CIS Benchmarks

CIS NGINX Benchmark

v1.0.0 - 02-28-2019

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

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 versions of Linux, please substitute the CentOS specific commands for the equivalent commands on the Linux distribution you are using.
To obtain the latest version of this guide, please visit http://benchmarks.cisecurity.org. If you have questions or 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, and 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:

Author
Alexander Sennhauser
James Scott

Contributor
Karen Scarfone

Recommendations

1 Initial Setup

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

1.1 Installation

1.1.1 Ensure NGINX is installed (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.

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, nginx is not installed.

Remediation:

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

1.1.2 Ensure NGINX is installed from source (Not Scored)

Profile Applicability:

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

Description:

Installing NGINX directly from source allows you to install NGINX without the use of a package manager.

Rationale:

Installing NGINX from source allows you to harden your instance of NGINX by minimizing modules. NGINX is unable to remove modules when installed using a package manager. By installing from source, you are able to minimize modules, however, some additional configuration will be required and updates will not be automated out of the box for you.

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, nginx is not installed.

Remediation:

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

Impact:

By installing NGINX from source, you will have to manually upgrade NGINX or automate upgrades yourself. The default values for NGINX may also vary from this guide using this method.

Default Value:

NGINX is not installed by default.

References:

1. https://docs.nginx.com/nginx/admin-guide/installing-nginx/installing-nginx-open-source/

CIS Controls:

Version 7

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

1.2 Configure Software Updates

1.2.1 Ensure package manager repositories are properly configured (Not Scored)

Profile Applicability:

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

Description:

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

Rationale:

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

Audit:

To 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 manager 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 (Not 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 exploitation of security vulnerabilities, such as the execution of malicious code.

Audit:

To verify your NGINX package is up to date, run the following command:

```
yum info nginx
```

Remediation:

To 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 (Not Scored)

Profile Applicability:

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

Description:

This NGINX installation comes with several modules out of the box. These modules are not all always needed. Installations of NGINX should be hardened to ensure only the necessary modules are installed.

Rationale:

Minimizing features and functionality built into NGINX can help to reduce the number of vulnerabilities your server has, which reduces the likelihood of a successful compromise by attackers.

Audit:

Audit the modules used in your current NGINX build by using the nginx verification command:

```
nginx -V
```

Remediation:

Consult the NGINX module documentation to determine which modules are needed for your specific installation.
Modules may be removed using the configure command.

References:

1. 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 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 WebDAV module is not installed (Scored)

Profile Applicability:

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

Description:

The http_dav_module enables HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV) as defined by RFC 4918. This enables file-based operations on your web server, such as the ability to create, delete, change and move files on your server. Most modern architectures have replaced this functionality with cloud-based object storage, in which case the module should not be installed.

Rationale:

WebDAV functionality opens up an unnecessary path for exploiting your web server. Through misconfigurations of WebDAV operations, an attacker may be able to access and manipulate files on the server.

Audit:

Run the following command to ensure the http_dav_module is not installed:

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

Ensure the output of the command is empty.

Remediation:

To remove the http_dav_module, recompile nginx from source without the --with-http_dav_module flag.

Default Value:

The HTTP WebDAV 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:

gzip is used for compression. Compression functionality should be disabled to prevent certain types of attacks from being performed successfully.

Rationale:

Compression has been linked with the Breach attack and others. While the Breach attack has been mitigated with modern usages of the HTTP protocol, disabling the use of compression is considered a defense-in-depth strategy to mitigate other attacks.

Audit:

Run the following command to ensure gzip modules are not installed:

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

Ensure the output of the command is empty.

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 be done without the --with-http_gzip_static_module configuration directive.

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

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 autoindex module processes requests ending with the slash character. This feature enables directory listing, which could be useful in attacker reconnaissance, so it should be disabled.

Rationale:

Automated directory listings may 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:

To determine if the autoindex module is disabled, search the NGINX configuration files (nginx.conf and any included configuration files) for autoindex directives:

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

Ensure there are no `autoindex on` directives present.

Remediation:

Perform the following to disable the autoindex module:

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

```
egrep -i '^\s*autoindex\s+' /etc/nginx/nginx.conf
egrep -i '^\s*autoindex\s+' /etc/nginx/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 (Not Scored)

Profile Applicability:

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

Description:

The nginx user directive designates which user account nginx worker processes run under. Ensuring a non-privileged, dedicated service account is used is a defense in depth measure to limit what an attacker who compromises the account can do.

Rationale:

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

Audit:

Run the following to verify nginx is being run by a dedicated non-privileged user account:
Step 1: Verify nginx is being run as a dedicated user:

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

If a user directive similar to the below is not found, this is not a dedicated user. If a user is found similar to the output shown below, continue to step 2. If the user does not exist, a user will need to be added.

```
user  nginx;
```

Step 2: Verify 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 if this user is not privileged:

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

Step 3: Verify 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 if this user is not part of any other groups than the primary group:

```
nginx : nginx
```

Remediation:

Add a system account for the nginx user with a home directory of /var/cache/nginx and a shell of /sbin/nologin so it does not have the ability to log in, 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 are 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 nginx user account should have a valid password, but the account should be locked.

NOTE: If a different account is used to run nginx, that account's name should be substituted for nginx in the audit and remediation procedures.

Rationale:

As a defense-in-depth measure, the nginx user account should be locked to prevent logins and to prevent someone 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, sudo should be used instead, which would not require the nginx account password.

Audit:

Verify the nginx service account is locked by running this command:

```
passwd -S nginx
```

The results should be similar to one of 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 service 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, so the /sbin/nologin shell should be set for 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. This prevents an attacker who compromises the account to log in with it.

Audit:

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

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

The shell 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 /sbin/nologin by using the following command:

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

Default Value:

The nginx user has a shell of /sbin/nologin 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 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 and its files should be root.

Rationale:

Setting 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 command to verify the ownership of the nginx configuration files:

```
stat /etc/nginx
```

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

```
Access: (0755/drwxr-xr-x)  Uid: (    0/    root)   Gid: (    0/    root)
```

Remediation:

Run the following command to ensure 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 to 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 should enforce the principle of least privilege.

Rationale:

This ensures that only users who need access to configuration files are able to view them, thus preventing unauthorized access. Other users will need to use sudo in order to access these files.

Audit:

To verify 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 permissions similar to the output below:

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

To verify 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 permissions similar to the output below:

```
-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 these commands:

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

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:

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 the NGINX process ID (PID) file is secured (Scored)

Profile Applicability:

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

Description:

The PID file stores the main process ID of the nginx process. This file should be protected from unauthorized modification.

Rationale:

The PID file should be owned by root and the group root. It should also be readable to everyone, but only writable by root (permissions 644). This will prevent unauthorized modification of the PID file, which could cause a denial of service.

Audit:

Run this command to verify the ownership and permissions of the nginx PID file:

```
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 PID file is not owned by root, issue this command:

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

If the PID file has permissions greater than 644, issue this 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 the core dump directory is secured (Not Scored)

Profile Applicability:

- Level 1 - Webserver

Description:

Core dumps are snapshots of memory. The `working_directory` directive is used to specify the directory NGINX attempts to create core dumps in. 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 and the group the NGINX process executes as, and is inaccessible to other users. Usually, production systems should not have this enabled.

Rationale:

Core dumps may contain sensitive information that should not be accessible by other accounts on the system.

Audit:

Run the following procedure to verify 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. It is not within the NGINX web document root.
2. It is owned by root and has a group ownership of the NGINX group.
3. It has 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. It is not within the NGINX web document root.
2. It is owned by root and has a group ownership of the NGINX group:

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

3. It has 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 any port, but it should be configured to listen on authorized ports only.

Rationale:

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

Audit:

Use this command to audit all listening ports on the server:

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

The ports being used should immediately follow the listen directive in the output. Ensure all ports that are actively listening and not commented out are authorized for use on the server. The output should look similar to this example:

```
/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 out or delete the associated configuration for 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 requests for unknown host names are rejected (Not Scored)

Profile Applicability:

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

Description:

Your host header should be part of a predefined whitelist of known good hosts, which enables blocking access to other hosts. You should treat the host header as another input to be validated, as it is defined by the user agent.

Rationale:

Whitelisting specific hosts and blocking access to all other hosts, you help to mitigate host header injection attacks against your server. Such attacks could be used by an attacker to redirect you to a rogue host and execute scripts or get you to input credentials.

Audit:

Run the following comment to verify 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 any kind of 200 response, this recommendation is not implemented.

Remediation:

Ensure your first server block mirrors the below in your nginx configuration, either at /etc/nginx/nginx.conf or any included file within your nginx config:

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

Then investigate each server block to ensure the server_name directive is explicitly defined. 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 to use for implementing 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 10 seconds or less, but not 0 (Scored)

Profile Applicability:

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

Description:

Persistent connections are leveraged by all modern browsers to facilitate greater web performance. The keep-alive timeout limits the time a 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 helps mitigate denial of service attacks that establish too many persistent connections, exhausting server resources.

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 the /etc/nginx directory. If none of these are present, the value is set at the default.

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

The output of the command should contain something similar to the following:

```
keepalive_timeout 10;
```

Remediation:

Find the HTTP or server block of your nginx configuration, and add the keepalive_timeout directive. Set it to 10 seconds or less, but not 0. This example command sets it to 10 seconds:

```
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 set to 10 seconds or less, but not 0 (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 helps mitigate slow HTTP denial of service attacks by ensuring 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 the /etc/nginx directory. If none of these are present, the value is set at the default.

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

The output of the command should be similar to the following:

```
send_timeout  10;
```

Remediation:

Find the HTTP or server block of your nginx configuration, and add the send_timeout directive. Set it to 10 seconds or less, but 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 NGINX version number and operating system version on error pages and in the `Server` HTTP response header field. This information should not be displayed.

Rationale:

Attackers can conduct reconnaissance on a website using these response headers, then target attacks for specific known vulnerabilities associated with the underlying technologies. Hiding the version will slow down and deter some potential attackers.

Audit:

In the NGINX configuration file nginx.conf, verify the `server_tokens` directive is set to `off`. To do this, check the response headers for the server header by issuing this command:

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

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

```
Server: nginx/1.14.0
```

Remediation:

To disable the `server_tokens` directive, set it 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 and index.html pages do not 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 they do not advertise the underlying infrastructure of the server.

Rationale:

By gathering information about the server, attackers can target attacks against its known vulnerabilities. Removing pages that disclose the server runs NGINX helps reduce targeted attacks on the server.

Audit:

Locate the error page and index directives in the location block of your server configuration. The default index and error pages in nginx are located at /usr/share/nginx/html/. Open these files and verify there are no references to NGINX. Issue the following commands to check the default pages and verify no results are returned:

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

Remediation:

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

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 (Not Scored)

Profile Applicability:

- Level 2 - Webserver

Description:

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

Rationale:

Disabling hidden files prevents an attacker from being able to reference a hidden file that may be put in your location and have sensitive information, like .git files.

Audit:

To verify hidden files are disabled, 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
```

Verify the output is:

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

Remediation:

Edit the `nginx.conf` file and add the 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 the NGINX reverse 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. The NGINX reverse proxy may pass these headers if not explicitly directed to remove them.

Rationale:

Attackers can conduct reconnaissance on a website using these response headers, then target attacks for specific known vulnerabilities associated with the underlying technologies. Removing these headers will reduce the likelihood of targeted attacks.

Audit:

Confirm that the headers are denied as part of the location block of the nginx configuration.
You may also have to check included files as part of this configuration.
Run this command:

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

The output should read:

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

Run this command:

```
grep proxy_hide_header
```

The output should read:

```
proxy_hide_header Server;
```

Remediation:

Implement the below directives as part of your location block. 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 logging is enabled (Not Scored)

Profile Applicability:

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

Description:

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

NOTE: Aim to keep sensitive information out of logs. For example, keep sensitive information out of query strings and URIs to avoid this.

Rationale:

Performing detailed logging ensures that incident responders, auditors, and others are able to clearly view the activity that has occurred on your server.

Audit:

Verify 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
'server="$server_name" host="$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";
```

Remediation:

Edit the log format directive in /etc/nginx/nginx.conf so it logs 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 consult the NGINX documentation and your organizational policy to ensure 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 - user agent of the client requesting access
$http_x_forwarded_for - client address a proxy or load balancer is forwarding
traffic for
```

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:

Load balancers are not source IP transparent. We must configure the X-Forwarded-For Header on the proxy and in the 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 access logging is enabled (Scored)

Profile Applicability:

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

Description:

The access_log directive should be on for every core site. It is enabled by default.

Rationale:

Access logging allows incident responders and auditors to investigate access to a system in the event of an incident.

Audit:

Run the following to verify access logging is enabled:

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

The output should show an access log configured and not disabled.
If the output is similar to the below, the nginx configuration file should be manually inspected to ensure you are logging access to all core sites and proxies.

```
access_log off;
```

Remediation:

Ensure the access_log directive is configured for every core site your organization requires logging for.
This should look similar to the below configuration snippet. 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 error logging is enabled and set to the info logging level (Scored)

Profile Applicability:

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

Description:

All errors for applications should be logged.

Rationale:

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

Audit:

Run the following to verify the 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, this recommendation is not implemented.

Remediation:

Edit /etc/nginx/nginx.conf so the error_log directive is present and not commented out. The error_log should be configured to the logging location of your choice. The configuration should 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 log files do not consume excessive disk space, potentially causing a denial of service.

Rationale:

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

Audit:

Run the below commands to verify the log rotation configuration. They should show that log compression occurs weekly and 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 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:

You should always comply with your organizational log retention 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 error logs are sent to a remote syslog server (Not Scored)

Profile Applicability:

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

Description:

Centralized log management helps ensure logs are forensically sound and are available at a central location for auditing and incident investigation.

Rationale:

A centralized logging solution aggregates logs from multiple systems to ensure logs can be referenced in the event systems are thought to be compromised. Centralized log servers are also often used to correlate logs for potential patterns of attack. If a centralized logging solution is not used and systems (and their logs) are believed to be compromised, then logs may not be permitted to be used as evidence.

Audit:

Use this command to verify your server is configured for central logging:

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

The output 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 log server, and 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, 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 access logs are sent to a remote syslog server (Not Scored)

Profile Applicability:

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

Description:

Centralized log management helps ensure logs are forensically sound and are available at a central location for auditing and incident investigation.

Rationale:

A centralized logging solution aggregates logs from multiple systems to ensure logs can be referenced in the event systems are thought to be compromised. Centralized log servers are also often used to correlate logs for potential patterns of attack. If a centralized logging solution is not used and systems (and their logs) are believed to be compromised, then logs may not be permitted to be used as evidence.

Audit:

Use this command to verify your server is configured for central logging:

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

The output should show the access 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 log server. The local logging facility may be 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, 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 and remote address headers help identify and separate the originating client IP address of the user agent and the proxy IP address. The two types of addresses are the 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:

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 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:

To ensure 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 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;
    }
}
```

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

Notes:

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

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 by leveraging HTTPS. Requests leveraging HTTP are unencrypted. Unencrypted requests should be redirected so they are encrypted. Any listening HTTP port on your web server should redirect to a server profile that uses encryption. The default HTTP (unencrypted) port is 80.

Rationale:

Redirecting user agent traffic to HTTPS helps to ensure all user traffic is encrypted. 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; users are able to access your website even if they lack the security awareness to use HTTPS over HTTP when requesting your website.

Audit:

To verify 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. The configuration file should return a 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:

Edit your web server or proxy configuration file to redirect 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 (Not Scored)

Profile Applicability:

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

Description:

Certificates and their trust chains are needed to establish the identity of a web server as legitimate and trusted. Certificate authorities validate a web server's identity and that you are the owner of that web server domain name.

Rationale:

Without a certificate and full trust chain installed on your web server, modern browsers will flag your web server as untrusted.

Audit:

Run this command to find the file location of your certificate:

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

The output of your command should look similar to the below output. If there is no output, you do not have a certificate installed.
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. It should include the full certificate chain.

```
-----BEGIN CERTIFICATE-----
Insert Your Web Server Certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Insert Your Certificate Authority Intermediate Certificate
```

```
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Insert Your Certificate Authority 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 server's private key and a certificate signing request.
The following command will create your certificate's 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 server's DNS
name
Email Address []: Your email address
```

Step 2: Obtain a signed certificate from your certificate authority.
Provide your chosen certificate authority with your certificate signing request. Follow your certificate authority's 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 the .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 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 with the following command:

```
sudo service nginx restart
```

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 from unauthorized access by limiting access based on the principle of least privilege.

Rationale:

A server's private key file should be restricted to 400 permissions. This ensures 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, 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:

Verify 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.

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

The output should show the permissions as 400 or -r--------. If this is not the case, 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
```

Default Value:

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

Notes:

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:

Only modern TLS protocols should be enabled in NGINX for all client connections and upstream connections. Removing legacy TLS and SSL protocols (SSL 3.0, TLS 1.0 and 1.1), and enabling emerging and stable TLS protocols (TLS 1.2), ensures users are able to take advantage of strong security capabilities and protects them from insecure legacy protocols.

Rationale:

Why disable SSL 3.0: The POODLE Vulnerability allowed attackers to exploit SSL 3.0 to obtain cleartext information by exploiting weaknesses in CBC in 2014. SSL 3.0 is also no longer FIPS 140-2 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: Because of the increased security associated with higher versions of TLS, TLS 1.0 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.

Audit:

You can verify which SSL/TLS protocols your server uses by issuing the below command to see the configured cipher suites on the server. If anything older than TLS 1.2 is implemented or nothing appears, 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 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;/" /etc/nginx/nginx.conf
```

Proxy:

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

If your ssl_protocols are not already configured, this can be accomplished manually by opening your web server or proxy server configuration file and manually adding the directives.

Web Server:

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

Proxy:

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

Impact:

Disabling certain TLS 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 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:

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 Disable weak ciphers (Scored)

Profile Applicability:

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

Description:

The ssl_ciphers directive should be used to configure the available ciphers on your web server, and the proxy_ssl_ciphers directive should be used to configure the available ciphers for your proxy. Weak ciphers should be disabled based on your company's policy or an industry best practice compliance profile.

The ssl_prefer_server_ciphers should be used to ensure the user agent respects the server's preferred cipher order and does not set its own. If you are using a proxy or load balancer, you should use the proxy_ssl_ciphers directive to ensure your upstream connections are negotiated using secure ciphers.

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 vulnerable way. 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 protect users from malicious actors on the client side.

Audit:

Use the following procedure to verify the ssl_cipher and proxy_ssl_cipher directives meet your company's policy.

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

This output will show the server's configured ciphers and cipher preference policy. If you have multiple server blocks or proxy passes, you should ensure the directive or directives appear for each.
The output should also contain the directive ssl_prefer_server_ciphers, as shown below:

```
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:
https://www.owasp.org/index.php/Testing_for_SSL-TLS_%28OWASP-CM-001%29

Remediation:

The following procedures may be used to implement 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.
Set the ssl_cipher directive as part of your server block, and set the proxy_ssl_ciphers directive 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 have ssl_ciphers or proxy_ssl_ciphers defined, add the directives to your proxy or web server configuration profile, then 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 are 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:

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

Custom Diffie-Hellman (DH) key exchange parameters should be used. DH Ephemeral (DHE) parameters with at least 2048 bits should be generated.

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; stronger alternatives should be used instead to provide better protection for data protected by encryption.

Audit:

Verify the option `ssl_dhparam` is explicitly provided:

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

Remediation:

Generate strong DHE (Ephemeral Diffie-Hellman) parameters using the following commands:

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

Alter the server configuration to use the new parameters:

```
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 allows a user's browser or another user agent to verify the certificate it is seeing is not revoked. OCSP stapling ensures your server presents this information to the user's browser in a way that best meets the performance and security needs of your website. It polls the Certificate Authority's (CA) OCSP server at regular intervals to ensure it is continuously kept up to date. OCSP stapling helps improve performance and security, so it should be enabled.

Rationale:

OCSP stapling 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.

Note: OCSP stapling, while a step forward from the older certificate revocation list model, does share similar risks. Between the time 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.

Audit:

Run the following command to verify 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 CA's OCSP server.
Your CA's OCSP server may be found on your CA's website and will 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. 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;
}
```

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:

HTTP Strict Transport Security (HSTS) headers instruct a user agent on how to communicate with a web server. HSTS headers ensure the strict transport security policies 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 recommended to be at least six months in length. This ensures the browser remembers your website should only be accessible via HTTPS for this amount of time.

Rationale:

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 this 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, this recommendation is not implemented. The header should also include the max-age directive with 15768000 seconds (six months) or longer configured.

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

Remediation:

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

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

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 is enabled (Not Scored)

Profile Applicability:

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

Description:

HTTP Public Key Pinning, also known as certificate pinning, allows a site to specify exactly which certificates the browser or another user agent should accept. HTTP Public Key Pinning allows for the certificate rotation to be scheduled using backup fingerprints to ensure that user agent has both certificates stored. HTTP Public Key Pinning should be enabled.

Rationale:

HTTP Public Key Pinning assists in preventing a user agent from falling victim to a forged certificate, such as man in the middle attacks.

Audit:

Verify HTTP Public Key Pinning is enabled by running this command:

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

If the output does not return anything, 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;
```

Default Value:

HTTP Public Key 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:

Client certificate validation allows the upstream server to authenticate the identity of the client connecting to it. This assists in the establishment of mutual authentication between the client and the server.

Rationale:

Using client certificate validation allows you to establish a trusted proxy server.

Audit:

To verify this recommendation, 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:

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

You should see output similar to the below. You may need to manually ensure 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 use /etc/nginx/ssl/cert.pem. Implement the configuration as part of the location block:

```
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 an NGINX web server, 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 is 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 the upstream traffic server certificate is trusted (Not Scored)

Profile Applicability:

- Level 2 - Proxy
- Level 2 - Loadbalancer

Description:

The NGINX server should be configured to validate the identity of the upstream server it is sending information to.

Rationale:

Configuring NGINX to validate the identity of the upstream server helps mitigate the risk of a man in the middle attack occurring against your server.

Audit:

To verify the configuration, follow this procedure:
Step 1: Verify 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 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:

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

Step 2: Verify the certificate trust chains for upstream servers are installed properly.
Verify the certificates installed in the location referenced by the proxy_ssl_trusted_certificate directive are valid.

Remediation:

Obtain the full certificate chain of the upstream server in .pem format. Then reference that file in the location block as part of the proxy_ssl_trusted_certificate directive. Implement the proxy_ssl_trusted_certificate and proxy_ssl_verify directives 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;
```

Default Value:

This is not set up 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 hardcodes it as only being accessible through HTTPS by browsers.

Note: Preloading should only be done with careful consideration! Your website and all its subdomains will be forced over HTTPS. If your website or any of its subdomains are not able to support preloading, 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.

Rationale:

Preloading your domain helps prevent HTTP downgrade attacks and increases trust.

Audit:

Visit https://hstspreload.org/ and type in your top-level domain name to verify it is 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 4.1.2.

2. Redirect from HTTP to HTTPS if using port 80.

This may be accomplished by following recommendation 4.1.1.

3. Configure all subdomains to 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. Configure an HSTS header on your base domain, as shown below for nginx.

If your base domain is nginx, you may accomplish this with several modifications from the HSTS recommendation. Change your header to include the preload directive and the includesubdomains directive, and make your max-length six months 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/.

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 (Scored)

Profile Applicability:

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

Description:

Session resumption for HTTPS sessions should be disabled so perfect forward secrecy can be achieved.

Rationale:

Perfect forward secrecy is an encryption mechanism that enables past session keys to not be compromised even if the server's 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. Allowing session resumption breaks perfect forward secrecy; this expands the surface area for an attacker to compromise past sessions and communications with a server if they are able to compromise the session.

Audit:

Run the following command to verify 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 should contain the following:

```
ssl_session_tickets off;
```

Remediation:

Turn off the ssl_session_tickets directive as part of any server block in your nginx configuration:

```
ssl_session_tickets off;
```

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:

HTTP/2.0 is an optimized and more secure version of the HTTP protocol. It should be enabled so users can take advantage of it.

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

Rationale:

HTTP/2.0 introduces both performance benefits through full multiplexing and several security benefits. HTTP/2.0 has improved cipher suite 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:

Verify that listening ports on the web server leverage HTTP/2.0 by running this 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 the output is commented out, this recommendation is not implemented.

Remediation:

Open the nginx server configuration file and configure all listening ports with http2, similar to that of this example:

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

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:

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 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. It is 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:

To verify IP-based restrictions are limiting access correctly, perform the following steps:

Step 1: Open your nginx config file and any files 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 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 only want user traffic to come from the load balancer.

Remediation:

Compile a list of network ranges or IP addresses you would want to access your web server or proxy. Then add these ranges with the allow directive. The deny directive should be included with all IP addresses implicitly denied.

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

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:

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 (Not Scored)

Profile Applicability:

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

Description:

HTTP methods (also known as verbs) allow different actions to be requested from the web server at a specified path. Only the necessary methods should be enabled.

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:

To verify this, use a tool like curl to send a request with each 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 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;
}
```

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 the client header and body are set correctly (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 the server will wait for the header or body to be sent from the client. If the client does not send the entire header in this predefined timeframe, the server will send back a 408 request timeout error.

Rationale:

Setting the client header and body timeouts help your server mitigate possible denial of service attacks. By timing out a request, the server is able to free up resources that may be waiting for the body or header.

Audit:

To verify the current settings 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, the value is set at the default.

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

The output should contain the following:

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

Remediation:

Find the HTTP or server block of your nginx configuration and add the client_header_timeout and client_body_timeout directives set to the configuration. The below example sets the timeouts to 10 seconds.

```
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 the maximum request body size is set correctly (Scored)

Profile Applicability:

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

Description:

The client_max_body_size directive sets the size of the request body that is allowed to read a client request. This defines the number of bytes 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 to perform buffer overflow attacks. This value should be set low enough to protect the application but high enough not to interfere with functionality and block legitimate request bodies.

Audit:

To verify 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, the value is set at the default.

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

Remediation:

Find the HTTP or server block of your nginx configuration and add the client_max_body_size set to 100K in this block. The appropriate value may be different based on your application's needs.

```
client_max_body_size 100K
```

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 the maximum buffer size for URIs is defined (Scored)

Profile Applicability:

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

Description:

The large_client_header_buffers directive 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. The large_client_header_buffers directive should be set to restrict buffer usage. The number of buffers should generally set to two and the length be set to 1K; however, this may not be a good fit for your application and may need to be set differently.

Rationale:

The large_client_header_buffers directive may assist in preventing buffer overflow attacks that leverage long URI query parameters.

Audit:

Run this command to verify that the large_client_header_buffers directive is configured properly:

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

The output should be 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 the server block for more specific configurations to meet your needs. Add the below line to implement this recommendation:

```
large_client_header_buffers 2 1k
```

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

Notes:

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

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 the number of connections per IP address is limited (Not Scored)

Profile Applicability:

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

Description:

The maximum number of simultaneous connections allowed from a single IP address to your server should be limited. It should be set to a value that meets your organizational policies.

Rationale:

Limiting the number of simultaneous connections is an effective way to prevent slow denial of service attacks that try to use as many server resources as possible. This can also help prevent brute force attacks on a login page.

Audit:

Verify the HTTP block and server block contain 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:

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 to allow 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 setting 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 address are set (Not Scored)

Profile Applicability:

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

Description:

Rate limiting should be enabled to limit the number of requests an IP address may make to a server in a given period of time. The configuration values should be set based on your application's needs and your organizational policy.

Rationale:

Rate limiting allows you to mitigate potential denial of service attacks as a defense in depth mechanism.

Audit:

Verify the HTTP block and server block contains the below configuration. You may also need to check files included in include statements within your nginx config.

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

Remediation:

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. 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 queued.

```
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 header should be set to allow specific websites or no sites at all to embed your website as an object within their own, depending on your organizational policy and application needs.

Rationale:

The X-Frame-Options header allows you to mitigate the risk of clickjacking attacks.

Audit:

Run the following to verify 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, this recommendation is not implemented.

```
add_header X-Frame-Options "SAMEORIGIN";
```

Remediation:

Add the below to your server blocks in your nginx configuration. The policy should be configured to meet your organization's needs.

```
add_header 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:

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 should be used to force supported user agents to check an HTTP response's content type header with 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 where a user agent is sniffing content types in responses.

Audit:

Run this command to verify 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 should be part of the output. If it is not, this recommendation is not implemented. This should be implemented on every server block. If there are multiple server blocks on the system, each should be checked.

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

Remediation:

Open the 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 the X-XSS-Protection Header is enabled and configured properly (Scored)

Profile Applicability:

- Level 1 - Webserver

Description:

The X-Xss-Protection Header allows you to leverage browser-based protections against cross-site scripting. 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 the browser blocks a page if cross-site scripting is detected.

Rationale:

X-Xss-Protection allows you to protect users whose browsers do not support Content Security Policy (generally older browsers), or protect users if you do not have a Content Security Policy.

Audit:

Verify the header is enabled and configured by issuing the following command:

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

The output should include the below 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 enabled and configured properly (Not Scored)

Profile Applicability:

- Level 2 - Webserver

Description:

Content Security Policy allows administrators to specify the locations from which allowable scripts may be executed, or if scripts may be executed at all. Content Security Policy should be used to improve user trust of your website.

Rationale:

Content Security Policies assist organizations in mitigating and reporting cross-site scripting (XSS) attacks.

Audit:

Run this command to verify the Content Security Policies header is enabled:

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

Output similar to the below shows this recommendation is implemented. It should be implemented on every server block. If there are multiple server blocks on the system, 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'";
```

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:

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 CSP may do and is not representative of how the policy should look for your site. Organizations should ensure that their CSP will meet their specific use case and needs.

If your CSP 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.

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 the Referrer Policy is enabled and configured properly (Not Scored)

Profile Applicability:

- Level 2 - Webserver

Description:

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

Rationale:

A Referrer header may expose sensitive data in another web server's 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, not using a properly configured Referrer Policy may allow session hijacking, credential gathering, or sensitive data exposure in a third party's logs.

Audit:

Verify your referrer policy is enabled and configured properly by running the following command. 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:

Add the below line to the server blocks within your nginx configuration. The policy should be customized for your specific organization's needs. The below policy will ensure 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 Mandatory Access Control

Mandatory Access Control (MAC) provides an additional layer of access restrictions to processes on top of the base Discretionary Access Controls. By restricting how processes can access files and resources on a system the potential impact from vulnerabilities in the processes can be reduced, reducing the attack surface of the system.

Impact: Mandatory Access Control (MAC) limits the capabilities of applications and daemons on a system, while this can prevent unauthorized access the configuration of MAC can be complex and difficult to implement correctly preventing legitimate access from occurring. If your MAC 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 MAC policy, and correct your policy if needed.

Appendix: Summary Table

Control		Set Correctly	
		Yes	No
1	**Initial Setup**		
1.1	**Installation**		
1.1.1	Ensure NGINX is installed (Scored)	□	□
1.1.2	Ensure NGINX is installed from source (Not Scored)	□	□
1.2	**Configure Software Updates**		
1.2.1	Ensure package manager repositories are properly configured (Not Scored)	□	□
1.2.2	Ensure the latest software package is installed (Not Scored)	□	□
2	**Basic Configuration**		
2.1	**Minimize NGINX Modules**		
2.1.1	Ensure only required modules are installed (Not Scored)	□	□
2.1.2	Ensure HTTP WebDAV 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 (Not 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 to NGINX directories and files is restricted (Scored)	□	□
2.3.3	Ensure the NGINX process ID (PID) file is secured (Scored)	□	□
2.3.4	Ensure the 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 requests for unknown host names are rejected (Not Scored)	□	□
2.4.3	Ensure keepalive_timeout is 10 seconds or less, but not 0 (Scored)	□	□
2.4.4	Ensure send_timeout is set to 10 seconds or less, but not 0 (Scored)	□	□
2.5	**Information Disclosure**		

2.5.1	Ensure server_tokens directive is set to `off` (Scored)	□	□
2.5.2	Ensure default error and index.html pages do not reference NGINX (Scored)	□	□
2.5.3	Ensure hidden file serving is disabled (Not Scored)	□	□
2.5.4	Ensure the NGINX reverse proxy does not enable information disclosure (Scored)	□	□
3	**Logging**		
3.1	Ensure detailed logging is enabled (Not Scored)	□	□
3.2	Ensure access logging is enabled (Scored)	□	□
3.3	Ensure error logging is enabled and set to the info logging level (Scored)	□	□
3.4	Ensure log files are rotated (Scored)	□	□
3.5	Ensure error logs are sent to a remote syslog server (Not Scored)	□	□
3.6	Ensure 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 (Not 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	Disable weak ciphers (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 is enabled (Not Scored)	□	□
4.1.10	Ensure upstream server traffic is authenticated with a client certificate (Scored)	□	□
4.1.11	Ensure the 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 (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 limit access to specific IP addresses (Not Scored)	□	□

5.1.2	Ensure only whitelisted HTTP methods are allowed (Not Scored)	□	□
5.2	**Request Limits**		
5.2.1	Ensure timeout values for reading the client header and body are set correctly (Scored)	□	□
5.2.2	Ensure the maximum request body size is set correctly (Scored)	□	□
5.2.3	Ensure the maximum buffer size for URIs is defined (Scored)	□	□
5.2.4	Ensure the number of connections per IP address is limited (Not Scored)	□	□
5.2.5	Ensure rate limits by IP address 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 the X-XSS-Protection Header is enabled and configured properly (Scored)	□	□
5.3.4	Ensure that Content Security Policy (CSP) is enabled and configured properly (Not Scored)	□	□
5.3.5	Ensure the Referrer Policy is enabled and configured properly (Not Scored)	□	□
6	**Mandatory Access Control**		

Appendix: Change History

Date	Version	Changes for this version
2/28/19	1.0.0	Initial Release