HARDENING GUIDE FOR MYSQL

	Hardening instructions	Commands/Instructions	Comments
1	**OS Hardening**		OS harden server hosting MySQL. Machine dedicated to running MySQL Limiting the number of services executing on the machine hosting MySQL will reduce the probability of the data within MySQL being compromised.
2	Run MySQL in Jail or Chroot On Unix, Linux machines		Running MySQL in a chroot environment may reduce the impact of a MySQL-born vulnerability by making portions of the file system inaccessible to the MySQL instance.
3	**Dedicated Account**		Dedicated non-administrative account for MySQL daemon/service
4	**Restrict network access**		Limiting the accessibility of the MySQL network socket

			may reduce the exposure to a MySQL-born vulnerability by preventing unauthorized hosts from communicating with the service.
5	**Database not on system partition**	**Auditing Guidance :** 1. Get data folder name "`show variables like 'datadir';`" 2. Verify that the database is not located on the root or system partition	Moving the database off the system partition (root partition) will reduce the probability of denial of service via the exhaustion of available disk space to the operating system.
6	**Command history**	Admin and DBA"s should disable command history by setting MYSQL_HISTFILE to /dev/null or linking .mysql_history to /dev/null	All commands run in the MySQL console application are saved to a history file. Disabling the MySQL command history reduces the probability of exposing sensitive information, such as passwords.
7	**MYSQL_PWD**	MySQL can read the database password from an environmental variable called `MYSQL_PWD`. Verify `MYSQL_PWD` environmental variable not used	The use of the `MYSQL_PWD` environment variable implies the clear text storage of MySQL credentials. Avoiding this may increase assurance that the confidentiality of MySQL

			credentials is preserved.
8	**MySQL User**	Disable interactive login For Windows: Deny the account the "Log on locally" right	Preventing the MySQL user from logging in interactively may reduce the impact of a compromised MySQL account.
9	**Windows Network Service Account**	MySQL should run as a network service account [Windows 2003, Windows XP]	Executing the MySQL user as the `NETWORK_SERVICE` account may reduce the impact of a MySQL-born vulnerability because this account has a restricted privilege set.
10	**Windows Platform Selection**	Do not install MySQL on a domain controller	Installing MySQL on a non-domain controller may reduce the impact of a MySQL-born vulnerability
11	**Data directory**	Read and write by MySQL user only. **Auditing Guidance:** 1. Locating directory: SQL: "`show variables like 'datadir';`" 2. Verify permissions	This is the location of the MySQL databases Limiting the accessibility of these objects will protect the confidentiality, integrity, and availability of the MySQL database.

10	**Binaries**	Verify and set permissions such that binaries are accessible only by database administrators and database users. Typically these are located on Unix systems in the `/usr/bin` and `/usr/sbin` folders. For Windows they are located in the installation folder. Can be found by locating the `mysqld`, `mysqladmin`, and `mysql` executables. **Auditing Guidance:** 1. Locate base directory: SQL: `"show variables like „basedir";"` 2. Verify permissions	Limiting the accessibility of these objects will protect the confidentiality, integrity, and availability of the MySQL database.
11	**Configuration File**	Set permissions so that configuration files are readable by database administrators and database users. Typically the MySQL configuration file on Unix systems is located in `/etc/mysql/my.cnf`. On Windows it will be located in the `%SYSTEMDIR%` or install folder.	Limiting the accessibility of these objects will protect the confidentiality, integrity, and availability of the MySQL database.

12	**Log files**	Permission log files to be readable and writeable by MySQL user and authorized administrators only. **Auditing Guidance:** 1. Find `log_bin` entry in configuration file (contains path to logs) 2. Verify permissions	Limiting the accessibility of these objects will protect the confidentiality, integrity, and availability of the MySQL logs.
13	**SSL files**	SSL files should be readable by MySQL user. No other read or write permissions. **Auditing Guidance:** 1. Locate files using the following variables: `ssl_ca`, `ssl_cert`, `ssl_key` 2. Include these variables in SQL statements such as "`show variables like „XXX";`" 3. Verify permissions	Limiting the accessibility of these objects will protect the confidentiality, integrity, and availability of the MySQL database.
14	**Error Logging Enabled**	**Auditing Guidance:** 1. SQL: "`show variables like „log_error";`" 2. Verify entry	The error log must be enabled. Enabling error logging may increase the ability to detect malicious attempts against MySQL.

15	**Logs not on system partition**	**Auditing Guidance:** 1. Verify "`show variables like „log_bin";`" is "ON"	Logs should be on a non-system partition
16	**Logs not on database partition**		Moving the MySQL logs off the database partition will reduce the probability of denial of service via the exhaustion of available disk space to MySQL.
17	**Do not use Update log**	Do not use --log-update **Auditing Guidance:** Verify that the "`--log-update`" option is not used on command line or in configuration files.	The update log is now deprecated and the binary log should be used instead. The update log is not transaction safe. Avoiding the `--log-update` option may increase the integrity and availability of MySQL log files
18	**Supported version of MySQL** Migrate to version 4.1 or 5.0	**Auditing Guidance:** SQL: "`show variables like „version";`"	

19	**Latest security patches**	Verify latest security patches. Determine current version of MySQL using "`mysql -h HOSTNAME -V`". Review changes in each revision greater than that running for security changes. See **Error! Not a valid result for table.** for links to change history.	Maintaining currency with MySQL patches will help protect the confidentiality, integrity, and availability of the data housed in MySQL.
20	**Upgrade fix privilege tables**	When upgrading always fix the privilege tables **Auditing Guidance for section 4.3:** Tables that will need to be checked: `mysql.user`, `mysql.host`, `mysql.db`, `mysql.tables_priv`, `mysql.columns_priv`, `mysql.func`, and `mysql.procs_priv`.	MySQL has a script for checking and upgrading the tables. Mysql_upgrade for v5.0+, mysql_fix_privilege_tables otherwise. Some revisions of MySQL have added privileges that did not exist in earlier versions. Ensuring that privileges are appropriately applied to MySQL objects will help ensure the confidentiality, integrity, and availability of the data housed in MySQL.

21	**Remove test database**	The default MySQL installation comes with a database called "`test`". Databases can be viewed using the "`SHOW DATABASES;`" command for example "`SHOW DATABASES like „test";`". Databases can be dropped using the "`DROP DATABASE xxx;`" syntax.	Removing unutilized components will eliminate an attacker‟s ability to leverage them.
22	**Change admin account name**	Change admin account from default ("root") to something else. **Auditing Guidance:** 1. SQL: "`select user from mysql.user where user = „root";`" 2. Verify no results were returned	Disabling the root user‟s ability to interact with MySQL will limit the use of this sensitive account for non-operating system administrative purposes. Additionally, avoiding the „root‟ account for MySQL interactions will reduce the possibility of compromising the system via a MySQL client-born vulnerability
23	**Complex Passwords**	Minimum 8 characters in length with characters from at least three of the following categories: uppercase, lowercase, numeric, non-alphanumeric	Complex passwords help mitigate dictionary, brute forcing, and other password attacks.

24			
25	**Verify Secure Password Hashes**	**Auditing Guidance:** 1. SQL: "`select User, Password from mysql.user where length(password) < 41;`" 2. Validate that no results are returned	Starting in v4.1 a stronger password hash is used that result in hashes 41 bytes long. Older password hashes were only 16 bytes. Utilizing the stronger hashing algorithm will ensure the confidentiality, integrity, and availability of the data housed within MySQL by protecting the confidentiality of authentication credentials.
26	**Single use accounts**	Each database user should be used for single purpose/person. Database user accounts should not be reused for multiple applications or users.	Utilizing unique database accounts across applications will reduce the impact of a compromised MySQL account.
27	**Wildcards in user hostname**	Verify if users have wildcard („%") in hostname **Auditing Guidance:** 1. SQL: "`select user from mysql.user where host = „%";`" 2. Verify that no results are returned	Avoiding the use of wildcards within hostnames will ensure that only trusted principals are capable of interacting with MySQL.