CIS Benchmarks

CIS ISC BIND DNS Server 9.11 Benchmark - ARCHIVE

v1.0.0 - 10-23-2020

Terms of Use

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

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

Table of Contents

Overview

This document is intended for system and application administrators, security specialists, auditors, help desk, and platform deployment personnel who plan to develop, deploy, assess, or secure solutions that incorporate ISC (Internet Systems Consortium) BIND (Berkeley Internet Name Domain) DNS Server 9.11 running on Linux.

There are several environment variables defined to identify the BIND configuration files and directory paths which may differ for each installation. The variables are referenced by audit and remediation steps in order to make the benchmark as independent of installation specifics as reasonable. The directory paths should not include a trailing slash after the directory name.

- `$CONFIG_FILES` – List of the primary configuration file and all included configuration files. Typically, `/etc/named.conf` and other included files. A recursive search for the "include" directive should locate all configuration files.
- `$ZONE_FILES` – All zone files referenced in the configuration files regardless of type.
- `$BIND_HOME` - Directory under which BIND runs, typically `/var/named` or a `chrooted` equivalent.
- `$RUNDIR` – Directory for temporary run time files, typically `/var/run/named`, `/run/named` or a `chrooted` equivalent.
- `$DYNDIR` – Directory for managed keys which are dynamically updated. Typically, `/var/named/dynamic` or a `chrooted` equivalent.
- `$SLAVEDIR` – Directory for dynamically updated slave zone files. Typically, `/var/named/slaves`.
- `$DATADIR` – Directory for run time statistics.
- `$LOGDIR` – Directory for log files. Typically, `/var/named/slaves`
- `$TMPDIR` – Directory for temporary files. Typically, `/tmp`
- `$KEYDIR` – Directory for signing key files.

Intended Audience

This document, CIS ISC BIND DNS Server Benchmark, provides prescriptive guidance for establishing a secure configuration posture for the ISC BIND DNS Server versions 9.11 running on Linux. This guide was tested using BIND version 9.11 installed from rpm packages on CentOS Linux 8.1. To obtain the latest version of this guide, please visit http://benchmarks.cisecurity.org. If you have questions, comments, or have identified ways to improve this guide, please write us at feedback@cisecurity.org.

Consensus Guidance

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

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

Typographical Conventions

The following typographical conventions are used throughout this guide:

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

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 Definitions

The following configuration profiles are defined by this Benchmark:

- **Authoritative Name Server Level 1**

  ISC BIND configured to be a master or slave authoritative name server, the server is authoritative for one or more domains. The name server is configured to not answer queries for any other domains for which it is not authoritative.

  Level 1 profiles intend to:

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

- **Caching Only Name Server Level 1**

  A caching only name server is not authoritative for any domain, but provides DNS service for a limited and well-defined set clients and systems. The Caching Only Name Server will perform recursive DNS queries on behalf of its clients, and will cache answers to improve performance.

  Level 1 profiles intend to:

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

- **Caching Only Name Server Level 2**

  A caching only name server is not authoritative for any domain, but provides DNS service for a limited and well-defined set clients and systems. The Caching Only Name Server will perform recursive DNS queries on behalf of its clients, and will cache answers to improve performance.

  Level 2 profiles 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.

- **Authoritative Name Server Level 2**

  ISC BIND configured to be a master or slave authoritative name server, the server is authoritative for one or more domains. The name server is configured to not answer queries for any other domains for which it is not authoritative.

  Level 2 profiles extends the "Level 1" profile. Items in this profile exhibit one or more of the following characteristics:

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

Acknowledgements

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

Editor
Tim Harrison, Center for Internet Security
Ralph Durkee GXPN, CISSP, GSEC, GCIH, GSNA, GPEN, C|EH, Durkee Consulting, Inc.

Recommendations

1 Planning and Architecture

DNS name servers are a foundational part of your network architecture. How many name servers you need and what roles they should play depends on your organization's network architecture. For this reason, it is critical that the DNS strategy be considered early on, while decisions about the network topology are being formed. Questions that should be answered include, "how is the e-mail going to be delivered?", "are there going to be DNS sub-domains for the organization?", "is DHCP going to be used?", and "is Microsoft Windows Active Directory going to be used?" Providing the detailed information needed to make a recommendation for every possible DNS architecture is beyond the scope of this CIS Benchmark. However, some important DNS architectural recommendations and principles are discussed in this section.

1.1 Use a Split-Horizon Architecture (Manual)

Profile Applicability:

- Authoritative Name Server Level 1

- Caching Only Name Server Level 1

Description:

Running a Split-Horizon DNS architecture refers to running authoritative DNS servers and services for external DNS queries separate from the internal authoritative DNS servers, which answer all queries originating from within the organization. The external servers are configured to provide only a limited amount of information for the services needed for communication with external clients and services. Typically, the information published in the externally available DNS is the minimal needed for the Internet services such as email, web and gateway systems such as VPNs. The separate internal DNS service typically provides a richer information set typically needed by internal clients. Likewise, names servers should physically and logically separated fit only one of the benchmark profiles, and not both:

- Authoritative Name Server
- Caching Only Name Server

Rationale:

The three goals of Split-Horizon are to:

1. Minimize the amount and type of externally available information.
2. Physical and logical separation of external and internal DNS services.
3. Servers roles are either an Authoritative Name Server, or a Caching Recursive Resolver, but not both.

Separating the external and internal DNS servers in this manner adheres to a defense-in-depth approach that limits the potential damage and impact should the external name server be compromised, since it does not service internal clients, nor does it have information on the internal systems and services.

BIND 9 Views can be used to provide different responses based on the source IP address, and have been suggested by some as a means to implement split-horizon without having to separate the internal and external servers. However, the usage of views without separating the servers does not accomplish the second goal. In addition, the usage of views often erroneously assumes that source IP addresses are a reliable security control and cannot be spoofed. Therefore, it is necessary that the internal DNS server be located internally in a way that firewalls and other network controls will ensure external malicious queries will not reach the internal server.

Audit:

Perform the following to audit for compliance:
Review the network and DNS architecture, and identify the external authoritative DNS servers along with the internal authoritative DNS servers to ensure they are separate and serve only external and only internal clients respectively. Review the external DNZ zones to ensure only minimal name information is included in the external zones. Perform a query of an internal only name to the external DNS servers to ensure they do not provide a positive response.

Remediation:

Perform the following for remediation:
Implement Split-Horizon Architecture to separate external and internal DNS services. The external DNS servers should respond only to names of approved external services, such as web, email and VPN services.

Default Value:

Not Applicable

References:

1. https://www.sans.org/reading-room/whitepapers/dns/current-issues-dns-32988
2. http://www.deer-run.com/~hal/EUGLUGBINDTalk.pdf

CIS Controls:

Version 6

12 Boundary Defense
Boundary Defense

Version 7

12 Boundary Defense
Boundary Defense

1.2 Do Not Install a Multi-Use System (Manual)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

Default server configurations often expose a wide variety of services unnecessarily increasing the risk to the system. Just because a server can perform many services doesn't mean it is wise to do so. The number of services and daemons executing on the ISC BIND DNS server should be limited to those necessary, with the DNS service being the only primary function of the server.

Rationale:

Maintaining a server for a single purpose increases the security of your system. The more services which are exposed to an attacker, the more potential vectors an attacker has to exploit the system and therefore the higher the risk for the server. A DNS server should function as only a name server and should not be mixed with other primary functions such as email, web, or database.

Audit:

Leverage the package or services manager for your OS to list enabled services and review to ensure each service is necessary and has a documented business need. On Red Hat systems, the following commands will produce the list of current services enabled. The `systemctl` command lists any `systemd` based services, which are common on current Linux systems while `chkconfig` will list any older traditional SysV based services.

```
# systemctl -t service --state enabled list-unit-files
# chkconfig --list | grep ':on'
```

Remediation:

Disable all unnecessary services or move necessary primary services other than DNS to another server. Leverage the package or services manager for your OS to uninstall or disable unneeded services. On Red Hat systems, the following commands may be used to uninstall a package or disable a service:

```
# yum erase <package name>
# systemctl disable <service name>
```

Default Value:

Depends on the platform

CIS Controls:

Version 6

9.5 Operate Critical Services On Dedicated Hosts (i.e. DNS, Mail, Web, Database)
Operate critical services on separate physical or logical host machines, such as DNS, file, mail, web, and database servers.

Version 7

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

1.3 Dedicated Name Server Role (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

A name server may be an authoritative name server for one or more domains for which it is configured to provide information. An authoritative-only name server only answers queries on the domains for which it is configured, and will reject queries for other domains. A caching name server will answer queries any domain. The caching name server gets answers by sending recursive DNS queries to other name servers and then storing the answer in its cache to provide a quicker response to the next query for that name. A caching-only name server is not authoritative for any domain. The BIND DNS names server should be configured to be either a caching-only or an authoritative-only name server, but not both.

Rationale:

DNS name servers are a foundational part of your network architecture and the security of other network services depend on their integrity. It is important to separate the roles of caching and authoritative name servers to minimize functionality and reduce risk for each server. Each name server role faces different threats in addition to direct attacks on the server. For example, the caching name server faces unique threats of malicious replies with bogus answers or over-sized answers intended to deny service. The authoritative name server is a critical part of the infrastructure should not be exposed to these additional attacks.

Audit:

Authoritative-Only Name Server:
To audit an authoritative name server, ensure it doesn't answer queries for other non-authoritative domain names. The following command may be run on a Linux system other than the name server to verify the query status is refused.

```
$ dig @<ip_address_of_nameserver> <non_authoritative_name> | grep status
;; ->>HEADER<<- opcode: QUERY, status: REFUSED, id: 52410
```

Alternatively, the BIND configuration file `name.conf` and all included configuration files may be searched for the allow-recursion statement. No configured occurrences other than localhost and local loopback should be found.

```
# grep allow-recursion $CONFIG_FILES
named.conf: allow-recursion { 127.0.0.1; };
```

Caching-Only Name Server:
To audit a caching-only name server, send a query with the no recursion specified for a valid authoritative name in the organization.

```
$ dig +norecurse @<ip_address_of_nameserver> <authoritative_name> | grep
status
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 12379
```

Alternatively, the BIND configuration file `name.conf` and all included configuration files may be searched for the zone statements. Zone that contain a local host name or a local loopback IP addresses should be allowed.

```
# grep -w zone $CONFIG_FILES

named.rfc1912.zones:zone "localhost.localdomain" IN {
named.rfc1912.zones:zone "localhost" IN {
named.rfc1912.zones:zone
"1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.ip6.arpa" IN
{
named.rfc1912.zones:zone "1.0.0.127.in-addr.arpa" IN {
named.rfc1912.zones:zone "0.in-addr.arpa" IN {
```

Remediation:

Authoritative-Only Name Server:
For the authoritative-only name server add or modify the allow-recursion statement to only include the localhost to as shown below, or add a recursion statement with a value of *no* as shown below.

```
options {
. . .
allow-rescursion { local; };
```

or

```
recursion no;
```

Caching-Only Name Server:
For the caching-only name server remove the non-local zone statements from the configuration file and restart the server.

CIS Controls:

Version 6

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

Version 7

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

1.4 Use Secure Upstream Caching DNS Servers (Manual)

Profile Applicability:

- Caching Only Name Server Level 2

Description:

Caching name servers often forward queries to another caching name server to allow the name service work to be aggregated and improve performance by taking advantage of the cache of an upstream name server. The default caching name server provided by the Internet service provider is often used in this manner. This may also be a security weakness by relying on insecure servers outside the organization's control and security policies.

Rationale:

The security of all of the external connections that your systems on your network depend in part on getting accurate IP addresses for external names. If the upstream caching name server is compromised, or has its cache poisoned with malicious records, then your entire network may be subject to an attack which may redirect web, email, or VPN traffic to malicious servers, or may cause denial of services attacks. Therefore, it is important to evaluate the security of the upstream caching name servers to reduce the risk of DNS attacks propagated to your network via the upstream provider. There are a number of security companies that offer secure caching DNS services that are worth considering. Features to look for and test include:

- Blocking of traffic to websites known to contain malware.
- Configurable categories for blocking inappropriate content, such as adult content.
- Detecting and blocking of malware communications to an external command and control server.
- Prevent DNS spoofing by ensuring the integrity and authenticity of all DNS responses.

Audit:

Perform the following for an audit:

- Check the network architecture and the identify all internal authorized caching DNS servers configured via DHCP or statically.
- Verify that there are network firewall and access controls rules that prevent internal systems from sending DNS queries directly to unauthorized external DNS servers. Only the authorized internal DNS servers should be allowed to send external DNS

queries and they should be configured to only use authorized external DNS servers. An example direct DNS query on a Microsoft Windows system can be done via `nslookup`, and should timeout as shown below.

```
C:\> nslookup cisecuritry.org 8.8.8.8
DNS request timed out.
timeout was 2 seconds.
Server: UnKnown
Address: 8.8.8.8
. . .
```

- Review the configuration of DNS forwarders for internal caching DNS servers to ensure the that only authorized DNS servers are configured. The following perl command maybe helpful in extracting forwarders directives from the configuration files.

```
perl -ne 'print if /^ *forwarders */i .. / *};/i' $CONFIG_FILES
```

- Review the service provider's agreements, policies and statements, or speak with the vendor and consider if the security of the approved external DNS servers, meet your organizations security standards and requirement. The following features and risk mitigations are recommended for consideration:
  - Prevent spoofing of external DNS replies to redirect traffic to malicious server
  - Prevent spoofing of DNS queries to solicit large DNS replies to perform a denial of service.
  - Blocking of known malicious or infected websites
  - Blocking known botnet C&C communications.
  - Reporting, alerting and configuration on blocked DNS traffic.

Remediation:

Perform the following for remediation:

- Select an external DNS provider that sufficiently mitigates malicious DNS traffic to meet your organizational requirements.
- Review network architectural, approved internal DNS servers, and prepare to block outbound DNS traffic, except to the approved DNS servers from the internal caching name servers.
- Review, test and document the approved external DNS servers.
- Configure the internal caching-only DNS servers to forward queries to the approved external caching DNS server. The forwarders directive similar to the example below may be placed in the server options directive.

```
        forwarders { acl_of_approved_servers; };
```

- Block outbound DNS traffic, except to the approved external DNS servers from the internal caching name servers.

CIS Controls:

Version 6

12 Boundary Defense
Boundary Defense

Version 7

7.4 Maintain and Enforce Network-Based URL Filters
Enforce network-based URL filters that limit a system's ability to connect to websites not approved by the organization. This filtering shall be enforced for each of the organization's systems, whether they are physically at an organization's facilities or not.

7.7 Use of DNS Filtering Services
Use DNS filtering services to help block access to known malicious domains.

1.5 Installing ISC BIND 9 (Automated)

Profile Applicability:

- Authoritative Name Server Level 1

- Caching Only Name Server Level 1

Description:

The ISC BIND Benchmark recommends using the binary packages provided by your platform vendor for most situations in order to reduce the effort and increase the effectiveness of maintenance and security patches. Red Hat Enterprise Linux 7 and 8 have been used for testing the benchmark.

Rationale:

The benefits of using the vendor supplied binaries include:

- Ease of installation.
- It is customized for your OS environment.
- It will be tested and have gone through QA procedures.
- Additional software you may need is likely to be included, such as `chroot` setup and startup scripts.
- Your vendor will tell you about security issues so you have to look in less places.
- Updates to fix security issues will be easier to apply.

However, building from source is suitable for those that want full control of the build process, prefer to build from source, or do not have a suitable package available for their platform. Source download and build information is available on the ISC website knowledge base at the URL reference below.

Audit:

Perform the following commands to check for an installed BIND rpm and to search the current path for the named executable, and to verify the version of bind.

```
# rpm -q bind
bind-9.11.-xx.xx.xx.xx

# which named
/sbin/named

# /sbin/named -v
BIND 9.11.4-xxxxx
```

Remediation:

Installation depends on the operating system platform. The following commands were tested on RHEL7 and RHEL8. On RHEL8 the yum command redirects to the newer dfm command.

```
# yum install bind
. . .
# yum install bind-chroot
. . .
```

References:

1. https://kb.isc.org/article/AA-00768/0/Getting-started-with-BIND-how-to-build-and-run-named-with-a-basic-recursive-configuration.html
2. https://www.isc.org/download/

CIS Controls:

Version 6

2 Inventory of Authorized and Unauthorized Software
Inventory of Authorized and Unauthorized Software

Version 7

2.1 Maintain Inventory of Authorized Software
Maintain an up-to-date list of all authorized software that is required in the enterprise for any business purpose on any business system.

2.2 Ensure Software is Supported by Vendor
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.

2 Restricting Permissions and Ownership

Security at the operating system (OS) level is the vital foundation required for a secure server. This section will focus on restricting platform permissions and privileges for the BIND DNS server.

2.1 Run BIND as a non-root User (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

To start BIND you must execute it as the root user. After the initial startup, BIND has the ability to change to a non-root user, allowing it to drop the root privileges.

Rationale:

The reason for configuring BIND to run as a non-root user is to limit the impact in case a future vulnerability is discovered and exploited. This is a common practice, which implements the principal of least privilege. This principle states that an entity, such as a service or user, should be granted only those specific privileges necessary to perform authorized actions. The server will still need to be started as root, but it should be configured to give up the root privilege after listening on port 53. The user ID under which named runs, needs to be created if it does not already exist and needs appropriate access to the DNS configuration and data files. Many systems including Red Hat Linux will come with a named user already created. Usage of the user and group id of 53 in the examples is arbitrary but is intended to be easier to recognize as it matches the listening port number.

Audit:

Perform the following to ensure the `named` account exists and has an appropriate non-root and non-user UID, and the `–u named` parameter is passed to `named` on startup.

1. Run the following commands to ensure the named account exists, and has been created with a UID greater than zero and less than `MIN_UID`.

```
# id named
uid=53(named) gid=53(named) groups=53(named)
```

```
# grep '^UID_MIN' /etc/login.defs
UID_MIN 1000
```

2. Verify that named service has been started and that the `-u named` option was passed when the daemon was executed, and that the user (first column) is equal to `named`.

```
# ps axu | grep named | grep -v 'grep'
named 423 0.0 1.0 236472 20512 ? Ssl Aug22 2:46 /usr/sbin/named -u
named -t /var/named/chroot
```

Remediation:

Create the named user and group if it does not already exist. Using a shell of /dev/null is best practice.

```
if ! id named; then
 groupadd -g 53 named
 useradd -m -u 53 -g 53 -c "BIND named" -d /var/named -s /dev/null named
fi 2>/dev/null
```

Add the `-u named` to the `OPTIONS` parameters in the `/etc/sysconfig/named` if not already present.

Default Value:

The default named startup parameters include the `-u named` value.

CIS Controls:

Version 6

5.1 Minimize And Sparingly Use Administrative Privileges
Minimize administrative privileges and only use administrative accounts when they are required. Implement focused auditing on the use of administrative privileged functions and monitor for anomalous behavior.

Version 7

4 Controlled Use of Administrative Privileges
Controlled Use of Administrative Privileges

2.2 Give the BIND User Account an Invalid Shell (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

The BIND user account, named by default, must not be used as a regular login account, and should be assigned an invalid or `nologin` shell to ensure that the account cannot be used to login.

Rationale:

Service accounts such as the `named` account represent a risk if they can be used to get a login shell to the system.

Audit:

Check the `named` login shell in the `/etc/passwd` file:

```
# grep named /etc/passwd
named:x:25:25:Named:/var/named:/sbin/nologin
```

The named account shell must be `/sbin/nologin` or `/bin/false` or `/dev/null` similar to the entry shown

Remediation:

Change the named account to use the `nologin` shell as shown:

```
# chsh -s /sbin/nologin named
```

Default Value:

`/bin/false`

CIS Controls:

Version 6

16 Account Monitoring and Control
Account Monitoring and Control

Version 7

4.7 Limit Access to Script Tools

Limit access to scripting tools (such as Microsoft PowerShell and Python) to only administrative or development users with the need to access those capabilities.

2.3 Lock the BIND User Account (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

The user account under which BIND runs should not have a valid password, but should be locked.

Rationale:

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

Audit:

Ensure the named account is locked using the following:

```
# passwd -S named
named LK 2020-02-11 -1 -1 -1 -1 (Password locked.)
```

The results must have the LK value in the second field, similar to the output shown above.

Remediation:

To remediate, lock the named account using the password command with the lock option as shown below.

```
# passwd -l named
Locking password for user named.
passwd: Success
```

Default Value:

Account is locked by default.

CIS Controls:

Version 6

16 Account Monitoring and Control
Account Monitoring and Control

2.4 Set root Ownership of BIND Directories (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

All of the directories under which ISC BIND runs should be owned by root. Of course, any files created at run time by BIND will still be owned by named, and the run time files will need to be in a directory writable by the named group.

Rationale:

Restricting ownership of the directories provides defense in depth and will reduce the probability of unauthorized modifications to those resources. If there was a BIND vulnerability that allowed code execution as the named user, then the code would not be able to modify permissions on the BIND directories owned by root.

Audit:

Ensure that the variable `$BIND_HOME` is set to the directory under which BIND runs, typically the directory `/var/named/`. In the case of a `chroot`'ed configuration, the daemon will likely run under `/var/named/chroot/`, however the upper level directory of `/var/named/` should still be used as it is specific to the BIND service, and will include the `chroot` directory. Also, the variable `$RUNDIR` should be set to the directory which is used to create run-time files such as the `pid` file and session-key. Perform the following to ensure the directory ownership:

```
# find $BIND_HOME $RUNDIR -type d \! -user root -ls
```

There should be NO directories listed in the output from the find command.

Remediation:

To correct the directory ownership, perform the following:

```
find $BIND_HOME $RUNDIR -type d \! -user root | xargs chown root
```

Note: it is important to remember that the run time files will need to be in a directory writable by the named group. Changing the directory ownership to root might cause permissions issues, if the group permissions are not writable.

Default Value:

The following directories are owned by `named` in the default package install

- `/var/named/dynamic`
- `/var/named/slaves`
- `/var/named/data`
- `/run/named`

CIS Controls:

Version 6

14.4 Protect Information With Access Control Lists
All information stored on systems shall be protected 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.

Version 7

14.6 Protect Information through Access Control Lists
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.

2.5 Set root Ownership of BIND Configuration Files (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

The configuration files in the ISC BIND directories should be owned by root. Of course, any files created at run time by BIND, such as `pid` files, log files and slave zone files will necessarily be owned by named.

Rationale:

Restricting ownership of the configuration files provides defense in depth and will reduce the probability of unauthorized modifications to those important files. If there was a BIND vulnerability that allowed code execution as the named user, then the code would not be able modify the configuration files.

Audit:

Run the command below to ensure that all BIND configuration files are owned by root, except for those found in the run-time directories. Ensure that the BIND benchmark variables used below are set as described in the benchmark overview, as these variables identify the run-time directories. `($DYNDIR, $SLAVEDIR, $DATADIR, $RUNDIR, $LOGDIR, $TMPDIR)` If a `chroot`'ed directory is not used, then `$LOGDIR` and `$TMPDIR` are not generally a subdirectory of `$BIND_HOME`, and the two directories may be omitted, however including them will not cause any errors or false positives.

```
# find $BIND_HOME -type f \! -user root | egrep -v \
\^$DYNDIR\|\^$SLAVEDIR\|\^$DATADIR\|\^$RUNDIR\|\^$LOGDIR\|\^$TMPDIR
```

There should be no files listed in the output from the find command.

Remediation:

Perform the following:

- Capture the output of the previous audit command to a file named `nonroot-files.txt` and review any files not owned by root to ensure the files are necessary and are not expected run-time files. Delete any unnecessary files, and ensure any run-time files are being created in the appropriate run-time directory.

```
# find $BIND_HOME -type f \! -user root | egrep -v \
\^$DYNDIR\|\^$SLAVEDIR\|\^$DATADIR\|\^$RUNDIR\|\^$LOGDIR\|\^$TMPDIR > \
$TMPDIR/nonroot-files.txt
```

- The remaining non-run-time files should be changed to be owned by root, with a command like the following:

```
\# cat $TMPDIR/nonroot-files.txt | xargs chown root
\# rm $TMPDIR/nonroot-files.txt
```

Default Value:

The default rpm has the following configuration files owned by named.

- `/var/named/named.ca`
- `/var/named/named.empty`
- `/var/named/named.localhost`
- `/var/named/named.loopback`

CIS Controls:

Version 6

14.4 Protect Information With Access Control Lists
All information stored on systems shall be protected 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.

Version 7

14.6 Protect Information through Access Control Lists
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.

2.6 Set Group named or root for BIND Directories and Files (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

All the BIND directories and files should have a group of either named or root.

Rationale:

In general, the BIND directories and files default to a group of named, however some system files may have a group of root. Examples of system files include `chroot`'ed system device files. Either group root or named is accepted, as the intent is to prevent unexpected group ids, from getting inappropriate access to BIND files. Run time directories to which BIND will need write access should have a group of named, so that write access may be granted via the group permissions.

Audit:

Ensure that the BIND benchmark variables used below are set as described in the benchmark overview. Run the command below to ensure that all BIND directories and files have a group of either named or root.

```
# find $BIND_HOME $RUNDIR \! \( -group root -o -group named \) -ls
```

There should be no files listed in the output from the find command.

Remediation:

Run the command below to change all BIND directories and files to the group named.

```
chgrp -R named $BIND_HOME $RUNDIR
```

Default Value:

The default rpm install has all directories and files in the BIND home and the run time directory with a group of named.

CIS Controls:

Version 6

14.4 Protect Information With Access Control Lists

All information stored on systems shall be protected 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.

Version 7

14.6 Protect Information through Access Control Lists

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.

2.7 Set Group Read-Only for BIND Files and Non-Runtime Directories (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

All of the BIND files and all of directories except the run-time directories into which BIND will create files should have group permissions set to not be writable. Any run-time files created by BIND will be owned by BIND, and therefore need not be group writable.

Rationale:

Restricting permissions on the directories and files provides defense in depth and will reduce the probability of unauthorized modifications to important files. If there was a BIND vulnerability that allowed code execution as the named user, then the code would not be able create or modify configuration files.

Audit:

Ensure the `BIND_HOME` and all the runtime directory variables are set as specified in the overview **without a trailing slash** after the directory name. Run the commands below to ensure that all BIND directories are read-only for group except for the expected run time directories where the named service will create files.

```
# find $BIND_HOME -type d -perm /020 | egrep -vx \
  $DYNDIR\|$SLAVEDIR\|$DATADIR\|$RUNDIR\|$LOGDIR\|$TMPDIR

# find $BIND_HOME -type f -perm /020
```

There should be no files listed in the output from the find commands.

Remediation:

Perform the following:

- Capture the output from the audit commands above into a file named `write-dirs.txt`
- Review the purpose for the identified directories and either delete them if the directory is not needed, or change the permissions of the directory to not be writable by group or other.

- The following command can be used to change the permissions of the directories that are appropriate.

```
xargs -a write-dirs.txt chmod g-w
```

Default Value:

The default rpm install has the following non-runtime directories with group write access.

- /var/named/

CIS Controls:

Version 6

14.4 Protect Information With Access Control Lists
All information stored on systems shall be protected 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.

Version 7

14.6 Protect Information through Access Control Lists
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.

2.8 Set Other Permissions Read-Only for All BIND Directories and Files (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

All the directories and files in BIND home and run time directories should have other permissions set to not be writable. Configuration files should, of course, not be writable by named, and any run time files created by BIND will be owned by named and writable by the user. A `chroot`'ed `tmp` directory only needs to be writable by the named group. Therefore, there are no exceptions required.

Rationale:

Restricting permissions on the files provides defense in depth and will reduce the probability of unauthorized modifications to important files. If there was a BIND vulnerability that allowed code execution as the named user, then the code would not be able to modify configuration files.

Audit:

Run the command below to ensure that all BIND directories and files are read-only for other. Note that a `chroot`'ed directory will have some special files which may need to be writable. Special files includes device files, like dev/null and a socket file for logging, but the `-type f` and `type d` restricts the find to just directories and regular files.

```
# find $BIND_HOME $RUNDIR -type f -perm /002

# find $BIND_HOME $RUNDIR -type d -perm /002
```

There should be no files listed in the output from the find commands.

Remediation:

Perform the following:

- Capture the output from the audit commands above into a file with the name `$TMPDIR/write-files.txt`

- Review the purpose for the identified files and either delete them if the file is not needed, or change the permissions of the file to not be writable by group or other.
- The following commands can be used to change the permissions of the appropriate files.

```
# find $BIND_HOME $RUNDIR -type f -perm /022 > $TMPDIR/write-files.txt
# xargs -a $TMPDIR/write-files.txt chmod go-w
# rm $TMPDIR/write-files.txt
```

Default Value:

The default rpm install has all BIND directories and files without group or other write access.

CIS Controls:

Version 6

14.4 <u>Protect Information With Access Control Lists</u>
All information stored on systems shall be protected 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.

Version 7

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.

2.9 Isolate BIND with chroot'ed Subdirectory (Automated)

Profile Applicability:

- Caching Only Name Server Level 2
- Authoritative Name Server Level 2

Description:

The `chroot()` system call causes an application to run with limited file system access so that a subdirectory becomes the root directory for the application environment. When this is done, the application is “jailed” and no longer has access to the entire file structure but is limited to the given subdirectory.

The chroot’d subdirectory and the recommendations in section "Enable SELinux to Restrict BIND Processes" provide similar controls, in that the DNS service is prevented from accessing and modifying inappropriate files. SELinux goes well beyond what the chroot is able to prevent, however for audit purposes either control, the chroot’d subdirectory or the SELinux in enforcing mode is sufficient.

Rationale:

Although there are ways that a `chroot` jail can be broken, most methods require that a process be running as root in order to escape. Since BIND should be run as a different user than root, a `chroot` is an effective defense, to limit access to sensitive system configuration files. In the event that BIND has a vulnerability that allows code execution, the attack will not have access to the real system files such as `/etc/password`, but will be limited to the files placed in the `chroot` subdirectory.

Audit:

Run the following two commands to find the root directory of the currently running named process. If the named process is `chroot`'ed, then the listing will show a symbolic link to the `chroot` subdirectory. If process is not `chroot`'ed, then the symbolic link will point to the real root directory `/`.

```
# NAMEDPID=$(pidof named)
# ls -ld /proc/$NAMEDPID/root
lrwxrwxrwx 1 named named 0 Sep 10 13:21 /proc/423/root -> /var/named/chroot
```

Note: SELinux in enforcing mode may be used as an audit alternative to the chroot'd subdirectory.

Remediation:

Perform the following:

- Stop the named service and install the `bind-chroot` package to provide the `chroot` directories.

```
# systemctl stop named.service
# yum install bind-chroot
```

- Edit the `/etc/sysconfig/named` configuration file to have a line similar to the one shown below that sets the `ROOTDIR` environment variable.

```
ROOTDIR="/var/named/chroot"
```

- Move all the configuration files and any master zone files into their respective directions under the subdirectory `/var/named/chroot/`
- It may be helpful to create symbolic links from a couple of system `/etc` files such as `/etc/named.conf` and `/etc/rndc.key` to the real files in the `chroot`'ed subdirectory, so that utilities like `rndc` will work as expected. **Do not create symbolic links or hard links from inside the chroot to external resources!** Instead use symbolic links to point from the outside resources into the `chroot`.
- Restart the named service and test the configuration.

```
# systemctl start named.service
```

Default Value:

The BIND service is not `chroot`'ed by default.

CIS Controls:

Version 6

14.4 Protect Information With Access Control Lists
All information stored on systems shall be protected 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.

Version 7

14.6 Protect Information through Access Control Lists
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.

3 Restricting Queries

Recommendations in this section pertain to configurable access control mechanisms that are available in ISC BIND to restrict queries.

3.1 Ignore Erroneous or Unwanted Queries (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

BIND can be configured to ignore requests originating from specified network segments. This is accomplished by implementing the `blackhole` option in `named.conf`. It is recommended that this feature be implemented to ignore requests that originate outside of expected network segments.

Rationale:

By ignoring traffic that originates from unexpected networks, the server's exposure to malicious entities is reduced.

Audit:

Attempt to query the server from an address that has been placed in the `blackhole` list. If properly configured, the query will fail.

```
nslookup www.google.com ns1.example.com
```

Remediation:

Add a `blackhole` option for multicast and link local addresses, and all private RFC 1918 addresses that are not being used.

```
blackhole {
    // Private RFC 1918 addresses
    10/8; 192.168/16; 172.16/12;
    // Multicast
    224/8;
    // Link Local
    169.254/16;
};
```

Default Value:

No networks are `blackhole`'d by default.

CIS Controls:

Version 6

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

Version 7

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

3.2 Restrict Recursive Queries (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

A recursive DNS query is your typical DNS query from a client to a caching DNS server. It places the burden of finding the answer on the caching DNS server which will recursively query other DNS servers authoritative for the domains, until it gets the answer which is then returned to the client. The DNS server will then cache the answer to that query until its time-to-live expires in order to provide a quick answer to future queries for the same name. BIND can be configured to restrict fulfillment of recursive lookups to only authorized network segments and hosts. This is made possible by the `allow-recursion` option. Caching non-authoritative name servers should only allow recursive queries from clients on their own authorized networks. Authoritative name servers should not allow recursive queries, except to the local host.

Rationale:

Recursive DNS queries are commonly used in malicious attacks, including DNS amplification attacks and DNS cache poisoning attacks. A DNS amplification attack is a form of a reflected distributed denial-of-service attack, where multiple publicly accessible servers are sent recursive queries with the source IP address spoofed to be that of the victim. A high volume of relatively large DNS responses then flood the victim. For a DNS cache poisoning attack, the attacker may perform a query, and then provide a bogus response for the server to store in the cache. The bogus response may redirect clients to a different IP address which is provided by the attacker. Once the cache is poisoned, then clients visiting web sites, connecting to mail servers or VPNs may be connected with a malicious server configured to attack the client or steal credentials.

Limiting recursive queries to trusted networks does not prevent all of the DNS attacks possible, but it does make the attacks much more difficult and dramatically limits the scope of possible attacks so that detection and response are manageable.

Audit:

From the command prompt on Windows or Linux, send a recursive DNS request for a domain name for which the server is not authoritative, and from a client that is not permitted to perform recursive queries:

```
$ nslookup www.cisecurity.com 10.10.3.53
. . .

** server can't find www.cisecurity.com: REFUSED
```

The expected result is for the query to be refused. If the query times out due to network blocking the request, then examine the configuration files and verify that an `allow-recursion` statement is present and has only the appropriate network and hosts IP addresses listed. Something like one of the two statements below:

```
allow-recursion { localhost; };

recursion no;
```

Remediation:

Authoritative Name Server:
For an authoritative name server, insert one of the following either into the global options or into every zone section.

```
allow-recursion { localhost; };

recursion no;
```

Caching Name Server:

- Define an ACL named `trusted_clients` which will identify the networks which are expected to use the DNS caching server, and will be allowed to send recursive DNS queries.

  ```
  acl trusted_clients { 10.19.4.0/28; . . . }
  ```

- Insert the following into the global options.

  ```
  allow-recursion { localhost; trusted_clients };
  ```

Default Value:

The allow-recursion option is not defined by default.

References:

1. https://www.us-cert.gov/ncas/alerts/TA13-088A
2. https://www.godaddy.com/help/what-risks-are-associated-with-recursive-dns-queries-1184

CIS Controls:

Version 6

9.1 Limit Open Ports, Protocols, and Services
Ensure that only ports, protocols, and services with validated business needs are running on each system.

Version 7

9.2 Ensure Only Approved Ports, Protocols and Services Are Running
Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.

3.3 Restrict Query Origins (Manual)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

BIND can be configured to restrict access to its query services based on source IP address. It is recommended that the `allow-query` option be used to restrict access to only the networks authorized to use the name server. For an external authoritative only name server, the authorized networks may include all networks, however for internal authoritative or caching name servers the authorized networks should be explicitly configured.

Rationale:

Using allow-query in conjunction with an ACL of trusted networks will reduce the risk of unauthorized access to name services content. Additionally, the exposure of vulnerabilities present in BIND's query handlers is reduced by this configuration as requests with an untrusted source will be rejected before the request is fully parsed by named. Keep in mind however, that the source IP addresses can be easily spoofed, and the firewall and network architecture also needs to protect internal name servers from external spoofed requests.

Audit:

Verify that the BIND configuration files contain a global allow-query option with only the predefined ACL `localhost` and an ACL of the explicitly authorized networks. For an external authoritative only name server, the authorized networks may be the ACL `any` which represents any IPv4 or IPV6 host, but for caching and internal name servers, the `authorized_networks` should be an ACL with an explicit list of networks. The name of the ACL does not have to be `authorized_networks`.

```
$ grep allow-query $CONFIG_FILES
    allow-query   { localhost; authorized_networks };
```

For an external authoritative only name server:

```
$ grep allow-query $CONFIG_FILES
    allow-query   { any };
```

Remediation:

For remediation:

- Create an ACL for the authorized trusted networks in the `named.conf` file.

```
acl authorized_networks { 10.10.32.0/24; 10.10.34.0/24; . . .  };
```

- Add the allow-query statement to the global options of the `named.conf` file with the localhost ACL and the authorized trusted networks ACL.

```
allow-query   { localhost; authorized_networks };
```

Default Value:

The default package install allows queries only from localhost.

CIS Controls:

Version 6

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

Version 7

14.7 Enforce Access Control to Data through Automated Tools
Use an automated tool, such as host-based Data Loss Prevention, to enforce access controls to data even when data is copied off a system.

3.4 Restrict Queries of the Cache (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

The BIND option `allow-query-cache` may be used to restrict or allow BIND to provide answers to queries from the current cache of previously resolved queries. An authoritative only name server should not allow cache queries, except from the localhost, A caching only name server should allow cache queries only from the list of authorized networks.

Rationale:

Caching only name servers are critical to the security of all of the clients and servers using them, only the local authorized networks should be allowed to perform queries of the server's cache. In addition to malicious malformed queries, an attacker could use information about what is or is not in the name servers cache to help setup a DNS attack against the systems using the caching name server.

Audit:

From the command prompt on a Microsoft Windows or Linux client not on the authorized network, send a non-recursive DNS request for a domain name for which the server is not authoritative, as shown below.

```
$ nslookup -norecurs www.cisecurity.org 10.3.5.53
Server: UnKnown
Address: 10.3.5.53
*** UnKnown can't find www.cisecurity.org: Query refused

-or-

** server can't find www.cisecurity.org: REFUSED
```

The correct response should say `REFUSED`, or `Query Refused` as shown above. If an IP address is returned, then the server accepted the quest and returned the value from the cache. If the reply includes any of the phrases `can't find` or `No Answer`, or `Server failed` similar to the samples below, then the request was allowed, and the value was not in the cache.

```
Example of Query responses that were allowed, but not in the cache

*** Can't find www.cisecurity.org: No answer

*** UnKnown can't find www.cisecurity.org: Server failed
```

Remediation:

Authoritative Only Name Server:
For an authoritative name, insert the following either into the global options or into every zone section.

```
allow-query-cache { localhost; };
```

Caching Only Name Server:
Use the previously defined an ACL named `trusted_clients` which will identify the networks which are expected to use the DNS caching server, and will be allowed to send DNS cache queries.

```
allow-query-cache { localhost; trusted_clients };
```

Default Value:

If the `allow-query-cache` option is not present in the configuration, the default value is the `allow-recursion` setting. If the `allow-recursion` setting is not present, then the `allow-query` setting is used, unless recursion is set to no. If recursion is set to no, then the default value is none. Otherwise, if `allow-query` is also not present then the default value is `localnets` and localhost.

References:

1. https://kb.isc.org/article/AA-00845/0/BIND-9.9-Administrator-Reference-Manual-ARM.html

CIS Controls:

Version 6

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

Version 7

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

4 Transaction Signatures -- TSIG

Transaction Signature (TSIG) is used by BIND to ensure the authenticity and integrity of DNS requests and responses. TSIG is implemented by generating a secure hash of the DNS data combined with a shared secret. Since TSIG depends on a shared secret between the two servers it is really only suitable for server to server communications such as authenticating your organization's or possible partnering organization's DNS servers. There are a few critical DNS functions for which using TSIG works well to provide authentication: zone transfers, notifications and dynamic updates. This section will discuss the secure generation and protection of the keys used for TSIG communication, and discussion of specific usage of TSIG for securing transfers and updates will follow.

4.1 Use TSIG Keys 256 Bits in Length (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

The TSIG secret keys used by the name server should be generated from a good source of entropy and should be at least 256 bits in length.

Rationale:

Weak cryptographic keys may allow cryptographic attacks to discover the key value, through repeated guesses. A strong HMAC key requires a good source of entropy and at least 256 bits in length.

Audit:

To check if the secret key is at least 256 bits long before encoding then the `base64` encoded value should be `44` characters long or longer with the padding and a trailing `=`. To easily count the number of characters, copy the key value into the quoted value in the command below.

```
$ echo -n "ezoZopbE4Q73HShuFYlf3FRvLWjtNXI5fd0TeQAYOug=" | wc -c
44
```

Remediation:

For remediation, replace any keys which are too short with a securely generated key with a length of 256 or 512. The tsig-keygen command below can be used to generate a key.

```
# tsig-keygen -a sha256 ns1-ns4.example.net > ns1-ns4.example.net.key
# cat ns1-ns4.example.net.key
key "ns1-ns4.example.net" {
        algorithm hmac-sha256;
        secret "ezoZopbE4Q73HShuFYlf3FRvLWjtNXI5fd0TeQAYOug=";
};
```

Ensure the key file has the appropriate file permissions, and include the file in the named configuration file.

Default Value:

The rndc key is generated as 256 bits during bind-utils package install, and the nsupdate session key is dynamically generated with a length of 256 bits.

CIS Controls:

Version 6

14.2 Encrypt All Sensitive Information Over Less-trusted Networks
All communication of sensitive information over less-trusted networks should be encrypted. Whenever information flows over a network with a lower trust level, the information should be encrypted.

Version 7

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

4.2 Include Cryptographic Key Files (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

Do not place keys directly in the BIND `named.conf`, but use separate configuration files for the keys and include them into the `named.conf` file, in order to protect the keys from unintentional disclosure.

Rationale:

Although the keys may be placed directly in the named.conf file, putting it in a separate file will limit the number of times it needs to be viewed, and make it independent of viewing and changes to the main configuration file.

Audit:

Use the grep command below to search the `named.conf` file to ensure it doesn't have any secret keys placed in the file.

```
# grep -C 3 secret /etc/named.conf

key host1-host2.cisecurity.org {
    algorithm hmac-sha256;
    secret "1R3DP9D81/yWXjqf3hlg2beRpti1883JnZ3s7RVb1HU=";
};
```

Remediation:

Move each key definition statement from the `named.conf` file into its own key file. It is recommended to name both the key and the key file after the two hosts that will be sharing the secret key, in order to avoid confusion. Then include the key files with include statements in the `named.conf`. An example is shown below with the key definition statement moved to a separate key file, however it is also accepted for only the secret statement to be moved to another file.

```
# grep -C 1 include /etc/named.conf

// Include the key file used for the host1 and host2 TSIG comms
include "/etc/private/host1-host2.cisecurity.org.key";
```

```
# cat /var/named/chroot/etc/private/host1-host2.cisecurity.org.key
key host1-host2.cisecurity.org {
 algorithm hmac-sha256;
 secret "1R3DP9D81/yWXjqf3hlg2beRpti1883JnZ3s7RVb1HU=";
};
```

Default Value:

During a default install an `rndc` key is generated in a separate file `/etc/rndc.key` and included in the `named.conf`.

CIS Controls:

Version 6

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

Version 7

13 Data Protection
Data Protection

4.3 Use Unique Keys for Each Pair of Hosts (Automated)

Profile Applicability:

- Authoritative Name Server Level 1

- Caching Only Name Server Level 1

Description:

A unique TSIG key should be used for each pair of communicating hosts. For example, if there is one master authoritative name server and three slave authoritative name servers that were updated by the master, then there would need to be a unique TSIG key for at least the following:

- Master <-> Slave1
- Master <-> Slave2
- Master <-> Slave3

Rationale:

Each communication channel should have a unique key, to reduce the risk of key disclosure. If one of the TSIG keys or one of the slave servers is compromised, then the remaining TSIG keys are not disclosed.

Audit:

To verify each key is unique, and has unique usage, perform the following:

- The sample command below will extract the secret keys from the configuration files and count the number of occurrences of each key value.

```
# cat $CONFIG_FILES | egrep -o "secret.*;" | sort | uniq -c
   1 secret "R/eBXL/5xso142dGZSGJixKAAW+bO1UHlIpxZAj92Cc=";
   2 secret "P3/AuCgxdt3buLyeb/QxRmPe9IfMwsXRrKyNvQSbN1k=";
   1 secret "SGNiICKGf86GbhzpDBZOkQ==";
   1 secret "gyxEId4g2gB+pVJSKXA=";
```

The count occurrences preceding each key should be one in the output.

- Search the configuration files for duplicate uses of the same key name. The command below will extract references to key names.

```
# egrep "keys +{.+}"  $CONFIG_FILES
named.conf:         allow { 127.0.0.1; } keys { "rndc-key"; };
```

```
ns1-ns2.cisecurity.org.key:        keys { "ns1-ns2.cisecurity.org"; };
ns1-ns3.cisecurity.org.key:        keys { "ns1-ns3.cisecurity.org"; };
```

Each key name should be referenced only once.

Remediation:

Generate unique keys for host to host communication. The command below can be used to generate 2 files, a *<name>.key* file and a *<name>.private* file with secret keys of suitable length with base64 encoding. The files themselves are not needed, and should be securely deleted once the values are copied into a key file for including in the named configuration.

```
$ dnssec-keygen -a HMAC-SHA256 -b 256 -n HOST ns1-ns2.cisecurity.org
Kns1-ns2.cisecurity.org.+163+13013

$ cat Kns1-ns2.cisecurity.org.+163+13013.key
ns1-ns2.cisecurity.org. IN KEY 512 3 163
9FQ2dYCQ17HJwDi/uHgANh2dlb8M7eb+F4AjML8tTdA=
```

Default Value:

The `rndc` key is automatically generated during package installation.

References:

1. http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-81-2.pdf

CIS Controls:

Version 6

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

Version 7

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

4.4 Restrict Access to All Key Files (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

The TSIG keys should be readable only by the named and root accounts. No other user accounts or groups should have read access. Note that BIND often creates a session key on startup for usage by `nsupdate -l`. Both the `$BIND_HOME` and `$RUNDIR` are included since the session key should also have the recommended permissions.

Rationale:

The secret key protects the authenticity and integrity of TSIG communications and disclosure of a key would allow an attacker to perform the authenticated operations such as `rndc` administrative operations, zone transfers or dynamic updates.

Audit:

Perform the following to audit the recommendation:

- Find all of the TSIG key files in the `$BIND_HOME` and `$RUNDIR` directory, and capture the list to a file named `key_file.txt` in a `tmp` directory with the command below. If the `RUNDIR` is a subdirectory of `BIND_HOME`, which is typical for a `chroot`'ed directory, then some key files may be found twice. Duplicates are removed by the final sort command.

```
# find $BIND_HOME $RUNDIR -type f | xargs fgrep -l secret | sort -u >
$TMPDIR/key_files.txt
```

- Check for appropriate ownership, group and permissions on the files with the following commands.

```
# find $(cat $TMPDIR/key_files.txt ) \! \( -user root -o -user named \)
-ls
# find $(cat $TMPDIR/key_files.txt ) \! \( -group root -o -group named
\) -ls
# find $(cat $TMPDIR/key_files.txt ) -perm /022
```

- There should be no output from the three find commands. Remove the temporary file when finished.

```
rm $TMPDIR/key_files.txt
```

Remediation:

Perform the following for remediation:

- Use the command below to find secret key files. Review the list of key files, and delete any unused or unnecessary key files. Recreate the file list, after deleting any unused files.

```
# find $BIND_HOME $RUNDIR -type f | xargs fgrep -l secret | sort -u >
$TMPDIR/key_files.txt
```

- Change the ownership, group and permissions on the key files.

```
# xargs -a $TMPDIR/key_files.txt chown -R root
# xargs -a $TMPDIR/key_files.txt chgrp -R named
# xargs -a $TMPDIR/key_files.txt chmod o-r
```

- Remove the temporary file,

```
rm $TMPDIR/key_files.txt
```

Default Value:

Ownership, Group and Permissions are correct for any default key files.

CIS Controls:

Version 6

14.4 Protect Information With Access Control Lists
All information stored on systems shall be protected 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.

Version 7

14.6 Protect Information through Access Control Lists
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.

4.5 Protect TSIG Key Files During Deployment (Manual)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

Do not expose the TSIG key files through insecure network transmission of the files when deployed, or via insecure permissions or shares on any intermediate systems used for the key deployment.

Rationale:

The secret key protects the authenticity and integrity of TSIG communications and disclosure of a key would allow an attacker to perform the authenticated operations such as `rndc` administrative operations, zone transfers or dynamic updates.

Audit:

Review the technical procedure for generating and deploying the TSIG keys to ensure the files are not inappropriately disclosed on the original systems where the key is generated, on any intermediate systems, or file shares. Also, ensure that the process does not allow the keys to be copied over the network via clear text of weak file transfer protocols, such as `telnet`, `ftp` or `rcp`.

Remediation:

Perform the following:

- Correct the deployment procedure to ensure secure transmission and intermediate storage protection of keys during deployment.
- Regenerate new keys via the corrected procedure and replace all previous TSIG keys.

CIS Controls:

Version 6

14.2 Encrypt All Sensitive Information Over Less-trusted Networks
All communication of sensitive information over less-trusted networks should be

encrypted. Whenever information flows over a network with a lower trust level, the information should be encrypted.

Version 7

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

5 Authenticate Zone Transfers and Updates

Recommendations in this section pertain to the configuration of secure DNS Zone transfers and dynamic updates to ensure the authenticity and integrity of the requests.

5.1 Securely Authenticate Zone Transfers (Automated)

Profile Applicability:

- Authoritative Name Server Level 2

Description:

A zone transfer is a mechanism commonly used by DNS deployments to replicate zone information from master/primary servers to slave/secondary servers. Each pair of name servers participating in zone transfers should authenticate the requests and ensure the integrity of the responses by using a unique shared secret TSIG key. BIND can be configured to respond only to authenticated transfer requests by using the `allow-transfer` statement with a key statement, that restricts the transfers to servers that provide a MAC using the named key.

Rationale:

A zone transfer is a popular information disclosure attack as it provides the entire list of resource records for a zone. There should be very few systems such as the slave name servers that should be authorized to perform a zone transfer for your domains. Authentication of transfer requests should not be made using only an IP address, since IP addresses can be spoofed, but rather by using TSIG keys.

Audit:

Perform the following:

- Search all of the included configuration files and zone files for the `allow-transfer` option.

```
grep -C 1 allow-transfer $CONFIG_FILES $ZONE_FILES
```

- If there are no `allow-transfer` statements found, then the configuration allows zone transfers, and is not compliant.
- If the only value in the address match list of all the `allow-transfer` statements is the value `none`, either with or without quotes, then the configuration is compliant. Examples output is shown below.

```
allow-transfer { none; };
allow-transfer {"none";};
```

- If **all** of the address list values of the `allow-transfer` statements have the keyword `key` followed by a name, then the configuration is compliant.

```
allow-transfer { key ns1-ns2.cisecurity.org.; };
```

- If the predefined address value of `any` appears in the `allow-transfer` statement, then the configuration is not compliant. If any of the address list values contains ACL names, IP addresses or network ranges, then the configuration is also not compliant.

```
allow-transfer { any; }
allow-transfer { 10.10.42.56; key ns1-ns2.cisecurity.org.; }
```

- Additionally, it is possible to confirm if a transfer is allowed to an IP address without a key, by performing the following command on the system with the suspected allowed IP address. An error of `Transfer failed` is the expected result. If a list of resource records is returned, then the transfer was allowed without a key, and the configuration is non-compliant.

```
$ dig @ns1.cisecurity.org cisecurity.org axfr
; <<>> DiG 9.9 . . .
; (1 server found)
;; global options: +cmd
; Transfer failed.
```

Remediation:

Generate TSIG keys 256 bits in length, unique for each host-to-host communication. Securely Transfer the keys and configure the keys to be required in all `allow-transfer` statements.

Default Value:

If the `allow-transfer` statement is missing, then transfers are allowed to any host.

CIS Controls:

Version 6

9.1 Limit Open Ports, Protocols, and Services
Ensure that only ports, protocols, and services with validated business needs are running on each system.

Version 7

16.4 Encrypt or Hash all Authentication Credentials

Encrypt or hash with a salt all authentication credentials when stored.

5.2 Securely Authenticate Dynamic Updates (Automated)

Profile Applicability:

- Authoritative Name Server Level 1

Description:

Dynamic updates are used to automate the updating of zones. Dynamic updates are typically used with DHCP; however, updates may include other records. The allow-update option allows deleting or adding any resource records of a zone except the SOA and NS records, and should not be used. Instead the update-policy option allows a more granular policy to be specified so that only specific resource record types and a specific sub-domain may be updated. The update-policy must be securely authenticated with a key identifier, rather than by an IP address. The key identifier may specify a `TSIG` key, a `GSS-TSIG` key, or a `SIG(0)` key.

Rationale:

Allowing other systems to make permanent updates to your zones is of course not allowed by default, and needs to be carefully secured. Consider the power of an attack that could update the zone to direct clients and servers to the malicious server of the attacker's choice. The attack would not be restricted to just HTTP, but every connection and protocol that uses a name and allows weak authentication may be subject to redirection and a variety of man-in-the-middle and protocol downgrade attacks. Therefore, it is important that all dynamic updates be securely authenticated using a cryptographic key, and not rely on an IP address.

Audit:

Perform the following steps:

- Search for the allow-update option in all of the included configuration files, and in the zone files. If any allow-update options are present, other than `none` or `localhost`, as shown below, then the configuration is not compliant.

```
# grep allow-update $CONFIG_FILES $ZONE_FILES
/etc/named.conf: allow-update { none; };
/. . . /data/cisecurity.org: allow-update { “localhost”; };
```

- Search for any update-policy options in all of the zone files. Any update policies found, should not contain any IP addresses, network CIDR notations, or any ACL names that represents an IP addresses. The only entries in the update-policy should be key identifiers or `local` as shown below. All of the following are compliant.

```
# grep update-policy $ZONE_FILES
/. . ./data/internal.org: update-policy { grant ns1-dhcp-update-key
name dyn.internal.org A; };
/. . ./data/cisecurity.local: update-policy { grant dyn_update_key self
office.cisecurity.local A; };
/. . ./data/test.local: update-policy { local; };
```

Remediation:

Perform the following steps for remediation:

- Remove any `allow-update` options from the global options configuration.
- Replace or add `allow-update` options to the zone files to specify a securely generated `TSIG` or `SIG(0)` key identifier, along with the appropriate domain or sub-domain, and the appropriate resource record type.

Default Value:

Dynamic updates are not allowed by default.

CIS Controls:

Version 6

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

Version 7

16.4 Encrypt or Hash all Authentication Credentials
Encrypt or hash with a salt all authentication credentials when stored.

5.3 Securely Authenticate Update Forwarding (Automated)

Profile Applicability:

- Authoritative Name Server Level 1

Description:

A secondary authoritative name server is allowed to accept zone updates on behalf of the primary name server, and forward them to the master name server, where the zone file can be updated. In this case, the authentication of the dynamic updates is configured with the allow-update-forwarding option. The update requests must be securely authenticated with a key identifier, rather than by an IP address. The key identifier may specify a `TSIG` key, a `GSS-TSIG`, or a `SIG(0)` key.

Rationale:

Of course, allowing unauthenticated updates to a zone should not be allowed. It is necessary for the secondary authoritative name server to carefully authenticate the update request before sending it on to the primary name server, to prevent malicious DNS updates be propagated via the secondary server.

Audit:

Search for the allow-update-forwarding option in all of the included configuration files, and in the zone files. If any allow-update-forwarding options are present, then verify that there are no IP addresses or networks used for authentication. Instead a key identifier should be used, or the value `none` may be used to disable dynamic updates. Note that the key identifiers, may reference a `TSIG` key, `GSS-TSIG` key or `SIG(0)` key. Use the grep command below to search for allow-update-forwarding options, and verify that either key identifiers or the value `none` are used in each.

```
# grep allow-update-forwarding $CONFIG_FILES $ZONE_FILES
/etc/named.conf: allow-update-forwarding { none; };
/. . ./dyn.internal.org: allow-update-forwarding { key dhcp-
server.internal.org.; };
```

Remediation:

Modify any `allow-update-forwarding` options to specify a securely generated `TSIG` or `SIG(0)` key identifier used by the DHCP server.

Default Value:

Dynamic updates are disabled by default.

CIS Controls:

Version 6

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

Version 7

16.4 Encrypt or Hash all Authentication Credentials
Encrypt or hash with a salt all authentication credentials when stored.

6 Information Leakage

Recommendations in this section are intended to limit the disclosure of potentially sensitive information.

6.1 Hide BIND Version String (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

BIND includes a built-in zone, `version.bind` which may be queried to get the version of the name server. The version may be set to a value of `none`, to disable reporting of the version information.

Rationale:

Making detailed BIND version information easy to obtain remotely helps attackers automate and target their attacks. The information is not necessary for the health of the server, and should not be disclosed.

Audit:

Use the dig command shown below to query the chaos class TXT record on `version.bind`. If there is no output from the command, or if a value of `No Info` or `None` is returned then the configuration is compliant.

```
$ dig @ns1.cisecurity.org version.bind chaos txt | grep '^version.bind.' |
grep TXT
version.bind. 0 CH TXT "No Info"

$ dig @ns2.cisecurity.org version.bind chaos txt | grep '^version.bind.' |
grep TXT
$
```

Remediation:

Add or modify the version option to have a value of `none` in the BIND global options, as shown below.

```
options {
version "none";
. . .
}
```

Default Value:

Default value returns the current BIND detailed version.

CIS Controls:

Version 6

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

Version 7

13.2 Remove Sensitive Data or Systems Not Regularly Accessed by Organization
Remove sensitive data or systems not regularly accessed by the organization from the network. These systems shall only be used as stand alone systems (disconnected from the network) by the business unit needing to occasionally use the system or completely virtualized and powered off until needed.

6.2 Hide Nameserver ID (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

The `server-id` option provides a server identifier that will be returned in response to an NSID query. An NSID query is described in RFC-5001, and is a method to identify servers in an environment where there are multiple DNS servers sharing the same IP address. With the use of load balancing and other IP sharing mechanisms, it can become difficult to discern exactly which name server is responding to a particular query. NSID allows a name server to respond with identifying information. The `server-id` option should be disabled with a value of `none`.

Rationale:

Enabling the NSID option may allow external parties to obtain information about the configuration and architecture of the DNS server. If it is found to be necessary to enable this service, then the identifying information should be generic. You should not use the server's geographic location, internal IP address or any other privileged information.

Audit:

Use the dig command below to send an NSID query, on the built-in zone `id.server`, for a chaos class TXT record. There should not be any output for a compliant configuration.

```
$ dig @ns2.cisecurity.local id.server chaos txt | grep '^id.server.' | grep
TXT
```

An example of a non-compliant response to an NSID query is shown below.

```
$ dig @ns1.cisecurity.local id.server chaos txt | grep '^id.server.' | grep
TXT
id.server.            0       CH      TXT     "cpe-172.lima.ny.us.local"
```

Remediation:

To explicitly disable NSID support, add or modify the `server-id` option in the global BIND options with a value of `none` as shown below.

```
server-id none;
```

Default Value:

NSID is disabled by default.

References:

1. https://tools.ietf.org/html/rfc5001

CIS Controls:

Version 6

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

Version 7

13.2 Remove Sensitive Data or Systems Not Regularly Accessed by Organization
Remove sensitive data or systems not regularly accessed by the organization from the network. These systems shall only be used as stand alone systems (disconnected from the network) by the business unit needing to occasionally use the system or completely virtualized and powered off until needed.

7 Secure Network Communications

Recommendations in this section pertain to the configuration of secure communications to ensure the authenticity and integrity of DNS related network traffic.

7.1 Do Not Define a Static Source Port (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

BIND can be configured to always use the same source port when communicating with other DNS servers. This capability is made possible through the `query-source` port option, and the `query-source-v6` port option. It is recommended that the source port be omitted if the `query-source` option is used, or that the port be specified as a `*`, so that the port will not be a static port number.

Rationale:

DNS attacks which involve spoofing a bogus DNS reply may require the attacker to guess the source port number of the request, if the attacker is unable to see the initial DNS query. Making the source port static makes the attack easier, as it eliminates the effort of getting the correct destination port number for the spoofed reply. Instead of a static source port, the port number should be selected randomly from the client ephemeral ports.

Audit:

Verify that a static port is not specified in a query-source option or a `query-source-v6` option, using the command below.

```
$ egrep '^\W*query-source' $CONFIG_FILES | grep port
/etc/named.conf:    query-source address 10.1.45.53 port *;
```

If there is no output from the grep or if the port number is specified as `*`, then the configuration is compliant. Examples of a non-compliant configurations are shown below.

```
query-source port 1053;

query-source port-v6 53;
```

Remediation:

Either remove the port specification from the `query-source` or the `query-source-v6` option or use an `*` for the port number.

Default Value:

The default is to not use a static source port for queries.

CIS Controls:

Version 6

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

Version 7

9.2 Ensure Only Approved Ports, Protocols and Services Are Running
Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.

7.2 Enable DNSSEC Validation (Automated)

Profile Applicability:

- Caching Only Name Server Level 1
- Authoritative Name Server Level 1

Description:

DNS Security Extensions or DNSSEC for short provides authentication of the name servers through public key cryptography. With DNSSEC, the name server signs its responses with its private key. This allows other name servers that have the public key of the name server to verify the integrity and authenticity of the response. DNSSEC also provides for signing of public keys so that delegated sub-domains may have their keys signed by a higher-level authority. This creates a chain of trust so that any name server that trusts the public key of the higher-level signing authority can trust the signed key. It is recommended that DNSSEC be enabled and be configured to validate domains that are signed. DNSSEC and validation are enabled via the options `dnssec-enable` and `dnssec-validation`, respectively.

Rationale:

DNSSEC reliably authenticates DNS responses to prevent the DNS spoofing and cache poisoning attacks.

Audit:

Perform the following to verify compliance.

- To verify that the name server will validate the trust for DNSSEC signed domains, perform the following `delv` command on the name server. The command queries the name isc.org, which has a valid trusted DNSSEC signature.

```
$ delv @127.0.0.1 isc.org
; fully validated
isc.org.        60      IN      A       149.20.1.66
isc.org.        60      IN      RRSIG   A 13 2 60 20200306020506
20200205011018 27566 isc.org.
VxlTxZDll9boCOG2jE+gt7w3memCLyyCd+thrwf5XRrKkrCjcJWL7cd0y82Pv5FHKqhgq4b
BKJOR4uNhUhWDg==
```

The compliant result should have `; fully validated` as the initial response indicating the recursive server is DNSSEC aware.

- To verify that the name server will properly reject DNSSEC signed domains with an invalid signature, perform the dig command below on the `www.dnssec-failed.org` name.

```
$ delv @127.0.0.1 www.dnssec-failed.org
;; resolution failed: SERVFAIL
```

  The status value should be `resolution failed: SERVFAIL`.

Remediation:

Perform the following for remediation:

- Check the BIND configuration files, and in the global options set the option `dnssec-enable` to yes, and option `dnssec-validation` to either yes or auto as shown below. The auto setting is generally preferred as the trust anchor will not need to be manually configured.

```
dnssec-enable yes
dnssec-validation auto
```

- Restart the named server.

Default Value:

DNSSEC and DNSSEC validation are enabled by default.

References:

1. https://kb.isc.org/docs/aa-01182

CIS Controls:

Version 6

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

Version 7

16.4 Encrypt or Hash all Authentication Credentials
Encrypt or hash with a salt all authentication credentials when stored.

7.3 Disable the dnssec-accept-expired Option (Automated)

Profile Applicability:

- Caching Only Name Server Level 1

- Authoritative Name Server Level 1

Description:

The `dnssec-accept-expired` option allows BIND to accept expired signatures during validation. The option should be disabled so that expired signatures will not be accepted.

Rationale:

Allowing expired signatures would leave the server vulnerable to replay attacks.

Audit:

Verify the `dnssec-accept-expired` option is not present in the configuration files, or is set to a value of `no`.

```
# grep dnssec-accept-expired $CONFIG_FILES
/var/named/chroot/etc/named.conf:    dnssec-accept-expired no;
```

Remediation:

Change the `dnssec-accept-expired` option to have a value of `no`, or remove the option from the configuration files.

Default Value:

The `dnssec-accept-expired` option is disabled by default.

CIS Controls:

Version 6

9 Limitation and Control of Network Ports, Protocols, and Services
Limitation and Control of Network Ports, Protocols, and Services

Version 7

16.7 Establish Process for Revoking Access
Establish and follow an automated process for revoking system access by disabling accounts immediately upon termination or change of responsibilities of an employee or

contractor . Disabling these accounts, instead of deleting accounts, allows preservation of audit trails.

7.4 Ensure Either SPF or DKIM DNS Records are Configured (Automated)

Profile Applicability:

- Authoritative Name Server Level 2

Description:

For each authoritative domain that receives SMTP email, add either an SPF (Sender Policy Framework) TXT record and/or add a DKIM (DomainKeys Identified Mail) TXT record.

Rationale:

The SPF record reduces spam and phishing usage of a domain name, by publishing the IP addresses of the authorized mail servers, allowed to send mail for the domain. SPF compliant mail servers may reject or treat as SPAM, any mail coming from other IP addresses.

The DKIM record publishes a public key which may be used to verify the authenticity and integrity of the message by using the key to verify a digital signature of the message stored in an SMTP header.

Either or both of the technologies is recommended to be configured for each domain, to reduced spoofing and phishing attacks that use the domains in a FROM address. Consider the risk of a spoofed phishing email coming from upper management with an urgent request which had a valid FROM address. It might be too easy for someone to trust the email and take the action requested. In addition to SPF and DKIM, configuring a DMARC record, after SPF and/or DKIM records are in place, is helpful for reporting and forensics on attempted usage of the domain name. Only the SPF and DKIM DNS records are audited in this recommendation.

Audit:

For each authoritative domain, perform the following:

- Check the domain for an MX record to verify that the domain receives SMTP email. If there no MX record, then the domain is considered compliant. The following host command will query the localhost DNS server for any MX records for the given domain. The output below shows a domain that does not have an MX record.

```
$ host -t mx example1.com 127.0.0.1
. . .
example1.com has no MX record
```

For domains which have an MX record, perform the following two tests to audit compliance. If either test passes, the domain is considered compliant.

- Query the DNS server, to check that an SPF TXT record is present and has either a strict (`-all`) or soft fail (`~all`) policy at the end of the record. The following command queries the `ns1.example.net` name server for an SPF record for the example.com domain.

```
$ host -t txt example.com ns1.example.org | grep 'v=spf1.*[~-]all'
example.com descriptive text "v=spf1 a mx a:mail.example.net
ip4:10.1.2.3 ~all"
```

  If there is no output, then either an SPF record does not exist, or it does not have the recommended policy.

- Query the DNS server to check for a default DKIM TXT record with a public key. The following command queries the `ns1.example.net` name server for an SPF record for the example.com domain.

```
$ host -t txt default._domainkey.example.com ns1.example.org | grep
'v=DKIM1; k=rsa; p='
default._domainkey.example.com descriptive text "v=DKIM1; k=rsa;
p=MIGf.  . .  "
```

  If there is no output, then a default DKIM record does not exist.

Remediation:

Add either an SPF TXT record and/or a default DKIM TXT record to the domains with the appropriate values. The SPF record should have a soft fail policy of `~all` or a strict policy of `-all` There are on-line resources and tools such as MX toolbox that will help in generating and testing SPF, DKIM and DMARC records as shown in the references.

Default Value:

No SPF or DKIM records are configured by default.

References:

1. https://mxtoolbox.com/NetworkTools.aspx
2. https://mxtoolbox.com/spf.aspx
3. https://mxtoolbox.com/dkim.aspx

CIS Controls:

Version 7

7.8 Implement DMARC and Enable Receiver-Side Verification
To lower the chance of spoofed or modified emails from valid domains, implement Domain-based Message Authentication, Reporting and Conformance (DMARC) policy and verification, starting by implementing the Sender Policy Framework (SPF) and the DomainKeys Identified Mail(DKIM) standards.

8 DNSSEC Digital Signatures for Authoritative Zones

Recommendations in this section pertain to the configuration of secure DNSSEC digital signatures for zones of an authoritative name server. The digital signatures ensure the authenticity and integrity of the name responses.

8.1 Install the Haveged Package for Enhanced Entropy (Automated)

Profile Applicability:

- Authoritative Name Server Level 2

Description:

Install the haveged package to provide enhanced entropy for generating cryptographic keys. Haveged is a user space entropy daemon which is not dependent upon the standard mechanisms for harvesting randomness for the system entropy pool. Haveged uses HAVEGE (HArdware Volatile Entropy Gathering and Expansion) to maintain a pool of 1 million random bytes used to fill /dev/random.

Rationale:

It is important for authoritative DNS servers deploying DNSSEC domains to have a good source of entropy to generate secure cryptographic keys. DNS servers are typically not multi-user systems and generally deployed as headless servers. In such situations generating keys without enhanced entropy can be painfully time-consuming, or may lack sufficient entropy. The `haveged` daemon ensures that keys can be generated timely and securely.

Audit:

To verify compliance, run the following `pgrep` command to ensure the `haveged` process is running.

```
$ pgrep -f $(which haveged)
2948
```

Output with any process ID is compliant. No output indicates the `haveged` process is not running.

Remediation:

Install the `haveged` package with the appropriate package manager and configure it to start, as shown below.

```
# yum install haveged

# systemctl enable haveged

# systemctl start haveged
```

Default Value:

The haveged package is not installed or enabled by default.

8.2 Ensure Signing Keys are Generated with a Secure Algorithm (Automated)

Profile Applicability:

- Authoritative Name Server Level 2

Description:

When Zone Signing Keys (ZSK) or Key Signing Keys (KSK) are generated there are several secure DNSSEC digital signature algorithms that are recommended. The algorithms are listed below with the standard DNSSEC algorithm number followed by the common name, and then the BIND 9 mnemonic name used by `dnssec-keygen`.

```
- 8     RSA/SHA-256    RSASHA256
- 10    RSA/SHA-512    RSASHA512
- 13    ECDSA/SHA-256  ECDSAP256SHA256
- 14    ECDSA/SHA-384  ECDSAP384SHA384
- 15    Ed25519        ED25519
```

Rationale:

A secure public key algorithm along with a secure hash algorithm, are part of the foundation for a secure digital secure. Weaknesses in older public key algorithms continue to develop, and it is important to use a recommended algorithm that is expected to be secure for the near future.

Audit:

To audit the key algorithms in use, search the private key files for the line that starts with `Algorithm:`, using the following commands.

```
# cd $BIND_HOME
# find . -name 'K*.private' -print0 | xargs -0 grep '^Algorithm:'
./keys/Kcisecurity.org.+008+42383.private:Algorithm: 8 (RSASHA256)
./keys/Kcisecurity.org.+008+11955.private:Algorithm: 8 (RSASHA256)
./keys/Kecckey.com.+013+49638.private:Algorithm: 13 (ECDSAP256SHA256)
./keys/Kweakkey.com.+003+36233.private:Algorithm: 3 (DSA)
```

Any algorithms other than `RSASHA256`, `RSASHA512`, `ECDSAP256SHA256`, `ECDSAP384SHA384` or `ED25519` are not compliant. Likewise, the algorithm number must be one of: `8`, `10`, `13`, `14` or `15`.

Remediation:

To remediate a weak key, perform the following:

- Generate a new key to replace the weak key using `dnssec-keygen` and one of the recommended algorithms. Examples commands are shown below.

```
# dnssec-keygen -a RSASHA256 -b 2048 example.com
# dnssec-keygen -a ECDSAP384SHA384 cisecurity.org
```

- Implement a rollover period to phase out the weak key and replace it with the newly generated key.
- Once the key is fully deleted from active use, remove the file.

Default Value:

The default algorithm is RSASHA1.

References:

1. https://tools.ietf.org/id/draft-ietf-dnsop-algorithm-update-01.html#rfc.section.1.1
2. https://bind.isc.org/doc/arm/9.11/man.dnssec-keygen.html
3. https://www.isc.org/dnssec/

CIS Controls:

Version 7

18.5 Use Only Standardized and Extensively Reviewed Encryption Algorithms
Use only standardized and extensively reviewed encryption algorithms.

8.3 Ensure Any Signing Keys using RSA Have a Length of 2048 or Greater (Automated)

Profile Applicability:

- Authoritative Name Server Level 2

Description:

If one of the RSA digital signature algorithms is used, then then key length should be at least 2048 bits. The Elliptic Curve algorithms have sufficient key length without any additional options, and will provide for smaller signed DNS responses then the RSA algorithms.

Rationale:

RSA keys of 1024 bits in length are no longer considered secure against brute force attacks. A key length of at least 2048 bits is required.

Audit:

To perform an audit, search the private key files for the RSA 'Prime1:' string and check that BASE64 encoded value is at least 172 characters long. The awk command below can be used to automate the check.

```
# find . -name 'K*.private' -print0 | xargs -0 awk -e '/^Prime1:/ {print
FILENAME,length($2)}'
'/^Prime1:/ {print FILENAME,length($2)}'
./Kcisecurity.org.+008+42363.private 172
./Kexample.org.+008+11725.private 172
./Kweak-rsa1024.com.+008+50106.private 88
```

Remediation:

To remediate a weak RSA key, perform the following:

- Generate a new key to replace the weak key using dnssec-keygen and one of the recommended algorithms and key lengths. Examples commands are shown below.

```
# dnssec-keygen -a RSASHA256 -b 2048 example.com
# dnssec-keygen -a ECDSAP384SHA384 cisecurity.org
```

- Implement a rollover period to phase out the weak key and replace it with the the newly generated key.
- Once the key is fully deleted from the active use, remove the file.

Default Value:

If an RSA algorithm is chosen the default key length is 1024 for the ZSK and 2048 for the KSK.

References:

1. https://en.wikipedia.org/wiki/Key_size#Asymmetric_algorithm_key_lengths
2. https://arstechnica.com/uncategorized/2007/05/researchers-307-digit-key-crack-endangers-1024-bit-rsa/

8.4 Restrict Access to Zone and Key Signing Keys (Automated)

Profile Applicability:

- Authoritative Name Server Level 2

Description:

The files and directories for Zone Signing Keys (ZSK) and Key Signing Keys (KSK) should be read-only by the named user, with no access to other.

Rationale:

The named daemon does not require write access to the key files or the directories, Implementing a minimal read-only access provides an additional layer of denfense, so that if the service was exploited, the exploit would not be able to modify signing keys. Likewise restricting read access to the keys will prevent inappropriate disclosure of the private keys.

Audit:

Ensure the KEYDIR variable is set to the top directory or directories that contains all of the key files for all of the authoritative zones. Then perform the following:

```
find $KEYDIR -perm /027 -ls
```

Any files or directories that are not compliant will be listed in the output along with their permissions.

Remediation:

Perform the following:

```
chmod  -R g-w,o-rwX $KEYDIR
```

Default Value:

The BIND signing key files and directory do not exist by default.

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
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.

8.5 Ensure each Zone has a Valid Digital Signature (Manual)

Profile Applicability:

- Authoritative Name Server Level 2

Description:

For each zone of the authoritative name server, verify that the signed zone file has a valid signature for each algorithm in the zone DNSKEY RRSet.

Rationale:

The zone must have a valid signature before it can be trusted by validating DNSSEC name resolvers.

Audit:

Perform the following command on each zone providing the signed zone file. The example file name is 'cisecurity.org.signed' for the domain cisecurity.org. If the signed zone file is not generated from inline signing, then the format may be ASCII, and the `-I raw` should be omitted.

```
# dnssec-verify -I raw -o cisecurity.org cisecurity.org.signed
Loading zone 'cisecurity.org' from file 'cisecurity.org.signed'
Verifying the zone using the following algorithms: ECDSAP256SHA256.
Zone fully signed:
Algorithm: ECDSAP256SHA256: KSKs: 1 active, 0 stand-by, 0 revoked
                            ZSKs: 1 active, 0 stand-by, 0 revoked
```

The compliant output will include the string `Zone fully signed:`.

Remediation:

Perform either of the following:

- Enable in-line signing in each zone configuration by setting `inline-signing` to `yes` value. For example:

```
zone "cisecurity.org" {
     type master;
     file "/etc/named/masters/cisecurity.org";
     key-directory "/etc/named/keys";
     inline-signing yes;
     auto-dnssec maintain;
};
```

- Reload the server configuration and zones.

```
rndc reload
```

Or if using manual or scripted zone signing instead of inline-signing, then perform the following.

- Include the signing keys at the end of the zone file to be signed. Such as:

```
$include Kcisecurity.com.+013+09768.key
$include Kcisecurity.com.+013+45248.key
```

- Then sign each zone file with the `dnssec-signzone` command such as:

```
dnssec-signzone -o cisecurity.com ../masters/cisecurity.com
Kcisecurity.com.+013+09768.key Kcisecurity.com.+013+45248.key
```

- Reload the configuration and zones.

```
rndc reload
```

CIS Controls:

Version 7

16.4 Encrypt or Hash all Authentication Credentials
Encrypt or hash with a salt all authentication credentials when stored.

8.6 Ensure Full Digital Chain of Trust can be Validated (Automated)

Profile Applicability:

- Authoritative Name Server Level 2

Description:

For each authoritative domain ensure the digital signatures are is fully trusted starting from the root zone.

Rationale:

In order for the the digital signatures to be trusted by other systems, The parent zone must be a DS (delegated signer) record that verifies the authenticity of the child zones KSK (key signing key). The delegated signature forms a chain of trust, delegated down from the root zone.

Audit:

To audit the chain of trust use the delv command to query an external independent validating name server. Such as:

```
$ delv @8.8.8.8 cisecurity.org
; fully validated
cisecurity.org. 21599   IN      A       10.1.2.3
cisecurity.org. 21599   IN      RRSIG   A 8 2 43200 20200328213257
20200224211144 42363 cisecurity.org. gqTNFiJ. . . 2n4Q==
```

The output of "fully validated" indicates the zone is compliant. Other answers are not compliant, and may include "no valid signature found" and "unsigned answer".

Remediation:

If the zone has a valid signature but the signature is not trusted, the delegation from the parent zone, or the registrar may not be properly configured. Check with your parent zone administrator or with your name registrar's process to be sure the required information has been provided and that sufficient time has been allowed for new DS record to propagate. Each registrar may have slightly different processes. Generating a DS record from the KSK will likley provide some of the required information.

```
# dnssec-dsfromkey -a SHA-256 Kexample.com.+013+09798.key
example.com. IN DS 9798 13 2 D9AA106E44 . . .
```

8.7 Ensure Signing Keys are Unique (Automated)

Profile Applicability:

- Authoritative Name Server Level 2

Description:

Each zone should have a unique Zone Signing Keys (ZSK) and a unique Key Signing Keys (KSK) that is different from all other keys.

Rationale:

The ZSK key typically has a shorter expiration date then the KSK, and should be unique from the KSK as well as keys used for other zones. If a private key is compromised, the damage is limited to unique key that was disclosed, rather the compromising multiple zones.

Audit:

To verify each key is unique, and has unique zone usage, perform the following:

- The sample command below will extract the public keys from the key files and count the number of occurrences of each key value

```
# find $KEYDIR -name '*.key' |  xargs grep -ho 'DNSKEY .*' | sort |
uniq -c
```

  The count occurrences preceding each key should be one in the output.

- To find the file names of duplicates key values, reuse the find command, but have the grep command report the file names for a given key value. For example:

```
# find $KEYDIR -name '*.key' |  xargs  grep -l ' <key value>
```

Remediation:

To remediate a duplicate key, perform the following:

- Generate a new key to replace the duplicate key using dnssec-keygen and one of the recommended algorithms. An example command is shown below:

```
# dnssec-keygen -a ECDSAP256SHA256 example.org
```

- Implement a rollover period to phase out the duplicate key and replace it with the the newly generated key.
- Once the key is fully deleted from the active use, remove the file.

8.8 Ensure Zones are Signed with NSEC or NSEC3 (Automated)

Profile Applicability:

- Authoritative Name Server Level 2

Description:

The NSEC records are used to prove that a name does not exist, by providing the name before it, and the name after it. NSEC3 records are similar, while using a hash to link records in order to make zone enumeration much more difficult. Either record type will securely validate a negative answer that a name does not exist.

Rationale:

The DNSSEC RRSIG records allows verification of the integrity and authenticity of answers for names which exist. However when the authoritative name server answers that a name does not exist. The nonexistent answer is not signed, and cannot be securely signed. An attacker could take advantage of this by spoofing nonexistent name answers to prevent resolving legitimate names. The NSEC and NSEC3 records provide a means for a DNSSEC validating resolver to verify the authenticity of a nonexistent answer.

Audit:

To audit the authoritative name server, use the delv command on a independent DNSSEC validating resolver to query a nonexistent name of the authoritative name server. The response should be a *"negative response, fully validated"* similar to the following example.

```
$  delv @8.8.8.8 nosuch-name.isc.org
;; resolution failed: ncache nxdomain
; negative response, fully validated
; nosuch-name.isc.org. 3200171710 IN  \-ANY   ;-$NXDOMAIN
; isc.org. SOA ns-int.isc.org. hostmaster.isc.org. 2020031004 7200 3600
24796800
. . .
```

A non-compliant response will not contain the *"fully validated"* response. For example:

```
$ delv @8.8.8.8 nosuch-name.example.com
;;   validating example.com/SOA: no valid signature found
;;   validating example.com/NSEC: no valid signature found
;; resolution failed: ncache nxdomain
; negative response, unsigned answer
; nosuch-name.example.com. 3200171710 IN \-ANY ;-$NXDOMAIN
. . .
```

Remediation:

An NSEC record and NSEC signatures are generated automatically by BIND for DNSSEC signed zones. If the audit fails, then verify that the zone has a valid signature and has delegated trust from the parent domain as in the previous recommendations “8.5 Ensure each Zone has a Valid Digital Signature” and “8.6 Ensure Full Digital Chain of Trust can be Validated” The signed zone file, or a zone transfer can also be checked for NSEC signatures. With a command such as:

```
$ dig @127.0.0.1 example.org AXFR | grep -w 'NSEC'
example.org.    86400   IN      NSEC    www.example.org. A NS SOA RRSIG NSEC
DNSKEY TYPE65534
example.org.    86400   IN      RRSIG    NSEC 8 2 86400 20200325222408
20200224212408 4236
. . .
```

Converting NSEC signing to NSEC3 signing is helpful to prevent zone walking of the linked NSEC records which easily reveal all of the names in a zone. The NSEC3 algorithm creates a linked list of signed hash values, instead of names, to prevent the simple disclosure of all names. The rndc signing command can be used to convert NSEC signing to NSEC3 signing. For example:

```
#  rndc signing -nsec3param 1 0 10  auto example.org
nsec3param request queued
<wait, check the named logs to ensure the zone has been re-signed>
. . .
zone example.org/IN (signed): sending notifies (serial 2020031005)
. . .

# dig @127.0.0.1 example.org AXFR +onesoa | grep NSEC
example.org.    0       IN      RRSIG  NSEC3PARAM 8 2 0 20200422125535
20200323125407 42363 example.org. CCHlbQud0W2XrNlmYHO. . . Kg7ltg==
example.org.    0       IN      NSEC3PARAM 1 0 10 74139101AD2E623E
. . .
```

Default Value:

For signed domains the NSEC records and signatures are generated by default.

References:

1. https://www.dnsinstitute.com/documentation/dnssec-guide/ch07s03.html#recipes-nsec-to-nsec3
2. https://weberblog.net/how-to-walk-dnssec-zones-dnsrecon/
3. https://www.digitalocean.com/community/tutorials/how-to-setup-dnssec-on-an-authoritative-bind-dns-server--2#securing-the-dnssec-setup-from-zone-walking

CIS Controls:

Version 7

8 Malware Defenses

Malware Defenses

9 Operations - Logging, Monitoring and Maintenance

This section provides recommendations for the BIND server configurations related to operations, updates, logging and monitoring.

9.1 Apply Applicable Updates (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

Over time, patches will be released to resolve defects in BIND. It is recommended that such patches be applied soon after they are available based on risk. High risk vulnerabilities should be patched within 30 days of availability.

Rationale:

By ensuring that BIND remains current and patched, the probability of an attacker successfully compromising BIND is reduced.

Audit:

Verify that the latest patch for the platform version of BIND is installed within 30 days of being available. Use the `-v` option to `named` to get the details version information.

```
$ /sbin/named -v
BIND 9.11.4-P2-RedHat-9.11.4-x.x.x
```

Remediation:

Update BIND to the most current revision available. Institute a patch process that aims to apply security updates within 30 days of their release. Subscribe to bind-announce@lists.isc.org on the https://www.isc.org web site to receive notifications of available BIND updates.

Default Value:

Not Applicable

References:

1. https://www.isc.org/

CIS Controls:

Version 6

4 Continuous Vulnerability Assessment and Remediation
Continuous Vulnerability Assessment and Remediation

Version 7

3 Continuous Vulnerability Management
Continuous Vulnerability Management

9.2 Configure a Logging File Channel (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

To capture logs to a local file, setup a channel for the file, in the logging configuration section. It's often helpful to have one log file for security related logs, and a second one with a dynamic severity level to be used as needed for debugging.

Rationale:

Logging security related events is critical for monitoring the security of the server in order to see any issues affecting the server, and to be able to respond to attacks.

Audit:

Perform the following:

- Search the logging options of the configuration file for configured log files

```
# grep -C 4 channel $CONFIG_FILES | egrep '\s+file\s+\"'
file "/var/log/named.log" versions 10 size 20m;
file "/var/log/secure.log" versions 10 size 20m;
```

- Perform a security related event such as a denied zone transfer to generate a log entry.

```
dig @ns2.cisecurity.org cisecurity.org axfr
```

- Check the log file to verify the attempt was logged.

```
tail /var/log/secure.log
30-Sep-2016 08:54:58.664 client 10.11.214.113#38401 (cisecurity.org):
zone transfer 'cisecurity.org/AXFR/IN' denied
```

Remediation:

In `named.conf`, configure a channel for a local security log file with the categories `config`, `dnssec`, `network`, `security`, `updates`, `xfer-in` and `xfer-out`. The local log file will be within the `chroot` directory.

```
logging {
. . .
    channel local_security_log {
        file "/var/run/named/secure.log" versions 10 size 20m;
        severity debug;
        print-time yes;
    };
    // Config file processing
    category config { local_security_log; };
    // Processing signed responses
    category dnssec { local_security_log; };
    // Network Operations
    category network { local_security_log; };
    // Approved or unapproved requests
    category security { local_security_log; };
    // dynamic updates
    category update { local_security_log; };
    // transfers to the name server
    category xfer-in { local_security_log; };
    // transfers from the name server
    category xfer-out { local_security_log; };
    // Optional debug log file, may be enabled dynamically.
    channel local_debug_log {
        file "/var/run/named/debug.log";
        severity dynamic;
        print-time yes;
    };
    category default { local_debug_log; };
    category general { local_debug_log; };
};
```

Default Value:

There is no security log by default.

CIS Controls:

Version 6

6.2 Ensure Audit Log Settings Support Appropriate Log Entry Formatting
Validate audit log settings for each hardware device and the software installed on it, ensuring that logs include a date, timestamp, source addresses, destination addresses, and various other useful elements of each packet and/or transaction. Systems should record logs in a standardized format such as syslog entries or those outlined by the Common Event Expression initiative. If systems cannot generate logs in a standardized format, log normalization tools can be deployed to convert logs into such a format.

Version 7

6.2 Activate audit logging
Ensure that local logging has been enabled on all systems and networking devices.

6.3 Enable Detailed Logging

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

9.3 Configure a Logging Syslog Channel (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

The `syslog` option of the logging configuration allows specification of the syslog facility to send log events. A syslog channel should be configured with the value of `daemon` or other appropriate syslog facility. The `default` and `general` categories should be included and the severity level should be info or lower.

Rationale:

Configuring a syslog channel allows BIND to log important information via the standard system syslog facility. It is important that the BIND logs be included with the system monitoring and response that is performed on other system logs, and the syslog facility is helpful to ensure that the important log information isn't lost, or ignored.

Audit:

Search the configuration file for a syslog logging channel, as shown below.

```
# grep -C 3 channel $CONFIG_FILES | egrep '\s+syslog\s+'
                syslog daemon;       # send to syslog's daemon facility
```

Usage of the syslog facility *daemon* is common practice, but other facilities may be configured.

Remediation:

Configure a syslog channel to capture at least the default and general categories of log events. For external authoritative name servers, the category `lame-servers` may be redirect to null, so that it is not logged. Using lame name servers is common for the domains used for SPAM and may overload the log with information that is not very useful.

```
logging {
. . .
        // Syslog
        channel default_syslog {
                syslog daemon;       # send to syslog's daemon facility
                severity info;       # only send priority info and higher
        };
        category default { default_syslog; };
        category general { default_syslog; };
        // Too many lame servers, especially from SPAM
        category lame-servers { null; };
```

Default Value:

There is no syslog channel by default.

CIS Controls:

Version 6

6.6 Deploy A SIEM OR Log Analysis Tools For Aggregation And Correlation/Analysis
Deploy a SIEM (Security Information and Event Management) or log analytic tools for log aggregation and consolidation from multiple machines and for log correlation and analysis. Using the SIEM tool, system administrators and security personnel should devise profiles of common events from given systems so that they can tune detection to focus on unusual activity, avoid false positives, more rapidly identify anomalies, and prevent overwhelming analysts with insignificant alerts.

Version 7

6.6 Deploy SIEM or Log Analytic tool
Deploy Security Information and Event Management (SIEM) or log analytic tool for log correlation and analysis.

6.8 Regularly Tune SIEM
On a regular basis, tune your SIEM system to better identify actionable events and decrease event noise.

9.4 Disable the HTTP Statistics Server (Automated)

Profile Applicability:

- Authoritative Name Server Level 1
- Caching Only Name Server Level 1

Description:

Starting in BIND 9.5.0 there was a new statistics web server included, that is a useful debugging tool in a non-production environment. The HTTP server provides data in XML format about the condition of a BIND 9 server. The statistics server provides the same statistics that are available to the statistics-file dump. This server should be left disabled.

Rationale:

A production name server should not have additional, unnecessary services running, as the additional services increases the risk of vulnerabilities.

Audit:

Verify that there is NOT a statistics channel statement:

```
# grep statistics-channel $CONFIG_FILES
```

No output is expected and confirms that the HTTP service is not enabled.

Remediation:

Remove the `statistics-channel` option from the configuration file.

Default Value:

The HTTP server is disabled by default.

CIS Controls:

Version 6

9.1 Limit Open Ports, Protocols, and Services
Ensure that only ports, protocols, and services with validated business needs are running on each system.

Version 7

9.2 Ensure Only Approved Ports, Protocols and Services Are Running

Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.

9.5 Response Rate Limiting and DDOS Mitigation (Automated)

Profile Applicability:

- Authoritative Name Server Level 1

Description:

Responses to excessive, nearly identical UDP requests can be controlled by configuring a response rate-limit clause in an options or view statement. At this time, Response Rate Limiting is only recommended for authoritative servers.

Rationale:

The response rate limiting mechanism keeps an authoritative name server from being effectively used to amplify reflected distributed denial of service (DDoS) attacks. Short truncated responses will be sent when the rate-limited is exceeded. Legitimate non-spoofed clients will react to a dropped or truncated response by retrying with UDP or with TCP respectively. While the truncated or dropped responses to spoofed requests intended will greatly diminished the effectiveness of the attack.

Audit:

To determine if the recommended state is implemented, check that the authoritative name server configuration file has an options statement with a rate-limit clause that sets the responses-per-second to a value of `5` or less. Keep in mind, the responses per second value only limits the responses to nearly identical requests from the same IP address.

Remediation:

To implement the recommended state, add or update a rate-limit clause in the server's options statement. Add a responses-per-second value of `5` or less, similar to the example below.

```
options {
 . . .
   rate-limit {
      // Limit Response to Rapid Identical Queries for DDOS mitigation
      responses-per-second 5;
   . . .
};
```

Default Value:

Default value is `0` or no rate limit.

References:

1. https://kb.isc.org/docs/aa-00994
2. https://www.us-cert.gov/ncas/alerts/TA13-088A
3. https://kb.isc.org/docs/aa-01148

CIS Controls:

Version 7

8 Malware Defenses
Malware Defenses

9.6 Ensure Signing Keys are Scheduled to be Replaced Periodically (Automated)

Profile Applicability:

- Authoritative Name Server Level 2

Description:

Implement a periodic key rollover process for both the Zone Signing Keys (ZSK) and the Key Signing Keys (KSK). The ZSK should be replaced within 2 years or less. The KSK should be replaced within 6 years or less. Keys are replaced by generating a new key before the existing key expires, and scheduling a rollover date when the new key will phase out and replace the old key.

Rationale:

Cryptographic keys like passwords need to be periodically replaced. By using strong key algorithms and appropriately long bit lengths, the lifetime for keys can be longer than a generally recommended for passwords. Typically, the Zone Signing Keys are rolled over more frequently than the Key Signing Keys.

Audit:

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

- Locate all of the ZSK key files for the name server by searching for the `256` key code using the following command.

```
# ZKEYS=$(find $KEYDIR -name '*.key' | xargs grep -l 'DNSKEY 256 3' )
```

- Check that the ZSK activation date is more recent then 2 years prior to the current date with the following commands. To automate the two year date calculation the -u option could be used for `dnssec-settime` to report the date in seconds since the start of the UNIX epoch.

```
# for zk in $ZKEYS; do echo -n "$zk:  "; dnssec-settime -pA $zk; done
./Kcisecurity.com.+013+45248.key:  Activate: Mon Mar  2 16:35:58 2020
```

- Locate all of the KSK key files for the name server by searching for the `257` key code using the following command.

```
# KKEYS=$(find $KEYDIR -name '*.key' | xargs grep -l 'DNSKEY 257 3' )
```

- Check that the KSK Activation date is more recent than six years prior to the current date. With the following command.

```
# for kk in $KKEYS; do echo -n "$kk:  "; dnssec-settime -pA $kk; done
./Kcisecurity.com.+013+45248.key:  Activate: Mon Mar  2 16:35:58 2020
```

If all ZSK activation dates are less than two years prior, and the KSK activation dates are less than six years prior, than the server is compliant.

Remediation:

To replace an aged key, perform the following:

- Generate a new key to replace the old key using `dnssec-keygen` and one of the recommended algorithms. An example command is shown below:

```
# dnssec-keygen -a ED25519 example.org
# dnssec-keygen -a ED25519 -f KSK example.org
```

- Implement a rollover period to phase out the old key and replace it with the newly generated key. The older key should have dates set for the keys to be inactive and then deleted.

```
# dnssec-settime -I +30d -D +60d  Kexample.org.+013+46651.key
```

- Once the date for key deletion has passed, and the key is no longer included in the zone, then remove the key files.

Default Value:

Signing key rollover is NOT implemented by default.

References:

1. https://www.dnsinstitute.com/documentation/dnssec-guide/ch06s04.html
2. https://downloads.isc.org/isc/dnssec-guide/dnssec-guide.pdf
3. https://tools.ietf.org/html/rfc7583

10 Enable SELinux to Restrict BIND Processes

Recommendations in this section provide mandatory access controls (MAC) using the SELinux kernel module in targeted mode. SELinux provides additional enforced security which will prevent access to resources, files and directories by the named processes even in cases where an application or server vulnerability might allow inappropriate access. The SELinux controls are advanced security controls that require significant effort to ensure they do not negatively impact the application and/or site functionality. It is highly recommended that the configuration states described in this section be tested thoroughly on test servers prior to deploying them to production servers. Depending on which Linux distribution is used, SELinux may already be installed or readily available as packages.

The recommendation "Isolate BIND with chroot'ed Subdirectory" and this section "Enable SELinux to Restrict BIND Processes" provide similar controls, in that the DNS service is prevented from accessing and modifying inappropriate files. SELinux goes well beyond what the chroot is able to prevent, however for audit purposes either control, the chroot'd subdirectory or the SELinux in enforcing mode is sufficient.

10.1 Ensure SELinux Is Enabled in Enforcing Mode (Automated)

Profile Applicability:

- Caching Only Name Server Level 2
- Authoritative Name Server Level 2

Description:

SELinux (Security-Enhanced Linux) is a Linux kernel security module that provides mandatory access control security policies with type enforcement that are checked after the traditional discretionary access controls. It was created by the US National Security Agency and can enforce rules on files and processes in a Linux system, and restrict actions, based on defined policies.

Rationale:

DNS servers act as a foundation for most of the internet and internal traffic. Web and mobile applications, email, cloud services and VPN connections, internal LAN connections all depend on DNS to translate names and route traffic to the correct destination. With DNS being such a critical service, it is a ripe target for attacks which may allow black-hat criminals to gain access to information and servers. The threat is especially high because DNS servers are often externally accessible and continue to have serious vulnerabilities. The SELinux mandatory access controls provide a much stronger security model which can be used to implement a deny-by-default model which only allows what is explicitly permitted.

Audit:

Perform the following steps to determine if the recommended state is implemented:
Use the *sestatus* command to check that SELinux is enabled and that both the current mode and the configured mode are set to `enforcing`.

```
$ sestatus | grep -i mode
Current mode:                   enforcing
Mode from config file:          enforcing
```

If there is no output, or both modes are not shown as enforcing, then the configuration is not compliant.

Note: The chroot'd subdirectory may be used as an audit alternative to the SELinux recommendations.

Remediation:

Perform the following to implement the recommended state:

If SELinux is not enabled in the configuration file, edit the file `/etc/selinux/config` and set the value of SELINUX as `enforcing` and reboot the system for the new configuration to be effective.

```
SELINUX=enforcing
```

If the current mode is not enforcing, and an immediate reboot is not possible, the current mode can be set to enforcing with the `setenable` command shown below.

```
# setenforce 1
```

Default Value:

SELinux is enforcing by default on some Linux distributions such as Red Hat Enterprise Linux 8.

References:

1. https://en.wikipedia.org/wiki/Security-Enhanced_Linux

CIS Controls:

Version 6

14.4 Protect Information With Access Control Lists
All information stored on systems shall be protected 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.

Version 7

14.6 Protect Information through Access Control Lists
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.

14.7 Enforce Access Control to Data through Automated Tools
Use an automated tool, such as host-based Data Loss Prevention, to enforce access controls to data even when data is copied off a system.

10.2 Ensure BIND Processes Run in the named_t Confined Context Type (Automated)

Profile Applicability:

- Caching Only Name Server Level 2

- Authoritative Name Server Level 2

Description:

SELinux includes customizable targeted policies that may be used to confine the BIND named server to enforce least privileges so that the server has only the minimal access to specified directories, files and network ports. Access is controlled by process types (domains) defined for the named process. There are about a dozen related types defined in a default named SELinux policy. The default SELinux policies work well for a default BIND installation, however testing of the SELinux polices with the specific BIND operations is highly recommended. All directories and files to be accessed by the named server process must have security labels with appropriate types. The following file context types are a sample of the most commonly used:

- `named_cache_t` - Directories and files with dynamically updated contents
- `named_conf_t` - Directories and Configuration files to be read, but not updated
- `named_exec_t` - BIND related Executables

The `seinfo` may be used list the types that are configured. For example, the following will list the relevant types that begin with named_

```
# seinfo -t | grep ' named_'
```

The `semanage fcontext` command may be used to list file context mapping. For example:

```
# semanage fcontext -l | grep ':named_'
```

Rationale:

With the proper implementation of SELinux, vulnerabilities in the BIND named server may be prevented from being exploited due to the additional restrictions. For example, a vulnerability that allows an attacker to read inappropriate system files may be prevented from execution by SELinux because the inappropriate files are not labeled with necessary `named` specific context. Likewise writing to an unexpected directory or execution of unexpected content can be prevented by similar mandatory security labels enforced by SELinux.

Audit:

Perform the following steps to determine if the recommended state is implemented:
Check that the BIND named process is confined to the `named_t` SELinux context. The type is listed in the third colon separated field and should be `named_t`.

```
# ps -eZ | grep named
system_u:system_r:named_t:s0       1792 ?        00:00:00 named
```

The service is non-compliant if the output does not show a process type or shows a process type other than `named_t`.

Remediation:

If the running named process is not confined to the `named_t` SELinux context. Then check the labeled context for the named binaries and set the binary files to have a context of `named_exec_t` as shown below. The `named-checkconf` executable should have `named_checkconf_exec_t` type.

```
# ls -Z  /usr/sbin/named /usr/sbin/named-checkconf /usr/sbin/unbound-anchor
          system_u:object_r:named_exec_t:s0 /usr/sbin/named
system_u:object_r:named_checkconf_exec_t:s0 /usr/sbin/named-checkconf
          system_u:object_r:named_exec_t:s0 /usr/sbin/unbound-anchor
```

If the executable files are not labeled correctly, they may be relabeled with the `chcon` command, as shown, however the file system labeling is based on the SELinux file context mapping polices and the file systems will on some occasions be relabeled according to the policy.

```
# chcon -t named_exec_t /usr/sbin/named /usr/sbin/unbound-anchor
# chcon -t named_checkconf_exec_t /usr/sbin/named-checkconf
```

Since the file system may be relabeled based on SELinux policy, it's best to check the SELinux policy with `semanage fcontext -l` option. If the policy is not present, then add the pattern to the policy using the `--add` option. The `restorecon` command shown below will restore the file context label according to the current policy, which is required if a pattern was added.

```
# ### Check the Policy
# semanage fcontext -l | grep '/usr/sbin/named*'
/usr/sbin/named            regular file system_u:object_r:named_exec_t:s0
/usr/sbin/named-checkconf  regular file
system_u:object_r:named_checkconf_exec_t:
s0

# semanage fcontext -l | grep /usr/sbin/unbound-anchor
/usr/sbin/unbound-anchor   regular file system_u:object_r:named_exec_t:s0
```

```
# ### Add to the policy, if not present
# semanage fcontext --add -f f -t named_exec_t /usr/sbin/named
# semanage fcontext --add -f f -t named_exec_t /usr/sbin/unbound-anchor
# semanage fcontext --add -f f -t named_checkconf_exec_t /usr/sbin/named-
checkconf

# ### Restore the file labeling accord to the SELinux policy
# restorecon -v /usr/sbin/named /usr/sbin/named-checkconf /usr/sbin/unbound-
anchor
```

Restarting the BIND named service will also be required.

Default Value:

The `name_t` is the default type for ISC BIND named if SELinux is enabled.

References:

1. https://wiki.centos.org/HowTos/SELinux

CIS Controls:

Version 6

14.4 Protect Information With Access Control Lists
All information stored on systems shall be protected 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.

Version 7

14.6 Protect Information through Access Control Lists
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.

10.3 Ensure the named_t Process Type is Not in Permissive Mode (Automated)

Profile Applicability:

- Caching Only Name Server Level 2
- Authoritative Name Server Level 2

Description:

In addition to setting the entire SELinux configuration in permissive mode, it is possible to set individual process types (domains) such as `named_t` into a permissive mode as well. The permissive mode will not prevent any access or actions, instead, any actions that would have been denied are simply logged.

Rationale:

Usage of the permissive mode is helpful for testing and ensuring that SELinux will not prevent access that is necessary for the proper function of the DNS server. However, inappropriate access will not be prevented in permissive mode by SELinux.

Audit:

Perform the following steps to determine if the recommended state is implemented:
Check that the `named_t` process type (domain) is not in permissive mode with the `semodule` command. There should be no output if the type is not set to `permissive`.

```
# semodule -l | grep permissive_named_t
```

Remediation:

Perform the following to implement the recommended state:

If the `named_t` type is in permissive mode; the customized permissive mode should be deleted with the following `semanage` command.

```
# semanage permissive -d named_t
```

Default Value:

The `named_t` type is not in permissive mode by default.

References:

1. https://man7.org/linux/man-pages/man8/semanage-permissive.8.html

CIS Controls:

Version 7

14.6 Protect Information through Access Control Lists
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.

10.4 Ensure Only the Necessary SELinux Booleans are Enabled (Automated)

Profile Applicability:

- Caching Only Name Server Level 2

- Authoritative Name Server Level 2

Description:

SELinux booleans allow or disallow specific behaviors. There are two boolean variables specific to the ISC BIND DNS server:

- **`named_tcp_bind_http_port`** - Allow named to tcp bind http port
- **`named_write_master_zones`** - Allow named to write master zones

The `named_tcp_bind_http_port` would allow enabling the BIND statistics http channel which is not recommended. The `named_write_master_zones` allows BIND to update the master files, which is necessary when dynamic updates are performed, or the server is automatically maintaining DNSSEC digital signatures.

Rationale:

Enabling only the necessary named related booleans provides a defense in depth approach, that will deny actions that are not in use or expected.

Audit:

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

Use `getsebool` to verify the `named_tcp_bind_http_port` boolean has a value of off.

```
# getsebool named_tcp_bind_http_port
named_tcp_bind_http_port --> off
```

The `named_write_master_zones` boolean is not audited as many BIND servers will require the boolean to be enabled.

Remediation:

Perform the following to implement the recommended state:

Disable the SELinux boolean using the `setsebool` command as shown below with the `-P` option to make the change persistent.

```
# setsebool -P named_tcp_bind_http_port off
```

Default Value:

The default value for `named_tcp_bind_http_port` is `off`.

CIS Controls:

Version 7

9.2 Ensure Only Approved Ports, Protocols and Services Are Running
Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.

Appendix: Summary Table

Control		Set Correctly	
		Yes	No
1	**Planning and Architecture**		
1.1	Use a Split-Horizon Architecture (Manual)	□	□
1.2	Do Not Install a Multi-Use System (Manual)	□	□
1.3	Dedicated Name Server Role (Automated)	□	□
1.4	Use Secure Upstream Caching DNS Servers (Manual)	□	□
1.5	Installing ISC BIND 9 (Automated)	□	□
2	**Restricting Permissions and Ownership**		
2.1	Run BIND as a non-root User (Automated)	□	□
2.2	Give the BIND User Account an Invalid Shell (Automated)	□	□
2.3	Lock the BIND User Account (Automated)	□	□
2.4	Set root Ownership of BIND Directories (Automated)	□	□
2.5	Set root Ownership of BIND Configuration Files (Automated)	□	□
2.6	Set Group named or root for BIND Directories and Files (Automated)	□	□
2.7	Set Group Read-Only for BIND Files and Non-Runtime Directories (Automated)	□	□
2.8	Set Other Permissions Read-Only for All BIND Directories and Files (Automated)	□	□
2.9	Isolate BIND with chroot'ed Subdirectory (Automated)	□	□
3	**Restricting Queries**		
3.1	Ignore Erroneous or Unwanted Queries (Automated)	□	□
3.2	Restrict Recursive Queries (Automated)	□	□
3.3	Restrict Query Origins (Manual)	□	□
3.4	Restrict Queries of the Cache (Automated)	□	□
4	**Transaction Signatures -- TSIG**		
4.1	Use TSIG Keys 256 Bits in Length (Automated)	□	□
4.2	Include Cryptographic Key Files (Automated)	□	□
4.3	Use Unique Keys for Each Pair of Hosts (Automated)	□	□
4.4	Restrict Access to All Key Files (Automated)	□	□
4.5	Protect TSIG Key Files During Deployment (Manual)	□	□
5	**Authenticate Zone Transfers and Updates**		
5.1	Securely Authenticate Zone Transfers (Automated)	□	□
5.2	Securely Authenticate Dynamic Updates (Automated)	□	□
5.3	Securely Authenticate Update Forwarding (Automated)	□	□
6	**Information Leakage**		
6.1	Hide BIND Version String (Automated)	□	□

6.2	Hide Nameserver ID (Automated)	□	□
7	**Secure Network Communications**		
7.1	Do Not Define a Static Source Port (Automated)	□	□
7.2	Enable DNSSEC Validation (Automated)	□	□
7.3	Disable the dnssec-accept-expired Option (Automated)	□	□
7.4	Ensure Either SPF or DKIM DNS Records are Configured (Automated)	□	□
8	**DNSSEC Digital Signatures for Authoritative Zones**		
8.1	Install the haveged Package for Enhanced Entropy (Automated)	□	□
8.2	Ensure Signing Keys are Generated with a Secure Algorithm (Automated)	□	□
8.3	Ensure Any Signing Keys using RSA Have a Length of 2048 or Greater (Automated)	□	□
8.4	Restrict Access to Zone and Key Signing Keys (Automated)	□	□
8.5	Ensure each Zone has a Valid Digital Signature (Manual)	□	□
8.6	Ensure Full Digital Chain of Trust can be Validated (Automated)	□	□
8.7	Ensure Signing Keys are Unique (Automated)	□	□
8.8	Ensure Zones are Signed with NSEC or NSEC3 (Automated)	□	□
9	**Operations - Logging, Monitoring and Maintenance**		
9.1	Apply Applicable Updates (Automated)	□	□
9.2	Configure a Logging File Channel (Automated)	□	□
9.3	Configure a Logging Syslog Channel (Automated)	□	□
9.4	Disable the HTTP Statistics Server (Automated)	□	□
9.5	Response Rate Limiting and DDOS Mitigation (Automated)	□	□
9.6	Ensure Signing Keys are Scheduled to be Replaced Periodically (Automated)	□	□
10	**Enable SELinux to Restrict BIND Processes**		
10.1	Ensure SELinux Is Enabled in Enforcing Mode (Automated)	□	□
10.2	Ensure BIND Processes Run in the named_t Confined Context Type (Automated)	□	□
10.3	Ensure the named_t Process Type is Not in Permissive Mode (Automated)	□	□
10.4	Ensure Only the Necessary SELinux Booleans are Enabled (Automated)	□	□

Appendix: Change History

Date	Version	Changes for this version
Oct 23, 2020	1.0.0	Initial Release