CIS Benchmarks

CIS MongoDB 3.6 Benchmark

v1.0.0 - 12-31-2019

Terms of Use

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

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

Table of Contents

Overview

This document, CIS MongoDB 3.6 Benchmark, provides prescriptive guidance for establishing a secure configuration posture for MongoDB version 3.6. This guide was tested against MongoDB 3.6 running on Ubuntu Linux and Windows but applies to other distributions as well. 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 to us at feedback@cisecurity.org.

Extracting Running Configuration File

To verify the MongoDB running configuration file we need to connect MongoDB instance using MongoDB client with valid username/password and execute this command:

```
db.runCommand( { getCmdLineOpts: 1 } )
```

The response will contain MongoDB running configuration file location. For example:

```
"config" : "/user/data/mongod.conf",
```

Important Information

- Mongod: The primary daemon process for the MongoDB system. It handles data requests, manages data access, and performs background management operations.
- Mongos: mongos is a routing service for MongoDB Sharded Clusters. mongos requires mongod config, which stores the metadata of the cluster. MongoDB Shard Utility, the controller and query router for sharded clusters. Sharding partitions the data-set into discrete parts.

Intended Audience

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

Consensus Guidance

This benchmark was created using a consensus review process comprised of subject matter experts. Consensus participants provide perspective from a diverse set of

backgrounds including consulting, software development, audit and compliance, security research, operations, government, and legal.

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

Typographical Conventions

The following typographical conventions are used throughout this guide:

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

Scoring Information

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

Scored

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

Not Scored

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

Profile Definitions

The following configuration profiles are defined by this Benchmark:

- **Level 1- MongoDB**

  Items in this profile apply to MongoDB and intend to:

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

- **Level 2 - MongoDB**

  This profile extends the “Level 1 - MongoDB” profile. Items in this profile apply to MongoDB and exhibit one or more of the following characteristics:

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

Acknowledgements

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

Author
Vinesh Redkar - Security Researcher
Pralhad Chaskar - Security Researcher

Editor
Tim Harrison - Center for Internet Security

Recommendations

1 Installation and Patching

This section provides guidance on ensuring that the MongoDB software is up to date to eliminate easily avoidable vulnerabilities.

1.1 Ensure the appropriate MongoDB software version/patches are installed (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

The MongoDB installation version, along with the patch level, should be the most recent that is compatible with the organization's operational needs.

Rationale:

Using the most recent MongoDB software version along with all applicable patches, helps limit the possibilities for vulnerabilities in the software. The installation version and/or patches applied should be selected according to the needs of the organization. At a minimum, the software version should be supported.

Audit:

On Ubuntu:
Run the following command from within the MongoDB shell to determine if the MongoDB software version complies with your organization’s operational needs:

```
> db.version()
```

On Windows:
Navigate to the Installation directory of Mongo DB on the server and run below command

```
mongod.exe --version
```

Remediation:

Upgrade to the latest version of the MongoDB software:

1. Backup the data set.
2. Download the binaries for the latest MongoDB revision from the MongoDB Download Page and store the binaries in a temporary location. The binaries download as compressed files that extract to the directory structure used by the MongoDB installation.
3. Shutdown the MongoDB instance.
4. Replace the existing MongoDB binaries with the downloaded binaries.
5. Restart the MongoDB instance.

Default Value:

Patches are not installed by default.

References:

1. https://docs.mongodb.com/v3.6/tutorial/upgrade-revision/
2. https://docs.mongodb.com/v3.6/release-notes/
3. https://www.mongodb.com/download-center#community
4. https://www.mongodb.com/support-policy

CIS Controls:

Version 6

4 Continuous Vulnerability Assessment and Remediation
Continuous Vulnerability Assessment and Remediation

Version 7

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 Authentication

This section contains recommendations for requiring authentication before allowing access to the MongoDB database.

2.1 Ensure Authentication is configured (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

This setting ensures that all clients, users, servers are required to authenticate before being granted access to the MongoDB database.

Authentication is the process of verifying the identity of a client. When access control, i.e. authorization, is enabled, MongoDB requires all clients to authenticate themselves in order to determine their access.

To authenticate as a user, you must provide a username, password, and the authentication database associated with that user.

Rationale:

Failure to authenticate clients, users, servers can enable unauthorized access to the MongoDB database and can prevent tracing actions back to their sources.

Audit:

Run the following command to verify whether an authorization is enabled on the MongoDB server.
On Ubuntu:

```
cat /etc/mongod.conf | grep "authorization"
```

On Windows:

```
type mongod.conf | findstr "authorization"
```

The value for `authorization` must be set to `enabled`.
To authenticate using the mongo shell use the following approach

- Use the mongo command-line authentication options (`--username`, `--password`, and `--authenticationDatabase`) when connecting to the mongod or mongos instance

  Or

- Connect first to the mongod or mongos instance, and then run the authenticate command or the `db.auth()` method against the authentication database.

Remediation:

The authentication mechanism should be implemented before anyone accesses the MongoDB Server.
To enable the authentication mechanism:

- Start the MongoDB instance without authentication.

```
mongod --port 27017 --dbpath /data/db1
```

  Or

```
mongod.exe --port 27017 --dbpath db1
```

- Create the system user administrator, ensuring that its password meets organizationally-defined password complexity requirements.

```
use admin
db.createUser(
  {
    user: "siteUserAdmin",
    pwd: "password",
    roles: [ { role: "userAdminAnyDatabase", db: "admin" } ]
  }
)
```

- Open `mongod.conf` and change for authorization value to enabled:

```
security:
    authorization: "enabled"
```

- Restart the MongoDB instance

```
service mongodb restart
```

Default Value:

By default, authorization is set to disable.

References:

1. https://docs.mongodb.com/v3.6/core/authentication/

CIS Controls:

Version 6

16 Account Monitoring and Control
Account Monitoring and Control

Version 7

16.3 Require Multi-factor Authentication
Require multi-factor authentication for all user accounts, on all systems, whether managed onsite or by a third-party provider.

2.2 Ensure that MongoDB does not bypass authentication via the localhost exception (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

MongoDB should not be set to bypass authentication via the localhost exception. The localhost exception allows the user to enable authorization before creating the first user in the system. When active, the localhost exception allows all connections from the localhost interface to have full access to that instance. The exception applies only when there are no users created in the MongoDB instance.

Note: This recommendation only applies when there are no users created in the MongoDB instance.

Rationale:

Disabling this exception will prevent unauthorized local access to the MongoDB database. It will also ensure the traceability of each database activity to a specific user. Localhost Exception allows direct connect to Mongod's without any UN/PW.

Audit:

Run the following command to extract the information about `enableLocalhostAuthBypass` setting on Configuration File.

On Ubuntu:

```
cat /etc/mongod.conf |grep "enableLocalhostAuthBypass"
```

On Windows:

```
type mongod.conf | findstr "enableLocalhostAuthBypass"
```

The value for `enableLocalhostAuthBypass` must be `false`.

Remediation:

To disable local authentication on the MongoDB database.

Type OS Console Command

```
mongod --setParameter enableLocalhostAuthBypass=0
```

or

To manually configure use the `setParameter` option in the mongo configuration file to set it to `false`.

```
setParameter:
  enableLocalhostAuthBypass: false
```

Default Value:

By default, localhost exception value (`enableLocalhostAuthBypass`) is `true`.

References:

1. https://docs.mongodb.com/v3.6/reference/parameters/#param.enableLocalhostAuthBypass

CIS Controls:

Version 6

16 Account Monitoring and Control
Account Monitoring and Control

Version 7

16.3 Require Multi-factor Authentication
Require multi-factor authentication for all user accounts, on all systems, whether managed onsite or by a third-party provider.

2.3 Ensure authentication is enabled in the sharded cluster (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

Authentication is enabled in a sharded cluster when the certificate or key files are created and configured for all components. This ensures that every client that accesses the cluster must provide credentials, to include MongoDB instances that access each other within the cluster.

With keyfile authentication, each mongod or mongos instance in the sharded cluster uses the contents of the keyfile as the shared password for authenticating other members in the deployment. Only mongod or mongos instances with the correct keyfile can join the sharded cluster.

For Production Environment: x.509 certificate authentication with secure TSL/SSL connection must be used for authentication.

For Development Purpose: Key file can be used as an authentication mechanism between the shared cluster. Keyfiles are bare-minimum forms of security and are best suited for testing or development environments.

Rationale:

Enforcing a key or certificate on a sharded cluster prevents unauthorized access to the MongoDB database and provides traceability of database activities to a specific user or component. A MongoDB sharded cluster can enforce user authentication as well as internal authentication of its components to secure against unauthorized access.

Audit:

Based on recommendations

- `PEMKeyFile`, `clusterFile`, `CAFile`must be configured.
- `clusterAuthMode`should be set to `x509`
- `authenticationMechanisms`should be set to `MONGODB-X509`.

Run the following command to verify that the certificate-based authentication is configured:

On Ubuntu:

```
cat /etc/mongod.conf | grep “PEMKeyFile”
cat /etc/mongod.conf | grep "CAFile"
cat /etc/mongod.conf | grep "clusterFile"
cat /etc/mongod.conf | grep "clusterAuthMode"
cat /etc/mongod.conf | grep “authenticationMechanisms:”
```

On Windows:

```
type mongod.conf | findstr “PEMKeyFile”
type mongod.conf | findstr “CAFile”
type mongod.conf | findstr “clusterFile”
type mongod.conf | findstr “clusterAuthMode”
type mongod.conf | findstr “authenticationMechanisms:”
```

Run the following command to verify that the `key` file-based authentication is configured: (*Only for Development Purpose*)

On Ubuntu:

```
cat /etc/mongod.conf | grep “keyFile=”
```

On Windows:

```
type mongod.conf | findstr “keyFile”
```

Remediation:

To authenticate to servers, clients can use x.509 certificates instead of usernames and passwords. MongoDB supports x.509 certificate authentication for use with a secure TLS/SSL connection. The x.509 client authentication allows clients to authenticate to servers with certificates rather than with a username and password.

Change the configuration file `/etc/mongod.conf` on each host, adding the following rows:

```
net:
   port: 27017
   ssl:
      mode: requireSSL
      PEMKeyFile: /etc/mongodb/ssl/server1.pem
      CAFile: /etc/mongodb/ssl/mongoCA.crt
      clusterFile: /etc/mongodb/ssl/server1.pem
   security:
      authorization: enabled
      clusterAuthMode: x509
```

Restart the daemon

```
sudo service mongodb restart
```

To enable authentication in the sharded cluster, perform the following steps:(*Only for Development Purpose*)

- Generate A Key File
- On each component in the shared cluster, enable authentication by editing the configuration file `/etc/mongod.conf`. Set the `keyFile` option to the key file's path and then start the component with this command:

```
keyFile = /srv/mongodb/keyfile
```

- When starting the component, set `--keyFile` option, which is an option for both `mongos` instances and `mongod` instances. Set the `--keyFile` to the key file's path.

Default Value:

Not configured

References:

1. https://docs.mongodb.com/v3.6/tutorial/enforce-keyfile-access-control-in-existing-sharded-cluster-no-downtime/
2. https://docs.mongodb.com/v3.6/tutorial/enforce-keyfile-access-control-in-existing-sharded-cluster/
3. https://docs.mongodb.com/v3.6/tutorial/configure-x509-member-authentication/

CIS Controls:

Version 6

16 Account Monitoring and Control
Account Monitoring and Control

Version 7

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

3 Access Control

This section contains recommendations for restricting access to MongoDB systems.

3.1 Ensure that Role-based access control (RBAC) is enabled and configured (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

Role-based access control (RBAC) is a method of regulating access to resources based on the roles of individual users within an enterprise. A user is granted one or more roles that determine the user's access to database resources and operations. Outside of role assignments, the user has no access to the system. MongoDB can use RBAC to govern access to MongoDB systems. MongoDB does not enable authorization by default.

Rationale:

When properly implemented, RBAC enables users to carry out a wide range of authorized tasks by dynamically regulating their actions according to flexible functions. This allows an organization to control employees' access to all database tables through RBAC.

Audit:

Connect to MongoDB with the appropriate privileges and run the following command:

```
mongo --port 27017 -u <username> -p password --authenticationDatabase
<database_name>
```

Identify users' roles and privileges:

```
> db.getUser()
> db.getRole()
```

Verify that the appropriate role or roles have been configured for each user.

Remediation:

1. Establish roles for MongoDB.
2. Assign the appropriate privileges to each role.
3. Assign the appropriate users to each role.

4. Remove any individual privileges assigned to users that are now addressed by the roles.
5. See the reference below for more information.

References:

1. https://docs.mongodb.com/v3.6/tutorial/manage-users-and-roles/
2. https://docs.mongodb.com/v3.6/core/authorization/

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.

3.2 Ensure that MongoDB only listens for network connections on authorized interfaces (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

Ensuring that MongoDB runs in a trusted network environment involves limiting the network interfaces on which MongoDB instances listen for incoming connections. Any untrusted network connections should be dropped by MongoDB.

Firewalls allow administrators to filter and control access to a system by providing granular control over network communications. For administrators of MongoDB, the following capabilities are important:

- Limiting incoming traffic on a specific port to specific systems
- Limiting incoming traffic from untrusted hosts.

On Linux systems, the `iptables` interface provides access to the underlying `netfilter` firewall. On Windows systems, `netsh` command line interface provides access to the underlying Windows Firewall.

Rationale:

This configuration blocks connections from untrusted networks, leaving only systems on authorized and trusted networks able to attempt to connect to the MongoDB. If not configured, this may lead to unauthorized connections from untrusted networks to MongoDB.

Audit:

On Ubuntu:

1. Verify that network exposure is limited, review the settings in the MongoDB configuration file:

```
cat /etc/mongod.conf |grep -A12 "net" | grep "bindIp"
```

2. Verify the relevant network settings on the Linux system itself:

```
iptables -L
```

On Windows:

```
type mongod.conf | findstr "bindIp"
```

Remediation:

Configure the MongoDB configuration file to limit its exposure to only the network interfaces on which MongoDB instances should listen for incoming connections.

Default Value:

Not configured

References:

1. https://docs.mongodb.com/v3.6/tutorial/configure-linux-iptables-firewall/
2. https://docs.mongodb.com/v3.6/tutorial/configure-windows-netsh-firewall/
3. https://docs.mongodb.com/v3.6/core/security-network/

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 Ensure that MongoDB is run using a Least Privileges, dedicated service account (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

The MongoDB service should not be run using a privileged account such as 'root' because this unnecessarily exposes the operating system to high risk.

This setting ensures that the monogd service runs as a least-privileged user.

Rationale:

Using a non-privileged, dedicated service account restricts the database from accessing the critical areas of the operating system which are not required by MongoDB. This will also mitigate the potential for unauthorized access via a compromised, privileged account on the operating system.

Anyone who has been a victim of viruses, worms, and other malicious software (malware) will appreciate the security principle of “least privilege.” If all processes ran with the minial set of privileges needed to perform the user's tasks, it would be more difficult for malware to infect a machine and propagate to other machines.

Audit:

Run the following command to get listing of all mongo instances, the PID number, and the PID owner.

On Ubuntu:

```
ps -ef | grep -E "mongos|mongod"
ps -aef | grep mongod
```

On Windows:

```
TASKLIST /V | findstr mongo
```

Remediation:

1. Create a user which is only used for running Mongodb and directly related processes. This user must not have administrative rights to the system. Steps to create user

```
useradd -m -d /home/mongodb -s /bin/bash -g mongodb -u 1234 mongodb
```

2. Set the Database data files, the keyfile, and the SSL private key files to only be readable by the mongod/mongos user and then set ownership to mongodb user only

```
sudo chown -R mongodb:mongodb /data/db
```

3. Set the log files to only be writable by the mongod/mongos user and readable only by root.

Default Value:

Not configured

References:

1. https://docs.mongodb.com/v3.6/tutorial/manage-users-and-roles/

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

4.3 Ensure the Use of Dedicated Administrative Accounts

Ensure that all users with administrative account access use a dedicated or secondary account for elevated activities. This account should only be used for administrative activities and not internet browsing, email, or similar activities.

3.4 Ensure that each role for each MongoDB database is needed and grants only the necessary privileges (Scored)

Profile Applicability:

- Level 2 - MongoDB

Description:

Reviewing all roles periodically and eliminating unneeded roles as well as unneeded privileges from necessary roles helps minimize the privileges for each user.

Rationale:

Although role-based access control (RBAC) has many advantages for regulating access to resources, over time, some roles may no longer be needed, and some roles may grant privileges that are no longer needed.

Audit:

Perform the following command to view all roles, including both built-in and user-defined roles as well as the privileges granted by each role, on the database on which the command runs. Ensure that only necessary roles are listed, and only the necessary privileges are listed for each role.

```
db.runCommand(
   {
      rolesInfo: 1,
      showPrivileges: true,
      showBuiltinRoles: true
   }
)
```

Remediation:

Revoke specified privileges from the user-defined role on the database where the command is run. The `revokePrivilegesFromRole` command has the following syntax:

```
{
revokePrivilegesFromRole: "<role>",
privileges:
   [
   { resource: { <resource> }, actions: [ "<action>", ... ] },
   ...
   ],
}
```

References:

1. https://docs.mongodb.com/v3.6/reference/method/db.revokePrivilegesFromRole/
2. https://docs.mongodb.com/v3.6/reference/method/db.revokePrivilegesFromRole/#db.revokePrivilegesFromRole

Notes:

You must have the `dropRole` action on a database to drop a role from that database.

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.

3.5 Review User-Defined Roles (Scored)

Profile Applicability:

- Level 2 - MongoDB

Description:

Reviewing all roles periodically and removing all users from those roles who do not require them helps minimize the privileges for each user.

Important Roles which should be reviewed periodically.

- `createRole`: Creates a role and specifies its privileges.
- `dropRole`: Deletes the user-defined role.
- `grantPrivilegesToRole`: Assigns privileges to a user-defined role.
- `grantRolesToRole`: Specifies roles from which a user-defined role inherits privileges.
- `updateRole`: Updates a user-defined role.

Rationale:

Although role-based access control (RBAC) has many advantages for regulating access to resources, over time some users may be assigned roles which are no longer necessary, e.g. a user changing jobs within the organization. Users who have excessive privileges pose an unnecessary risk to the organization.

Audit:

Check each role for each database using one of the following commands.

To specify a role from the current database, specify the role by its name:

```
db.runCommand( { rolesInfo: "<rolename>" } )
```

To specify a role from another database, specify the role by a document that specifies the role and database:

```
db.runCommand( { rolesInfo: { role: "<rolename>", db: "<database>" } } )
```

Remediation:

To remove a user from one or more roles on the current database, use the following command:

```
>use dbName
>db.revokeRolesFromUser("<username>",[<roles>])
```

References:

1. https://docs.mongodb.com/v3.6/reference/method/db.revokeRolesFromUser/
2. https://docs.mongodb.com/v3.6/reference/privilege-actions/
3. https://docs.mongodb.com/v3.6/reference/command/nav-role-management/

Notes:

Logged-in user must have the `revokeRole` action on a database to revoke a role on that database. Also, `roleInfo` works for both user-defined roles and built-in roles.

CIS Controls:

Version 6

16.1 Perform Regular Account Reviews
Review all system accounts and disable any account that cannot be associated with a business process and owner.

Version 7

16.8 Disable Any Unassociated Accounts
Disable any account that cannot be associated with a business process or business owner.

3.6 Review Superuser/Admin Roles (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

Roles provide several advantages that make it easier to manage privileges in a database system. Security administrators can control access to their databases in a way that mirrors the structure of their organizations (they can create roles in the database that map directly to the job functions in their organizations). The assignment of privileges is simplified. Instead of granting the same set of privileges to each individual user in a particular job function, the administrator can grant this set of privileges to a role representing that job function and then grant that role to each user in that job function.

The following roles provide the ability to assign any user any privilege on any database, which means that users with one of these roles can assign themselves any privilege on any database:

- `dbOwner` role, when scoped to the admin database
- `userAdmin` role, when scoped to the admin database
- `userAdminAnyDatabase` role

Rationale:

Reviewing the Superuser/Admin roles within a database helps minimize the possibility of privileged unwanted access.

Audit:

Superuser roles provide the ability to assign any user any privilege on any database, which means that users with one of these roles can assign themselves any privilege on any database:

```
db.runCommand( { rolesInfo: "dbOwner" } )
db.runCommand( { rolesInfo: "userAdmin" } )
db.runCommand( { rolesInfo: "userAdminAnyDatabase" } )
```

Root role provides access to the operations and all the resources of the `readWriteAnyDatabase`, `dbAdminAnyDatabase`, `userAdminAnyDatabase`, `clusterAdmin` roles, restore combined.

```
db.runCommand( { rolesInfo: "readWriteAnyDatabase" } )
db.runCommand( { rolesInfo: "dbAdminAnyDatabase" } )
db.runCommand( { rolesInfo: "userAdminAnyDatabase" } )
db.runCommand( { rolesInfo: "clusterAdmin" } )
```

Cluster Administration Roles are used for administering the whole system rather than just a single database.

```
db.runCommand( { rolesInfo: "hostManager" } )
```

Remediation:

To remove a user from one or more roles on the current database.

```
use <dbName>
db.revokeRolesFromUser( "<usernamem>", [ <roles> ])
```

References:

1. https://docs.mongodb.com/v3.6/reference/built-in-roles/
2. https://docs.mongodb.com/v3.6/reference/method/db.revokeRolesFromUser/

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.

16.1 Perform Regular Account Reviews

Review all system accounts and disable any account that cannot be associated with a business process and owner.

Version 7

4.3 Ensure the Use of Dedicated Administrative Accounts

Ensure that all users with administrative account access use a dedicated or secondary account for elevated activities. This account should only be used for administrative activities and not internet browsing, email, or similar activities.

16.8 Disable Any Unassociated Accounts

Disable any account that cannot be associated with a business process or business owner.

4 Data Encryption

This section contains recommendations for securing data at rest (stored) and data in motion (transiting) for MongoDB.

4.1 Ensure Encryption of Data in Transit TLS/SSL (Transport Encryption) (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

Use TLS or SSL to protect all incoming and outgoing connections. This should include using TLS or SSL to encrypt communication between the mongod and mongos components of a MongoDB client as well as between all applications and MongoDB.

MongoDB supports TLS/SSL (Transport Layer Security/Secure Sockets Layer) to encrypt all of MongoDB's network traffic. TLS/SSL ensures that MongoDB network traffic is only readable by the intended client.

Rationale:

This prevents sniffing of cleartext traffic between MongoDB components or performing a man-in-the-middle attack for MongoDB.

Audit:

To verify that the server requires SSL or TLS (`net.ssl.mode` value set to `requireSSL`), run one of the following commands:

On Ubuntu:

```
cat /etc/mongod.conf | grep –A20 'net' | grep –A10 'ssl' | grep 'mode'
```

On Windows:

```
type mongod.conf | findstr –A20 'net' | findstr –A10 'ssl' | findstr 'mode'
```

Remediation:

Configure MongoDB servers to require the use of SSL or TLS to encrypt all MongoDB network communications.

To implement SSL or TLS to encrypt all MongoDB network communication, perform the following steps:

For mongod ("Primary daemon process for the MongoDB system")

In the configuration file `/etc/mongod.conf`, set the `PEMKeyFile` option to the certificate file's path and then start the component with this command:

```
ssl:
   mode: requireSSL
   PEMKeyFile: /etc/ssl/mongodb.pem
   CAFile: /etc/ssl/ca.pem
```

And restart monogdb instance with

```
mongod --config /etc/mongod.conf
```

Or

```
mongod --sslMode requireSSL --sslPEMKeyFile /etc/ssl/mongodb.pem --sslCAFile
/etc/ssl/ca.pem
```

Default Value:

Not configured

References:

1. https://docs.mongodb.com/v3.6/core/security-transport-encryption/
2. https://docs.mongodb.com/v3.6/tutorial/configure-ssl/
3. https://docs.mongodb.com/v3.6/tutorial/configure-ssl-clients/
4. https://docs.mongodb.com/v3.6/tutorial/configure-x509-client-authentication/
5. https://docs.mongodb.com/v3.6/tutorial/configure-x509-member-authentication/

Notes:

Value	Description
disabled	The server does not use TLS/SSL.
allowSSL	Connections between servers do not use TLS/SSL. For incoming connections, the server accepts both TLS/SSL and non-TLS/non-SSL.
preferSSL	Connections between servers use TLS/SSL. For incoming connections, the server accepts both TLS/SSL and non-TLS/non-SSL.
requireSSL	The server uses and accepts only TLS/SSL encrypted connections.

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 Ensure Federal Information Processing Standard (FIPS) is enabled (Scored)

Profile Applicability:

- Level 2 - MongoDB

Description:

The Federal Information Processing Standard (FIPS) is a computer security standard used to certify software modules and libraries that encrypt and decrypt data securely. You can configure MongoDB to run with a FIPS 140-2 certified library for OpenSSL.

FIPS is a property of the encryption system and not the access control system. However, the environment requires FIPS compliant encryption and access control. Organizations must ensure that the access control system uses only FIPS-compliant encryption.

Rationale:

FIPS is an industry standard which dictates how data should be encrypted at rest and during transmission.

Audit:

On Ubuntu:

To verify that the server uses FIPS Mode (`net.ssl.FIPSMode` value set to `true`), run following commands:

```
mongod --config /etc/mongod.conf

net:
 ssl:
 FIPSMode: true
```

Or

To verify FIPS mode is running, check the server log file for a message that FIPS is active:

```
FIPS 140-2 mode activated
```

On Windows:

Check `FIPSMode` is `true`

```
type mongod.conf | findstr "FIPSMode"
```

Remediation:

Configuring FIPS mode, ensure that your certificate is FIPS compliant. Run mongod or mongos instance in FIPS mode.

Make changes to configuration file, to configure your mongod or mongos instance to use FIPS mode, shut down the instance and update the configuration file with the following setting:

```
net:
   ssl:
      FIPSMode: true
```

Start mongod or mongos instance with a configuration file.

```
mongod --config /etc/mongod.conf
```

Default Value:

Not configured

References:

1. https://docs.mongodb.com/v3.6/tutorial/configure-fips/

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.

14.5 Encrypt At Rest Sensitive Information
Sensitive information stored on systems shall be encrypted at rest and require a secondary authentication mechanism, not integrated into the operating system, in order to access the information.

Version 7

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

14.8 Encrypt Sensitive Information at Rest
Encrypt all sensitive information at rest using a tool that requires a secondary

authentication mechanism not integrated into the operating system, in order to access the information.

4.3 Ensure Encryption of Data at Rest (Scored)

Profile Applicability:

- Level 2 - MongoDB

Description:

Encryption of data at rest must be enabled to ensure compliance with security and privacy standards including HIPAA, PCI-DSS, and FERPA.

Encryption at rest, when used in conjunction with transport encryption and good security policies that protect relevant accounts, passwords, and encryption keys.

Rationale:

Unauthorized users, such as intruders who are attempting security attacks, cannot read the data from storage and back up media unless they have the master encryption key to decrypt it.

Audit:

Remediation:

It is recommended to enable the data at rest encryption to protect the data.
Protecting Data at Rest Including following steps.

- Generating a master key.
- Generating keys for each database.
- Encrypting data with the database keys.
- Encrypting the database keys with the master key.

Only the master key is external to the server and requires external management. To manage the master key, MongoDB's encrypted storage engine supports two key management options:

- Integration with a third-party key management appliance via the Key Management Interoperability Protocol (KMIP). Recommended
- Use of local key management via a keyfile.

The encryption occurs transparently in the storage layer; i.e. all data files are fully encrypted from a filesystem perspective, and data only exists in an unencrypted state in memory and during transmission.

To enable Encryption on Database follow below step mentioned in below Link

https://docs.mongodb.com/v3.6/tutorial/configure-encryption/

Rotation of Key is also important. This can be enabled by following mentioned steps in below link.

https://docs.mongodb.com/v3.6/tutorial/rotate-encryption-key/

References:

1. https://docs.mongodb.com/v3.6/core/security-encryption-at-rest/
2. https://docs.mongodb.com/v3.6/tutorial/configure-encryption/

Notes:

Available in MongoDB Enterprise only.

CIS Controls:

Version 7

14.8 Encrypt Sensitive Information at Rest

Encrypt all sensitive information at rest using a tool that requires a secondary authentication mechanism not integrated into the operating system, in order to access the information.

5 Auditing

This section contains recommendations related to configuring audit logging in MongoDB.

5.1 Ensure that system activity is audited (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

Track access and changes to database configurations and data. MongoDB Enterprise includes a system auditing facility that can record system events (e.g. user operations, connection events) on a MongoDB instance. These audit records permit forensic analysis and allow administrators to verify proper controls.

Rationale:

System level logs can be handy while troubleshooting an operational problem or handling a security incident.

Audit:

To verify that system activity is being audited for MongoDB, run the following command to confirm the `auditLog.destination` value is set correctly:

On Ubuntu:

```
cat /etc/mongod.conf |grep -A4 "auditLog" | grep "destination"
```

On Windows:

```
type mongod.conf | findstr -A4 "auditLog" | findstr "destination"
```

Remediation:

Set the value of `auditLog.destination` to the appropriate value from the following options:

Syslog
To enable auditing and print audit events to syslog

```
mongod --dbpath data/db --auditDestination syslog
```

Console
To enable auditing and print audit events to standard output (i.e., `stdout`)

```
mongod --dbpath data/db --auditDestination console
```

Json File
To enable auditing and print audit events to a file in JSON format. Printing audit events to file in JSON format degrades server performance more than printing to a file in BSON format.

```
mongod --dbpath data/db --auditDestination file --auditFormat JSON --
auditPath data/db/auditLog.json
```

Bson File
To enable auditing and print audit events to a file in BSON binary format

```
mongod --dbpath data/db --auditDestination file --auditFormat BSON --
auditPath data/db/auditLog.bson
```

Default Value:

By default, Audit Logs are not configured.

References:

1. https://docs.mongodb.com/v3.6/tutorial/configure-auditing/

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.

5.2 Ensure that audit filters are configured properly (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

MongoDB Enterprise supports auditing of various operations. When enabled, the audit facility, by default, records all auditable operations as detailed in Audit Event Actions, Details, and Results. To specify which events to record, the audit feature includes the `--auditFilter` option. This check is only for Enterprise editions.

Rationale:

All operations carried out on the database are logged. This helps in backtracking and tracing any incident that occurs.

Audit:

To verify that audit filters are configured on MongoDB as per the organization's requirements, run the following command:

On Ubuntu:

```
cat /etc/mongod.conf |grep -A10 "auditLog" | grep "filter"
```

On Windows:

```
type mongod.conf | findstr -A10 "auditLog" | findstr "filter"
```

Remediation:

Set the audit filters based on the organization's requirements.

Default Value:

Not configured

References:

1. https://docs.mongodb.com/v3.6/reference/audit-message/
2. https://docs.mongodb.com/v3.6/tutorial/configure-audit-filters/

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.

5.3 Ensure that logging captures as much information as possible (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

The `SystemLog.quiet` option stops logging of information such as:

- connection events
- authentication events
- replication sync activities
- evidence of some potentially impactful commands being run (eg: `drop`, `dropIndexes`, `validate`)

This information should be logged whenever possible. This check is only for Enterprise editions.

Rationale:

The use of `SystemLog.quiet` makes troubleshooting problems and investigating possible security incidents much more difficult.

Audit:

To verify that the `SystemLog.quiet` option is disabled (i.e.; `false`), run the following command:

On Ubuntu:

```
cat /etc/mongod.conf |grep "quiet"
```

On Windows:

```
type mongod.conf | findstr "quiet"
```

Remediation:

Set `SystemLog.quiet` to `false` in the `/etc/mongod.conf` file to disable it.

```
systemLog:
   quiet: false
```

References:

1. https://docs.mongodb.com/v3.6/reference/configuration-options/#systemLog.quiet

Notes:

`systemLog.quiet` is not recommended for production systems as it may make tracking problems during particular connections much more difficult.

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.

5.4 Ensure that new entries are appended to the end of the log file (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

By default, new log entries will overwrite old entries after a restart of the mongod or mongos service. Enabling the `systemLog.logAppend` setting causes new entries to be appended to the end of the log file rather than overwriting the existing content of the log when the mongod or mongos instance restarts.

Rationale:

Allowing old entries to be overwritten by new entries instead of appending new entries to the end of the log may destroy old log data that is needed for a variety of purposes.

Audit:

To verify that new log entries will be appended to the end of the log file after a restart (`systemLog.logAppend` value set to `true`), run the following command:

On Ubuntu:

```
cat /etc/mongod.conf | grep -A10 'systemLog' | grep 'logAppend'
```

On Windows:

```
type mongod.conf | findstr -A10 'systemLog' | findstr 'logAppend'
```

Remediation:

Set `systemLog.logAppend` to `true` in the `/etc/mongod.conf` file.

References:

1. https://docs.mongodb.com/v3.6/reference/configuration-options/#systemLog.logAppend

CIS Controls:

Version 6

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

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

Version 7

6.4 Ensure adequate storage for logs

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

6 Operating System Hardening

This section contains recommendations related to hardening the operating system running below MongoDB.

6.1 Ensure that MongoDB uses a non-default port (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

Changing the default port used by MongoDB makes it harder for attackers to find the database and target it.

Rationale:

Standard ports are used in automated attacks and by attackers to verify which applications are running on a server.

Audit:

To verify the port number used by MongoDB, execute the following command and ensure that the port number is not `27017`:

On Ubuntu:

```
cat /etc/mongod.conf |grep "port"
```

On Windows:

```
type mongod.conf | findstr "port"
```

Remediation:

Change the port for MongoDB server to a number other than `27017`.

Impact:

Hackers frequently scan IP addresses for commonly used ports, so it's not uncommon to use a different port to "fly under the radar". This is just to avoid detection, other than that there is no added safety by using a different port.

References:

1. https://docs.mongodb.com/v3.6/reference/default-mongodb-port/

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.

6.2 Ensure that operating system resource limits are set for MongoDB (Not Scored)

Profile Applicability:

- Level 2 - MongoDB

Description:

Operating systems provide ways to limit and control the usage of system resources such as threads, files, and network connections on a per-process and per-user basis

Rationale:

These `ulimits` prevent a single user from consuming too many system resources.

Audit:

To verify the resource limits set for MongoDB, run the following commands.

Extract the process ID for MongoDB:

```
ps -ef|grep mongod
```

View the limits associated with that process number:

```
cat /proc/1322/limits
```

Remediation:

Every deployment may have unique requirements and settings. Recommended thresholds and settings are particularly important for MongoDB deployments:

- `f` (file size): unlimited
- `t` (cpu time): unlimited
- `v` (virtual memory): unlimited [1]
- `n` (open files): 64000
- `m` (memory size): unlimited [1] [2]
- `u` (processes/threads): 64000

Restart the mongod and mongos instances after changing the `ulimit` settings to ensure that the changes take effect.

Default Value:

Not configured

References:

1. https://docs.mongodb.com/v3.6/reference/ulimit/#recommended-ulimit-settings

6.3 Ensure that server-side scripting is disabled if not needed (Not Scored)

Profile Applicability:

- Level 2 - MongoDB

Description:

MongoDB supports the execution of JavaScript code for certain server-side operations: `mapReduce`, `group`, and `$where`. If you do not use these operations, server-side scripting should be disabled.

Rationale:

If server-side scripting is not needed and is not disabled, this introduces unnecessary risk which may allow an attacker to take advantage of insecure coding.

Audit:

If server-side scripting is not required, verify that it is disabled (`javascriptEnabled` value of `false`) using the following command:

On Ubuntu:

```
cat /etc/mongod.conf | grep -A10 "security" | grep "javascriptEnabled"
```

On Windows:

```
type mongod.conf | findstr -A10 "security" | findstr "javascriptEnabled"
```

Remediation:

If server-side scripting is not required, disable it by using the `--noscripting` option on the command line.

Default Value:

Enabled

References:

1. https://docs.mongodb.com/v3.6/reference/configuration-options/#security.javascriptEnabled

CIS Controls:

Version 6

18.9 Sanitize Deployed Software Of Development Artifacts

For in-house developed applications, ensure that development artifacts (sample data and scripts; unused libraries, components, debug code; or tools) are not included in the deployed software, or accessible in the production environment.

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 File Permissions

This section provides recommendations for setting permissions for the key file and the database file.

7.1 Ensure authentication file permissions are set correctly (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

In the Shared Cluster, the certificate or keyfile is utilized for authentications. Implementing proper file permissions on the certificate or keyfile will prevent unauthorized access to it.

Rationale:

Protecting the certificate/keyfile strengthens authentication in the sharded cluster and prevents unauthorized access to the MongoDB database.

Audit:

Find the location of certificate/ keyfile using the following commands:

On Ubuntu:

```
cat /etc/mongod.conf | grep "keyFile:"
cat /etc/mongod.conf | grep "PEMKeyFile:"
cat /etc/mongod.conf | grep "CAFile:"
```

On Windows:

```
type mongod.conf | findstr "keyFile:"
type mongod.conf | findstr "PEMKeyFile:"
type mongod.conf | findstr "CAFile:"
```

Check the permission of the file using:

```
ls -l certificate_file_locations
ls -l keyfile_locations
```

Remediation:

Set the `keyFile` ownership to mongodb user and remove other permissions by executing these commands:

```
chmod 600 /keyfile
sudo chown mongodb:mongodb /keyfile
```

Default Value:

Not configured

References:

1. https://docs.mongodb.com/v3.6/tutorial/enforce-keyfile-access-control-in-existing-replica-set/

CIS Controls:

Version 6

16.14 Encrypt/Hash All Authentication Files And Monitor Their Access

Verify that all authentication files are encrypted or hashed and that these files cannot be accessed without root or administrator privileges. Audit all access to password files in the system.

Version 7

16.4 Encrypt or Hash all Authentication Credentials

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

7.2 Ensure that database file permissions are set correctly (Scored)

Profile Applicability:

- Level 1- MongoDB

Description:

MongoDB database files need to be protected using file permissions.

Rationale:

This will restrict unauthorized users from accessing the database.

Audit:

To verify that the permissions for the MongoDB database file are configured securely, run the following commands.

Find out the database location using the following command:

On Ubuntu:

```
cat /etc/mongod.conf |grep "dbpath"
```

On Windows:

```
type mongod.conf | findstr "dbpath"
```

Use the database location as part of the following command to view and verify the permissions set for the database file:

```
ls –l /var/lib/mongodb
```

Remediation:

Set ownership of the database file to mongodb user and remove other permissions using the following commands:

```
chmod 770 /var/lib/mongodb
sudo chown mongodb:mongodb /var/lib/mongodb
```

Default Value:

Not configured

References:

1. https://docs.mongodb.com/v3.6/reference/configuration-options/#storage.dbPath

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.

Appendix: Summary Table

Control		Set Correctly	
		Yes	No
1	**Installation and Patching**		
1.1	Ensure the appropriate MongoDB software version/patches are installed (Scored)	☐	☐
2	**Authentication**		
2.1	Ensure Authentication is configured (Scored)	☐	☐
2.2	Ensure that MongoDB does not bypass authentication via the localhost exception (Scored)	☐	☐
2.3	Ensure authentication is enabled in the sharded cluster (Scored)	☐	☐
3	**Access Control**		
3.1	Ensure that Role-based access control (RBAC) is enabled and configured (Scored)	☐	☐
3.2	Ensure that MongoDB only listens for network connections on authorized interfaces (Scored)	☐	☐
3.3	Ensure that MongoDB is run using a Least Privileges, dedicated service account (Scored)	☐	☐
3.4	Ensure that each role for each MongoDB database is needed and grants only the necessary privileges (Scored)	☐	☐
3.5	Review User-Defined Roles (Scored)	☐	☐
3.6	Review Superuser/Admin Roles (Scored)	☐	☐
4	**Data Encryption**		
4.1	Ensure Encryption of Data in Transit TLS/SSL (Transport Encryption) (Scored)	☐	☐
4.2	Ensure Federal Information Processing Standard (FIPS) is enabled (Scored)	☐	☐
4.3	Ensure Encryption of Data at Rest (Scored)	☐	☐
5	**Auditing**		
5.1	Ensure that system activity is audited (Scored)	☐	☐
5.2	Ensure that audit filters are configured properly (Scored)	☐	☐
5.3	Ensure that logging captures as much information as possible (Scored)	☐	☐
5.4	Ensure that new entries are appended to the end of the log file (Scored)	☐	☐
6	**Operating System Hardening**		
6.1	Ensure that MongoDB uses a non-default port (Scored)	☐	☐
6.2	Ensure that operating system resource limits are set for MongoDB (Not Scored)	☐	☐

6.3	Ensure that server-side scripting is disabled if not needed (Not Scored)	☐	☐
7	**File Permissions**		
7.1	Ensure authentication file permissions are set correctly (Scored)	☐	☐
7.2	Ensure that database file permissions are set correctly (Scored)	☐	☐

Appendix: Change History

Date	Version	Changes for this version
Dec 31, 2019	1.0.0	Initial Release