command(s)

CIS Controls:

Version 7

3.4 Deploy Automated Operating System Patch Management Tools

Deploy automated software update tools in order to ensure that the operating systems are running the most recent security updates provided by the software vendor.

3.5 Deploy Automated Software Patch Management Tools

Deploy automated software update tools in order to ensure that third-party software on all systems is running the most recent security updates provided by the software vendor.

2 Basic Configuration

2.1 Minimize NGINX Modules

2.1.1 Ensure only Required Modules are Installed (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

This NGINX installation comes with several modules out of the box. These modules are not all always needed in order to ensure the functionality of your application. Installations of NGINX should be hardened to ensure the minimum necessary number of modules are installed.

Rationale:

Minimizing features and functionality built into nginx can help to ensure that your server is not exposed to any vulnerability that is not necessary for your application's functionality.

Audit:

Modules used in your current nginx build can be audited using the nginx verification command.

```
nginx -V
```

Remediation:

Consult the NGINX module documentation to ensure that all modules configured are needed for your specific installation.
The NGINX module documentation can be found at the nginx documentation.
Modules may be removed using the configure command.

References:

1. http://nginx.org/en/docs/configure.html

Notes:

NOTE: NGINX does not support the removal of modules using the yum method of installation. In order to remove modules from NGINX, you will need to compile it from source.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.1.2 Ensure HTTP DAV Module is not Installed (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The http_dav_module enables HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV) as defined by RFC4918. This enables file-based operations on your web server, such as the ability to create, delete, change and move files on your server.

Rationale:

WebDAV functionality opens up an unnecessary potential for exploitation on your web server. Through misconfigurations of WebDAV operations through this module an attacker may be able to access and manipulate files on the server. Most modern architectures have replaced this functionality with cloud-based object storage.

Audit:

Run the following procedure to ensure that the http_dav_module is not installed:
Step 1: Check if the module is present

```
nginx -V 2>&1 | grep http_dav_module
```

Ensure the output of the command is empty. If this is the case then the HTTP DAV module is not installed.

Remediation:

To remove the http_dav_module you will need to recompile nginx from source without the --with-http_dav_module flag.

Impact:

You will not be able to use standard WebDAV HTTP methods.

Default Value:

The HTTP DAV module is not installed by default when installing from source. It does come by default when installed using yum.

References:

1. http://nginx.org/en/docs/configure.html
2. https://tools.ietf.org/html/rfc4918

Notes:

NGINX does not support the removal of modules using the yum method of installation. In order to remove modules from NGINX, you will need to compile from source.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.1.3 Ensure Modules With GZIP Functionality are Disabled (Scored)

Profile Applicability:

- Level 2 - Webserver
- Level 2 - Proxy
- Level 2 - Loadbalancer

Description:

The disabling of compression functionality is a means of mitigating against vulnerabilities, such as breach.

Rationale:

Compression has been linked with a number of attacks, most notably the Breach attack. While breach has been mitigated against with modern usages of the HTTP protocol, disabling compression and its use is considered a defense in depth strategy to mitigate against the breach attack.

Audit:

Run the following procedure to ensure that gzip modules are not installed:
Step 1: Check if the module is present

```
nginx -V | grep 'http_gzip_module\|http_gzip_static_module'
```

Ensure the output of the command is empty. If this is the case then the gzip modules are not installed.

Remediation:

In order to disable the http_gzip_module nginx must be recompiled from source. This can be accomplished using the below command in the folder you used during your original compilation. This must also be done without the --with-http_gzip_static_module configuration directive.

```
./configure --without-http_gzip_module
```

Impact:

This ensures that the server will not be able to leverage gzip functionality.

Default Value:

The http_gzip_module is enabled by default in the source build and the http_gzip_static_module is not. Both are enabled by default in the yum package.

References:

1. http://nginx.org/en/docs/configure.html
2. http://nginx.org/en/docs/configure.html

Notes:

NGINX does not support the removal of modules using the yum method of installation. In order to remove modules from NGINX, you will need to compile from source.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.1.4 Ensure the Autoindex Module is Disabled (Scored)

Profile Applicability:

- Level 1 - Webserver

Description:

The nginx auto index module processes requests ending with the slash character. This feature enables directory listing which could be useful in attacker reconnaissance.

Rationale:

Automated directory listings should not be enabled as it will also reveal information helpful to an attacker such as naming conventions and directory paths. Directory listings may also reveal files that were not intended to be revealed.

Audit:

Perform the following to determine if the recommended state is implemented:

1. Search the NGINX configuration files (nginx.conf and any included configuration files) to find any autoindex directives:

```
egrep -i '^\s*autoindex\s+' $NGINX_PREFIX/nginx.conf
egrep -i '^\s*autoindex\s+' $NGINX_PREFIX/conf.d/*
```

2. Ensure there are no autoindex directives present, or there are, the values are set to off.

Remediation:

Perform the following to implement the recommended state:

1. Search the nginx configuration files (nginx.conf and any included configuration files) to find autoindex directives.

```
egrep -i '^\s*autoindex\s+' $NGINX_PREFIX/nginx.conf
egrep -i '^\s*autoindex\s+' $NGINX_PREFIX/conf.d/*
```

2. Set the value for all autoindex directives to off or remove those directives.

Default Value:

This module is not enabled by default.

References:

1. http://nginx.org/en/docs/http/ngx_http_autoindex_module.html

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.2 Account Security

2.2.1 Ensure that NGINX is Run Using a Non-Privileged, Dedicated Service Account (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The nginx user directive designates what user nginx worker processes run under. Ensuring that a dedicated service account is used is a defense in depth measure in case the user is compromised.

Rationale:

Running a web server under a dedicated service account helps mitigate the risk of lateral movement to other services or processes in the event that the user running the web services is compromised. The default user nobody is typically used for several processes and if this is compromised could allow an attacker to have access to all processes running as that user. Running as a dedicated non-privileged user makes it harder for an attacker to propagate throughout the system after an initial compromise.

Audit:

Run the following procedure to determine if nginx is being run by a dedicated non-privileged user account:
Step 1: Check to see if nginx is being run as a dedicated user:

```
grep "user[^;]*;" /etc/nginx/nginx.conf
```

If the user directive similar to the below is not found then this is not a dedicated user. If a user is found similar to the output shown below then continue to step 2. If the user does not exist we will need to add a user.

```
user  nginx;
```

Step 2: Check to ensure the nginx dedicated user is not privileged:
Run the below command replacing nginx with any designated user you may have assigned:

```
sudo -l -U nginx
```

The output should look similar to the below and identify that this user is not privileged:

```
sudo -l -U nginx
User nginx is not allowed to run sudo
```

Step 3: Check to ensure the nginx dedicated user is not part of any unexpected groups:
Run the below command replacing nginx with any designated user you may have assigned:

```
groups nginx
```

The output should look similar to the below and identify that this user is not part of any other groups than the primary group:

```
nginx : nginx
```

Remediation:

The following procedure will add a system account for the nginx user with a home directory of /var/cache/nginx and a shell of /sbin/nologin to ensure that it does not have the ability to log in. It will then add the nginx user to be used by nginx.

```
user add nginx -r -g nginx -d /var/cache/nginx -s /sbin/nologin
```

Then add the nginx user to /etc/nginx/nginx.conf by adding the user directive as shown below:

```
user nginx;
```

Default Value:

By default, if nginx is compiled from source, the user and group are nobody. If downloaded from yum the user and group nginx and the account is not privileged.

References:

1. http://nginx.org/en/docs/ngx_core_module.html#user

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.2.2 Ensure the NGINX Service Account is Locked (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The user account under which nginx runs should have a valid password but should be locked.

Rationale:

As a defense-in-depth measure, the nginx user account should be locked to prevent logins and to prevent a user from switching users to nginx using the password. In general, there shouldn't be a need for anyone to have to su as nginx, and when there is a need, then sudo should be used instead, which would not require the nginx account password.

Audit:

Ensure the nginx account is locked using the following:

```
passwd -S nginx
```

The results will be similar to the following:

```
nginx LK 2010-01-28 0 99999 7 -1 (Password locked.)
```

Or

```
nginx L 07/02/2012 -1 -1 -1 -1
```

Remediation:

Use the `passwd` command to lock the `nginx` account:

```
passwd -l nginx
```

Impact:

This ensures the nginx user account may not be used by a human user.

Default Value:

The nginx user is locked by default.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.2.3 Ensure the NGINX Service Account has an Invalid Shell (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The nginx account should not have the ability to log in and should only be used for the nginx service. To ensure this account does not have login capabilities we must set the /sbin/nologin shell to the account.

Rationale:

The account used for nginx should only be used for the nginx service and does not need to have the ability to log in. Ensuring that this service is unable to log in is a simple hardening task.

Audit:

Check the nginx login account shell in the /etc/passwd file using the following command:

```
grep nginx /etc/passwd
```

The nginx must be /sbin/nologin similar to the example output shown below:

```
nginx:x:997:994:nginx user:/var/cache/nginx:/sbin/nologin
```

Remediation:

Change the login shell for the nginx account to ensure that it uses /sbin/nologin using the following command:

```
chsh -s /sbin/nologin nginx
```

Impact:

This will ensure that the nginx account can't be used as a login account.

Default Value:

The nginx user has a shell of /sbin/nologin by default.

References:

1. CIS Apache Server Benchmark

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.3 Permissions and Ownership

2.3.1 Ensure NGINX Directories and Files are Owned by Root (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The owner and group of the /etc/nginx directory should be root.

Rationale:

Reducing principles and ownership to only those users in the root group and the root user will reduce the likelihood of unauthorized modifications to the nginx configuration files.

Audit:

Run the following procedure to check the ownership of the nginx configuration files.

```
ls -ld /etc/nginx
```

The output should show the ownership and group as root similar to the below output.

```
drwxr-xr-x. 4 root root 188 Nov 28 23:22 /etc/nginx/
```

Remediation:

Run the following procedure to ensure that ownership and group ownership is set to root.

```
chown -R root:root /etc/nginx
```

Default Value:

The default ownership and group for nginx is root.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.3.2 Ensure Access on NGINX Directories and Files is Restricted (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

Permissions on the /etc/nginx directory are more permissive than required by default.

Rationale:

Permissions on the file system should always be hardened to least privilege. This ensures that only users on the operating system who need access to these configuration files are able to view them. Other users will need to use sudo in order to access these files.

Audit:

To ensure that the nginx directory has other read and execute permissions revoked look at the permissions by running the below command:

```
find /etc/nginx -type d | xargs ls -ld
```

The output should show the permissions similar to the below output:

```
drwxr-x---. 4 root root 188 Nov 28 23:22 /etc/nginx
```

To ensure that the nginx configuration files have other read and execute permissions revoked look at the permissions by running the below command:

```
find /etc/nginx -type f | xargs ls -l
```

The output should show the permissions similar to the below output.

```
-rw-r-----. 1 root root 2192 Nov 11  2017 /etc/nginx/nginx.conf
```

Remediation:

To set permissions to least privilege on the nginx configuration files issue the below commands:

```
find /etc/nginx -type d | xargs chmod 750
find /etc/nginx -type f | xargs chmod 640
```

Impact:

Users on the operating system who are not the owner or group will not be able to read, write or execute any files in the /etc/nginx directory.

Default Value:

Permissions are set with the ability to read as other by default: -rw-r--r--

References:

1. https://dev-sec.io/baselines/nginx/

Notes:

Note: You should always check your private key permissions after implementing this recommendation. This recommendation assumes the private key has not yet been created or is not in the /etc/nginx directory.

CIS Controls:

Version 6

14 Controlled Access Based on the Need to Know
Controlled Access Based on the Need to Know

2.3.3 Ensure NGINX Process ID (PID) File is Secured (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The NGINX PID file is the file that stores the main process ID of the NGINX process. This file should be protected to prevent a denial of service attack.

Rationale:

The NGINX PID should be owned by root and the group root. It should also be readable to everyone but only writable by root. Permissions of the PID file should be 644. This will prevent unauthorized modification of the PID file. If the PID file is altered by an unauthorized user it will cause a denial of service attack.

Audit:

Run the below command to check the ownership and permissions of the nginx PID file. Ownership should be root:root and permissions should be 644.

```
ls -l /var/run/nginx.pid
```

The output should show that the PID file is owned by root and has the group root as shown below.

```
-rw-r--r--. 1 root root 6 Nov 12 01:06 /var/run/nginx.pid
```

If this is not the location of the PID file the PID file location can be found using the output of the below command.

```
nginx -V
```

Remediation:

If the NGINX PID file is not owned by root issue the below command:

```
chown root:root /var/run/nginx.pid
```

If the PID file has permissions greater than 644 issue the below command:

```
chown 644 /var/run/nginx.pid
```

Default Value:

The PID file is owned by root by default.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.3.4 Ensure Core Dump Directory Is Secured (Not Scored)

Profile Applicability:

- Level 1 - Webserver

Description:

The `working_directory` directive is used to specify the directory NGINX attempts to create the core dump. Core dumps will be disabled if the directory is not writable by the NGINX user. It is recommended that the `working_directory` directive be set to a directory that is owned by the root user, owned by the group the NGINX process executes as, and be inaccessible to other users. Usually, production systems should not have this enabled, however, if it is required for debugging the following procedure should be conducted.

Rationale:

Core dumps are snapshots of memory and may contain sensitive information that should not be accessible by other accounts on the system.

Audit:

Run the following procedure to determine if the core dump configuration is secured:
Step 1: Check to see if the `working_directory` directive is configured:

```
grep working_directory /etc/nginx/nginx.conf
```

Step 2: If the `working_directory` directive is enabled it needs to meet the following requirements:

1. `working_directory` is not within the NGINX web document root.
2. Must be owned by root and have a group ownership of the NGINX group.
3. Must have no read-write-search access permission for other users. (e.g. o=rwx)

Remediation:

Either remove the `working_directory` directive from the NGINX configuration files or ensure that the configured directory meets the following requirements.

1. `working_directory` is not to be within the NGINX web document root
2. Must be owned by root and have a group ownership of the NGINX group.

```
chown root:nginx /var/log/nginx
```

3. Must have no read-write-search access permission for other users.

```
chmod o-rwx /var/log/nginx
```

Default Value:

The `working_directory` value is not set by default.

References:

1. https://www.nginx.com/resources/wiki/start/topics/tutorials/debugging/#core-dump

CIS Controls:

Version 7

5.1 Establish Secure Configurations
Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.4 Network Configuration

2.4.1 Ensure NGINX only Listens for Network Connections on Authorized Ports (Not Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

NGINX can be configured to listen on many ports. It is capable of listening on any port of your choice. NGINX should be configured to only listen on authorized ports.

Rationale:

Limiting the listening ports to only those that are authorized helps to ensure that no unauthorized services are running through the use of nginx.

Audit:

Use the below command in order to audit all listening ports on the server.

```
grep -ir listen /etc/nginx
```

Ports used should immediately follow the listen directive. Ensure that all ports that are actively listening and not commented out are authorized for use on the server.
Output should look similar to the below:

```
/etc/nginx/conf.d/default.conf:    listen 80 default_server;
/etc/nginx/conf.d/default.conf:    listen 443 ssl http2;
/etc/nginx/conf.d/default.conf:    listen [::]:443 ssl http2;
```

Remediation:

If any ports are listening that are not authorized comment or delete the associated configuration with that listener.

Default Value:

Only port 80 is listening by default.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.4.2 Ensure that Requests for Unknown Host Names are Rejected (Not Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

This configuration ensures that your host header is part of a pre-defined whitelist of known good hosts.

Rationale:

By whitelisting specific hosts and blocking access to all other hosts you help to mitigate host header injection on your server. This helps prevent injection of your host header, which may be used in your application code or your nginx configuration to direct you to websites. This could be used by an attacker to redirect you to a rogue host and execute scripts or get you to input credentials.

You should treat the host header as another input validation as it is defined by the user agent.

Audit:

Run the following procedure to check to ensure this is configured.

```
curl -k -v https://127.0.0.1 -H 'Host: invalid.host.com'
```

If you do not receive a 404 error or receive any kind of 200 response then this recommendation is not implemented.

Remediation:

Ensure that your first server block mirrors the below in your nginx configuration:

```
server {
    return 404;
}
```

Then investigate each server block to ensure that the server_name directive is explicitly defined. For example, each server block should look similar to the below with the defined hostname of the associated server block in the server_name directive. For example, if your server is cisecurity.org the configuration should look like the below example.

```
server {
    listen       443;
    server_name  cisecurity.org;
    .....
}
```

Impact:

If you are in an environment, such as the cloud, you should not put an IP address or default hostname as your server_name because these addresses are often ephemeral in nature. Additionally, you will be blocked from accessing your site if you use a means of access that does not directly reference names in the server_name directive. You should reserve a DNS name for use to implement this recommendation.

Default Value:

This is not set by default.

References:

1. https://www.acunetix.com/blog/articles/automated-detection-of-host-header-attacks/
2. https://hackerone.com/reports/94637
3. https://stackoverflow.com/questions/9824328/why-is-nginx-responding-to-any-domain-name

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.4.3 Ensure keepalive_timeout is Configured (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

Persistent connections are leveraged by all modern browsers in order to facilitate greater web performance. The keep-alive timeout limits the time that that persistent connection may remain open. Setting the keep-alive timeout allows this timeout to be controlled on the server side.

Rationale:

Setting a keep-alive timeout on the server side allows you to protect yourself from denial of service attacks. This helps mitigate against slow denial of service attacks. If too many persistent connections are established on the server, this may exhaust server resources and ultimately cause a denial of service.

Audit:

To check the current setting for the keepalive_timeout directive issue the below command. You should also manually check your nginx configuration for include statements that may be located outside of the /etc/nginx directory. If this is not present then the value is set at the default.

```
grep -ir keepalive_timeout /etc/nginx
```

The output of the command should contain the following output.

```
keepalive_timeout 10;
```

Remediation:

Find the HTTP or server block of your nginx configuration and add the keepalive_timeout directive and set it to no more than 10 seconds and not 0.

```
keepalive_timeout 10;
```

Default Value:

By default, this timeout is dictated by the user agent and varies. It is not set on the server side by default.

References:

1. http://nginx.org/en/docs/http/ngx_http_core_module.html#keepalive_timeout

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.4.4 Ensure send_timeout is configured (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The send_timeout directive sets a timeout for transmitting a response to the client between two successive write operations.

Rationale:

Setting the send_timeout directive on the server side allows you to protect yourself from slow HTTP denial of service attacks. This helps mitigate against denial of service attacks by ensuring that write operations taking up large amounts of time are closed.

Audit:

To check the current setting for the send_timeout directive issue the below command. You should also manually check your nginx configuration for include statements that may be located outside of the /etc/nginx directory. If this is not present then the value is set at the default.

```
grep -ir send_timeout /etc/nginx
```

The output of the command should contain the following output and be set to a reasonably low number, such as 10.

```
send_timeout  10;
```

Remediation:

Find the HTTP or server block of your nginx configuration and add the send_timeout directive and set it to no more than 10 seconds and not 0.

```
send_timeout   10;
```

Default Value:

send_timeout 60s;

References:

1. https://www.owasp.org/index.php/SCG_WS_nginx
2. http://nginx.org/en/docs/http/ngx_http_core_module.html#send_timeout

2.5 Information Disclosure

2.5.1 Ensure server_tokens directive is set to Off (Scored)

Profile Applicability:

- Level 1 - Webserver

Description:

The `server_tokens` directive is responsible for displaying the NGNIX version number and Operating system on error pages, and in the `Server` HTTP response header field

Rationale:

Attackers are able to conduct reconnaissance on a website using these response headers. This header could be used to target attacks for specific known vulnerabilities associated with the underlying technology. Removing this header will prevent the targeting of your application for specific exploits by non-determined attackers.

Potential attackers may check if your version of NGINX contains known vulnerabilities. Hiding the version will slow down and mitigate potential attackers.

Audit:

In the NGINX configuration file nginx.conf ensure the server_tokens directive is set to off. Check the response headers for the server header by issuing the below command:

```
curl -I 127.0.0.1 | grep -i server
```

The output should not contain the server header providing your server version and server such as the below:

```
Server: nginx/1.14.0
```

Remediation:

To disable the server_tokens directive, set if to off inside a server block in your nginx.conf:

```
server {
    ...
    server_tokens        off;
    ...
}
```

Default Value:

The default value of server_tokens is on.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.5.2 Ensure Default Error Pages and index.html Pages don't reference NGINX (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The default error and index.html pages for nginx reveal that the server is nginx. These default pages should be removed or modified so that they do not advertise the underlying infrastructure of the server.

Rationale:

Information about the underlying infrastructure is a useful part of the reconnaissance phase of an attack. By gathering information about the server attackers can target attacks against you with known vulnerabilities. Removing pages that disclose that the server run with nginx helps mitigate against targeted attacks on your server.

Audit:

Locate the error page and index directives in the location block of your server configuration. Open these files and check for references of nginx. If these files reference nginx they should be removed and customized.
The default index and error pages in nginx are located at /usr/share/nginx/html/
Issue the following commands to check the default pages:

```
grep -i nginx /usr/share/nginx/html/index.html
grep -i nginx /usr/share/nginx/html/50x.html
```

Ensure that no results are returned

Remediation:

Edit `/usr/share/nginx/html/index.html` and `usr/share/nginx/html/50x.html` and remove any lines that reference `NGNIX`

Impact:

You will have to customize these files based on your use case.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.5.3 Ensure Hidden File Serving Is Disabled (Scored)

Profile Applicability:

- Level 2 - Webserver

Description:

Disabling hidden files is a defense in depth mechanism to use in order to help prevent accidentally exposing sensitive information.

Rationale:

By disabling hidden files, you prevent an attacker from being able to reference a hidden file that may be put in your location. This may help you prevent an attacker from being able to read sensitive hidden information, like .git files.

Audit:

To ensure that this recommendation is implemented, open your nginx configuration file and search for the below string or another regex pattern that denies access to files with a dot as the first character in the file path.
Run the following command:

```
grep location /etc/nginx/nginx.conf
```

and ensure the output is:

```
location ~ /\.  { deny all; return 404; }
```

Remediation:

To implement this recommendation, edit the `nginx.conf` file and add following line:

```
location ~ /\.  { deny all; return 404; }
```

Impact:

This may break well known hidden files that are needed for functionality, for example, it may prevent functionality used by LetsEncrypt. To enable configure a location exception like that shown below.

```
location ~ /\.well-known\/acme-challenge {

    allow all;
```

```
}
```

Default Value:

This is not set by default.

References:

1. https://programming-review.com/nginx-disable-access-to-htaccess-file/

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

2.5.4 Ensure Proxy does not Enable Information Disclosure (Scored)

Profile Applicability:

- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The server and x-powered-by header may specify the underlying technology used by an application. Attackers are able to conduct reconnaissance on a website using these response headers. This header could be used to target attacks for specific known vulnerabilities associated with the underlying technology. Removing this header will prevent the targeting of your application for specific exploits by non-determined attackers.

The nginx reverse proxy may pass these headers if not explicitly directed to remove them.

Rationale:

While this is not the only way to fingerprint a site through the response headers, it makes it harder and prevents some potential attackers.

Audit:

The below headers should be denied as part of the location block of your nginx configuration. You may also have to check included files as part of this configuration. Check to identify if the below directives are part of your nginx configuration.
Run the following commands:

```
grep proxy_hide_header /etc/nginx/nginx.conf
```

The output should read:

```
proxy_hide_header X-Powered-By;
```

AND

```
grep proxy_hide_header
```

The output should read:

```
proxy_hide_header Server;
```

Remediation:

Implement the below directives as part of your location block as shown below.
Edit `/etc/nginx/nginx.conf` and add the following:

```
location /docs {
....
proxy_hide_header X-Powered-By;
proxy_hide_header Server;
....
}
```

Default Value:

This is not implemented by default.

References:

1. http://nginx.org/en/docs/http/ngx_http_proxy_module.html

CIS Controls:

Version 7

5.1 Establish Secure Configurations
Maintain documented, standard security configuration standards for all authorized operating systems and software.

3 Logging

3.1 Ensure Detailed Access and Request Logging Is Enabled (Not Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

System logging should always be configured to meet your organizational security and privacy policies. Enabling detailed system logging to include information about events, event sources, timestamps, users, and other useful elements may assist in incident response activities.

Rationale:

Configuring detailed access logging ensures that incident responders and auditors are able to clearly trace back the activity that occurs on your server. This information is useful in the event of an incident or audit and can be used to trace back an attacker's activity when accessing your system.

Audit:

Find or implement the log format directive in /etc/nginx/nginx.conf. Ensure that you are logging everything needed to meet your organizational policies.
The following variables may be considered as useful examples include in your log_format with descriptive logging. You should always consult the nginx documentation and your organizational policy to ensure that you are logging sufficient information and removing sensitive information where needed.

```
$remote_addr - client address.
$remote_user - the user if basic authentication is used.
$status - the HTTP response status.
$content_type - Content-Type request header field.
$time_local - local time in the Common Log Format.
$request_method - request method, usually GET or POST.
$request - full original request line.
$uri - normalized URI in request.
$server_port - port of the server which accepted a request.
$server_name - name of the server which accepted a request.
$http_user_agent - The user agent of the client requesting access.
```

```
$http_x_forwarded_for - The client address a proxy or load balancer is
forwarding traffic for.
```

Remediation:

Ensure your log format meets your organizational security and privacy policies. All necessary logging variables should contain descriptive definitions at /etc/nginx/nginx.conf. An example can be found below:

```
log_format main
'sever="$server_name" host="$host" dest_port="$server_port"'
'src="$remote_addr" ip="$realip_remote_addr" user="$remote_user" '
'time_local="$time_local" http_status="$status" '
'http_referer="$http_referer" http_user_agent="$http_user_agent" '
'http_x_forwarded_for="$http_x_forwarded_for" '
'http_x_header="$http_x_header" uri_query="$query_string" uri_path="$uri" '
'request=$request http_method="$request_method";
```

Impact:

We should always aim to keep sensitive information out of our logs. Keep sensitive information out of the query string or URI to avoid this. It is always best to check your logs for sensitive information to ensure that an application is designed to not log or contain this information in query strings or logs. Additionally, all information logged should be explicitly stated in your site privacy policy.

Default Value:

```
log_format  main  '$remote_addr - $remote_user [$time_local]
"$request" ' '$status $body_bytes_sent "$http_referer" '

'"$http_user_agent" "$http_x_forwarded_for"';
```

References:

1. http://nginx.org/en/docs/http/ngx_http_log_module.html#log_format

Notes:

Note: Load Balancers are not source IP transparent. We must configure the X-Forwarded-For Header on our proxy and in our logs to show where the request is coming from.

CIS Controls:

Version 7

6.3 Enable Detailed Logging

Enable system logging to include detailed information such as a event source, date, user, timestamp, source addresses, destination addresses, and other useful elements.

3.2 Ensure that Access Logging is Enabled (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The access_log directive should not be turned off on any core site. This is enabled by default and must be explicitly turned off.

Rationale:

Access Logging allows incident responders and auditors to investigate access to your system in the event of an incident. Access logging should always be turned on for any core site you host.

Audit:

Run the following procedure in order to ensure that access logging is enabled.

```
grep -ir access_log /etc/nginx
```

The output of this command should show an access log configured and that the access log is not set to off.
Any output similar to the below should be manually inspected in the nginx configuration file to ensure that you are logging access to all core sites and proxies.

```
access_log off;
```

Remediation:

Ensure the access_log directive is configured for any core site that your organization requires logging for.
This should look similar to the below snippet of configuration. You may use different log file locations based on your needs.

```
access_log  /var/log/nginx/host.access.log  main;
```

Default Value:

The access log is enabled by default.

References:

1. http://nginx.org/en/docs/http/ngx_http_log_module.html#log_format

CIS Controls:

Version 7

6.3 Enable Detailed Logging

Enable system logging to include detailed information such as a event source, date, user, timestamp, source addresses, destination addresses, and other useful elements.

3.3 Ensure that Error Logging is Enabled and Set to the Info Logging Level (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

Ensure that error logging is configured to ensure that all errors for the applications are logged.

Rationale:

Error logging can be useful in proactively identifying an attacker attempting to exploit a system. It can also be useful in recreating an attacker's steps. Error logging also helps with identifying possible issues with your application.

Audit:

Run the following procedure to check for error logging configuration in /etc/nginx/nginx.conf:

```
grep error_log /etc/nginx/nginx.conf
```

If there is no output, the output is commented out or the logging level is set to anything other than info then this recommendation is not implemented.

Remediation:

Ensure that the error_log directive is present and not commented out in /etc/nginx/nginx.conf. This should be configured to the logging location of your choice. The error log configuration will look similar to the below:

```
error_log  /var/log/nginx/error.log info;
```

CIS Controls:

Version 7

6.3 Enable Detailed Logging

Enable system logging to include detailed information such as a event source, date, user, timestamp, source addresses, destination addresses, and other useful elements.

3.4 Ensure Log Files are Rotated (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

Log rotation ensures that your log files do not consume excessive disk space potentially denying service.

Rationale:

Log files are important to track activity that occurs on your server, however, they take up significant amounts of space. Log rotation should be configured in order to ensure that your server logs do not consume so much disk space that logging becomes unavailable.

Audit:

Run the below command in order to check log rotation configuration. This should show that the log compression occurs weekly and the log rotation occurs every 13 weeks.

```
cat /etc/logrotate.d/nginx | grep weekly
cat /etc/logrotate.d/nginx | grep rotate
```

Remediation:

Follow the below procedure in order to change the default configuration to the recommended log rotation configuration. You may need to manually edit or change the below command if the configuration is not the default.
To change Log Compression from daily to Weekly:

```
sed -i "s/daily/weekly/" /etc/logrotate.d/nginx
```

to change Log Rotation From Every Year to Every 13 weeks:

```
sed -i "s/rotate 52/rotate 13/" /etc/logrotate.d/nginx
```

Default Value:

```
cat /etc/logrotate.d/nginx
```

```
/var/log/nginx/*.log {
        daily
        missingok
        rotate 52
        compress
        delaycompress
        notifempty
        create 640 nginx adm
        sharedscripts
        postrotate
                if [ -f /var/run/nginx.pid ]; then
                        kill -USR1 `cat /var/run/nginx.pid`
                fi
        endscript
}
```

Notes:

Note: You should always comply with your organizational log retention policy. If you are not sending logs to a central syslog server that facilitates your compliance with this policy, you should check to ensure that this configuration meets your organizational policy.

CIS Controls:

Version 6

6.3 Ensure Audit Logging Systems Are Not Subject To Loss (i.e. rotation/archive)

Ensure that all systems that store logs have adequate storage space for the logs generated on a regular basis, so that log files will not fill up between log rotation intervals. The logs must be archived and digitally signed on a periodic basis.

Version 7

6.4 Ensure adequate storage for logs

Ensure that all systems that store logs have adequate storage space for the logs generated.

3.5 Ensure NGINX Error Logs are Sent to a Remote Syslog Server (Not Scored)

Profile Applicability:

- Level 2 - Webserver
- Level 2 - Proxy
- Level 2 - Loadbalancer

Description:

Central Log management helps ensure that logs are forensically sound and that logs are available at a central location for auditing and incident investigation.

Rationale:

A central logging solution aggregates logs from multiple systems to ensure logs can be referenced in the event logs are thought to be compromised. Central logging servers are also often used to correlate logs for potential patterns of attack. If a central logging solution is not used and logs are believed to be compromised then logs may be thrown out in the court of law and not be allowed to be used as documentary evidence. It can also help you to more effectively comply with your log retention policy.

Audit:

Follow the below procedure in order to check if your server is configured for central logging.

```
grep -ir syslog /etc/nginx
```

The output of the command should show the error logs being sent to a central server similar to the output of the command below. 192.168.2.1 should be replaced with your central logging server. The logging level should be set to info.

```
error_log syslog:server=192.168.2.1 info;
```

Remediation:

To enable central logging for your error logs you should add the below line to your server block in your server configuration file.
192.168.2.1 should be replaced with the location of your central log server.

```
error_log syslog:server=192.168.2.1 info;
```

Default Value:

Syslog is not configured by default.

References:

1. http://nginx.org/en/docs/syslog.html

CIS Controls:

Version 7

6.5 Central Log Management

Ensure that appropriate logs are being aggregated to a central log management system for analysis and review.

3.6 Ensure NGINX Access Logs are Sent to a Remote Syslog Server (Not Scored)

Profile Applicability:

- Level 2 - Webserver
- Level 2 - Proxy
- Level 2 - Loadbalancer

Description:

Central Log management helps ensure that logs are forensically sound and that logs are available at a central location for auditing and incident investigation. Access logs should be sent to this solution.

Rationale:

A central logging solution aggregates logs from multiple systems to ensure logs can be referenced in the event logs are thought to be compromised. Central logging servers are also often used to correlate logs for potential patterns of attack. If a central logging solution is not used and logs are believed to be compromised then logs may be thrown out in the court of law and not be allowed to be used as documentary evidence.

Audit:

Follow the below procedure in order to check if your server is configured for central logging.

```
grep -ir syslog /etc/nginx
```

The output of the command should show the error logs being sent to a central server similar to the output of the command below. 192.168.2.1 should be replaced with your central logging server. The local logging facility may be changed to any unconfigured facility on your server.

```
access_log syslog:server=192.168.2.1,facility=local7,tag=nginx,severity=info
combined;
```

Remediation:

To enable central logging for your access logs you should add the below line to your server block in your server configuration file.

192.168.2.1 should be replaced with the location of your central log server.
The local logging facility may be changed to any unconfigured facility on your server.

```
access_log syslog:server=192.168.2.1,facility=local7,tag=nginx,severity=info
combined;
```

Default Value:

Syslog is not set up by default.

References:

1. http://nginx.org/en/docs/syslog.html

CIS Controls:

Version 7

6.5 Central Log Management

Ensure that appropriate logs are being aggregated to a central log management system for analysis and review.

3.7 Ensure Proxies Pass Source IP Information (Scored)

Profile Applicability:

- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The x-forwarded-for header and remote address header help identify and separate the originating client IP address of the user agent and the proxy IP address. The X-Forwarded-For Address and the Remote Address are the exact same and one should always be present.

Rationale:

Being able to identify the originating client IP address can help, auditors or incident responders identify where the corresponding user came from. This may be useful in the event of an attack to analyze if the IP address is a good candidate for blocking. It may also be useful to correlate an attacker's actions.

Audit:

To audit this procedure, open the nginx configuration file and the associated included files in that configuration. Check all location blocks for the presence of the proxy_pass directive. The proxy_pass directive should be followed by one of the below two directives to ensure that the client IP address is passed to the endpoint the proxy is serving traffic to.

```
proxy_set_header X-Real-IP $remote_addr;
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
```

Remediation:

In order to ensure that your proxy or load balancer will forward information about the client and the proxy to the application, you must set the below headers in your location block.
Edit your location block so that it shows the proxy_set_header directives for the client and the proxy as shown below. These headers are the exact same and there is no need to have both present.

```
server {
    ...
 location / {
     proxy_pass (Insert Application URL here);
     proxy_set_header X-Real-IP $remote_addr;
```

```
     proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}
```

Impact:

Users privacy should be kept in mind when deploying this header. If this is deployed you should ensure that your privacy policy includes that you collect IP address information about them.

Default Value:

This is not set by default.

References:

1. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Forwarded-For
2. http://nginx.org/en/docs/http/ngx_http_proxy_module.html

CIS Controls:

Version 6

6.4 Regularly Monitor Logs For Anomalies

Have security personnel and/or system administrators run biweekly reports that identify anomalies in logs. They should then actively review the anomalies, documenting their findings.

Version 7

6.7 Regularly Review Logs

On a regular basis, review logs to identify anomalies or abnormal events.

4 Encryption

4.1 TLS / SSL Configuration

4.1.1 Ensure HTTP is Redirected to HTTPS (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

Browsers and clients establish encrypted connections with servers leveraging https. Requests leveraging HTTP are unencrypted. Allowing unencrypted traffic to your website could allow for unauthorized access to a website users information through network monitoring.

Rationale:

Redirecting user agent traffic to HTTPS helps to ensure that all user traffic is encrypted to your site. This helps protect sensitive user information. Encryption in transit is also a good business practice. This ensures that users are able to trust your site with their information. Modern Browsers alert users that your website is insecure when HTTPS is not used. This can decrease user trust in your website and ultimately result in decreased use of your web services. Redirection from HTTP to HTTPS couples security with usability for users. This ensures that users are always able to access your website even if they lack the security awareness to use HTTPS over HTTP when requesting your website.

Audit:

Ensure that any listening port on your web server redirects to a server profile that uses encryption. The default unencrypted port is 80. To check your server listening configuration check your web server or proxy configuration file. The default web server configuration file is /etc/nginx/conf.d/default.conf and the default proxy configuration file is /etc/nginx/nginx.conf. This configuration file should a return statement redirecting to HTTPS. This should be similar to the code below where cisecurity.org is used as an example.

```
server {
    listen 80;

    server_name cisecurity.org;

    return 301 https://$host$request_uri;
}
```

Remediation:

Ensure that your website redirects all unencrypted listening ports, such as port 80 using a redirection through the return directive (cisecurity.org is used as an example server name).

```
server {
    listen 80;

    server_name cisecurity.org;

    return 301 https://$host$request_uri;
}
```

Impact:

Use of HTTPS does result in a performance reduction in traffic to your website, however, due to the increased value of the security, many businesses consider this to be a cost of doing business.

Default Value:

NGINX is not configured to use HTTPS or redirect to it by default.

References:

1. https://serversforhackers.com/c/redirect-http-to-https-nginx

CIS Controls:

Version 7

5.1 Establish Secure Configurations
Maintain documented, standard security configuration standards for all authorized operating systems and software.

14.4 Encrypt All Sensitive Information in Transit
Encrypt all sensitive information in transit.

4.1.2 Ensure a Trusted Certificate and Trust Chain is Installed (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

Certificates and their trust chains are used to establish the identity of your web server as trusted.

Rationale:

Certificates and their trust chains are used to establish the identity of your web server as legitimate. Certificate Authorities validate your web server identity and that you are the owner of that web server domain name. Without a certificate and full trust chain installed on your web server, modern browsers will flag your web server as untrusted.

Audit:

Run the below command to find the file location of your certificate. If there is no output to this command then you do not have one installed.

```
grep -ir ssl_certificate /etc/nginx/
```

The output of your command should look similar to the below output.
Web Server:

```
/etc/nginx/nginx.conf:    ssl_certificate /etc/nginx/cert.pem;
/etc/nginx/nginx.conf:    ssl_certificate_key /etc/nginx/nginx.key;
```

Open the file to the right of the ssl_certificate directive using the following command:

```
cat /etc/nginx/cert.pem
```

The output of your command should look similar to the below. If the output does not have the full certificate chain then the full chain should be installed.

```
-----BEGIN CERTIFICATE-----
Insert Your Web Servers Certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
```

```
Insert Your Certificate Authorities Intermediate Certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Insert Your Certificate Authorities Root Certificate
-----END CERTIFICATE-----
```

Remediation:

Use the following procedure to install a certificate and its signing certificate chain onto your web server, load balancer or proxy.
Step 1: Create the servers private key and a certificate signing request.
The following command will create your certificates private key with 2048-bit key strength. Optionally, this parameter may be changed to 4096 for greater security. It will also output your certificate signing request to the nginx.csr file in your present working directory.

```
openssl req -new -newkey rsa:2048 -keyout nginx.key -out nginx.csr
```

Enter the below information about your private key:

```
Country Name (2 letter code) [XX]: Your Country
State or Province Name (full name) []: Your State
Locality Name (eg, city) [Default City]: Your City
Organization Name (eg, company) [Default Company Ltd]: Your City
Organizational Unit Name (eg, section) []: Your Organizational Unit
Common Name (eg, your name or your server's hostname) []: Your servers DNS
name
Email Address []: Your e-mail address
```

Step 2: Obtain a signed certificate from your certificate signing authority.
Provide your chosen certificate authority with your certificate signing request. Follow your certificate authorities signing procedures in order to obtain a certificate and the certificate's trust chain. A full trust chain is typically delivered in .pem format.
Step 3: Install certificate and signing certificate chain on your web server.
Place your provided .pem file from your certificate authority into the directory of your choice. Locate your created key file from the command you used to generate your certificate signing request. Open your website configuration file and edit your encrypted listener to leverage the ssl_certificate and ssl_certificate_key directives for a web server as shown below. You should also inspect include files inside of your nginx.conf. This should be part of the server block.

```
server {
    listen 443 ssl http2;
    listen [::]:443 ssl http2;
    ssl_certificate /etc/nginx/cert.crt;
    ssl_certificate_key /etc/nginx/nginx.key;
```

```
    ...
    }
```

After editing this file, you must recycle nginx services for these changes to take effect. This can be done leveraging the following command:

```
sudo service nginx restart
```

Impact:

Without a full certificate chain installed on your server, your user's web browsers may flag your website in modern browsers as untrusted.

Default Value:

No certificate is installed by default.

References:

1. http://nginx.org/en/docs/http/configuring_https_servers.html#chains
2. https://www.digicert.com/csr-ssl-installation/nginx-openssl.htm
3. https://support.globalsign.com/customer/portal/articles/1290470-install-certificate---nginx

CIS Controls:

Version 6

14.2 Encrypt All Sensitive Information Over Less-trusted Networks

All communication of sensitive information over less-trusted networks should be encrypted. Whenever information flows over a network with a lower trust level, the information should be encrypted.

Version 7

14.4 Encrypt All Sensitive Information in Transit

Encrypt all sensitive information in transit.

4.1.3 Ensure Private Key Permissions are Restricted (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The server's private key should be protected to decrease the risk that traffic to your web server is decrypted. You should limit the permissions to the server's private key to the least privilege needed for the server to operate.

Rationale:

A servers private key file should be restricted to 400 permissions. This ensures that only the owner of the private key file can access it. This is the minimum necessary permissions for the server to operate. If the private key file is not protected then an unauthorized user with access to the server may be able to find the private key file and use it to decrypt traffic sent to your server.

Audit:

Check the permissions on the key file by running the following procedure:
Check to ensure the permissions on the key file are 400. This can be found by running the following command. You should replace /etc/nginx/nginx.key with the location of your key file before checking permissions.

```
ls -l /etc/nginx/nginx.key
```

The output of the permissions on this file should show the permissions as 400 or -r--------. If this is not the case then excessive permissions have been granted to this file.

Remediation:

Run the following command on your key file to ensure its permissions are set to 400. The file name /etc/nginx/nginx.key should be replaced with the location of your key file.

```
sudo chmod 400 /etc/nginx/nginx.key
```

Impact:

User permissions will be restricted on the private key so that only the owner of the private key may read it.

Default Value:

The default permissions on the server's private key are 644 or -rw-r--r--.

Notes:

Important Note: This recommendation should be applied to both the keys of your server certificate and the key of your client certificate if you are looking to mutually authenticate a proxy server.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

4.1.4 Ensure Only Modern TLS Protocols are Used (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

This recommendation ensures that only modern SSL/TLS protocols are enabled. Removing legacy TLS protocols and enabling emerging and stable ones ensure users are able to take advantage of emerging security capabilities and protects them from insecure legacy protocols.

The following tasks are associated with this recommendation:

1. Disable SSL 3.0
2. Disable TLS 1.0
3. Disable TLS 1.1
4. Enable TLS 1.2
5. Enable TLS 1.3

This should be done in NGINX for all client connections and upstream connections.

Rationale:

Why disable SSL 3.0: The POODLE Vulnerability allowed attackers to exploit SSL3.0 to obtain cleartext information by exploiting weaknesses in CBC in 2014. SSL3.0 is also no longer FIPS140-2 compliant.

Why disable TLS 1.0: TLS 1.0 was deprecated from use when PCI DSS Compliance mandated that it not be used for any applications processing credit card numbers in June 2018. TLS 1.0 does not make use of modern protections and almost all user agents that do not support TLS 1.2 or higher are no longer supported by their Vendor.

Why disable TLS 1.1: About 0.5 percent of HTTPS connections are still made using TLS 1.0 and 1.1. In lieu of this small population and the increased security associated with higher versions of TLS, this should be disabled. Modern browsers will begin to flag TLS 1.1 as deprecated in early 2019.

Why enable TLS 1.2: TLS 1.2 takes advantage of several security features including modern cipher suites, perfect forward security, and authenticated encryption.

Why enable TLS 1.3: TLS 1.3 removes insecure cipher suites that are present in TLS 1.2 and adds more secure cipher suites. It also adds protection by removing compression and renegotiation from the protocol.

Audit:

You can check the SSL/TLS protocols used by your server by issuing the below command. Run the following command to check the configured cipher suites on the server. If anything other than TLSv1.2 and TLSv1.3 is implemented or nothing appears then this recommendation is not implemented.

```
grep -ir ssl_protocol /etc/nginx
```

Note: Depending on your configuration you may see different results. The directive ssl_protocols should always be part of your server block. If your NGINX server is also a proxy or load balancer you should also check for the presence of the proxy_ssl_protocols directive as part of the location block of your nginx configuration. This ensures that your proxy follows a specific set of negotiation rules for encrypting traffic with your upstream server.

Remediation:

Run the following commands to change your ssl_protocols if they are already configured. This remediation advice assumes your nginx configuration file does not include server configuration outside of /etc/nginx/nginx.conf. You may have to also inspect the include files in your nginx.conf to ensure this is properly implemented.
Web Server:

```
sed -i "s/ssl_protocols[^;]*;/ssl_protocols TLSv1.2 TLSv1.3;/"
/etc/nginx/nginx.conf
```

Proxy:

```
sed -i "s/proxy_ssl_protocols[^;]*;/proxy_ssl_protocols TLSv1.2 TLSv1.3;/"
/etc/nginx/nginx.conf
```

If these are not already configured this can also be accomplished manually by opening up your web server or proxy server configuration file and manually adding the directives.
Web Server:

```
server {
    ssl_protocols TLSv1.2 TLSv1.3;
}
```

Proxy:

```
location / {
      proxy_pass cisecurity.org;
      proxy_ssl_protocols TLSv1.2 TLSv1.3;
    }
```

Impact:

Disabling certain TLS protocols may not allow legacy user agents to connect to your server. Disabling negotiation of specific protocols with your backend server may also limit your ability to connect with legacy servers. You should always consider if you need to support legacy user agents or servers when selecting your TLS/SSL protocols.

Default Value:

By default, NGINX does not specify the TLS protocol and accepts all TLS versions. ssl_protocols TLSv1.0 TLSv1.1 TLSv1.2 proxy_ssl_protocols TLSv1.0 TLSv1.1 TLSv1.2

References:

1. https://webkit.org/blog/8462/deprecation-of-legacy-tls-1-0-and-1-1-versions/
2. https://www.cloudflare.com/learning-resources/tls-1-3/

Notes:

Note: TLS configuration should always be set to your organizational policy. This recommendation is designed to meet best practices.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

4.1.5 Restrict Weak Ciphers From Being Used (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

Disable weak ciphers leveraging your company's policy or an industry best practice compliance profile. The ssl_ciphers directive may be used to configure the available ciphers on your web server and the proxy_ssl_ciphers directive may be used to configure the available ciphers for your proxy. The ssl_prefer_server_ciphers should also be used to ensure that the user agent respects the servers preferred cipher order and does not set their own. This should be enabled both for client communications and upstream communications if using a proxy or load balancer.

Rationale:

The use of strong ciphers is critical to maintaining strong encryption on your web server, load balancer or proxy. Weak Ciphers may compromise the security of your site or your users by allowing legacy user agents to connect to your site in a way that has been found to be vulnerable. You may also meet compliance concerns by ensuring that your upstream connections meet the same level of security if using a proxy or load balancer. The server should enforce the cipher preference on the server side to ensure to protect users from malicious actors on the client side.

The ssl_cipher directive should be used on all server blocks in order to ensure that clients connect to the server securely. If you are using a proxy or load balancer you should also use the proxy_ssl_ciphers directive to ensure that your upstream connections are negotiated using secure ciphers.

Audit:

Use the following procedure to check the ssl_cipher and proxy_ssl_cipher directive configuration for configuration compliance that meets your company's policy.

```
grep -ir ssl_ciphers /etc/nginx/
grep -ir ssl_prefer_server_ciphers /etc/nginx
```

This output will show that servers configured ciphers and its cipher preference policy. If you have multiple server blocks or proxy passes then you should ensure that this appears for each.
The output should also contain the directive ssl_prefer_server_ciphers as shown below to ensure that these ciphers cannot be changed in preference on the client side.

```
ssl_prefer_server_ciphers on;
```

In your environment, you may have to check all include files in your nginx configuration or the nginx configuration itself manually. The server ciphers may be located as part of the server block and the proxy ciphers may be located as part of the location block for your upstream traffic.
Openssl may also be used to check compatible ciphers following the procedure found at OWASP at the following link:
https://www.owasp.org/index.php/Testing_for_SSL-TLS_%28OWASP-CM-001%29

Remediation:

The following procedures may be used to implement several industry standard cipher profiles if you have an existing profile defined. These profiles may be modified to meet the requirements defined in your company's policy. This procedure assumes that all server blocks will be in /etc/nginx/nginx.conf and not inside any included files in the configuration.
You will want to ensure that the ssl_cipher directive is set as part of your server block and that the proxy_ssl_ciphers directive is set as part of the location block for your upstream server.
This should look similar to the below examples:
Server block configuration for client connectivity to web server, proxy or load balancer:

```
server {
  ssl_ciphers ALL:!EXP:!NULL:!ADH:!LOW:!SSLv2:!SSLv3:!MD5:!RC4;
}
```

Proxy or Load Balancer Configuration for defined upstream negotiation:

```
location / {
  proxy_pass https://cisecurity.org;
  proxy_ssl_ciphers ALL:!EXP:!NULL:!ADH:!LOW:!SSLv2:!SSLv3:!MD5:!RC4;
}
```

The below procedure assumes the default configuration profile. If you do not yet have the ssl_ciphers or proxy_ssl_ciphers defined, add the directives to your proxy or web server configuration profile and run the below commands to configure them to your selected

profile.

FIPS 140-2 Compliant Proxy:

```
sed -i "s/proxy_ssl_ciphers[^;]*;/proxy_ssl_ciphers
ALL:!EXP:!NULL:!ADH:!LOW:!SSLv2:!SSLv3:!MD5:!RC4;/" /etc/nginx/nginx.conf
```

Or if ssl_prefer_server_ciphers is not already enabled

```
sed -i "s/proxy_ssl_ciphers[^;]*;/proxy_ssl_ciphers
ALL:!EXP:!NULL:!ADH:!LOW:!SSLv2:!SSLv3:!MD5:!RC4;\nssl_prefer_server_ciphers
on;/" /etc/nginx/nginx.conf
```

FIPS 140-2 Compliant Web Server:

```
sed -i "s/ssl_ciphers[^;]*;/ssl_ciphers
ALL:!EXP:!NULL:!ADH:!LOW:!SSLv2:!SSLv3:!MD5:!RC4;/" /etc/nginx/nginx.conf
```

Or if ssl_prefer_server_ciphers is not already enabled

```
sed -i "s/ssl_ciphers[^;]*;/ssl_ciphers
ALL:!EXP:!NULL:!ADH:!LOW:!SSLv2:!SSLv3:!MD5:!RC4;\nssl_prefer_server_ciphers
on;
/" /etc/nginx/nginx.conf
```

No Weak Ciphers SSLLABS Proxy Configuration

```
sed -i "s/proxy_ssl_ciphers[^;]*;/proxy_ssl_ciphers
HIGH:!aNULL:!CAMELLIA:!SHA:!RSA;/" /etc/nginx/nginx.conf
```

Or if ssl_prefer_server_ciphers is not already enabled

```
sed -i "s/proxy_ssl_ciphers[^;]*;/proxy_ssl_ciphers
HIGH:!aNULL:!CAMELLIA:!SHA:!RSA;\nssl_prefer_server_ciphers on;/"
/etc/nginx/nginx.conf
```

No Weak Ciphers SSLLABS Web Server Configuration:

```
sed -i "s/ssl ciphers[^;]*;/ssl ciphers HIGH:!aNULL:!CAMELLIA:!SHA:!RSA;/"
/etc/nginx/nginx.conf

#Or if  ssl_prefer_server_ciphers is not already enabled

sed -i "s/ssl_ciphers[^;]*;/ssl_ciphers
HIGH:!aNULL:!CAMELLIA:!SHA:!RSA;\nssl_prefer_server_ciphers on;/"
/etc/nginx/nginx.conf
```

Mozilla Modern Profile Proxy:

```
sed -i "s/proxy_ssl_ciphers[^;]*;/proxy_ssl_ciphers 'ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-
```

```
CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-
SHA256:ECDHE-RSA-AES128-SHA256';/" /etc/nginx/nginx.conf
```

Or if ssl_prefer_server_ciphers is not already enabled

```
sed -i "s/proxy_ssl_ciphers[^;]*;/proxy_ssl_ciphers 'ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-
CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-
SHA256:ECDHE-RSA-AES128-SHA256';\nssl_prefer_server_ciphers on;/"
/etc/nginx/nginx.conf
```

Mozilla Modern Profile Web Server:

```
sed -i "s/ssl_ciphers[^;]*;/ssl_ciphers 'ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-
RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-
POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-
ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-
RSA-AES128-SHA256';/" /etc/nginx/nginx.conf
```

Or if ssl_prefer_server_ciphers is not already enabled

```
sed -i "s/ssl_ciphers[^;]*;/ssl_ciphers 'ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-
RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-
POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-
ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-
RSA-AES128-SHA256';\nssl_prefer_server_ciphers on;/" /etc/nginx/nginx.conf
```

For changes to take effect you must recycle nginx.

```
service nginx restart
``
```

Impact:

Strong cipher configurations may not allow legacy user agents or user agents with weak configurations to connect to your site. If your server must also pass to a legacy upstream server this may prevent it from being able to negotiate a cipher upstream.

Default Value:

These directives are not specified by default and set to the default of HIGH:!aNULL:!MD5.

References:

1. CIS Apache HTTP Server Benchmark
2. https://ssllabs.com
3. https://mozilla.github.io/server-side-tls/ssl-config-generator/

4. http://nginx.org/en/docs/http/ngx_http_ssl_module.html
5. https://www.owasp.org/index.php/Testing_for_SSL-TLS_%28OWASP-CM-001%29
6. https://www.acunetix.com/blog/articles/tls-vulnerabilities-attacks-final-part/
7. https://www.gracefulsecurity.com/tls-ssl-vulnerabilities/

Notes:

Note: TLS configuration should always be set to your organizational policy. This recommendation is designed to meet best practices and offers several profiles that your organization may align to.

CIS Controls:

Version 7

5.1 Establish Secure Configurations
Maintain documented, standard security configuration standards for all authorized operating systems and software.

14.4 Encrypt All Sensitive Information in Transit
Encrypt all sensitive information in transit.

4.1.6 Ensure Custom Diffie-Hellman Parameters are Used (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The CIS NGINX Benchmark recommends using custom Diffie-Hellman (DH) key exchange parameters.

Generate DHE (Ephemeral Diffie-Hellman) parameters with at least 2048 bits.

Rationale:

Backward compatible Perfect Forward Secrecy (PFS) ciphers (e.g. DHE-RSA-AES128-SHA256) should use strong and unique parameters.

By default, NGINX will generate 1024-bit RSA keys for PFS ciphers.

Audit:

Ensure the option `ssl_dhparam` is explicitly provided.

```
grep ssl_dhparam /etc/nginx/nginx.conf
```

Remediation:

Generate strong DHE (Ephemeral Diffie-Hellman) parameters.

```
mkdir /etc/nginx/ssl
openssl dhparam -out /etc/nginx/ssl/dhparam.pem 2048
chmod 400 /etc/nginx/ssl/dhparam.pem
```

Adapt server configuration.

```
http {
    server {
        ssl_dhparam /etc/nginx/ssl/dhparam.pem;
    }
}
```

References:

1. https://weakdh.org/sysadmin.html

CIS Controls:

Version 6

14.2 Encrypt All Sensitive Information Over Less-trusted Networks

All communication of sensitive information over less-trusted networks should be encrypted. Whenever information flows over a network with a lower trust level, the information should be encrypted.

Version 7

14.4 Encrypt All Sensitive Information in Transit

Encrypt all sensitive information in transit.

4.1.7 Ensure Online Certificate Status Protocol (OCSP) Stapling is Enabled (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

OCSP Protocol is the protocol that allows a user's browser or another user agent to check to ensure that the certificate it is seeing is not revoked. OSCP Stapling allows a server to cache a status of the server's certificate is revoked. This polls the Certificate Authority's (CA) OSCP server at regular intervals to ensure that this is continuously kept up to date and allows users browser or other user agents to evaluate if a certificate has been revoked. OCSP stapling helps improve performance while improving security.

Rationale:

OCSP allows a user's browser or other user agent to check that your certificate has not been revoked. This protects your users from accessing a website where a private key is believed to be compromised. If a private key is compromised an attacker may be able to obtain unauthorized access to the encrypted data transmitted by a user. OCSP Stapling ensures that your server presents this information to the user's browser in a way that best meets the performance and security needs of your website.

Audit:

Run the following command to verify if OCSP stapling is enabled.

```
grep -ir ssl_stapling /etc/nginx
```

If the output does not contain your proxy or web server configuration file with the below two directives, this recommendation is not implemented.

```
ssl_stapling on;
ssl_stapling_verify on;
```

Remediation:

Follow this procedure to enable OCSP validation:
Step 1: Ensure your NGINX server has access to your Certificate Authority's (CA's) OCSP server.
Your CA's OCSP server may be found on your CA's website and will be vary depending on your CA vendor. Issue the following command in order to check your connectivity to their site.

```
curl -I "insert certificate authority ocsp server here"
```

If you get a 200 code response your server has access.
Step 2: Enable OCSP on nginx.
Implement the ssl_stapling and ssl_stapling_verify directives in order to ensure that your server. The directive ssl_stapling enables ocsp stapling and the directive ssl_stapling_verify enables verification of the OCSP responses on nginx.

```
server {
  ssl_stapling on;
  ssl_stapling_verify on;
}
```

Impact:

OCSP Stapling, while a step forward from the older certificate revocation list model does share similar risks. Between the time that a certificate is revoked and the point where a new signed OCSP profile is requested if a server's certificate has been revoked a user agent may not be informed.

Default Value:

OCSP Stapling is not enabled by default.

References:

1. https://www.digicert.com/ssl-support/nginx-enable-ocsp-stapling-on-server.htm

CIS Controls:

Version 7

5.1 Establish Secure Configurations
Maintain documented, standard security configuration standards for all authorized operating systems and software.

4.1.8 Ensure HTTP Strict Transport Security (HSTS) is Enabled (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

HSTS headers instruct a user agent on how to communicate with a web server. HSTS Headers ensure that the strict transport security policy's built into browsers and other user agents are informed only to communicate over HTTPS. HSTS with long validity periods should be used to most effectively secure your user population.

Strict-Transport-Security should have a long max-age, which is above 18 weeks and recommended to be at least one year in length. This ensures that the browser remembers your website should only be accessible via HTTPS for this amount of time.

Rationale:

The HSTS Header is an important part of browser security. HSTS Headers help protect a server's users from accessing the server over unencrypted protocols. This header helps to prevent HTTP downgrade attacks.

Audit:

Issue the below command to check for HSTS headers on your website.

```
grep -ir Strict-Transport-Security /etc/nginx
```

If your output does not include the following directive associated with your server configuration file, then this recommendation is not implemented. This header should also include the max-age directive with 15768000 seconds or longer configured.

```
add_header Strict-Transport-Security "max-age=15768000;";
```

Remediation:

Ensure that the below snippet of code can be found in your server configuration for your proxy or web server. This will ensure that the HSTS header is set with a validity period of 6 months, or 15768000 seconds.

```
server {
  add_header Strict-Transport-Security "max-age=15768000;";
}
```

Impact:

Your site will be forced over https. If your website is not configured for https this may cause impact to the accessibility of your site.

Default Value:

HSTS Headers are not set by default.

References:

1. https://www.globalsign.com/en/blog/what-is-hsts-and-how-do-i-use-it/
2. https://mozilla.github.io/server-side-tls/ssl-config-generator/
3. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security#Preloading_Strict_Transport_Security
4. https://hstspreload.org
5. https://tools.ietf.org/html/rfc6797

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

14.4 Encrypt All Sensitive Information in Transit

Encrypt all sensitive information in transit.

4.1.9 Ensure HTTP Public Key Pinning Extension is Enabled (Not Scored)

Profile Applicability:

- Level 2 - Webserver
- Level 2 - Proxy
- Level 2 - Loadbalancer

Description:

HTTP Public Key Pinning allows the site to specify exactly what certificate the browser or another user agent should accept. This helps prevent a user agent from falling victim to a forged certificate.

Rationale:

HTTP Public Key Pinning is a defense in depth strategy, which allows the server to instruct the user's browser or another user agent to only allow specific certificates when accessing the site. This assists in preventing man in the middle attacks. HTTP Public Key Pinning allows for the certificate rotation to be scheduled using backup fingerprints to ensure that user agent has both certificates stored.

Audit:

Ensure that certificate pinning is enabled by running the below command:

```
grep -ir Public-Key-Pins /etc/nginx
```

If the output does not return anything then the header is not implemented.

Remediation:

Find the fingerprint of your certificate by referencing the fingerprint section of your certificate details. Take down the SHA256 fingerprint in this section as well as that of a backup certificate or the next scheduled certificate for the website.
Insert your SHA256 fingerprint along with the below header to your server configuration.

```
add_header Public-Key-Pins 'pin-
sha256="base64+primary==InsertPrimaryCertificateSHA256FingerPrintHere"; pin-
sha256="base64+backup==InsertBackupCertificateSHA256FingerPrintHere"; max-
age=5184000;
```

Impact:

HTTP Certificate Pinning should be considered carefully during your renewal processes for certificates. If this is failed to be updated as part of a certificate renewal then browsers will not trust the certificate.

Default Value:

HTTP Certificate Pinning is not enabled by default.

References:

1. https://developer.mozilla.org/en-US/docs/Web/HTTP/Public_Key_Pinning

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

4.1.10 Ensure Upstream Server Traffic is Authenticated with a Client Certificate (Scored)

Profile Applicability:

- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

Mutual Authentication allows your upstream server to authenticate your identity.

Rationale:

Client certificate validation allows the upstream server to authenticate the identity of the client connecting to it. This allows you to establish yourself as a trusted proxy server. This assists in the establishment of mutual authentication between the client and the server.

Audit:

In order to validate this recommendation, you may either validate that the below configuration is in the location block of your nginx configuration that is sending traffic to an upstream location. The command below may be helpful in determining if this is set up at all.

```
grep -ir proxy_ssl_certificate /etc/nginx
```

You should see output similar to the below. You may need to manually ensure that this is configured properly by investigating your location block for the output as well.

```
proxy_ssl_certificate /etc/nginx/ssl/nginx.pem;
proxy_ssl_certificate_key /etc/nginx/ssl/nginx.key;
```

Remediation:

In order to implement this recommendation, you must create a client certificate to be authenticated against and have it signed. Once you have a signed certificate place the certificate in a location of your choice. In the below example we will use /etc/nginx/ssl/cert.pem. Implement the below configuration as part of the location block as shown below.

```
proxy_ssl_certificate /etc/nginx/ssl/nginx.pem;
proxy_ssl_certificate_key /etc/nginx/ssl/nginx.key;
```

Default Value:

This is not authenticated by default.

References:

1. https://docs.nginx.com/nginx/admin-guide/security-controls/securing-http-traffic-upstream/
2. http://www.staticshin.com/programming/proxy-ssl-cert-in-nginx.html
3. http://nginx.org/en/docs/http/ngx_http_proxy_module.html#proxy_ssl_certificate

Notes:

Your upstream server must also be configured to verify the client certificate. If your upstream web server is a nginx web server then you may accomplish this by adding the client certificate into a file referenced by the directive ssl_client_certificate. This should be part of your server block. An example can be found below.

```
ssl_client_certificate      /etc/nginx/ssl/ca.cert;

ssl_verify_client          on;
```

CIS Controls:

Version 6

1.6 Use Of Client Certificates For System Authentication
Use client certificates to validate and authenticate systems prior to connecting to the private network.

Version 7

1.8 Utilize Client Certificates to Authenticate Hardware Assets
Use client certificates to authenticate hardware assets connecting to the organization's trusted network.

4.1.11 Ensure Upstream Traffic Server Certificate is Trusted (Not Scored)

Profile Applicability:

- Level 2 - Proxy
- Level 2 - Loadbalancer

Description:

Your nginx server should always be configured to validate the identity of the upstream server that it is sending information to.

Rationale:

By configuring nginx to validate the identity of the upstream server, you help mitigate the risk of a man in the middle attack occurring against your server.

Audit:

To audit this recommendation, run the below procedure.
Step 1: Check to ensure your nginx proxy or load balancer is set up to validate server identity.
You should check for the presence of the below directives as part of the location block you are using to send traffic to your upstream server. The below two commands should help you to identify if this is implemented, however, you may also want to manually check through include files as well that can be found in your nginx configuration. As part of this configuration check, you should also ensure that the proxy_ssl_verify directive is set to on.

```
grep -ir proxy_ssl_trusted_certificate /etc/nginx
grep -ir proxy_ssl_verify /etc/nginx
```

The output and directives you should look for are below:

```
proxy_ssl_trusted_certificate /etc/nginx/trusted_ca_cert.crt;
proxy_ssl_verify        on;
```

Step 2: Check to ensure that certificate trust chains for upstream servers are installed properly
You should check the certificates installed in the location referenced by the proxy_ssl_trusted_certificate directive to ensure that the certificate you reference is valid for your upstream server.

Remediation:

To ensure this recommendation is implemented you must obtain the full certificate chain of the upstream server in .pem format. You should then reference that file in the location block as part of the
proxy_ssl_trusted_certificate directive. Implement the proxy_ssl_trusted_certificate directive and the proxy_ssl_verify directive as shown below as part of the location block you are using to send traffic to your upstream server.

```
proxy_ssl_trusted_certificate /etc/nginx/trusted_ca_cert.crt;
proxy_ssl_verify        on;
```

Impact:

You should check to ensure that this configuration is updated with the correct certificates when your certificate is rotated. If this is not done properly you may experience an outage.

Default Value:

This is not setup by default.

References:

1. https://docs.nginx.com/nginx/admin-guide/security-controls/securing-http-traffic-upstream/
2. http://nginx.org/en/docs/http/ngx_http_proxy_module.html#proxy_ssl_trusted_certificate

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

4.1.12 Ensure your Domain is Preloaded (Not Scored)

Profile Applicability:

- Level 2 - Webserver
- Level 2 - Proxy
- Level 2 - Loadbalancer

Description:

Preloading your domain hard codes your domain as only being accessible through https into browsers.

Rationale:

Preloading your domain helps protect your businesses digital users from accessing your server over unencrypted protocols. This helps to prevent HTTP downgrade attacks and increase trust in your business.

Audit:

To ensure that you are preloaded visit https://hstspreload.org/ to validate that your top-level domain is preloaded. Type in your top-level domain into the site to see if you are already preloaded.

Remediation:

In order to successfully preload your website you must meet the below criteria:

1. Serve a valid certificate.

This may be accomplished by following recommendation 4.1.3.

2. Redirect from HTTP to HTTPS if using port 80

This may be accomplished by following recommendation 4.1.1.

3. Ensure all subdomains support HTTPS only.

This will require you to configure all subdomains for HTTPS only. For example, a subdomain of cissecurity.org is workbench.cissecurity.org and would need to be configured for https only.

4. Ensure you have an HSTS header configured on your base domain as shown below for nginx.

If your base domain is nginx then you may accomplish this with several modifications from the HSTS recommendation. Change your header to include the preload directive, the includesubdomains directive and make your max-length a year or longer. The header should be modified similar to the below snippet.

```
add_header Strict-Transport-Security "Strict-Transport-Security: max-age=31536000; includeSubDomains; preload";
```

After you have met these requirements add your site to the list by following the instructions at https://hstspreload.org/.

Impact:

Preloading should only be done with careful consideration!

Your website and all of its subdomains will be forced over https. If your website or any of its subdomains are not able to support preloading then you should not preload your site.

Preloading should be opt-in only and if done may impact more sites than the nginx instance you are working on. Removing preloading can be "slow and painful," and should only be done with careful consideration according to https://hstspreload.org

Default Value:

Your website is not preloaded by default.

References:

1. https://hstspreload.org/

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

4.1.13 Ensure Session Resumption Is Disabled To Enable Perfect Forward Security (Not Scored)

Profile Applicability:

- Level 2 - Webserver
- Level 2 - Proxy
- Level 2 - Loadbalancer

Description:

Perfect forward secrecy is an encryption mechanism that enables past session keys to not be compromised even if the servers private key is compromised. If an attacker recorded all traffic to a server and stored it and then obtained the private key without perfect forward secrecy all communications would be compromised. With perfect forward secrecy, session keys are generated using Diffie-Hellman for every session a user initiates, which isolates session compromise to only that communication session.

Rationale:

Allowing session resumption expands the surface area for an attacker to compromise past sessions and your communication with a server if they are able to compromise the session.

Audit:

Run the following command in order to check to see if the ssl_session_tickets directive is turned off. You should always investigate your nginx configuration file for included file locations outside of /etc/nginx to ensure you are properly investigating each server block for the presence of the
ssl_session_tickets directive being turned off.

```
grep -ir ssl_session_tickets /etc/nginx
```

The output of this command should contain the following output:

```
ssl_session_tickets off;
```

Remediation:

Ensure that the ssl_session_tickets directive is turned off as part of any server block in your nginx configuration.

```
ssl_session_tickets off;
```

Impact:

Implementing this recommendation will cause a minor performance impact.

Default Value:

Perfect forward security is not enabled by default.

References:

1. https://www.imperialviolet.org/2013/06/27/botchingpfs.html
2. https://scotthelme.co.uk/perfect-forward-secrecy/

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

4.1.14 Ensure HTTP/2.0 is Used (Not Scored)

Profile Applicability:

- Level 2 - Webserver

Description:

Leveraging HTTP/2.0 allows users to take advantage of an optimized and more secure version of the HTTP protocol.

Rationale:

HTTP/2.0 introduces not only performance benefits through full multiplexing, but several increased security benefits. By leveraging HTTP/2.0 we help protect compatible user agents. HTTP/2.0 comes with improved cipher suites requirements and blacklists. It also disables session renegotiation and TLS compression. This helps protect against vulnerabilities like CRIME and ensures we have stronger encryption.

Audit:

Check to ensure that listening ports on the web server leverage HTTP/2.0. This can be done by running the below command.

```
grep -ir http2 /etc/nginx
```

If there is no output, the output does not cover all running ports on the server that are not redirecting to another port or that output is commented out then this recommendation is not implemented.

Remediation:

Open the nginx server configuration file and ensure that all listening ports are configured with http2 similar to that of the below example:

```
server {
    listen 443 ssl http2;
}
```

Impact:

Legacy user agents may not be able to connect to a server using HTTP/2.0

Default Value:

By default, HTTP/1.1 is used.

References:

1. https://mozilla.github.io/server-side-tls/ssl-config-generator/
2. http://http2.github.io/http2-spec/

Notes:

NOTE: HTTP/2.0 is not supported for proxies at the time of the writing of this recommendation.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

5 Request Filtering and Restrictions

5.1 Access Control

5.1.1 Ensure Allow and Deny Filters are Used to Limit Access to Specific IP Addresses (Not Scored)

Profile Applicability:

- Level 2 - Webserver
- Level 2 - Proxy
- Level 2 - Loadbalancer

Description:

IP Based restrictions act as a defense in depth mechanism. They allow you to whitelist legitimate paths to your applications and explicitly deny IP addresses you believe to be malicious.

Rationale:

IP restrictions help you to only allow traffic based on the concept of least privilege. You may specify vlans, countries, or specific servers that may be allowed or denied on your site. Its recommended that you implicitly deny all traffic and only allow those with a legitimate use case to access your website if choosing to take this approach. This allows you to limit the surface area an attack may come from.

Audit:

Step 1: Open your nginx config file and any fines that are appended in an include statement.
Step 2: Check the location context of your server block for the allow and deny directives. The output should look similar to the below snippet and may be expressed in CIDR notation, or single addresses.

```
location / {
    allow 10.1.1.1;
    deny all;
}
```

Step 3: Ensure that the allowed IP addresses are not too permissive for your use case. For example, In the above snippet, 10.1.1.1 may be a load balancer connecting to your proxy and you would only like user traffic to come from the load balancer.

Remediation:

To ensure this recommendation is implemented you should compile a list of network ranges or IP addresses that you would want to access your web server or proxy. Then ensure that you add these ranges with the allow directive. The deny directive is recommended to always be included with all IP addresses implicitly denied.

```
location / {
    allow 10.1.1.1;
    deny all;
}
```

Impact:

This may block a legitimate IP address from being able to access your web server, load balancer or proxy if it is not configured in the allow list.

Default Value:

This is not configured by default.

References:

1. https://help.dreamhost.com/hc/en-us/articles/222784068-The-most-important-steps-to-take-to-make-an-nginx-server-more-secure
2. http://nginx.org/en/docs/http/ngx_http_access_module.html

Notes:

Note: If you do not want to restrict this to a specific network range this recommendation may not fit your use case.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

9.5 Implement Application Firewalls

Place application firewalls in front of any critical servers to verify and validate the traffic going to the server. Any unauthorized traffic should be blocked and logged.

5.1.2 Ensure only Whitelisted HTTP Methods are Allowed (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

HTTP methods (or verbs) allow for different actions to be requested from the web server at a specified path.

Rationale:

Most websites only require the methods GET, POST and HEAD to function correctly. Web applications may also require other verbs (e.g. DELETE). In order to narrow vectors of attack, it is recommended to only enable the required verbs.

Audit:

Use a tool like curl to send a request with a method which should not be supported (e.g. DELETE) and compare the output to a supported method (e.g. GET).

```
# curl -X DELETE http://localhost/index.html
curl: (52) Empty reply from server
# curl -X GET http://localhost/index.html
```

Remediation:

To remove unneeded methods and only allow required HTTP methods add the following into a server or location block in your nginx.conf. The below snippet assumes only the methods GET, HEAD and POST are required for an application. The reason for 444 as a response is because it contains no information and can help mitigate automated attacks.

```
if ($request_method !~ ^(GET|HEAD|POST)$) {
   return 444;
}
```

Impact:

This will disable HTTP methods not included in your if statement.

Default Value:

All methods are allowed.

References:

1. https://www.acunetix.com/blog/articles/nginx-server-security-hardening-configuration-1/

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

5.2 Request Limits

5.2.1 Ensure Timeout Values for Reading Client Request Header and Client Request Body are Set (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The client_header_timeout and client_body_timeout directives define the time that the server will wait in between for the header or body to be sent from the client. If the client does not send the entire header in this pre-defined timeframe then the server will send back a 408 request timeout error.

Rationale:

Setting the client request header and body timeouts help your server mitigate possible denial of service attacks. By timing out this request the server is able to free up resources that may be waiting for the body or header. Setting this timeout helps to mitigate the slow HTTP denial of service attack, which slowly sends bodies and headers in an attempt to consume server resources.

Audit:

To check the current setting for the client_body_timeout and client_header_timeout directives issue the below command. You should also manually check your nginx configuration for include statements that may be located outside of the /etc/nginx directory. If this is not present then the value is set at the default.

```
grep -ir timeout /etc/nginx
```

The output of the command should contain the following output.

```
client_body_timeout   10;
client_header_timeout 10;
```

Remediation:

Find within the HTTP or server block of your nginx configuration and add the client_body_timeout and client_header_timeout directives set to 10 seconds for both timeouts in this block.

```
client_body_timeout   10;
client_header_timeout 10;
```

Default Value:

client_header_timeout 60; client_body_timeout 60;

References:

1. https://www.owasp.org/index.php/SCG_WS_nginx
2. https://blog.qualys.com/securitylabs/2011/11/02/how-to-protect-against-slow-http-attacks

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

5.2.2 Ensure a Limit for Reading Client Request Body is Set (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The client_max_body_size directive allows you to set the size of the request body that is allowed to read a client request. This defines the number of bytes that are allowed in a request and is equivalent to the Content-Length request header field.

Rationale:

Limiting the size of the request body helps prevent unexpectedly long or large client requests from being passed to an application. This is a defense in depth mechanism to ensure to help protect against a vulnerable application. This value should be set to a value low enough to protect the application but high enough not to interfere with functionality and block legitimate request bodies. This can help prevent buffer overflow attacks.

Audit:

To check the current setting for the client_max_body_size directive issue the below command. You should also manually check your nginx configuration for include statements that may be located outside of the /etc/nginx directory. If this is not present then the value is set at the default.

```
grep -ir client_max_body_size /etc/nginx
```

Remediation:

Find within the HTTP or server block of your nginx configuration and add the client_max_body_size set to 100K in this block. This may be different based on your applications needs.

```
client_max_body_size 100K
```

Impact:

If this value is not customized correctly for your application's needs then you may block legitimate client requests.

Default Value:

client_max_body_size 1m;

References:

1. https://www.cyberciti.biz/tips/linux-unix-bsd-nginx-webserver-security.html
2. http://nginx.org/en/docs/http/ngx_http_core_module.html#client_body_temp_path
3. https://www.acunetix.com/blog/articles/nginx-server-security-hardening-configuration-1/
4. https://www.tecmint.com/nginx-web-server-security-hardening-and-performance-tips/

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

5.2.3 Ensure Limits for Reading Large Client Request Headers are Set (Scored)

Profile Applicability:

- Level 1 - Webserver
- Level 1 - Proxy
- Level 1 - Loadbalancer

Description:

The large_client_header_buffers defines the number and size of buffers used within the URI. A request cannot exceed the size of this buffer when this directive is configured.

Rationale:

The large_client_header_buffers directive allows you to set the maximum size of a given URI. This may assist in preventing buffer overflow attacks that leverage long URI query parameters.

Audit:

Ensure that the large_client_header_buffers directive is set and defined. It is recommended that the number of buffers is set to two and the length be set to 1K. This may not always be a good fit for your application and may need to vary based on your applications needs. Run the following command to check the configuration of this directive.

```
grep -ir large_client_header_buffers /etc/nginx/
```

The output should contain output similar to the below.

```
large_client_header_buffers 2 1k
```

Remediation:

Open your nginx.conf file and locate your server or HTTP blocks. This may be added to the HTTP block for all configurations or server for more specific configurations to meet your needs. Add the below line to implement this recommendation.

```
large_client_header_buffers 2 1k
```

Impact:

If this directive is not set correctly for your applications needs, your users may receive a 414 Request-URI Too Large error.

Default Value:

large_client_header_buffers 4 8k;

References:

1. https://www.cyberciti.biz/tips/linux-unix-bsd-nginx-webserver-security.html
2. https://www.owasp.org/index.php/Denial_of_Service_Cheat_Sheet
3. http://nginx.org/en/docs/http/ngx_http_core_module.html#large_client_header_buffers

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

5.2.4 Ensure Number of Connections are Limited by IP Address (Not Scored)

Profile Applicability:

- Level 2 - Webserver
- Level 2 - Proxy
- Level 2 - Loadbalancer

Description:

Limiting the connections from a single IP address to your server helps to mitigate denial of service attacks.

Rationale:

Limiting the number of simultaneous connections is an effective way to prevent slow denial of service attacks. These attacks try to use as many server resources as possible. Limiting the number of connections that are allowed by a single IP address is an effective way to minimize the number of resources that an attacker may take up in the event of an attack. This is also defense in depth mechanism that can be used to help prevent brute force attacks on a login page. You should set this value to a value that meets your organizational policies.

Audit:

To check for this recommendation check the HTTP block and server block for the below configuration. You may also need to check files included in include statements within your nginx config.

```
http {
  limit_conn_zone $binary_remote_addr zone=limitperip:10m;
  server {
    limit_conn limitperip 10;
  }
}
```

Remediation:

In order to meet this recommendation implement the below directives under the HTTP and server blocks of your nginx configuration or any include files. The below configuration creates a memory zone of 10 megabytes called limitperip. It will limit the number of

connections per IP address to 10 simultaneous connections. The number of simultaneous connections may be different depending on your organization's policies and use cases.

```
http {
  limit_conn_zone $binary_remote_addr zone=limitperip:10m;
  server {
    limit_conn limitperip 10;
  }
}
```

Impact:

Users of your system that are behind a corporate web proxy using network address translation or a proxy service such as tor may have an increased chance of being blocked due to this configuration. This is because multiple users in these scenarios come from the same IP address. You should always consider your user base when performing a connection limit.

Default Value:

This value is not set by default.

References:

1. https://www.nginx.com/resources/library/complete-nginx-cookbook/
2. http://nginx.org/en/docs/http/ngx_http_limit_conn_module.html
3. https://scotthelme.co.uk/mitigating-http-get-dos-attack/

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

5.2.5 Ensure Rate Limits by IP Addresses are Set (Not Scored)

Profile Applicability:

- Level 2 - Webserver
- Level 2 - Proxy
- Level 2 - Loadbalancer

Description:

Rate limiting allows you to protect yourself from denial of service attacks by limiting the number of requests a user may make in a given period of time.

Rationale:

Rate limiting allows you to mitigate potential denial of service attacks as a defense in depth mechanism. It allows you to set the policy for how frequently an IP address may access your server.

Audit:

To check for this recommendation check the HTTP block and server block for the below configuration. You may also need to check files included in include statements within your nginx config. This configuration should be customized to your applications needs and your organizational policy.

```
http {
  limit_req_zone $binary_remote_addr zone=ratelimit:10m rate=5r/s;
  server {
    location / {
      limit_req zone=ratelimit burst=10 nodelay;
    }
  }
}
```

Remediation:

In order to meet this recommendation implement the below directives under the HTTP and server blocks of your nginx configuration or any include files. The below configuration creates a memory zone of 10 megabytes called "ratelimit" and sets the number of requests per second that can be sent by any given IP address to 5 requests per second. Further, this configuration sets a burst of 10 to ensure that requests may come more frequently and sets no delay to ensure that the bursting may be all at once and not queue. The number of

simultaneous connections may be different depending on your applications use case. You should consult the nginx documentation to determine what is right for you.

```
http {
  limit_req_zone $binary_remote_addr zone=ratelimit:10m rate=5r/s;
  server {
    location / {
      limit_req zone=ratelimit burst=10 nodelay;
    }
  }
}
```

Impact:

If you serve a high traffic API, this may prevent users from being able to call your website. You may also limit users behind a corporate web proxy or a proxy service such as tor if they use your website heavily.

Default Value:

This is not set by default.

References:

1. https://scotthelme.co.uk/mitigating-http-get-dos-attack/
2. https://www.nginx.com/blog/rate-limiting-nginx/

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

5.3 Browser Security

5.3.1 Ensure X-Frame-Options Header is Configured and Enabled (Scored)

Profile Applicability:

- Level 1 - Webserver

Description:

The X-Frame-Options allow you to decide if your website is allowed to be embedded within another site. It allows you to authorize each site that is allowed on an individual level.

Rationale:

The X-Frame-Options Header allows you to mitigate the risk of clickjacking attacks. This can be set to allow only specific websites or no sites at all to allow other sites to embed your website as an object within their own.

Audit:

Run the following procedure to ensure this header is implemented on your site.

```
grep -ir X-Frame-Options /etc/nginx
```

The output should look similar to the below, but customized to your use case. If there is no output from this command then this recommendation is not implemented.

```
X-Frame-Options: "sameorigin";
```

Remediation:

To resolve this add the below output to your server blocks in your nginx configuration. The policy should be configured to meet your organization's needs.

```
X-Frame-Options: "sameorigin";
```

Impact:

Implementing this may block legitimate partner sites from embedding your website if this header is not configured properly.

Default Value:

This is not configured by default.

References:

1. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
2. https://scotthelme.co.uk/hardening-your-http-response-headers/
3. https://www.veracode.com/blog/2014/03/guidelines-for-setting-security-headers
4. https://www.owasp.org/index.php/OWASP_Secure_Headers_Project

Notes:

Important Note: The configuration in this recommendation recommends that this header is set to the same origin, however, this does not mean that if it is not set so that this is done incorrectly. This header may also be configured to allow from specific origins or deny from all origins.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

5.3.2 Ensure X-Content-Type-Options Header is Configured and Enabled (Scored)

Profile Applicability:

- Level 1 - Webserver

Description:

The X-Content-Type-Options Header forces supported user agents to check an HTTP response's content type header what is expected from the destination of the request.

Rationale:

Implementing the X-Content-Type-Options header with the "nosniff" directive helps to prevent drive-by download attacks. This helps to prevent a user agent from mime sniffing content types in responses.

Audit:

Run the following procedure to check that the X-Content-Type-Options Header is enabled and set to not allow content type sniffing:

```
grep -ir X-Content-Type-Options /etc/nginx
```

The below output should be part of the output. If it is not then this recommendation is not implemented. This should be implemented on every server block. If there are multiple server blocks on the system then each should be checked. The below output should be part of each server block and part of the output of the above audit command.

```
add_header X-Content-Type-Options "nosniff";
```

Remediation:

Open your nginx configuration file that contains your server blocks. Add the below line into your server block to add X-Content-Type-Options header and direct your user agent to not sniff content types.

```
add_header X-Content-Type-Options "nosniff";
```

Default Value:

This header is not implemented by default.

References:

1. https://scotthelme.co.uk/hardening-your-http-response-headers/
2. https://www.veracode.com/blog/2014/03/guidelines-for-setting-security-headers
3. https://www.owasp.org/index.php/OWASP_Secure_Headers_Project
4. https://fetch.spec.whatwg.org/#x-content-type-options-header
5. https://www.iana.org/assignments/message-headers/message-headers.xml#perm-headers

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

7 Email and Web Browser Protections

Email and Web Browser Protections

5.3.3 Ensure that X-XSS-Protection Header is Configured and Enabled (Scored)

Profile Applicability:

- Level 1 - Webserver

Description:

The X-XSS-Protection Header allows you to protect users of older browsers who must access your website. It uses browser-based protection in order to detect and block reflected cross-site scripting.

Rationale:

The X-XSS-Protection Header is a defense-in-depth mechanism that allows you to leverage browser-based protections against cross-site scripting. While content security policy may also provide this protection X-XSS-Protection allows you to protect users whose browsers do not support Content Security Policy or as a means to protect users if you do not have a Content Security Policy. This should be implemented on your web servers to protect your users and increase user trust in your site. Your policy should be set in blocking mode when possible to ensure that the browser blocks a page if XSS is detected.

Audit:

```
grep -ir X-XSS-Protection /etc/nginx
```

The output of the command should render the below output at a minimum:

```
add header X-XSS-Protection: "1; mode=block";
```

Optionally you may configure your policy to report to a reporting URI when violations of this policy occur. You can do this by leveraging the report directive.

Remediation:

Open your nginx configuration file that contains your server blocks. Add the below line into your server block to add Content-Security-Policy and direct your user agent to block reflected cross-site scripting.

```
add header X-XSS-Protection: "1; mode=block";
```

Default Value:

This header is not enabled by default.

References:

1. https://www.owasp.org/index.php/OWASP_Secure_Headers_Project
2. https://scotthelme.co.uk/hardening-your-http-response-headers/
3. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-XSS-Protection

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

5.3.4 Ensure that Content Security Policy (CSP) is Configured and Enabled (Not Scored)

Profile Applicability:

- Level 2 - Webserver

Description:

Content Security Policy is a browser-based instruction on what locations are allowed to load scripts onto a page. It allows administrators to protect their users from possible malicious activity.

Rationale:

Content Security Policies assist organizations in mitigating and reporting cross-site scripting (XSS) attacks. Content Security Policy allows administrators to specify the locations from which allowable scripts may be executed, or if scripts may not be executed at all. This will protect your users from cross-site scripting attacks and ultimately improve user trust of your website.

Audit:

Run the following procedure to check that the Content Security Policies Header is enabled:

```
grep -ir Content-Security-Policy /etc/nginx
```

Output similar to the below output should show that this recommendation is implemented. This should be implemented on every server block. If there are multiple server blocks on the system then each should be checked.

```
add_header Content-Security-Policy: "default-src 'self'";
```

Remediation:

Open your nginx configuration file that contains your server blocks. Add the below line into your server block to add Content-Security-Policy and direct your user agent to accept documents from only specific origins.

```
add_header Content-Security-Policy: "default-src 'self'";
```

Impact:

If your content security policy is not continuously updated as your application adds resources that come from different origins or if the CSP is not correct the first time you may block execution from legitimate origins.

Default Value:

This is not enabled by default.

References:

1. https://scotthelme.co.uk/hardening-your-http-response-headers/
2. https://www.owasp.org/index.php/OWASP_Secure_Headers_Project
3. https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP
4. https://www.w3.org/TR/CSP3/

Notes:

Important Note: Content Security Policy may be customized for a significant number of use cases, including upgrading insecure requests, directing the origin of executables and reporting violations of the policy. This is a simplistic version that does not go into the depth of what a content security policy may do and is not representative of how the policy should look for your site. Organizations should investigate that their content security policy will meet their specific use case and needs.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

5.3.5 Ensure Referrer Policy is Configured and Enabled (Not Scored)

Profile Applicability:

- Level 2 - Webserver

Description:

The Referrer Policy enables organizations to define what sites should see that a referral came from your site. When an origin site directs a user to another site a referrer is sent, which identifies the URL that a user came from. Depending on your sites specific use this may present a privacy concern to your users.

Rationale:

A Referrer policy is a useful feature to help protect privacy-minded users. It allows you to identify if your site should send information to sites that it redirects where your user came from. Depending on your websites use case and design this may present privacy concerns to your users. For example, the referrer header may expose sensitive data in another web servers log if you use sensitive data in your URL parameters, such as personal information, username, and password or persistent sessions.

Ultimately, depending on your application design, this may allow session highjacking, credential gathering, or sensitive data exposure in a third party's logs.

Audit:

To check to see if your referrer policy is implemented run the following procedure. You should check to ensure that the header is part of the server block of all sites.

```
grep -r Referrer-Policy /etc/nginx
```

The output should look similar to the below. The policy may differ depending on your organization's needs.

```
add_header Referrer-Policy "no-referrer";
```

Remediation:

To implement the referrer policy, add the below line to your server blocks within your nginx configuration. The policy should be customized for your specific organization's needs. The below policy will ensure that your website is never allowed in a referrer.

```
add_header Referrer-Policy "no-referrer";
```

Default Value:

This policy is not set by default.

References:

1. https://scotthelme.co.uk/a-new-security-header-referrer-policy/
2. https://www.w3.org/TR/referrer-policy/

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

6 Linux Kernel Security

6.1 SELinux

6.1.1 Ensure SELINUX is Set to Enforcing (Scored)

Profile Applicability:

- Level 2 - Webserver
- Level 2 - Proxy
- Level 2 - Loadbalancer

Description:

SELinux is a Linux kernel mechanism that enforces mandatory access control. It works by assigning labels to files and using these to enforce pre-defined access control policies to files on the host server.

Rationale:

If an attacker exploits a server-side vulnerability, SELinux will help to prevent an attacker's lateral movement throughout the system by making it harder to access other system resources outside of the compromised realm of access.

SELinux may be set to only log violations to its policies or to enforce the policy by blocking access. Setting the policy to enforce blocks actions that violate the policy.

Audit:

Run the following procedure in order to test to ensure SELinux is set to enforcing.

```
getenforce
```

The output of the command should show as "Enforcing". If it is not Enforcing then this recommendation is not implemented.

Remediation:

Run the following procedure to ensure that SELinux is set to enforcing mode.

```
sudo setenforce 1
```

Impact:

This will enforce your SELinux policy. If your policy is not configured correctly, you may block access to legitimate files needed by legitimate users on your host system.

Default Value:

SELinux is turned on by default.

References:

1. https://www.getpagespeed.com/server-setup/nginx/nginx-selinux-configuration
2. https://www.nginx.com/blog/using-nginx-plus-with-selinux/

Notes:

Note: While default settings for SELinux allow nginx to operate you may need to change the policy if you add additional features or customizations.

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

6.1.2 Ensure the httpd_t Type is Not in Permissive Mode (Scored)

Profile Applicability:

- Level 2 - Webserver
- Level 2 - Proxy
- Level 2 - Loadbalancer

Description:

SELinux by default labels nginx files with the type httpd_t. It is possible that a specific type may have a different permission than the global policy.

Rationale:

Ensuring that the httpd_t type is set to enforcing and that the nginx runs under this type helps to ensure that nginx has the proper SELinux policies enforced on your server.

Audit:

To check if the httpd_t type is set to run in a permissive mode with a custom policy issue the below command.

```
semanage permissive -l | grep httpd_t
```

If there is no output then this recommendation is implemented.

Remediation:

To ensure that SELinux sets the httpd_t type to enforcing issue the below command:

```
semanage permissive -d httpd_t
```

Impact:

If your SELinux policy is not configured correctly this may block legitimate access by users. You should check your audit.d file for legitimate traffic that may have been blocked by your SELinux policy and correct your policy if needed prior to implementing this recommendation.

Default Value:

httpd_t is set to enforcing by default.

References:

1. https://www.getpagespeed.com/server-setup/nginx/nginx-selinux-configuration
2. https://www.nginx.com/blog/using-nginx-plus-with-selinux/

CIS Controls:

Version 7

5.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software.

Appendix: Summary Table

Control		Set Correctly	
		Yes	No
1	**Initial Setup**		
1.1	**Installation**		
1.1.1	Installing NGINX (Scored)	☐	☐
1.2	**Configure Software Updates**		
1.2.1	Ensure Package Manager Repositories are Configured (Scored)	☐	☐
1.2.2	Ensure the Latest Software Package is Installed (Scored)	☐	☐
2	**Basic Configuration**		
2.1	**Minimize NGINX Modules**		
2.1.1	Ensure only Required Modules are Installed (Scored)	☐	☐
2.1.2	Ensure HTTP DAV Module is not Installed (Scored)	☐	☐
2.1.3	Ensure Modules With GZIP Functionality are Disabled (Scored)	☐	☐
2.1.4	Ensure the Autoindex Module is Disabled (Scored)	☐	☐
2.2	**Account Security**		
2.2.1	Ensure that NGINX is Run Using a Non-Privileged, Dedicated Service Account (Scored)	☐	☐
2.2.2	Ensure the NGINX Service Account is Locked (Scored)	☐	☐
2.2.3	Ensure the NGINX Service Account has an Invalid Shell (Scored)	☐	☐
2.3	**Permissions and Ownership**		
2.3.1	Ensure NGINX Directories and Files are Owned by Root (Scored)	☐	☐
2.3.2	Ensure Access on NGINX Directories and Files is Restricted (Scored)	☐	☐
2.3.3	Ensure NGINX Process ID (PID) File is Secured (Scored)	☐	☐
2.3.4	Ensure Core Dump Directory Is Secured (Not Scored)	☐	☐
2.4	**Network Configuration**		
2.4.1	Ensure NGINX only Listens for Network Connections on Authorized Ports (Not Scored)	☐	☐
2.4.2	Ensure that Requests for Unknown Host Names are Rejected (Not Scored)	☐	☐
2.4.3	Ensure keepalive_timeout is Configured (Scored)	☐	☐
2.4.4	Ensure send_timeout is configured (Scored)	☐	☐
2.5	**Information Disclosure**		
2.5.1	Ensure server_tokens directive is set to Off (Scored)	☐	☐
2.5.2	Ensure Default Error Pages and index.html Pages don't	☐	☐

	reference NGINX (Scored)		
2.5.3	Ensure Hidden File Serving Is Disabled (Scored)	☐	☐
2.5.4	Ensure Proxy does not Enable Information Disclosure (Scored)	☐	☐
3	**Logging**		
3.1	Ensure Detailed Access and Request Logging Is Enabled (Not Scored)	☐	☐
3.2	Ensure that Access Logging is Enabled (Scored)	☐	☐
3.3	Ensure that Error Logging is Enabled and Set to the Info Logging Level (Scored)	☐	☐
3.4	Ensure Log Files are Rotated (Scored)	☐	☐
3.5	Ensure NGINX Error Logs are Sent to a Remote Syslog Server (Not Scored)	☐	☐
3.6	Ensure NGINX Access Logs are Sent to a Remote Syslog Server (Not Scored)	☐	☐
3.7	Ensure Proxies Pass Source IP Information (Scored)	☐	☐
4	**Encryption**		
4.1	**TLS / SSL Configuration**		
4.1.1	Ensure HTTP is Redirected to HTTPS (Scored)	☐	☐
4.1.2	Ensure a Trusted Certificate and Trust Chain is Installed (Scored)	☐	☐
4.1.3	Ensure Private Key Permissions are Restricted (Scored)	☐	☐
4.1.4	Ensure Only Modern TLS Protocols are Used (Scored)	☐	☐
4.1.5	Restrict Weak Ciphers From Being Used (Scored)	☐	☐
4.1.6	Ensure Custom Diffie-Hellman Parameters are Used (Scored)	☐	☐
4.1.7	Ensure Online Certificate Status Protocol (OCSP) Stapling is Enabled (Scored)	☐	☐
4.1.8	Ensure HTTP Strict Transport Security (HSTS) is Enabled (Scored)	☐	☐
4.1.9	Ensure HTTP Public Key Pinning Extension is Enabled (Not Scored)	☐	☐
4.1.10	Ensure Upstream Server Traffic is Authenticated with a Client Certificate (Scored)	☐	☐
4.1.11	Ensure Upstream Traffic Server Certificate is Trusted (Not Scored)	☐	☐
4.1.12	Ensure your Domain is Preloaded (Not Scored)	☐	☐
4.1.13	Ensure Session Resumption Is Disabled To Enable Perfect Forward Security (Not Scored)	☐	☐
4.1.14	Ensure HTTP/2.0 is Used (Not Scored)	☐	☐
5	**Request Filtering and Restrictions**		
5.1	**Access Control**		
5.1.1	Ensure Allow and Deny Filters are Used to Limit Access to Specific IP Addresses (Not Scored)	☐	☐

5.1.2	Ensure only Whitelisted HTTP Methods are Allowed (Scored)	☐	☐
5.2	**Request Limits**		
5.2.1	Ensure Timeout Values for Reading Client Request Header and Client Request Body are Set (Scored)	☐	☐
5.2.2	Ensure a Limit for Reading Client Request Body is Set (Scored)	☐	☐
5.2.3	Ensure Limits for Reading Large Client Request Headers are Set (Scored)	☐	☐
5.2.4	Ensure Number of Connections are Limited by IP Address (Not Scored)	☐	☐
5.2.5	Ensure Rate Limits by IP Addresses are Set (Not Scored)	☐	☐
5.3	**Browser Security**		
5.3.1	Ensure X-Frame-Options Header is Configured and Enabled (Scored)	☐	☐
5.3.2	Ensure X-Content-Type-Options Header is Configured and Enabled (Scored)	☐	☐
5.3.3	Ensure that X-XSS-Protection Header is Configured and Enabled (Scored)	☐	☐
5.3.4	Ensure that Content Security Policy (CSP) is Configured and Enabled (Not Scored)	☐	☐
5.3.5	Ensure Referrer Policy is Configured and Enabled (Not Scored)	☐	☐
6	**Linux Kernel Security**		
6.1	**SELinux**		
6.1.1	Ensure SELINUX is Set to Enforcing (Scored)	☐	☐
6.1.2	Ensure the httpd_t Type is Not in Permissive Mode (Scored)	☐	☐

Appendix: Change History

Date	Version	Changes for this version
12/10/2018	1.0.0	Early Draft Release