

CIS NGINX Benchmark

v2.0.1 - 06-15-2023

Terms of Use

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

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

Table of Contents

Terms of Use	1
Table of Contents	2
Overview	5
Intended Audience.....	5
Consensus Guidance	6
Typographical Conventions.....	7
Recommendation Definitions.....	8
Title.....	8
Assessment Status.....	8
Automated	8
Manual.....	8
Profile	8
Description.....	8
Rationale Statement	8
Impact Statement.....	9
Audit Procedure.....	9
Remediation Procedure.....	9
Default Value.....	9
References	9
CIS Critical Security Controls® (CIS Controls®).....	9
Additional Information.....	9
Profile Definitions	10
Acknowledgements	12
Recommendations	13
1 Initial Setup	13
1.1 Installation	14
1.1.1 Ensure NGINX is installed (Automated).....	15
1.1.2 Ensure NGINX is installed from source (Manual).....	17
1.2 Configure Software Updates	19
1.2.1 Ensure package manager repositories are properly configured (Manual).....	20
1.2.2 Ensure the latest software package is installed (Manual)	22
2 Basic Configuration.....	24
2.1 Minimize NGINX Modules.....	25
2.1.1 Ensure only required modules are installed (Manual)	26
2.1.2 Ensure HTTP WebDAV module is not installed (Automated).....	28
2.1.3 Ensure modules with gzip functionality are disabled (Automated)	30

2.1.4 Ensure the autoindex module is disabled (Automated)	32
2.2 Account Security	34
2.2.1 Ensure that NGINX is run using a non-privileged, dedicated service account (Automated)	35
2.2.2 Ensure the NGINX service account is locked (Automated)	38
2.2.3 Ensure the NGINX service account has an invalid shell (Automated)	40
2.3 Permissions and Ownership	43
2.3.1 Ensure NGINX directories and files are owned by root (Automated)	44
2.3.2 Ensure access to NGINX directories and files is restricted (Automated)	46
2.3.3 Ensure the NGINX process ID (PID) file is secured (Automated)	48
2.3.4 Ensure the core dump directory is secured (Manual)	50
2.4 Network Configuration	52
2.4.1 Ensure NGINX only listens for network connections on authorized ports (Manual)	53
2.4.2 Ensure requests for unknown host names are rejected (Automated)	55
2.4.3 Ensure keepalive_timeout is 10 seconds or less, but not 0 (Automated)	57
2.4.4 Ensure send_timeout is set to 10 seconds or less, but not 0 (Automated)	59
2.5 Information Disclosure	61
2.5.1 Ensure server_tokens directive is set to `off` (Automated)	62
2.5.2 Ensure default error and index.html pages do not reference NGINX (Automated)	64
2.5.3 Ensure hidden file serving is disabled (Manual)	66
2.5.4 Ensure the NGINX reverse proxy does not enable information disclosure (Automated)	68
3 Logging	70
3.1 Ensure detailed logging is enabled (Manual)	71
3.2 Ensure access logging is enabled (Manual)	75
3.3 Ensure error logging is enabled and set to the info logging level (Automated)	77
3.4 Ensure log files are rotated (Automated)	79
3.5 Ensure error logs are sent to a remote syslog server (Manual)	81
3.6 Ensure access logs are sent to a remote syslog server (Manual)	83
3.7 Ensure proxies pass source IP information (Manual)	85
4 Encryption	87
4.1 TLS / SSL Configuration	88
4.1.1 Ensure HTTP is redirected to HTTPS (Manual)	89
4.1.2 Ensure a trusted certificate and trust chain is installed (Manual)	91
4.1.3 Ensure private key permissions are restricted (Automated)	94
4.1.4 Ensure only modern TLS protocols are used (Automated)	96
4.1.5 Disable weak ciphers (Manual)	99
4.1.6 Ensure custom Diffie-Hellman parameters are used (Automated)	103
4.1.7 Ensure Online Certificate Status Protocol (OCSP) stapling is enabled (Automated)	105
4.1.8 Ensure HTTP Strict Transport Security (HSTS) is enabled (Automated)	107
4.1.9 Ensure upstream server traffic is authenticated with a client certificate (Automated)	109
4.1.10 Ensure the upstream traffic server certificate is trusted (Manual)	111
4.1.11 Ensure your domain is preloaded (Manual)	113
4.1.12 Ensure session resumption is disabled to enable perfect forward security (Automated)	115
4.1.13 Ensure HTTP/2.0 is used (Automated)	117
4.1.14 Ensure only Perfect Forward Secrecy Ciphers are Leveraged (Manual)	119
5 Request Filtering and Restrictions	121
5.1 Access Control	122
5.1.1 Ensure allow and deny filters limit access to specific IP addresses (Manual)	123
5.1.2 Ensure only approved HTTP methods are allowed (Manual)	125
5.2 Request Limits	127
5.2.1 Ensure timeout values for reading the client header and body are set correctly (Automated)	128

5.2.2 Ensure the maximum request body size is set correctly (Automated).....	130
5.2.3 Ensure the maximum buffer size for URIs is defined (Automated).....	132
5.2.4 Ensure the number of connections per IP address is limited (Manual)	134
5.2.5 Ensure rate limits by IP address are set (Manual)	136
5.3 Browser Security	138
5.3.1 Ensure X-Frame-Options header is configured and enabled (Automated).....	139
5.3.2 Ensure X-Content-Type-Options header is configured and enabled (Automated)	141
5.3.3 Ensure that Content Security Policy (CSP) is enabled and configured properly (Manual)	143
5.3.4 Ensure the Referrer Policy is enabled and configured properly (Manual)	145
6 Mandatory Access Control.....	147
<i>Appendix: Summary Table</i>	<i>148</i>
<i>Appendix: Change History</i>	<i>153</i>

Overview

All CIS Benchmarks focus on technical configuration settings used to maintain and/or increase the security of the addressed technology, and they should be used in **conjunction** with other essential cyber hygiene tasks like:

- Monitoring the base operating system for vulnerabilities and quickly updating with the latest security patches
- Monitoring applications and libraries for vulnerabilities and quickly updating with the latest security patches

In the end, the CIS Benchmarks are designed as a key **component** of a comprehensive cybersecurity program.

This document, CIS NGINX Benchmark, provides prescriptive guidance for establishing a secure configuration posture for NGINX version 1.22.x running on Redhat.

This guide was tested against NGINX version 1.22.0 with default installation location, using the packages installed using the Red Hat Enterprise Linux package managers.

This Benchmark was written using commands for and tested on, Red Hat Enterprise Linux release 8.0. For other versions of Linux, please substitute the distribution-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 to 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 CIS Benchmark was created using a consensus review process comprised of a global community of subject matter experts. The process combines real world experience with data-based information to create technology specific guidance to assist users to secure their environments. 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
<code>Stylized Monospace font</code>	Used for blocks of code, command, and script examples. Text should be interpreted exactly as presented.
<code>Monospace font</code>	Used for inline code, commands, or examples. Text should be interpreted exactly as presented.
<i><italic font in brackets></i>	Italic texts set in angle brackets denote a variable requiring substitution for a real value.
<i>Italic font</i>	Used to denote the title of a book, article, or other publication.
Note	Additional information or caveats

Recommendation Definitions

The following defines the various components included in a CIS recommendation as applicable. If any of the components are not applicable it will be noted or the component will not be included in the recommendation.

Title

Concise description for the recommendation's intended configuration.

Assessment Status

An assessment status is included for every recommendation. The assessment status indicates whether the given recommendation can be automated or requires manual steps to implement. Both statuses are equally important and are determined and supported as defined below:

Automated

Represents recommendations for which assessment of a technical control can be fully automated and validated to a pass/fail state. Recommendations will include the necessary information to implement automation.

Manual

Represents recommendations for which assessment of a technical control cannot be fully automated and requires all or some manual steps to validate that the configured state is set as expected. The expected state can vary depending on the environment.

Profile

A collection of recommendations for securing a technology or a supporting platform. Most benchmarks include at least a Level 1 and Level 2 Profile. Level 2 extends Level 1 recommendations and is not a standalone profile. The Profile Definitions section in the benchmark provides the definitions as they pertain to the recommendations included for the technology.

Description

Detailed information pertaining to the setting with which the recommendation is concerned. In some cases, the description will include the recommended value.

Rationale Statement

Detailed reasoning for the recommendation to provide the user a clear and concise understanding on the importance of the recommendation.

Impact Statement

Any security, functionality, or operational consequences that can result from following the recommendation.

Audit Procedure

Systematic instructions for determining if the target system complies with the recommendation

Remediation Procedure

Systematic instructions for applying recommendations to the target system to bring it into compliance according to the recommendation.

Default Value

Default value for the given setting in this recommendation, if known. If not known, either not configured or not defined will be applied.

References

Additional documentation relative to the recommendation.

CIS Critical Security Controls® (CIS Controls®)

The mapping between a recommendation and the CIS Controls is organized by CIS Controls version, Safeguard, and Implementation Group (IG). The Benchmark in its entirety addresses the CIS Controls safeguards of (v7) "5.1 - Establish Secure Configurations" and (v8) "4.1 - Establish and Maintain a Secure Configuration Process" so individual recommendations will not be mapped to these safeguards.

Additional Information

Supplementary information that does not correspond to any other field but may be useful to the user.

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:

Contributor

Alexander Sennhauser
Eric Pinnell
James Scott
Karen Scarfone
Greg MacLean
Krishna Rayavaram

Editor

Eric Pinnell
Krishna Rayavaram

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 (Automated)

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:

1. Configure and setup Nginx


```

sudo su
dnf update -y && dnf install dnf-utils -y
cat << EOF > /etc/yum.repos.d/nginx.repo
[nginx-stable]
name=nginx stable repo
baseurl=http://nginx.org/packages/rhel/8/\$basearch/
gpgcheck=1
enabled=1
gpgkey=https://nginx.org/keys/nginx_signing.key
module_hotfixes=true
EOF
dnf 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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	2.2 <u>Ensure Authorized Software is Currently Supported</u> Ensure that only currently supported software is designated as authorized in the software inventory for enterprise assets. If software is unsupported, yet necessary for the fulfillment of the enterprise's mission, document an exception detailing mitigating controls and residual risk acceptance. For any unsupported software without an exception documentation, designate as unauthorized. Review the software list to verify software support at least monthly, or more frequently.			
v7	2.2 <u>Ensure Software is Supported by Vendor</u> Ensure that only software applications or operating systems currently supported by the software's vendor are added to the organization's authorized software inventory. Unsupported software should be tagged as unsupported in the inventory system.			

1.1.2 Ensure NGINX is installed from source (Manual)

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.

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.

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"](https://docs.nginx.com/nginx/admin-guide/installing-nginx/installing-nginx-open-source/).







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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>2.2 Ensure Authorized Software is Currently Supported</u> Ensure that only currently supported software is designated as authorized in the software inventory for enterprise assets. If software is unsupported, yet necessary for the fulfillment of the enterprise's mission, document an exception detailing mitigating controls and residual risk acceptance. For any unsupported software without an exception documentation, designate as unauthorized. Review the software list to verify software support at least monthly, or more frequently.			
v7	<u>2.2 Ensure Software is Supported by Vendor</u> Ensure that only software applications or operating systems currently supported by the software's vendor are added to the organization's authorized software inventory. Unsupported software should be tagged as unsupported in the inventory system.			

1.2 Configure Software Updates

1.2.1 Ensure package manager repositories are properly configured (Manual)

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 commands:

Redhat:

```
dnf repolist -v nginx-stable
```

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

Additional Information:

Package update and installation commands are based on Red Hat Enterprise Linux 8. If using a different Linux distribution, please substitute with the appropriate command(s).

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>7.3 Perform Automated Operating System Patch Management</u> Perform operating system updates on enterprise assets through automated patch management on a monthly, or more frequent, basis.			
v8	<u>7.4 Perform Automated Application Patch Management</u> Perform application updates on enterprise assets through automated patch management on a monthly, or more frequent, basis.			
v7	<u>3.4 Deploy Automated Operating System Patch Management Tools</u> 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.			
v7	<u>3.5 Deploy Automated Software Patch Management Tools</u> 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 (Manual)

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:

Redhat:

```
dnf info nginx
```

Remediation:

To install the latest NGINX package, run the following command:

Redhat:

```
dnf update nginx -y
```













References:

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

Additional Information:

Package update and installation commands are based on Red Hat Enterprise Linux 8. If using a different Linux distribution, please substitute with the appropriate command(s).

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>7.3 Perform Automated Operating System Patch Management</u> Perform operating system updates on enterprise assets through automated patch management on a monthly, or more frequent, basis.			
v8	<u>7.4 Perform Automated Application Patch Management</u> Perform application updates on enterprise assets through automated patch management on a monthly, or more frequent, basis.			
v7	<u>3.4 Deploy Automated Operating System Patch Management Tools</u> 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.			
v7	<u>3.5 Deploy Automated Software Patch Management Tools</u> 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 (Manual)

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>

Additional Information:

NOTE: NGINX does not support the removal of modules using the dnf method of installation. In order to remove modules from NGINX, you will need to compile it from source. Consider the tradeoff between security hardening and ease of upgrade prior to attempting to build from source.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	2.6 Allowlist Authorized Libraries Use technical controls to ensure that only authorized software libraries, such as specific .dll, .ocx, .so, etc., files, are allowed to load into a system process. Block unauthorized libraries from loading into a system process. Reassess bi-annually, or more frequently.			
v7	2.8 Implement Application Whitelisting of Libraries The organization's application whitelisting software must ensure that only authorized software libraries (such as *.dll, *.ocx, *.so, etc) are allowed to load into a system process.			

2.1.2 Ensure HTTP WebDAV module is not installed (Automated)

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 `dnf`.




References:

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

Additional Information:

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

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	2.6 <u>Allowlist Authorized Libraries</u> Use technical controls to ensure that only authorized software libraries, such as specific .dll, .ocx, .so, etc., files, are allowed to load into a system process. Block unauthorized libraries from loading into a system process. Reassess bi-annually, or more frequently.			
v7	2.8 <u>Implement Application Whitelisting of Libraries</u> The organization's application whitelisting software must ensure that only authorized software libraries (such as *.dll, *.ocx, *.so, etc) are allowed to load into a system process.			

2.1.3 Ensure modules with gzip functionality are disabled (Automated)

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 2>&1 | grep -E '(http_gzip_module|http_gzip_static_module)'
```

Ensure the output of the command is empty.

Remediation:

In order to disable the http_gzip_module and the http_gzip_static_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 or --with-http_gzip_module configuration directives.

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

Default Value:

The http_gzip_module is enabled by default in the source build, and the http_gzip_static_module is not. Only the http_gzip_static_module is enabled by default in the dnf package.

References:




1. <http://nginx.org/en/docs/configure.html>
2. <http://nginx.org/en/docs/configure.html>
3. http://nginx.org/en/docs/http/nginx_http_gzip_module.html

4. http://nginx.org/en/docs/http/nginx_http_gzip_static_module.html

Additional Information:

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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	2.6 Allowlist Authorized Libraries Use technical controls to ensure that only authorized software libraries, such as specific .dll, .ocx, .so, etc., files, are allowed to load into a system process. Block unauthorized libraries from loading into a system process. Reassess bi-annually, or more frequently.			
v7	2.8 Implement Application Whitelisting of Libraries The organization's application whitelisting software must ensure that only authorized software libraries (such as *.dll, *.ocx, *.so, etc) are allowed to load into a system process.			

2.1.4 Ensure the autoindex module is disabled (Automated)

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/nginx_autoindex_module.html

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	2.6 Allowlist Authorized Libraries Use technical controls to ensure that only authorized software libraries, such as specific .dll, .ocx, .so, etc., files, are allowed to load into a system process. Block unauthorized libraries from loading into a system process. Reassess bi-annually, or more frequently.			
v7	2.8 Implement Application Whitelisting of Libraries The organization's application whitelisting software must ensure that only authorized software libraries (such as *.dll, *.ocx, *.so, etc) are allowed to load into a system process.			

2.2 Account Security

2.2.1 Ensure that NGINX is run using a non-privileged, dedicated service account (Automated)

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 -Pi -- '^h*user\h+[\^;\n\r]+\h*;*.*$' /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:

```
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:

```
useradd 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 dnf, the user and group nginx and the account are not privileged.

References:

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

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>5.4 Restrict Administrator Privileges to Dedicated Administrator Accounts</u> Restrict administrator privileges to dedicated administrator accounts on enterprise assets. Conduct general computing activities, such as internet browsing, email, and productivity suite use, from the user's primary, non-privileged account.			
v7	<u>4.3 Ensure the Use of Dedicated Administrative Accounts</u> Ensure that all users with administrative account access use a dedicated or secondary account for elevated activities. This account should only be used for administrative activities and not internet browsing, email, or similar activities.			

2.2.2 Ensure the NGINX service account is locked (Automated)

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.

Impact:

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

Audit:

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

```
passwd -S "$(awk '$1~/^\s*user\s*$/ {print $2}' /etc/nginx/nginx.conf | sed -r 's/;.*//g')"
```

The result should be similar to the following:

```
nginx LK 2022-09-06 -1 -1 -1 -1 (Password locked.)
```

Remediation:







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

```
passwd -l "$(awk '$1~/^\s*user\s*$/ {print $2}' /etc/nginx/nginx.conf | sed -r 's/;.*//g')"
```

Default Value:

The nginx user is locked by default.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.3 <u>Configure Data Access Control Lists</u> Configure data access control lists based on a user's need to know. Apply data access control lists, also known as access permissions, to local and remote file systems, databases, and applications.			
v7	14.6 <u>Protect Information through Access Control Lists</u> Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.			

2.2.3 Ensure the NGINX service account has an invalid shell (Automated)

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:

Run the following script to verify the nginx service account has an invalid shell:

```
#!/usr/bin/env bash

{
    l_output="" l_output2="" l_out=""
    if [ -f /etc/nginx/nginx.conf ]; then
        l_user="$(awk '1~/^\s*user\s*$/ {print $2}' /etc/nginx/nginx.conf |
sed -r 's/;.*//g')"
        l_valid_shells="^($( sed -rn '/^\s*{s/,,\s*\s*,g;p}' /etc/shells | paste
-s -d '|' - ))$"
        l_out="$(awk -v pat="$l_valid_shells" -v ngusr="$l_user" -F: '($ (NF) ~
pat && $1==ngusr) { $(NF-1) }' /etc/passwd)"
        if [ -z "$l_out" ]; then
            l_output=" - NGINX user account: \"$l_user\" has an invalid shell"
        else
            l_output2=" - NGINX user account: \"$l_user\" has a valid shell:
\"$l_out\""
        fi
    else
        l_output2=" - NGINX user account can not be determined.\n - file:
\"/etc/nginx/nginx.conf\" is missing"
    fi
    if [ -z "$l_output2" ]; then
        echo -e "\n- Audit Result:\n  ** PASS **\n$l_output\n"
    else
        echo -e "\n- Audit Result:\n  ** FAIL **\n - Reason(s) for audit
failure:\n$l_output2\n"
    fi
}
```

Remediation:




Change the login shell for the nginx account to /sbin/nologin by using the following command:




```
usermod -s /sbin/nologin nginx
```

Default Value:

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

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	4.1 Establish and Maintain a Secure Configuration Process Establish and maintain a secure configuration process for enterprise assets (end-user devices, including portable and mobile, non-computing/IoT devices, and servers) and software (operating systems and applications). Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			

Controls Version	Control	IG 1	IG 2	IG 3
v7	5.1 <u>Establish Secure Configurations</u> 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 (Automated)

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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.3 <u>Configure Data Access Control Lists</u> Configure data access control lists based on a user's need to know. Apply data access control lists, also known as access permissions, to local and remote file systems, databases, and applications.			

Controls Version	Control	IG 1	IG 2	IG 3
v7	<p>14.6 <u>Protect Information through Access Control Lists</u></p> <p>Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.</p>			

2.3.2 Ensure access to NGINX directories and files is restricted (Automated)

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 write permissions revoked, look at the permissions by running the below command:

```
find /etc/nginx -type d -exec stat -Lc "%n %a" {} +
```

Verify The output directories are mode 755 or more restrictive:

Example

```
/etc/nginx 755
```

To verify the nginx configuration files have other read, write and execute permissions revoked, look at the permissions by running the below command:

```
find /etc/nginx -type f -exec stat -Lc "%n %a" {} +
```

Verify The output files are mode 660 or more restrictive:

Example:

```
/etc/nginx/nginx.conf 660
```

Remediation:

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

Permissions are set with the ability to read and execute as other by default on all directories: drwxr-xr-x

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

```
find /etc/nginx -type d -exec chmod go-w {} +  
find /etc/nginx -type f -exec chmod ug-x,o-rwx {} +
```

Default Value:

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

Permissions are set with the ability to read and execute as other by default on all directories: drwxr-xr-x







References:

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

Additional Information:

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

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.3 <u>Configure Data Access Control Lists</u> Configure data access control lists based on a user's need to know. Apply data access control lists, also known as access permissions, to local and remote file systems, databases, and applications.			
v7	14.6 <u>Protect Information through Access Control Lists</u> Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.			

2.3.3 Ensure the NGINX process ID (PID) file is secured (Automated)

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:

```
stat -L -c "%U:%G" /var/run/nginx.pid && stat -L -c "%a" /var/run/nginx.pid
```

The output should show that the PID file is owned by root and has the group root and that the permissions are 644 as shown below:

```
root:root
644
```

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:

```
chmod u-x,go-wx /var/run/nginx.pid
```

Default Value:

The PID file is owned by root and has permissions 644 by default when building using dnf.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.3 <u>Configure Data Access Control Lists</u> Configure data access control lists based on a user's need to know. Apply data access control lists, also known as access permissions, to local and remote file systems, databases, and applications.			
v7	14.6 <u>Protect Information through Access Control Lists</u> Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.			

2.3.4 Ensure the core dump directory is secured (Manual)

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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.3 <u>Configure Data Access Control Lists</u> Configure data access control lists based on a user's need to know. Apply data access control lists, also known as access permissions, to local and remote file systems, databases, and applications.			
v7	14.6 <u>Protect Information through Access Control Lists</u> Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.			

2.4 Network Configuration

2.4.1 Ensure NGINX only listens for network connections on authorized ports (Manual)

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/nginx.conf.rpmsave:    listen      80;
/etc/nginx/nginx.conf.rpmsave:    listen 443 ssl http2;
/etc/nginx/conf.d/default.conf:   listen      80;
```

Any files included in the `include` directive of the nginx configuration file or the nginx configuration file itself should be checked.

To check which files are included in the nginx configuration file run the following command:

```
grep include /etc/nginx/nginx.conf
```

The output below shows that any file matching the pattern `/etc/nginx/conf.d/*.conf/` in the above audit check should be considered an auditable port.





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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>16.10 Apply Secure Design Principles in Application Architectures</u> Apply secure design principles in application architectures. Secure design principles include the concept of least privilege and enforcing mediation to validate every operation that the user makes, promoting the concept of "never trust user input." Examples include ensuring that explicit error checking is performed and documented for all input, including for size, data type, and acceptable ranges or formats. Secure design also means minimizing the application infrastructure attack surface, such as turning off unprotected ports and services, removing unnecessary programs and files, and renaming or removing default accounts.			
v7	<u>9.2 Ensure Only Approved Ports, Protocols and Services Are Running</u> Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.			

2.4.2 Ensure requests for unknown host names are rejected (Automated)

Profile Applicability:

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

Description:

Your host header should be part of a predefined allowlist 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:

Allowlisting 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.

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.

Audit:

Run the following command to verify this is configured:

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

If you do not receive a 400 series 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;
    ....
}
```







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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	4.1 Establish and Maintain a Secure Configuration Process Establish and maintain a secure configuration process for enterprise assets (end-user devices, including portable and mobile, non-computing/IoT devices, and servers) and software (operating systems and applications). Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	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 (Automated)

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/nginx_http_core_module.html#keepalive_timeout

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	16.1 <u>Establish and Maintain a Secure Application Development Process</u> Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	18.1 <u>Establish Secure Coding Practices</u> Establish secure coding practices appropriate to the programming language and development environment being used.			

2.4.4 Ensure `send_timeout` is set to 10 seconds or less, but not 0 (Automated)

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/nginx_http_core_module.html#send_timeout

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	16.1 <u>Establish and Maintain a Secure Application Development Process</u> Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	18.1 <u>Establish Secure Coding Practices</u> Establish secure coding practices appropriate to the programming language and development environment being used.			

2.5 Information Disclosure

2.5.1 Ensure `server_tokens` directive is set to `off` (Automated)

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/1.22.0
```

Remediation:





To disable the `server_tokens` directive, set it to `off` inside of every server block in your `nginx.conf` or in the `http` block:

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

Default Value:

The default value of `server_tokens` is `on`.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<p><u>16.1 Establish and Maintain a Secure Application Development Process</u></p> <p>Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.</p>			
v7	<p><u>18.1 Establish Secure Coding Practices</u></p> <p>Establish secure coding practices appropriate to the programming language and development environment being used.</p>			

2.5.2 Ensure default error and index.html pages do not reference NGINX (Automated)

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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<p>16.1 <u>Establish and Maintain a Secure Application Development Process</u></p> <p>Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.</p>			
v7	<p>18.1 <u>Establish Secure Coding Practices</u></p> <p>Establish secure coding practices appropriate to the programming language and development environment being used.</p>			

2.5.3 Ensure hidden file serving is disabled (Manual)

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.

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;  
}
```

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; }
```





Default Value:

This is not set by default.

References:

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

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>16.1 Establish and Maintain a Secure Application Development Process</u> Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	<u>18.1 Establish Secure Coding Practices</u> Establish secure coding practices appropriate to the programming language and development environment being used.			

2.5.4 Ensure the NGINX reverse proxy does not enable information disclosure (Automated)

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;  
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/nginx_http_proxy_module.html

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	16.1 <u>Establish and Maintain a Secure Application Development Process</u> Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	18.1 <u>Establish Secure Coding Practices</u> Establish secure coding practices appropriate to the programming language and development environment being used.			

3 Logging

3.1 Ensure detailed logging is enabled (Manual)

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. CIS control 8.5: “Collect Detailed Audit Logs” recommends that you configure detailed audit logging for enterprise assets containing sensitive data. It further recommends you include event source, date, username, timestamp, source addresses, destination addresses, and other useful elements that could assist in a forensic investigation.

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`.

Edit the log format directive in `/etc/nginx/nginx.conf` so it logs everything needed to meet your organizational policies.

The final format should resemble something like the below example:


```

log_format main 'Event Source Information - Server Name: "$server_name"
Server_Protocol: "$server_protocol" Hostname: "$hostname" Host: "$host" '
                'Date and Timestamp Information - Local Time:
"$time_local" ISO8601 Time: "$time_iso8601" '
                'Username Information - Basic Authentication User:
"$remote_user" '
                'Source Address Information - Source Address:Port:
"$remote_addr:$remote_port" '
                'Destination Address Information - Destination
Address:Port "$server_addr:$server_port" '
                'Request Information - Request: "$request" HTTP
Response Status: "$status" Referer: "$http_referer" Content Type:
"$content_type" Body Bytes Sent: "$body_bytes_sent" User Agent:
"$http_user_agent" ';

```

The following variables may be considered as useful examples to 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. The NGINX documentation on supported log variables may be found here: <http://nginx.org/en/docs/varindex.html>

Please review the NGINX documentation for log variables to include in your logs in alignment with your use case and logging policy. Below are some relevant log variables to help focus your review:

Event Source Variables: The event source is the name of the software that logs the event. It may frequently be the name of the application or application component that receives the request.

```

$server_name - Logs the name of the server which accepted a request
$server_protocol - Logs the request protocol, usually "HTTP/1.0", "HTTP/1.1",
or "HTTP/2.0"
$hostname - Logs the hostname
$host - Logs the host name from the request line, or host name from the
"Host" request header field, or the server name matching a request in that
order of precedence.

```

Date & Timestamp Variables: The date and timestamp of the HTTP request used for forensic investigations.

```

$time_local - Logs the local time in the Common Log Format.
$time_iso8601 - Logs the local time in the ISO 8601 standard format.

```

Username Variables:

```

$remote_user - Logs the user name supplied with basic authentication if basic
authentication is used by NGINX. Nothing will be logged if the request is not
part of an application using basic authentication with NGINX.

```

Source Address Variables: Identify the source of the HTTP request

```
$remote_addr - Logs the client IP address of the HTTP request
$remote_port - Logs the client port used in the HTTP requests
```

Destination Address Variables: Identify the destination of the HTTP request.

```
$server_addr - Logs an address of the server which accepted a request
$server_port - Logs the port of the server which accepted a request
```

Other useful elements that could assist in a forensic investigation:

```
$request - Logs an address of the server which accepted a request, which may
be useful in forensic investigations
$status - Logs the response status code of the HTTP request, which may be
useful in forensic investigations
$http_referer - Logs the response status code of the HTTP request, which may
be useful in forensic investigations
$http_user_agent - Logs the user agent of the client behind the request,
which may be useful in forensic investigations
$pid - Logs the process ID of the worker process of NGINX.
```

Remediation:

Edit the log format directive in `/etc/nginx/nginx.conf` so it logs everything needed to meet your organizational policies.

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/nginx_http_log_module.html#log_format
2. <http://nginx.org/en/docs/varindex.html>
3. <https://workbench.cisecurity.org/sections/698300/recommendations/1142990>

Additional Information:

Note: 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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	8.5 <u>Collect Detailed Audit Logs</u> Configure detailed audit logging for enterprise assets containing sensitive data. Include event source, date, username, timestamp, source addresses, destination addresses, and other useful elements that could assist in a forensic investigation.			
v7	6.3 <u>Enable Detailed Logging</u> Enable system logging to include detailed information such as an event source, date, user, timestamp, source addresses, destination addresses, and other useful elements.			

3.2 Ensure access logging is enabled (Manual)

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/nginx_http_log_module.html#log_format

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	8.5 <u>Collect Detailed Audit Logs</u> Configure detailed audit logging for enterprise assets containing sensitive data. Include event source, date, username, timestamp, source addresses, destination addresses, and other useful elements that could assist in a forensic investigation.			
v7	6.3 <u>Enable Detailed Logging</u> Enable system logging to include detailed information such as an 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 (Automated)

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.log info;
```

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	8.5 Collect Detailed Audit Logs Configure detailed audit logging for enterprise assets containing sensitive data. Include event source, date, username, timestamp, source addresses, destination addresses, and other useful elements that could assist in a forensic investigation.			

Controls Version	Control	IG 1	IG 2	IG 3
v7	6.3 <u>Enable Detailed Logging</u> Enable system logging to include detailed information such as an event source, date, user, timestamp, source addresses, destination addresses, and other useful elements.		●	●

3.4 Ensure log files are rotated (Automated)

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
}
```

Additional Information:

You should always comply with your organizational log retention policy.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	8.3 <u>Ensure Adequate Audit Log Storage</u> Ensure that logging destinations maintain adequate storage to comply with the enterprise's audit log management process.			
v7	6.4 <u>Ensure adequate storage for logs</u> 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 (Manual)

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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	8.9 Centralize Audit Logs Centralize, to the extent possible, audit log collection and retention across enterprise assets.		●	●
v7	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 (Manual)

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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	8.9 Centralize Audit Logs Centralize, to the extent possible, audit log collection and retention across enterprise assets.			
v7	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 (Manual)

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/nginx_http_proxy_module.html

Additional Information:

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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	8.11 Conduct Audit Log Reviews Conduct reviews of audit logs to detect anomalies or abnormal events that could indicate a potential threat. Conduct reviews on a weekly, or more frequent, basis.			
v7	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 (Manual)

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.

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.

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 ciscurrency.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 (ciscurrency.org is used as an example server name).

```
server {
    listen 80;

    server_name ciscurrency.org;

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





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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.			

4.1.2 Ensure a trusted certificate and trust chain is installed (Manual)

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 systemctl restart nginx
```





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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 Encrypt Sensitive Data in Transit Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 Encrypt All Sensitive Information in Transit Encrypt all sensitive information in transit.			

4.1.3 Ensure private key permissions are restricted (Automated)

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.

```
find /etc/nginx/ -name '*.key' -exec stat -Lc "%n %a" {} +
```

The output should show mode 400 or more restrictive

Example:

```
/etc/nginx/nginx.key 400
```

Remediation:

Run the following command to remove excessive permissions on key files in the `/etc/nginx/` directory.

Note: The directory `/etc/nginx/` should be replaced with the location of your key file.

```
find /etc/nginx/ -name '*.key' -exec chmod u-wx,go-rwx {} +
```







Default Value:

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

Additional Information:

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

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.3 <u>Configure Data Access Control Lists</u> Configure data access control lists based on a user's need to know. Apply data access control lists, also known as access permissions, to local and remote file systems, databases, and applications.			
v7	14.6 <u>Protect Information through Access Control Lists</u> Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.			

4.1.4 Ensure only modern TLS protocols are used (Automated)

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, and TLS 1.3), 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.

Why enable TLS 1.3: TLS 1.3 improves security by removing several insecure cipher suites by default and adding several more secure algorithms. All public-key exchange mechanisms support perfect forward secrecy in this version of TLS. Additionally, TLS 1.3 makes drastic performance improvements by removing a full round trip in the TLS handshake.

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.

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 TLSv1.3;/"  
/etc/nginx/nginx.conf
```

Proxy:

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

If 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 TLSv1.3;  
}
```

Proxy:

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

Default Value:

By default, NGINX does not specify the TLS protocol and accepts all TLS versions, except for TLS 1.3, which must be enabled by an administrator to take effect.

Defaults: 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/>
3. <https://tools.ietf.org/html/rfc8446>





Additional Information:

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

Compatibility Note: TLS 1.3 will only be available if a version of OpenSSL greater than 1.1.0 is installed on your server. You can check the version of OpenSSL you are using by leveraging the following command:

```
openssl version
```

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.			

4.1.5 Disable weak ciphers (Manual)

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.

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.

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.

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.

No weak ciphers SSLLABS proxy configuration

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

No weak ciphers SSLLABS web server configuration:

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

For changes to take effect, you must recycle nginx:

```
systemctl restart nginx
```

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/nginx_ssl_module.html
5. [https://www.owasp.org/index.php/Testing_for_SSL-TLS %28OWASP-CM-001%29](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/>

Additional Information:

Note: 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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.			

4.1.6 Ensure custom Diffie-Hellman parameters are used (Automated)

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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.			

4.1.7 Ensure Online Certificate Status Protocol (OCSP) stapling is enabled (Automated)

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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).		●	●
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.		●	●

4.1.8 Ensure HTTP Strict Transport Security (HSTS) is enabled (Automated)

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;" always;
```

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;" always;  
}
```





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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.			

4.1.9 Ensure upstream server traffic is authenticated with a client certificate (Automated)

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/nginx_http_proxy_module.html#proxy_ssl_certificate

Additional Information:

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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.			

4.1.10 Ensure the upstream traffic server certificate is trusted (Manual)

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/nginx_http_proxy_module.html#proxy_ssl_trusted_certificate

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.			

4.1.11 Ensure your domain is preloaded (Manual)

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 one year or longer. The header should be modified similar to the below snippet.

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

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





Default Value:

Your website is not preloaded by default.

References:

1. <https://hstspreload.org/>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.			

4.1.12 Ensure session resumption is disabled to enable perfect forward security (Automated)

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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.			

4.1.13 Ensure HTTP/2.0 is used (Automated)

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 denylists. 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/>

Additional Information:

1. HTTP/2.0 is not supported for proxies at the time of the writing of this recommendation.
2. Versions of NGINX prior to 1.17.2 are affected by several HTTP/2 vulnerabilities. More information on these on these vulnerabilities can be found at: <https://www.nginx.com/blog/nginx-updates-mitigate-august-2019-http-2-vulnerabilities/> and http://nginx.org/en/security_advisories.html

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.			

4.1.14 Ensure only Perfect Forward Secrecy Ciphers are Leveraged (Manual)

Profile Applicability:

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

Description:

Perfect forward secrecy protects users of your website by ensuring that even if your private key is compromised that your user's sessions are not able to be compromised. This improves upon other ciphers where if your private key was compromised all user sessions can also be compromised retroactively.

Rationale:

Perfect Forward Secrecy (PFS) helps to reduce the impact of a private key compromise.

Audit:

To check that only PFS ciphers are used to ensure that only ciphers that are ECDHE/EECDH ciphers and DHE/EDH ciphers are enabled. To audit the ciphers setup on NGINX run the below two commands:

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

Remediation:

Ensure that only ciphers that are compatible with perfect forward secrecy are used. ECDHE/EECDH ciphers and DHE/EDH ciphers support this capability. Its recommended to leverage ECDHE ciphers unless you need to support legacy clients because they are considered stronger and faster. An example configuration that may be used is: "EECDH:EDH:!NULL:!SSLv2:!RC4:!aNULL:!3DES:!IDEA".

The below configuration will only enable ciphers compatible with perfect forward secrecy.

Web Server:

```
ssl_ciphers EECDH:EDH:!NULL:!SSLv2:!RC4:!aNULL:!3DES:!IDEA;
```

Proxy:


```
proxy_ssl_ciphers ECDH:EDH:!NULL:!SSLv2:!RC4:!aNULL:!3DES:!IDEA;
```





Default Value:

Perfect Forward Secrecy is not the only default negotiable cipher suite.

References:

1. <https://scotthelme.co.uk/perfect-forward-secrecy/>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	3.10 <u>Encrypt Sensitive Data in Transit</u> Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).			
v7	14.4 <u>Encrypt All Sensitive Information in Transit</u> Encrypt all sensitive information in transit.			

5 Request Filtering and Restrictions

5.1 Access Control

5.1.1 Ensure allow and deny filters limit access to specific IP addresses (Manual)

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 allowlist 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/nginx_http_access_module.html

Additional Information:

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

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	13.10 Perform Application Layer Filtering Perform application layer filtering. Example implementations include a filtering proxy, application layer firewall, or gateway.			●
v7	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 approved HTTP methods are allowed (Manual)

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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<p><u>16.10 Apply Secure Design Principles in Application Architectures</u></p> <p>Apply secure design principles in application architectures. Secure design principles include the concept of least privilege and enforcing mediation to validate every operation that the user makes, promoting the concept of "never trust user input." Examples include ensuring that explicit error checking is performed and documented for all input, including for size, data type, and acceptable ranges or formats. Secure design also means minimizing the application infrastructure attack surface, such as turning off unprotected ports and services, removing unnecessary programs and files, and renaming or removing default accounts.</p>			
v7	<p><u>9.2 Ensure Only Approved Ports, Protocols and Services Are Running</u></p> <p>Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.</p>			

5.2 Request Limits

5.2.1 Ensure timeout values for reading the client header and body are set correctly (Automated)

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>
3. http://nginx.org/en/docs/http/nginx_http_core_module.html#client_header_timeout

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	16.1 <u>Establish and Maintain a Secure Application Development Process</u> Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	18.1 <u>Establish Secure Coding Practices</u> Establish secure coding practices appropriate to the programming language and development environment being used.			

5.2.2 Ensure the maximum request body size is set correctly (Automated)

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/nginx_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/>
5. http://nginx.org/en/docs/http/nginx_http_core_module.html#client_max_body_size

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<p>16.1 <u>Establish and Maintain a Secure Application Development Process</u></p> <p>Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.</p>			
v7	<p>18.1 <u>Establish Secure Coding Practices</u></p> <p>Establish secure coding practices appropriate to the programming language and development environment being used.</p>			

5.2.3 Ensure the maximum buffer size for URIs is defined (Automated)

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

Additional Information:

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

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<u>16.1 Establish and Maintain a Secure Application Development Process</u> Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	<u>18.1 Establish Secure Coding Practices</u> Establish secure coding practices appropriate to the programming language and development environment being used.			

5.2.4 Ensure the number of connections per IP address is limited (Manual)

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.

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.

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;
    }
}
```

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/nginx_http_limit_conn_module.html
3. <https://scotthelme.co.uk/mitigating-http-get-dos-attack/>

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	16.1 <u>Establish and Maintain a Secure Application Development Process</u> Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.		●	●
v7	18.1 <u>Establish Secure Coding Practices</u> Establish secure coding practices appropriate to the programming language and development environment being used.		●	●

5.2.5 Ensure rate limits by IP address are set (Manual)

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.

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.

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;
        }
    }
}

```





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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	16.1 <u>Establish and Maintain a Secure Application Development Process</u> Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	18.1 <u>Establish Secure Coding Practices</u> Establish secure coding practices appropriate to the programming language and development environment being used.			

5.3 Browser Security

5.3.1 Ensure X-Frame-Options header is configured and enabled (Automated)

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.

Impact:

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

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" always;
```

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" always;
```

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

Additional Information:

Important Note: The configuration in this recommendation recommends that this header is set to the same origin; however, this does not mean that if it is not set so, that this is done incorrectly. This header may also be configured to allow from specific origins or deny from all origins. The always parameter ensures that this header is applied no matter what the http response code is.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<p>16.1 Establish and Maintain a Secure Application Development Process</p> <p>Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.</p>			
v7	<p>18.1 Establish Secure Coding Practices</p> <p>Establish secure coding practices appropriate to the programming language and development environment being used.</p>			

5.3.2 Ensure X-Content-Type-Options header is configured and enabled (Automated)

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" always;
```

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" always;
```

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>

Additional Information:

The always parameter ensures that the header is applied to the HTTP response no matter what the HTTP response code is.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	<p>16.1 <u>Establish and Maintain a Secure Application Development Process</u></p> <p>Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.</p>			
v7	<p>18.1 <u>Establish Secure Coding Practices</u></p> <p>Establish secure coding practices appropriate to the programming language and development environment being used.</p>			

5.3.3 Ensure that Content Security Policy (CSP) is enabled and configured properly (Manual)

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'" always;
```

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'" always;
```

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/>

Additional Information:

Important Note: Content Security Policy may be customized for a significant number of use cases, including upgrading insecure requests, directing the origin of executables, and reporting violations of the policy. This is a simplistic version that does not go into the depth of what a 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. The always parameter ensures that the header is returned in the response no matter what the HTTP response code is.

Note: 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:

Controls Version	Control	IG 1	IG 2	IG 3
v8	2.7 <u>Allowlist Authorized Scripts</u> Use technical controls, such as digital signatures and version control, to ensure that only authorized scripts, such as specific .ps1, .py, etc., files, are allowed to execute. Block unauthorized scripts from executing. Reassess bi-annually, or more frequently.			●
v7	2.9 <u>Implement Application Whitelisting of Scripts</u> The organization's application whitelisting software must ensure that only authorized, digitally signed scripts (such as *.ps1, *.py, macros, etc) are allowed to run on a system.			●

5.3.4 Ensure the Referrer Policy is enabled and configured properly (Manual)

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/>

Additional Information:

Important Note: This header should be customized to your application and "no-referrer" is not always appropriate. This may vary depending on your circumstances. The always parameter ensures that the header is returned in the response no matter what the HTTP response code is.

CIS Controls:

Controls Version	Control	IG 1	IG 2	IG 3
v8	16.1 <u>Establish and Maintain a Secure Application Development Process</u> Establish and maintain a secure application development process. In the process, address such items as: secure application design standards, secure coding practices, developer training, vulnerability management, security of third-party code, and application security testing procedures. Review and update documentation annually, or when significant enterprise changes occur that could impact this Safeguard.			
v7	18.1 <u>Establish Secure Coding Practices</u> Establish secure coding practices appropriate to the programming language and development environment being used.			

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

CIS Benchmark Recommendation		Set Correctly	
		Yes	No
1	Initial Setup		
1.1	Installation		
1.1.1	Ensure NGINX is installed (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
1.1.2	Ensure NGINX is installed from source (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
1.2	Configure Software Updates		
1.2.1	Ensure package manager repositories are properly configured (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
1.2.2	Ensure the latest software package is installed (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
2	Basic Configuration		
2.1	Minimize NGINX Modules		
2.1.1	Ensure only required modules are installed (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
2.1.2	Ensure HTTP WebDAV module is not installed (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
2.1.3	Ensure modules with gzip functionality are disabled (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
2.1.4	Ensure the autoindex module is disabled (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
2.2	Account Security		
2.2.1	Ensure that NGINX is run using a non-privileged, dedicated service account (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
2.2.2	Ensure the NGINX service account is locked (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
2.2.3	Ensure the NGINX service account has an invalid shell (Automated)	<input type="checkbox"/>	<input type="checkbox"/>

CIS Benchmark Recommendation		Set Correctly	
		Yes	No
2.3	Permissions and Ownership		
2.3.1	Ensure NGINX directories and files are owned by root (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
2.3.2	Ensure access to NGINX directories and files is restricted (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
2.3.3	Ensure the NGINX process ID (PID) file is secured (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
2.3.4	Ensure the core dump directory is secured (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
2.4	Network Configuration		
2.4.1	Ensure NGINX only listens for network connections on authorized ports (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
2.4.2	Ensure requests for unknown host names are rejected (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
2.4.3	Ensure keepalive_timeout is 10 seconds or less, but not 0 (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
2.4.4	Ensure send_timeout is set to 10 seconds or less, but not 0 (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
2.5	Information Disclosure		
2.5.1	Ensure server_tokens directive is set to `off` (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
2.5.2	Ensure default error and index.html pages do not reference NGINX (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
2.5.3	Ensure hidden file serving is disabled (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
2.5.4	Ensure the NGINX reverse proxy does not enable information disclosure (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
3	Logging		

CIS Benchmark Recommendation		Set Correctly	
		Yes	No
3.1	Ensure detailed logging is enabled (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
3.2	Ensure access logging is enabled (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
3.3	Ensure error logging is enabled and set to the info logging level (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
3.4	Ensure log files are rotated (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
3.5	Ensure error logs are sent to a remote syslog server (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
3.6	Ensure access logs are sent to a remote syslog server (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
3.7	Ensure proxies pass source IP information (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
4	Encryption		
4.1	TLS / SSL Configuration		
4.1.1	Ensure HTTP is redirected to HTTPS (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
4.1.2	Ensure a trusted certificate and trust chain is installed (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
4.1.3	Ensure private key permissions are restricted (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
4.1.4	Ensure only modern TLS protocols are used (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
4.1.5	Disable weak ciphers (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
4.1.6	Ensure custom Diffie-Hellman parameters are used (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
4.1.7	Ensure Online Certificate Status Protocol (OCSP) stapling is enabled (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
4.1.8	Ensure HTTP Strict Transport Security (HSTS) is enabled (Automated)	<input type="checkbox"/>	<input type="checkbox"/>

CIS Benchmark Recommendation		Set Correctly	
		Yes	No
4.1.9	Ensure upstream server traffic is authenticated with a client certificate (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
4.1.10	Ensure the upstream traffic server certificate is trusted (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
4.1.11	Ensure your domain is preloaded (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
4.1.12	Ensure session resumption is disabled to enable perfect forward security (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
4.1.13	Ensure HTTP/2.0 is used (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
4.1.14	Ensure only Perfect Forward Secrecy Ciphers are Leveraged (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
5	Request Filtering and Restrictions		
5.1	Access Control		
5.1.1	Ensure allow and deny filters limit access to specific IP addresses (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
5.1.2	Ensure only approved HTTP methods are allowed (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
5.2	Request Limits		
5.2.1	Ensure timeout values for reading the client header and body are set correctly (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
5.2.2	Ensure the maximum request body size is set correctly (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
5.2.3	Ensure the maximum buffer size for URIs is defined (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
5.2.4	Ensure the number of connections per IP address is limited (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
5.2.5	Ensure rate limits by IP address are set (Manual)	<input type="checkbox"/>	<input type="checkbox"/>

CIS Benchmark Recommendation		Set Correctly	
		Yes	No
5.3	Browser Security		
5.3.1	Ensure X-Frame-Options header is configured and enabled (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
5.3.2	Ensure X-Content-Type-Options header is configured and enabled (Automated)	<input type="checkbox"/>	<input type="checkbox"/>
5.3.3	Ensure that Content Security Policy (CSP) is enabled and configured properly (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
5.3.4	Ensure the Referrer Policy is enabled and configured properly (Manual)	<input type="checkbox"/>	<input type="checkbox"/>
6	Mandatory Access Control		

Appendix: Change History

Date	Version	Changes for this version
Oct 13, 2017		_Default Value, Remediation Procedure, Listing Order, Section_ on **[Recommendation] 4.2 Give the NGINX User Account an Invalid Shell** were updated.
Nov 10, 2018	1.0.0	_Listing Order_ on **[recommendation] 2.2.2P Ensure the NGINX service account is locked** was updated.
Feb 15, 2018	1.0.0	**[recommendation] 1.1.1P Installing Nginx** was created.
Nov 10, 2018	1.0.0	_Listing Order, Status, references, Default Value, Impact Statement, Audit Procedure, Remediation Procedure, Rationale Statement, Description, Scoring Status, Title_ on **[recommendation] 4.1.10 Ensure a Trusted Certificate and Trust Chain is Installed**

Date	Version	Changes for this version
Nov 10, 2018	1.0.0	_Listing Order, Status, references, Default Value, Impact Statement, Audit Procedure, Remediation Procedure, Rationale Statement, Description, Scoring Status, Title_ on **[recommendation] 4.1.11 Ensure Permissions on Servers Private Key is Protected** wer
Nov 11, 2018	1.0.0	**[section] 9 Ensure session resumption is enabled to improve https performance** was created.
Nov 11, 2018	1.0.0	**[recommendation] 4.1.4P Ensure Permissions on Servers Private Key is Protected** was created.
Nov 13, 2018	1.0.0	**[recommendation] 4.1.4P Restrict Weak Ciphers From Being Used** was created.
Dec 10, 2018	1.0.0	_Listing Order_ on **[recommendation] 1.2.1P Ensure the Latest Software Package is Installed** was updated.
Dec 10, 2018	1.0.0	_Listing Order_ on **[recommendation] 1.1.1P Installing NGINX** was updated.
Dec 10, 2018	1.0.0	_Listing Order_ on **[recommendation] 1.2.1P Ensure Package Manager Repositories are Configured** was updated.

Date	Version	Changes for this version
Dec 10, 2018	1.0.0	_Listing Order, Rationale Statement, Description_ on **[recommendation] 2.5.2P Ensure server_tokens directive is set to Off** were updated.
Dec 10, 2018	1.0.0	_Listing Order_ on **[recommendation] 2.5.2P Ensure Default Error Pages and index.html Pages don't reference NGINX** was updated.
Dec 10, 2018	1.0.0	_Listing Order_ on **[recommendation] 2.5.2P Ensure Hidden File Serving Is Disabled** was updated.
Dec 10, 2018	1.0.0	_Listing Order_ on **[recommendation] 2.5.2P Ensure Proxy does not Enable Information Disclosure** was updated.
Dec 10, 2018	1.0.0	_Listing Order, Status_ on **[recommendation] 2.1.5 New recommendation being proposed by jscott9321** were updated.
Jan 7, 2019	1.0.0	_Listing Order_ on **[recommendation] 1.1.1P Installing NGINX** was updated.
Sep 8, 2022	2.0.0	Update to NGINX Install instructions to show 16.1 (Ticket 9029)
Sep 8, 2022	2.0.0	No longer score this as the industry deprecates it. (Ticket 9030)

Date	Version	Changes for this version
Sep 19, 2022	2.0.0	Advice for <code>ssl_prefer_server_ciphers</code> is outdated (Ticket 12709)
Oct 3, 2022	2.0.0	Update Artifact (Ticket 16386)
Oct 3, 2022	2.0.0	Update Artifact (Ticket 16388)
Oct 3, 2022	2.0.0	Manual or Automated (Ticket 16389)
Oct 7, 2022	2.0.0	Verify the Username (Ticket 16390)
Oct 11, 2022	2.0.0	UPDATE - 2.3.2 Ensure access to NGINX directories and files is restricted (Ticket 16392)
Oct 12, 2022	2.0.0	Question about using Error code 404 in 2.4.2 Ensure requests for unknown host names are rejected (Ticket 15400)
Oct 12, 2022	2.0.0	Update prose (Ticket 16398)
Oct 12, 2022	2.0.0	Remove SHA from recommended hash algorithms in cipher suites. (Ticket 16405)
Oct 13, 2022	2.0.0	Review these links with community on next update (Ticket 12371)
Oct 28, 2022	2.0.0	False positive on Centos 8, zabbix appliance (Ticket 12389)
Oct 28, 2022	2.0.0	NGINX False Positives on Debian 9 (Ticket 10847)

Date	Version	Changes for this version
Oct 28, 2022	2.0.0	NGINX benchmark reports false positives on Ubuntu 18.04 (Ticket 10791)
Oct 28, 2022	2.0.0	Can't forget OWASP Mod Security (Ticket 7397)
June 15, 2023	2.0.1	Addressed AAC bug and republished as v2.0.1