A Guide to MySQL and PCI Data Security Standard Compliance

A MySQL White Paper

Table of Contents

Introduction

Many organizations struggle to achieve compliance with the Payment Card Industry Data Security Standard (PCI DSS), the payment card industry mandate to protect cardholder data and prevent fraud. The latest PCI Security Standards Council (PCI SSC) is version 3.1 published in April 2015.

Although the standard is more prescriptive than most government or industry security directives, with its 12 requirements and supporting guidance, compliance is often far more resource-intensive, expensive and error prone than necessary. Given the effort to manually collect and analyze log data, compile reports, and maintain and validate security controls and policies through inefficient labor-intensive processes, average costs can reach beyond hundreds of thousands of dollars per year depending on your organization's tier level.

Access control, data protection and configuration management policies are difficult to implement and even harder to maintain, manage and enforce. Organizations are looking to implement an efficient, repeatable and evolving security program that satisfies the technical requirements of their PCI obligations while protecting critical cardholder data. This white paper covers the fundamentals of a PCI compliance program using MySQL Enterprise Edition, focusing on how to:

- **Protect cardholder data** from unauthorized use
- **Enforce strong controls** around privileged users and data access
- **Implement centralized, automated role-based** access control, authorization, and authentication
- **Provide system and database auditing**, and database activity monitoring

This whitepaper will discuss how specific features within MySQL Enterprise Edition, as well as other complimentary Oracle products, can be used to meet the database specific PCI requirements.

- **Requirement 2:** Not Using Vendor Default Passwords and Security Settings
- **Requirement 3:** Protect Stored Cardholder Data
- **Requirement 6:** Develop and Maintain Secure Systems and Applications
- **Requirement 7:** Restrict Access to Cardholder Data by Business Need to Know
- **Requirement 8:** Identify and Authenticate Access to System Components
- **Requirement 10:** Track and Monitor Access to Cardholder Data

Its important to note that some of the requirements are not related to MySQL or databases in general. For example, requirements 1, 4, 5, 9, 11, and 12, are associated with anti-virus, network firewall deployments, encrypted transmission of cardholder data over public networks, physical security controls, testing and maintaining administrative policies.

The Challenges of PCI Data Protection

Cardholder Data at Risk

PCI DSS requires organizations to protect cardholder data. Essentially, that covers the primary account number (PAN), cardholder name, service code and expiration date — if stored in conjunction with the card numbers (as they typically are, in order to be of practical business use).

News reports have been filled with stories about how external hackers and malicious insiders have stolen millions of credit card records stored in backend databases, selling them on the international Internet black markets and/or using them to purchasing high-priced items.

Unauthorized and more typically, unnecessary database and/or operating system access, exposes cardholder data, often to trusted privileged users, such as database and system administrators, as well as developers and other employees who have no need to access or see this information. Plus, unsecure database configurations, often the result of flawed deployments, leave cardholder data highly susceptible to theft.

Lets examine some of the PCI requirements and how MySQL Enterprise Edition can help organizations meet those requirements.

Secure Configurations, Security Settings & Patching

Requirement 2: Do Not Use Vendor Supplied Defaults for System Passwords and other Security Parameters

Attackers can and will exploit configuration weaknesses in database instances, as this is one of several ways malicious outsiders can harvest cardholder data. Weak configuration settings, lack of configuration enforcement standards, missing patches, errors and configuration “drift” as changes are made, including unauthorized changes made outside of change management procedures, all can leave cardholder data vulnerable to attack.

To identify and fix security such vulnerabilities, PCI DSS addresses the need for secure configuration practices. For example:

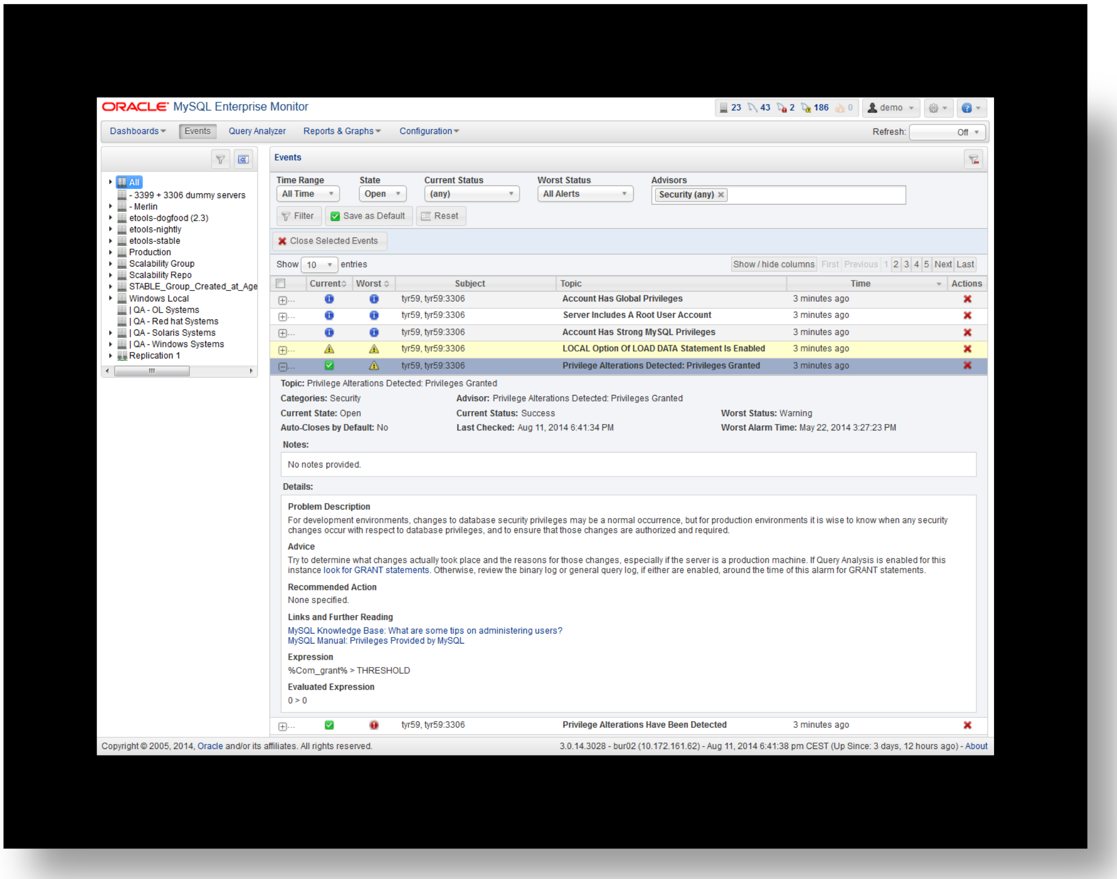
- Requirement 2 directs organizations to change vendor-supplied defaults such as passwords and to eliminate any unnecessary accounts.
- Section 2.2 requires development of configuration standards that are consistent with accepted hardening standards and that address known configuration vulnerabilities.
- Requirement 6 mandates up-to-date patching

PCI requires database servers be security hardened, and that the secure hardened sever is rechecked, as well as upgraded with further security hardening options as they become available. MySQL Enterprise Monitor additionally provides not just configuration monitoring, but also monitoring and alert facilities for object, schema, and user changes.

MySQL Enterprise Monitor Security Advisors

MySQL Enterprise Monitor provides hundreds of best practice Advisors alerting you to potential security issues. For example, the MySQL Enterprise Monitor Security Advisors will alert you about the following:

- Policy-Based Password Validation Is Weak
- Policy-Based Password Validation Not Enabled
- Server Has Accounts Without A Password
- Root Account Without Password
- Server Includes A Root User Account
- SHA-256 Password Authentication Not Enabled
- Users Can View All Databases On MySQL Server
- Policy-Based Password Validation Does Not Perform Dictionary Checks



Time Range: All Time
State: Open
Current Status: (any)
Worst Status: All Alerts
Advisors: Security (any) ×

Show 10 entries

□	Current	Worst	Subject	Topic	Time	Actions
⊞...	ⓘ	ⓘ	tyr59, tyr59:3306	**Account Has Global Privileges**	3 minutes ago	✖
⊞...	ⓘ	ⓘ	tyr59, tyr59:3306	**Server Includes A Root User Account**	3 minutes ago	✖
⊞...	ⓘ	ⓘ	tyr59, tyr59:3306	**Account Has Strong MySQL Privileges**	3 minutes ago	✖
⊞...	⚠	⚠	tyr59, tyr59:3306	**LOCAL Option Of LOAD DATA Statement Is Enabled**	3 minutes ago	✖
⊟...	☑	⚠	tyr59, tyr59:3306	**Privilege Alterations Detected: Privileges Granted**	3 minutes ago	✖

Topic: Privilege Alterations Detected: Privileges Granted

Categories: Security **Advisor:** Privilege Alterations Detected: Privileges Granted

Current State: Open **Current Status:** Success **Worst Status:** Warning

Auto-Closes by Default: No **Last Checked:** Aug 11, 2014 6:41:34 PM **Worst Alarm Time:** May 22, 2014 3:27:23 PM

Notes:

No notes provided.

Details:

Problem Description

For development environments, changes to database security privileges may be a normal occurrence, but for production environments it is wise to know when any security changes occur with respect to database privileges, and to ensure that those changes are authorized and required.

Advice

Try to determine what changes actually took place and the reasons for those changes, especially if the server is a production machine. If Query Analysis is enabled for this instance look for GRANT statements. Otherwise, review the binary log or general query log, if either are enabled, around the time of this alarm for GRANT statements.

Recommended Action

None specified.

Links and Further Reading

MySQL Knowledge Base: What are some tips on administering users?
MySQL Manual: Privileges Provided by MySQL

Expression

%Com_grant% > THRESHOLD

Evaluated Expression

0 > 0

⊞...	☑	●	tyr59, tyr59:3306	**Privilege Alterations Have Been Detected**	3 minutes ago	✖

Copyright © 2005, 2014, Oracle and/or its affiliates. All rights reserved.
3.0.14.3028 - bur02 (10.172.161.62) - Aug 11, 2014 6:41:38 pm CEST (Up Since: 3 days, 12 hours ago) - About

MySQL Enterprise Monitor Dashboard delivers critical security events.

Protecting Cardholder Data – Strong Cryptography

Requirement 3: Protect Stored Cardholder Data

Open networked environments are susceptible to a variety of communication security problems such as man-in-the-middle attacks and other security threats. Requirement 3 of the PCI DSS states:

- Mask Personal Account Number (PAN) when displayed
- Render PAN unreadable anywhere it is stored
- Document and implement procedures to protect keys used to store cardholder data against disclosure and misuse
- Generation of strong cryptographic keys
- Secure cryptographic key distribution
- Secure cryptographic key storage
- And more.

MySQL Enterprise Edition preserves data privacy and confidentiality by:

- Preventing data sniffing, data loss, replay and man-in-the-middle attacks.
- Enabling data-at-rest encryption by encrypting the physical files of the database.
- Providing both native network encryption and SSL/TLS based encryption for enterprises with PKI based infrastructures.
- Rejecting connections from clients that do not encrypt data or optionally it can selectively allow unencrypted connections for deployment flexibility.

MySQL Enterprise Masking and De-identification

Cardholder information and other production data is important for development and testing activities, potentially putting large amounts of cardholder data in the hands of customer support, developers, QA personnel, etc. who should not be allowed to see it according to the PCI requirements. Requirement section 6.4.3 specifically prohibits the use of live PANs for development. To mitigate this scenario, developers will sometimes generate fake data to simulate live production data, but this is not always as reliable, especially for testing purposes.

In any of these cases, the trick is to mask the information so that the visible data can neither compromise security nor privacy. Data masking simply substitutes false values for real ones, keeping the data formats, regardless of the number and type of fields. MySQL Enterprise Edition Masking and De-identification replaces credit card numbers and associated cardholder information with realistic but false values, allowing production data to be safely used for transactions, development, testing, or sharing in conjunction with out-sourced or off-shore partners for nonproduction purposes. MySQL Enterprise Edition Masking and De-identification functions transform data for securely testing applications without exposing sensitive data.

MySQL Enterprise Encryption – Public Key Cryptography

To protect sensitive data throughout its lifecycle, MySQL Enterprise Encryption provides industry standard functionality for asymmetric encryption (Public Key Cryptography). MySQL Enterprise Encryption provides encryption, key generation, digital signatures and other cryptographic features to help organizations protect confidential data and comply with regulatory requirements including the PCI Data Security Standard.

MySQL Enterprise Encryption allows your enterprise to:

- **Secure data using combination of public, private, and symmetric keys** to encrypt and decrypt data.
- **Encrypt data stored in MySQL** using RSA, DSA, or DH encryption algorithms.
- **Protect replicated data** by encrypting the MySQL Binlog and Redo Logs.
- **Digitally sign messages to confirm the authenticity** of the sender (non-repudiation) and the integrity of the message.
- **Eliminate unnecessary exposure to data** by enabling DBAs to manage encrypted data.
- **Interoperate with other cryptographic systems** and appliances without changing existing applications.
- **Avoid exposure of asymmetric keys** within client applications or on disk.

MySQL Enterprise Encryption gives DBAs and Developers the tools they need for:

- Asymmetric Public Key Encryption (RSA)
- Asymmetric Private Key Decryption (RSA)
- Generate Public/Private Key (RSA, DSA, DH)
- Derive Symmetric Keys from Public and Private Key pairs (DH)
- Digitally Sign Data (RSA, DSA)
- Verify Data Signature (RSA, DSA)
- Validation Data Authenticity (RSA, DSA)
- This enables software developers to encrypt data by using RDS, DHS and DH encryption algorithms without changing existing applications.

Oracle Key Vault – Encryption Key Management

Generating and protecting encryption keys, while maintaining data availability has traditionally been a major barrier to implementing encryption on an enterprise scale.

Key management is complex and challenging, and often fails because issuance, storage and renewal are difficult. In addition, lost keys can make important data permanently unrecoverable.

Key management often becomes a hollow security control, as IT managers give way to more pressing priorities and pressure from the business managers to relax key management controls. As a consequence, keys become widely available to multiple users, rendering encryption ineffective.

MySQL, together with Key vaults such as Oracle Key Vault enable companies to quickly deploy encryption and other security solutions by centrally managing encryption keys. Using key vaults in conjunction with encryption keys and other security related information, provides much in the way of the protections required within PCI.

MySQL Enterprise Transparent Data Encryption (TDE)

MySQL Enterprise TDE enables data-at-rest encryption by encrypting the physical files of the database. Data is encrypted automatically, in real time, prior to writing to storage and decrypted when read from storage. As a result, hackers and malicious users are unable to read sensitive data from tablespace files, database backups or disks. MySQL Enterprise TDE uses industry standard AES algorithms.

MySQL Enterprise TDE uses a two-tier encryption key architecture, consisting of a master encryption key and tablespace keys, which provides easy key management and rotation. Tablespace keys are managed automatically behind the scenes while the master encryption key is stored in a centralized key management solution such as Oracle Key Vault, which enforces clear separation of keys from encrypted data. Centralized key management solutions automate key rotation and storing historical keys for decrypting database backups.

MySQL Enterprise Backup - Encrypting Database Backups

PCI requires encryption of backup cardholder information to protect against lost or stolen tapes or other backup media.

MySQL Enterprise Backup supports strong AES 256 based encryption of backup images, as well as support for direct backup to media servers which also support added encryption and protection. For example, MySQL Enterprise Backup is certified with Oracle Secure Backup, which encrypts tape data and provides centralized tape backup management.

User Access and Authorization

The concern over privileged user access and authorization is reflected heavily in the PCI DSS. The PCI DSS requirements underscore the importance of access control, user authentication, and user tracking of privileged accounts on a global basis. Organizations will have won a hollow victory if they assert a reasonable level of access control over the general user population but fail to bring privileged user accounts into line.

Requirement 7: Restrict Cardholder Data Access by Business Need to Know

Requirement 7 gives directives to ensure critical data can only be accessed by authorized personnel. Section 7.1 explicitly directs organizations to restrict access rights of privileged user IDs to the least privileges necessary to perform job responsibilities. Privileges should be based on job classification and function. It is also critical to review and, if necessary, redefine these job classifications, particularly for high-privilege accounts where the risk of data abuse is the greatest.

Privileged Accounts - The Enemy Within

Important capabilities, such as privileged user access controls, privileged user authorization, and separation of duties are difficult to manage and maintain when you are left to rely on manual procedures. This is especially true when you consider common factors like changing personnel, changing requirements, and acquiring new assets. Unfortunately, organizations react by sharing user IDs across administrators (prohibited by requirement 8.5), and by authorizing privileges in violation of policy, or outside of required workflow processes. To exacerbate the issue, organizations fail to appropriately review and monitor logs for unauthorized or inappropriate activity because it is time-consuming, resource-intensive, and prone to error.

Ironically, organizations tend to do a better job managing general user account security than privileged accounts such as system administrators and DBAs, even though these accounts present a higher security risk. This risk is due to their ability to access sensitive information, such as cardholder data, configure systems, modify databases and grant privileges to others. As a result hackers target privileged user accounts for their potentially wide ranging access to systems, such as databases containing sensitive cardholder data.

MySQL Enterprise Edition Security Alerts

MySQL Enterprise Edition gives organizations the power to manage and monitor privileged users to meet the regulatory requirements. For example, the MySQL Enterprise Monitor generates security alerts on excessive user privilege assignment and strong privileges, such as:

- **Account Has Global Privileges –** Identify user accounts with privileges on all databases and tables (*.*).In most cases global privileges should be allowed only for the MySQL root user, and possibly for users that you trust or use for backup purposes.
- **Account Has Strong MySQL Privileges -** Certain account privileges can be dangerous and should only be granted to trusted users when necessary. For example, the FILE privilege allows a user to read and write files on the database server (which includes sensitive operating system files), the PROCESS privilege allows currently executing statements to be monitored, and the SHUTDOWN privilege allows a user to shut down the server.
- **Non-Authorized User Has GRANT Privileges On All Databases -** The GRANT privilege, when given on all databases as opposed to being limited to a few specific databases, enables a user to give to other users those privileges that the grantor possesses on all databases.
- **Non-Authorized User Has Server Admin Privileges -** Certain privileges, such as SHUTDOWN and SUPER, are primarily used for server administration. Some of these privileges can have a dramatic effect on a system because they allow

someone to shutdown the server or kill running processes. Such operations should be limited to a small set of users.

- **Non-Authorized User Has DB, Table, Or Index Privileges On All Databases -** Privileges such as SELECT, INSERT, ALTER, and so forth allow a user to view and change data, as well as impact system performance. Such operations should be limited to only those databases to which a user truly needs such access so the user cannot inadvertently affect other people's applications and data stores.
- **Privilege Alterations Have Been Detected -** For development environments, changes to database security privileges may be a normal occurrence, but for production environments it is wise to know when any security changes occur with respect to database privileges, and to ensure that those changes are authorized and required.

Identity Management Enables Compliance

PCI Requirements for Identity and Access Management

Requirement 8: Identify and Authenticate Access to System Components

Requirement 8 defines identity management requirements including user management, authentication management and password management. More specifically, Requirement 8 requires:

- All access to databases containing cardholder data must be authenticated
- A unique user ID be assigned each person with computer access
- Two-factor authentication where deemed appropriate (mandatory for remote access)
- Rules governing passwords

The requirement also covers provisioning including important functions, such as:

- **Control addition, deletion, and modification** of user IDs, credentials and other identifier objects and limiting management to a small group with specific authority
- **Immediately revoke access** for any terminated users
- **Remove/disable** inactive user accounts at least every 90 days
- **Enable accounts** used by vendors for remote maintenance only during the time period needed

The identity management-related requirements of PCI DSS are perhaps the most difficult to implement, and they are even more difficult to manage and maintain. These problems are among the chief reasons that organizations are unable to sustain compliance efforts on a continuous basis. They may lead organizations to spend unreasonable time, manpower and money gathering and analyzing data (if it is available) in attempt to fix broken systems and remediate violations.

Let’s examine some of these challenges, which generally are the result of trying to enforce good policies with inefficient manual processes.

Access control, Authorization and Authentication

Access control, authorization and authentication is, out of necessity, generally performed on a per application basis rather than being centrally managed. In the absence of centralized, policy-based management system, it is difficult to apply appropriate authentication controls consistently across users, groups, applications and data access. The outcome is uneven policy enforcement, manpower-intensive administration, slow/inefficient response to changing business requirements, error prone results, and weakened security.

Role-based access control is excellent in theory, but difficult to implement and maintain. It requires an enormous commitment of time and resources to define roles and establish a role-based approach across the enterprise. This is exacerbated because information about users, responsibilities and lines of reporting typically resides in silos throughout the organization. It is nearly impossible to manage and enforce consistent policy across a complex, distributed enterprise.

Provisioning and De-provisioning

Provisioning and de-provisioning users and user authorizations is often slow and impedes the business. Spreadsheet-based administration can effectively enforce policies but can become a bottleneck because of slow response due to busy administrators and manager approvals.

The provisioning and de-provisioning process is generally error prone for a variety of reasons. Role-based controls are difficult to define and manage without automated systems, so individuals may be given too little, or more likely, too many privileges based on coarse-grained individual and/or group assignments. Administrators are likely to err on the side of excessive authorization to ensure that the individual has what he or she needs to perform their job. This is of even greater concern if the individual is a contractor or a vendor with high privilege inside of your organization.

In this kind of environment, “ghost” (a.k.a. rogue) accounts persist long after an individual has left, or temporary privilege authorization for an existing account was never withdrawn. The account might have belonged to a former employee who still has access to corporate systems and data, a contractor whose assignment is finished or an existing employee who was granted temporary rights or has changed jobs within the organization.

Your plan for identity management should include:

- **Centrally managed access control** separated from individual applications so that controls can be maintained efficiently, according to policy, across the enterprise.
- **Well-defined, granular role-based access control (RBAC).** Roles are created for particular job functions and the necessary permissions defined for each role. Roles can then be assigned to individuals, making it easy to add or change responsibilities.

- **Timely and accurate provisioning and de-provisioning** of employees, contractors and authorization privileges based on a well-defined evaluation and approval workflow.
- **Real-time monitoring and alerting**, comprehensive and timely auditing, and strong reporting.

Identity Management Solutions

MySQL Enterprise Security – External Authentication

There are various options that can be implemented towards solving Identity Management challenges. MySQL Enterprise Edition includes support for external authentication to various sources with abstractions supporting both individual users to accounts and privileges or external user groups with group to privilege mapping.

MySQL users can be authenticated using PAM or Microsoft Windows Active Directory.

- **MySQL External Authentication for LDAP** - Enables you directly authenticate users to LDAP.
- **MySQL External Authentication for PAM** - Enables you to configure MySQL to use PAM to authenticate users on Unix/Linux, and other systems.
- **MySQL External Authentication for Windows** - Enables you to configure MySQL to use native Windows services to authenticate client connections. Users who have logged in to Windows can connect from MySQL client programs to the server based on the token information in their environment without specifying an additional password.

Additionally MySQL accounts are supported within the Oracle suite of identity management products, providing a fully integrated, centralized, managed and automated solution. It covers access control, authorization & authentication, granular role-based controls, and provisioning capabilities — all to help meet and exceed the directives in PCI requirements 7, 8 and 10.

Oracle Identity Management

Oracle’s identity management products provide powerful monitoring, auditing and reporting capabilities to effectively meet PCI requirement 10 terms for monitoring and audit. Oracle Access Management Suite sets and enforces policy-based authentication and monitors user access activity. Identity Manager detects rogue accounts and changes user access privileges.

Oracle Access Management Suite’s auditing services provide detailed and flexible logging of monitored events such as authentication success or failure. Audit logs can be written either to a flat file or to a database and exported to any third-party reporting tool to produce comprehensive auditing reports.

Monitoring and Tracking Access to Cardholder Data

Strong data security policies and controls around cardholder data requires continuous monitoring, tracking and auditing to assure that they are operating as intended and are being properly enforced. Having the ability to verify that controls are effective and detect and address unauthorized activity, completes a robust cardholder data protection program and enables organizations to verify to Qualified Security Assessors (QSAs) that their policies and controls are in full force.

Requirement 10: Track and Monitor Access to Cardholder Data

PCI DSS requirement 10 requires organizations to track and monitor access to network resources and cardholder data. . Note that PCI DSS does not focus solely on direct access to the data, recognizing that cardholder information exists in a live, dynamic production environment with many players and a lot of moving, changing parts.

The standard places strong emphasis on audit capabilities, in particular implement automated audit trails for:

- All system components
- For each individual users
- All actions taken by privileged users
- And secure audit trails so they cannot be altered

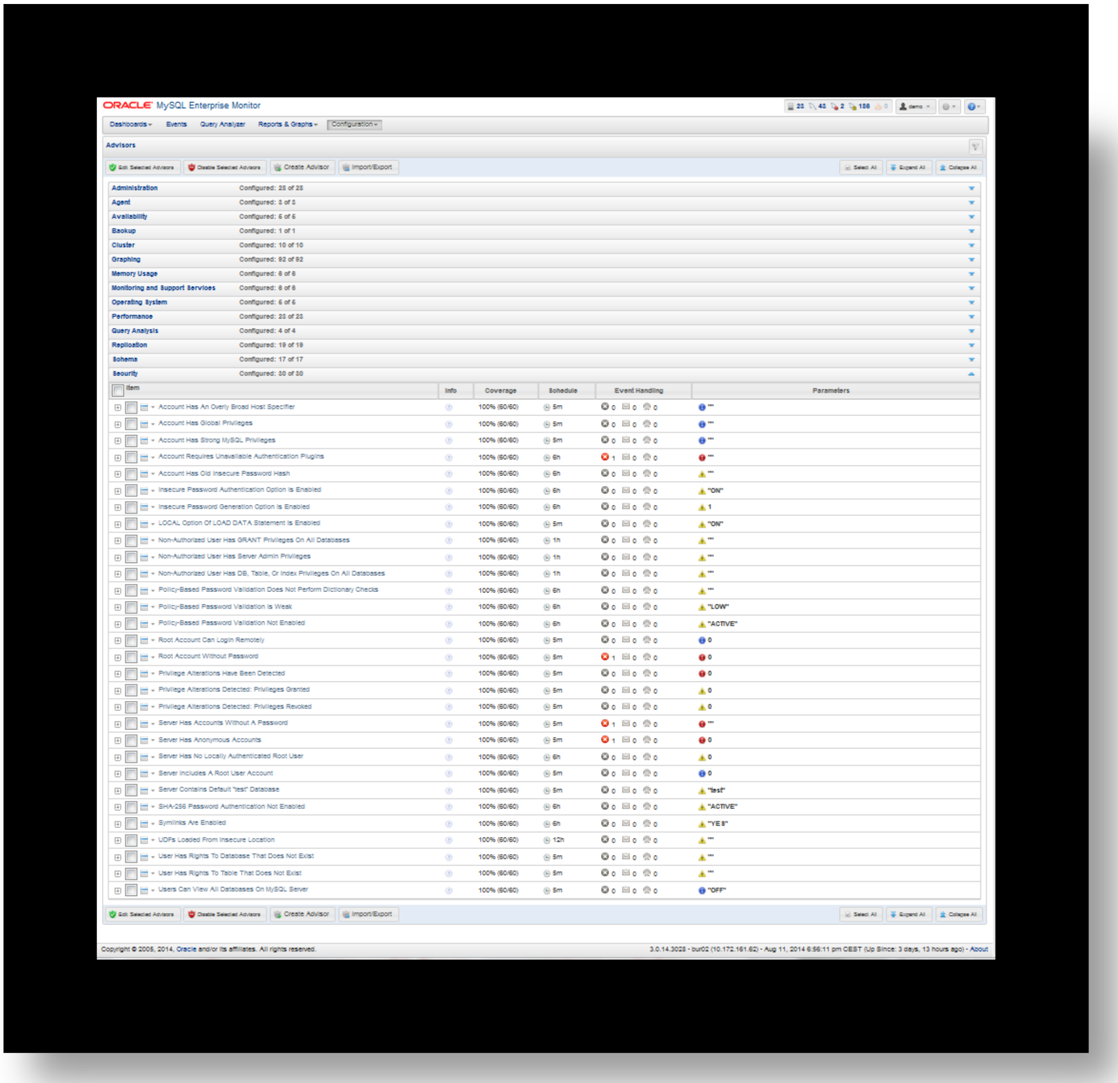
It also is essential to note that first comes access control, authorization, provisioning and authentication rules, then monitoring user and application activities. If you do not have policies and processes that control correct, authorized activity, then how can you monitor for incorrect or unauthorized activity?

MySQL Enterprise Monitor

Monitoring user activity is difficult. In most cases, there is no real-time monitoring and alerting capability. Because access control and, hence, monitoring is managed separately for each application or system, it is close to impossible to monitor individuals as a practical matter.

MySQL Enterprise Monitor and MySQL Enterprise Audit enable organizations to implement automated process that monitor individual users access to system components and provide audit trails to reconstruct all user access to cardholder data. Access to audit trails, invalid access attempts, initialization of audit logs, and creation and deletion of system-level objects also are covered in the requirements.

Once again, MySQL Enterprise Monitor reports on user accounts, privileges providing the ability to track and monitor MySQL database access and usage.



Administration	Configured: 28 of 28
Agent	Configured: 3 of 3
Availability	Configured: 5 of 5
Backup	Configured: 1 of 1
Cluster	Configured: 10 of 10
Graphing	Configured: 92 of 92
Memory Usage	Configured: 8 of 8
Monitoring and Support Services	Configured: 8 of 8
Operating System	Configured: 5 of 5
Performance	Configured: 23 of 23
Query Analysis	Configured: 4 of 4
Replication	Configured: 19 of 19
Schema	Configured: 17 of 17
Security	Configured: 30 of 30

Item	Info	Coverage	Schedule	Event Handling	Parameters
Account Has An Overly Broad Host Specifier	?	100% (60/60)	5m	0 0 0	ℹ ""
Account Has Global Privileges	?	100% (60/60)	5m	0 0 0	ℹ ""
Account Has Strong MySQL Privileges	?	100% (60/60)	5m	0 0 0	ℹ ""
Account Requires Unavailable Authentication Plugins	?	100% (60/60)	6h	1 0 0	❗ ""
Account Has Old Insecure Password Hash	?	100% (60/60)	6h	0 0 0	⚠ ""
Insecure Password Authentication Option Is Enabled	?	100% (60/60)	6h	0 0 0	⚠ "ON"
Insecure Password Generation Option Is Enabled	?	100% (60/60)	6h	0 0 0	⚠ 1
LOCAL Option Of LOAD DATA Statement Is Enabled	?	100% (60/60)	5m	0 0 0	⚠ "ON"
Non-Authorized User Has GRANT Privileges On All Databases	?	100% (60/60)	1h	0 0 0	⚠ ""
Non-Authorized User Has Server Admin Privileges	?	100% (60/60)	1h	0 0 0	⚠ ""
Non-Authorized User Has DB, Table, Or Index Privileges On All Databases	?	100% (60/60)	1h	0 0 0	⚠ ""
Policy-Based Password Validation Does Not Perform Dictionary Checks	?	100% (60/60)	6h	0 0 0	⚠ ""
Policy-Based Password Validation Is Weak	?	100% (60/60)	6h	0 0 0	⚠ "LOW"
Policy-Based Password Validation Not Enabled	?	100% (60/60)	6h	0 0 0	⚠ "ACTIVE"
Root Account Can Login Remotely	?	100% (60/60)	5m	0 0 0	ℹ 0
Root Account Without Password	?	100% (60/60)	5m	1 0 0	❗ 0
Privilege Alterations Have Been Detected	?	100% (60/60)	5m	0 0 0	❗ 0
Privilege Alterations Detected: Privileges Granted	?	100% (60/60)	5m	0 0 0	⚠ 0
Privilege Alterations Detected: Privileges Revoked	?	100% (60/60)	5m	0 0 0	⚠ 0
Server Has Accounts Without A Password	?	100% (60/60)	5m	1 0 0	❗ ""
Server Has Anonymous Accounts	?	100% (60/60)	5m	1 0 0	❗ 0
Server Has No Locally Authenticated Root User	?	100% (60/60)	6h	0 0 0	⚠ 0
Server Includes A Root User Account	?	100% (60/60)	5m	0 0 0	ℹ 0
Server Contains Default "test" Database	?	100% (60/60)	5m	0 0 0	⚠ "test"
SHA-256 Password Authentication Not Enabled	?	100% (60/60)	6h	0 0 0	⚠ "ACTIVE"
Symlinks Are Enabled	?	100% (60/60)	6h	0 0 0	⚠ "YES"
UDFs Loaded From Insecure Location	?	100% (60/60)	12h	0 0 0	⚠ ""
User Has Rights To Database That Does Not Exist	?	100% (60/60)	5m	0 0 0	⚠ ""
User Has Rights To Table That Does Not Exist	?	100% (60/60)	5m	0 0 0	⚠ ""
Users Can View All Databases On MySQL Server	?	100% (60/60)	5m	0 0 0	ℹ "OFF"

Copyright © 2005, 2014, Oracle and/or its affiliates. All rights reserved.

3.0.14.3028 - bur02 (10.172.161.62) - Aug 11, 2014 6:56:11 pm CEST (Up Since: 3 days, 13 hours ago) - About

And as discussed earlier, Oracle Audit Vault and Database Firewall provides robust monitoring, auditing and reporting capabilities, which can, of course, be applied to all types of privileged users.

MySQL Enterprise Audit

Auditing can be a very time consuming process, as is reporting for PCI audits. Meeting regulatory and internal requirements often is inefficient because logs are split among applications and systems. This requires manual information gathering, analysis and reporting, as well as correlation when multiple applications and systems are involved.

MySQL Enterprise Audit allows users to quickly and seamlessly add policy-based auditing compliance to PCI applications. It supports selective, dynamically configurable tracking of user level activity so you can implement activity-based policies and manage audit log files. Templates or highly customized filters using simple JSON filter definition can be used for fine-grained auditing. Filter on connections, users, table access, access type, statement status (success/failure), query content, and more.

MySQL Enterprise Firewall

MySQL Enterprise Firewall is an instance-based SQL firewall that provides security personnel with the ability to block, count, log and alert on a per user basis, activities that may indicate attempts to gain unauthorized access and/or abuse system privileges. It continuously assesses SQL statements, evaluating the activities against defined expected patterns, including system events such as changes to application tables and creating privileged users.

Oracle Audit Vault and Database Firewall

Additionally, for Audit log management or SQL firewall capabilities, MySQL is also integrated and supported by the Oracle Audit Vault and Database Firewall product which provides an additional option for security personnel to detect and alert on activities that may indicate attempts to gain unauthorized access and/or abuse system privileges. It continuously monitors the audit data collected, evaluating the activities against defined alert conditions, including system events such as changes to application tables and creating privileged users. Oracle Audit Vault and Database Firewall collects audit trails from MySQL and Oracle and non-Oracle databases, as well as operating systems, directories, and other sources. This includes Oracle Database fine grained and conditional Audits.

Oracle Audit Vault and Database Firewall also provides powerful built-in reports to monitor a wide range of activity to support QSA assessments, internal audits, security programs and operational requirements. Rules can be put in place to automatically highlight specific rows so that report users can quickly spot suspicious or unauthorized activity. Out-of-the-box reports include information on database account management, roles and privileges, object management, and login failures.

Oracle Audit Vault and Database Firewall provides centralized management of Oracle database audit settings, simplifying the job of the IT security and internal auditors in managing audit settings across the enterprise and demonstrating compliance and repeatable controls to QSAs.

MySQL and PCI Solutions Map

Below is a summary table that shows how MySQL Enterprise Edition solutions as well some other Oracle Product integrations map onto specific sections of the PCI DSS, helping to address our customer's compliance needs.

CHAPTER		PCI 3.0 REQUIREMENT	MATCHING MySQL CAPABILITY
	BUILD AND MAINTAIN A SECURE NETWORK AND SYSTEMS		
2:	DO NOT USE VENDOR-SUPPLIED DEFAULTS FOR SYSTEM PASSWORDS AND OTHER SECURITY PARAMETERS		
	Malicious individuals (external and internal to a company) often use vendor default passwords and other vendor default settings to compromise systems. These passwords and settings are well known by hacker communities and easily determined via public information.		
	2.1:	Always change vendor-supplied defaults and remove or disable unnecessary default accounts **before** installing a system on the network. This applies to ALL default passwords, including but not limited to those used by operating systems, software that provides security services, application and system accounts, *point-of-sale* (POS) terminals, simple network management protocol (SNMP), community strings, etc.	MySQL packages both setup or contain tools for secure initial creation of accounts and passwords during installation. Passwords for root accounts are prompted for during Windows installation. The MySQL validate password Component Plugin allows users to set policies with requirements for password length and other password strength requirements (case, dictionary check, numbers, special characters). MySQL also supports password expiration and password resets in supporting client applications.
	2.2:	Develop configuration standards for all system components. Assure that these standards address all known security vulnerabilities and are consistent with industry-accepted system hardening standards may include, but are not limited to: • Center for Internet Security (CIS) • International Organization for Standardization (ISO) • SysAdmin Audit Network Security (SANS) Institute • National Institute of Standards Technology (NIST).hardening standards. Sources of industry-accepted	MySQL Enterprise Monitor provides out-of-the-box security best practices and vulnerability scans, alerting you to any potential security issues, and providing recommendations to address them. Additionally, custom checks that map to more specific requirements can be added to assure hardening is in place and maintained.

CHAPTER		PCI 3.0 REQUIREMENT		MATCHING MYSQL CAPABILITY
		2.2.4	Configure system security parameters to prevent misuse.	Follow MySQL security guidelines or various guides relate to the Steps to Secure. Monitor, Report, Alert with MySQL Enterprise Monitor. Oracle Audit Vault and Database Firewall consolidates and continually analyzes audit data generated by the MySQL Enterprise Audit. Oracle Audit Vault and Database Firewall can report and alert on threats detected within MySQL audit data.
		2.2.5	Remove all unnecessary functionality, such as scripts, drivers, features, subsystems, file systems, and unnecessary web servers	MySQL Enterprise Server allows specific components to be installed or removed. Based on the OS target setting are generically “secure”, however based on a users environment typically permissions can be further tightened.
	2.3:	Encrypt all non-console administrative access using strong cryptography. Use technologies such as SSH, VPN, or SSL/TLS for web-based management and other non-console administrative access		MySQL Enterprise database provides network encryption (SSL/TLS and native) to encrypt all traffic from the mysql client, Workbench, between the middle tiers and the database, between clients/connectors and the database, and between database servers themselves. Additionally, MySQL users can be restricted to the use of SSL enabled connections to ensure use of strong cryptography for their network connections.
	2.6:	Shared hosting providers must protect each entity's hosted environment and cardholder data. These providers must meet specific requirements as detailed in *Appendix A: "Additional PCI DSS Requirements for Shared Hosting Providers"*		See Appendix A

PROTECT CARDHOLDER DATA		
3:	PROTECT STORED CARDHOLDER DATA	
CHAPTER	PCI 3.0 REQUIREMENT	MATCHING MYSQL CAPABILITY
Protection methods such as encryption, truncation, masking, and hashing are critical components of cardholder data protection. If an intruder circumvents other security controls and gains access to encrypted data, without the proper cryptographic keys, the data is unreadable and unusable to that person. Other effective methods of protecting stored data should also be considered as potential risk mitigation opportunities. For example, methods for minimizing risk include not storing cardholder data unless absolutely necessary, truncating cardholder data if full PAN is not needed, and not sending unprotected PANs using end-user messaging technologies, such as e-mail and instant messaging. Please refer to the *PCI DSS Glossary of Terms, Abbreviations, and Acronyms* for definitions of "strong cryptography" and other PCI DSS terms.		
3.3:	Mask PAN when displayed (the first six and last four digits are the maximum number of digits to be displayed), such that only personnel with a legitimate business need to see the full PAN *Note: This requirement does not supersede stricter requirements in place for displays of cardholder data—for example legal or payment card brand requirements for point-of-sale (POS) receipts*	MySQL Enterprise Masking and De-identification provides specific functions to mask PAN data whether Strict or Relaxed . These can also be used in conjunction with SQL, permissions, column and view level security to limit data access.
3.4:	Render PAN unreadable anywhere it is stored (including on portable digital media, backup media, and in logs) by using any of the following approaches: • One-way hashes based on strong cryptography, (hash must be of the entire PAN) • Truncation (hashing cannot be used to replace the truncated segment of PAN) • Index tokens and pads (pads must be securely stored) • Strong cryptography with associated key-management processes and procedures *Note: It is a relatively trivial effort for a malicious individual to reconstruct original PAN data if they have access to both the truncated and hashed version of a PAN. Where hashed and truncated versions of the same PAN are present in an entity's environment, additional controls should be in place to ensure that the hashed and truncated versions cannot be correlated to reconstruct the original PAN.*	MySQL Enterprise Encryptions and MySQL Enterprise Masking and De-identification provide functions to both symmetrically and asymmetrically (PKI) encrypt, hash, pad, mask or truncate the PAN (Primary Account Numbers) MySQL Enterprise Transparent Data Encryption (TDE) protects critical data by enabling data-at-rest encryption in the database. MySQL Enterprise Backup can encrypt (and compress) the entire backup on disk.

CHAPTER	PCI 3.0 REQUIREMENT		MATCHING MYSQL CAPABILITY
			Database backups files can be securely shared with the receiving party. MySQL Enterprise backup includes support for Oracle Secure Backup which provides a solution for backing up and encrypting directly to tape storage. Encryption algorithms supported includes AES 256 in MySQL 5.6 and above.
3.5:	Document and implement procedures to protect keys used to secure stored cardholder data against disclosure and misuse: *Note: This requirement applies to keys used to encrypt stored cardholder data, and also applies to key-encrypting keys used to protect data-encrypting keys—such key-encrypting keys must be at least as strong as the data-encrypting key.*		MySQL Enterprise TDE table, tablespace, redo/undo, binlog, and audit file keys are stored in the database and encrypted using a separate master encryption key that is stored in a Key Vault. MySQL Enterprise Edition Key Vault support (KMIP - includes Oracle Key Vault and other key management products plus additional key management protocols) provides centralized key management, auditing and controls for keys and other secrets.
	3.5.1	Restrict access to cryptographic keys to the fewest number of custodians necessary.	` MySQL use of key vaults removes the need to have access to the master encryption key. Designated individuals (DBA or Database Security Administrator DSA) who require the master password will have very controlled access via the vendors Key Vault access controls.
	3.5.2	Store secret and private keys used to encrypt/decrypt cardholder data in one (or more) of the following forms at all times: • Encrypted with a key-encrypting key that is at least as strong as the data-encrypting key, and that is stored separately from the data- encrypting key • Within a secure cryptographic device (such as a hardware security module (HSM) or PTS-approved point of interaction device) • As at least two full-length key components or key shares, in accordance with an industry-accepted method *Note: It is not required that public keys be stored in one of these forms.*	Key Vault support (KMIP - includes Oracle Key Vault and other key management products plus additional key management protocols) provides centralized key management, auditing and controls for keys and other secrets.

	3.6:	3.6.1	Generation of strong cryptographic keys	MySQL Enterprise Encryption provides supports the latest generation of strong cryptographic keys as well as their usage within both symmetric and asymmetric encryption and decryption.
		3.6.2	Secure cryptographic key distribution	Key Vault support (KMIP - includes Oracle Key Vault and other key management products plus additional key management protocols) provides centralized and secure methods meeting PCI key distribution and management requirements
		3.6.3	Secure cryptographic key storage	Key Vault support (KMIP - includes Oracle Key Vault and other key management products plus additional key management protocols) provides centralized and secure methods key distribution and management
		3.6.4	Cryptographic key changes for keys that have reached the end of their cryptoperiod (for example, after a defined period of time has passed and/or after a certain amount of cipher-text has been produced by a given key), as defined by the associated application vendor or key owner, and based on industry best practices and guidelines (for example, NIST Special Publication 800-57).	Key Vault support (KMIP - includes Oracle Key Vault and other key management products plus additional key management protocols) manages key lifecycle stages including creation, rotation, and expiration.

MAINTAIN A VULNERABILITY MANAGEMENT PROGRAM		
6:	**DEVELOP AND MAINTAIN SECURE SYSTEMS AND APPLICATIONS**	
	Unscrupulous individuals use security vulnerabilities to gain privileged access to systems. Many of these vulnerabilities are fixed by vendor-provided security patches, which must be installed by the entities that manage the systems. All systems must have all appropriate software patches to protect against the exploitation and compromise of cardholder data by malicious individuals and malicious software. *Note: Appropriate software patches are those patches that have been evaluated and tested sufficiently to determine that the patches do not conflict with existing security configurations. For in-house developed applications, numerous vulnerabilities can be avoided by using standard system development processes and secure coding techniques.*	
CHAPTER	**PCI 3.0 REQUIREMENT**	**MATCHING MYSQL CAPABILITY**

	6.1:	Establish a process to identify security vulnerabilities, using reputable outside sources for security vulnerability information, and assign a risk ranking (for example, as “high,” medium,” or “low”) to newly discovered security vulnerabilities. *Note: Risk rankings should be based on industry best practices as well as consideration of potential impact.*		Oracle follows the Common Vulnerability Scoring System (CVSS) when providing severity ratings for bug fixes released in Critical Patch Updates (CPUs) http://www.oracle.com/technetwork/topics/security/cvssscoringsystem-091884.html Subscribe to Oracle Critical Patch Updates, Security Alerts and Third Party Bulletin RSS feed: http://www.oracle.com/technetwork/topics/security/alerts-086861.html.
	6.2:	Ensure that all system components and software are protected from known vulnerabilities by installing all vendor-supplied security patches. Install critical security patches within one month of release.		MySQL frequent release updates include any critical patch updates which customers can implement.
	6.4:	6.4.1	Separate development/test environments from production environments, and enforce the separation with access controls	MySQL Enterprise supports various Authentication Plugins that provide the ability to centrally manage database users and their authorizations in one central place. MySQL is also supported by Oracle Identity Management, giving the ability to centrally manage database users and their authorizations in one central place.
		6.4.3	Production data (live PANs) are not used for testing or development	MySQL Enterprise Masking and De-identification provides functions that allow developers obfuscate payment card numbers, and other sensitive information, for testing and development environments.

		6.4.5	Change control procedures for the implementation of security patches and software modifications must include the following: 6.4.1 Documentation of impact 6.4.2 Documented change approval by authorized parties 6.4.3 Functionality testing to verify that the change does not adversely impact the security of the system 6.4.4 Back-out procedures	Database change control procedures, checks, and controlled schema migration can be performed with MySQL Workbench. MySQL Enterprise Monitor also provides alerts and advisors that allow you to receive notifications any time that database changes are made.
	6.5:	6.5 Address common coding vulnerabilities in software-development processes as follows: • Train developers in secure coding techniques, including how to avoid common coding vulnerabilities, and understanding how sensitive data is handled in memory. • Develop applications based on secure coding guidelines. *Note: The vulnerabilities listed at 6.5.1 through 6.5.10 were current with industry best practices when this version of PCI DSS was published. However, as industry best practices for vulnerability management are updated (for example, the OWASP Guide, SANS CWE Top 25, CERT Secure Coding, etc.), the current best practices must be used for these requirements*		MySQL Enterprise Firewall provides methods to learn normal SQL statements patterns and block unexpected patterns that could be exploited if developers have a gap in their secure coding techniques – thus blocking attacks coming from vulnerability exploits on applications. MySQL supports use of Oracle Audit Vault and Database Firewall for inspecting inbound SQL statements for SQL injection and other statement based attacks, esp. but not limited to SQL injection exploits. MySQL Enterprise edition enables encrypted network connections to and from the MySQL database using OpenSSL with SSL/TLS.
		6.5.1	Injection flaws, particularly SQL injection. Also consider OS Command Injection, LDAP and XPath injection flaws as well as other injection flaws	
		6.5.3	Insecure cryptographic storage	
		6.5.4	Insecure communications	
IMPLEMENT STRONG ACCESS CONTROL MEASURES				
7:	**RESTRICT ACCESS TO CARDHOLDER DATA BY BUSINESS NEED TO KNOW**			
	To ensure critical data can only be accessed by authorized personnel, systems and processes must be in place to limit access based on need to know and according to job responsibilities. "Need to know" is when access rights are granted to only the least amount of data and privileges needed to perform a job.			

	7.1:	Limit access to system components and cardholder data to only those individuals whose job requires such access.		
		7.1.1	Define access needs for each role, including: • System components and data resources that each role needs to access for their job function • Level of privilege required (for example, user, administrator, etc.) for accessing resources.	MySQL Database roles, object privileges and database user access controls provide basic security. MySQL Enterprise Monitor will alert you when it identifies any user account that has unnecessarily broad permissions. MySQL Enterprise Security provides External Authentication support which can be used to limit access and control individuals with access. MySQL Enterprise Monitor provides alerts and reports related to database access, rights, and overly promiscuous privileges. MySQL Roles, Proxy user accounts, Access Controls and integration with External Authentication can be configure to limit assigned access to map to job needs. MySQL Enterprise Monitor will alert you when it identifies any user account that has unnecessarily broad permissions as well as changes to database objects, schemas, and user accounts. MySQL Masking and De-identification removes or masks sensitive application data fields based on organizational and regulatory policy combined with the requestor’s entitlements.
		7.1.2	Restrict access to privileged user IDs to least privileges necessary to perform job responsibilities.	
		7.1.3	Assign access based on individual personnel's job classification and function.	
	7.2:	Establish an access control system for systems components with multiple users that restricts access based on a user's need to know, and is set to "deny all" unless specifically allowed. This access control system must include the following:		MySQL Roles, Proxy user accounts, Access Controls and integration with External Authentication can be configure to limit assigned access to map to job needs.

		7.2.1	Coverage of all system components	
		7.2.2	Assignment of privileges to individuals based on job classification and function	
		7.2.3	Default “deny-all” setting	
8:	**IDENTIFY AND AUTHENTICATE ACCESS TO SYSTEM COMPONENTS**			
	Assigning a unique identification (ID) to each person with access ensures that each individual is uniquely accountable for their actions. When such accountability is in place, actions taken on critical data and systems are performed by, and can be traced to, known and authorized users and processes. The effectiveness of a password is largely determined by the design and implementation of the authentication system—particularly, how frequently password attempts can be made by an attacker, and the security methods to protect user passwords at the point of entry, during transmission, and while in storage.			
	Note: These requirements are applicable for all accounts, including point-of-sale accounts, with administrative capabilities and all accounts used to view or access cardholder data or to access systems with cardholder data. This includes accounts used by vendors and other third parties (for example, for support or maintenance). However, requirements 8.1.1, 8.2, 8.5, 8.2.3 through 8.2.5, and 8.1.6 through 8.1.8 are not intended to apply to user accounts within a point-of-sale payment application that only have access to one card number at a time in order to facilitate a single transaction (such as cashier accounts).			
	8.1:	Define and implement policies and procedures to ensure proper user identification management for non-consumer users and administrators on all system components as follows:		MySQL Database authentication supports dedicated user accounts, and strong authentication capabilities including LDAP, Linux PAMs and Microsoft AD. This allows you to automatically adopt your already established login related security polices to MySQL.
		8.1.1	Assign all users a unique ID before allowing them to access system components or cardholder data.	

		8.1.2	Control addition, deletion, and modification of user IDs, credentials, and other identifier objects	MySQL Enterprise Monitor can also monitor your servers and alert you to any user account changes. The MySQL Audit Plugin allows you to know when your users are connecting, and what they are executing. You can then use this information to remove or disable user accounts when they have been inactive or should otherwise be terminated.
		8.1.3	Immediately revoke access for any terminated users	
		8.1.4	Remove/disable inactive user accounts at least every 90 days	
		8.1.5	Manage IDs used by vendors to access, support, or maintain system components via remote access as follows: • Enabled only during the time period needed and disabled when not in use. • Monitored when in use.	
		8.1.6	Limit repeated access attempts by locking out the user ID after not more than six attempts.	
		8.1.7	Set the lockout duration to a minimum of 30 minutes or until an administrator enables the user ID.	
		8.1.8	If a session has been idle for more than 15 minutes, require the user to re-authenticate to re-activate the terminal session.	
	8.2:	In addition to assigning a unique ID, ensure proper user-authentication management for non-consumer users and administrators on all system components by employing at least one of the following methods to authenticate all users: • Something you know, such as a password or passphrase • Something you have, such as a token device or smart card • Something you are, such as a biometric		MySQL provides two-factor authentication via X.509 certificates as an addition to passwords. It also support SASL authentication when using LDAP external authentication. MySQL provides the ability to define per user authentication policies for different levels of security requirements.
		8.2.1	Use strong cryptography, render all authentication credentials (such as passwords/phrases) unreadable during transmission and storage on all system components.	MySQL Authentication includes SHA256 support. MySQL provides network encryption (SSL/TLS), and strong authentication and supports X.509.

		8.2.3	Passwords/phrases must meet the following: • Require a minimum length of at least seven characters • Contain both numeric and alphabetic characters. Alternatively, the passwords/phrases must have complexity and strength at least equivalent to the parameters specified above.	MySQL Enterprise includes support for self-service password resets and the ability to specify policies that can meet the complexity requirements of PCI DSS 3.0.
		8.2.4	Change user passwords/passphrases at least every 90 days.	MySQL provides a password-expiration capability, which enables database administrators to expire account passwords and thus prevent subsequent logins until the password is updated.
		8.2.5	Do not allow an individual to submit a new password/phrase that is the same as any of the last four passwords/phrases he or she has used.	MySQL provides configurable policies for controlling password reuse.

	8.3:	Incorporate two-factor authentication for remote network access originating from outside the network by personnel (including users and administrators) and all third parties, (including vendor access for support or maintenance). *Note: Two-factor authentication requires that two of the three authentication methods (see Requirement 8.2 for descriptions of authentication methods) be used for authentication. Using one factor twice (for example, using two separate passwords) is not considered two-factor authentication.* *Examples of two-factor technologies include remote authentication and dial-in service (RADIUS) with tokens; terminal access controller access control systems (TACACS) with tokens; and other technologies that facilitate two-factor authentication.*	MySQL support usage of various options related to this. 1. Use X.509 2. To provide 2 factor access over the network many require both MySQL authentication as well as ssh authentication for remote access. This is supported within MySQL Workbench which is a common tool for MySQL database support and maintenance. 3. MySQL External Authentication options related to this.
	8.5:	Do not use group, shared, or generic IDs, passwords, or other authentication methods as follows: • Generic user IDs are disabled or removed. • Shared user IDs do not exist for system administration and other critical functions. Shared and generic user IDs are not used to administer any system components.	(Customer internal policy); MySQL Database dedicated user accounts; MySQL Authentication Plugins for External Authentication MySQL DBAs should not use root account but instead named accounts with "Super" or more limited privileges.

	8.7	All access to any database containing cardholder data (Including access by applications, administrators, and all other users) Is restricted as follows: • All user access to, user queries of, and user actions on databases are through programmatic methods. • Only database administrators have the ability to directly access or query databases. • Application IDs for database applications can only be used by the applications (and not by individual users or other non-application processes).	Enabled by proper usage of MySQL authentication or External Authentication in combination with properly restricted access controls related to the associated usage and accounts.

REGULARLY MONITOR AND TEST NETWORKS	
10	***TRACK AND MONITOR ALL ACCESS TO NETWORK RESOURCES AND CARDHOLDER DATA***
	Logging mechanisms and the ability to track user activities are critical in preventing, detecting, or minimizing the impact of a data compromise. The presence of logs in all environments allows thorough tracking, alerting, and analysis when something does go wrong. Determining the cause of a compromise is very difficult, if not impossible, without system activity logs.

	10.1:	Implement audit trails to link all access to system components to each individual user.		(Customer internal policy); Establish dedicated DBA accounts in the database. MySQL Enterprise Audit creates audit trails within the MySQL Server. MySQL audit trails can be collected in Oracle Audit Vault and Database Firewall. Audit trails can be encrypted and compressed, in XML or JSON formats.
	10.2:	Implement automated audit trails for all system components to reconstruct the following events:		(see below)
		10.2.1	All individual user accesses to cardholder data	MySQL Enterprise Audit
		10.2.2	All actions taken by any individual with root or administrative privileges	Establish dedicated DBA accounts in the database.
				Oracle Audit Vault and Database Firewall audit data consolidation for enterprise reports and alerting.
		10.2.3	Access to all audit trails	MySQL audit data can be stored in the Audit Vault.

		10.2.4	Invalid logical access attempts	MySQL general logging can audit failed login attempts. MySQL provides lockout after n-number of incorrect login attempts, and prepares an audit trail Audit Vault and Database Firewall can alert on invalid logical access attempts for privileged users
		10.2.5	Use of and changes to identification and authentication mechanisms—including but not limited to creation of new accounts and elevation of privileges—and all changes, additions, or deletions to accounts with root or administrative privileges.	MySQL authentication and audits MySQL Enterprise Monitor – assesses and alerts on identification and authentification changes. Oracle Audit Vault and Database Firewall can assess audit logs and alert for these changes. Oracle Audit Vault and Database Firewall can assess audit logs and alert for these changes.
		10.2.7	Creation and deletion of system-level objects	MySQL auditing, Comparisons reports using MySQL Workbench. MySQL Enterprise Monitor can alert you when any database objects are modified.
	10.3	Record at least the following audit trail entries for all system components for each event:		
		10.3.1	User identification	MySQL auditing. Filter on connections, users, table access, access type, statement status (success/failure), query content, and more.
		10.3.2	Type of event	
		10.3.3	Date and time	Oracle Audit Vault and Database Firewall audit data consolidation, reporting, alerting and protection MySQL connection user and session information include this data
		10.3.4	Success or failure indication	
		10.3.5	Origination of event	
		10.3.6	Identity or name of affected data, system component, or resource	

	10.5:	Secure audit trails so they cannot be altered.		Using MySQL Auditing and Oracle Audit Vault and Database Firewall audit data consolidation protects audit data in transit and storage, and helps meet chain of custody requirements for use in criminal prosecution. MySQL audit files can be encrypted at rest with AES 256.
		10.5.1	Limit viewing of audit trails to those with a job- related need.	MySQL audit data stored in Oracle Audit Vault and Database Firewall provides separation of duties limits MySQL audit files can be encrypted at rest with AES 256. access to audit data.
		10.5.2	Protect audit trail files from unauthorized modifications.	Oracle Audit Vault and Database Firewall separation of duties prevents access and modification of audit data by administrators (DBA)
		10.5.3	Promptly back-up audit trail files to a centralized log server or media that is difficult to alter.	MySQL audit data stored in Oracle Audit Vault and Database Firewall audit data consolidation provides a scalable and secure audit warehouse
		10.5.5	Use file-integrity monitoring or change-detection software on logs to ensure that existing log data cannot be changed without generating alerts (although new data being added should not cause an alert)	Oracle Audit Vault and Database Firewall audit data consolidation protects audit data in transit Oracle Audit Vault and Database Firewall separation of duties prevents access and modification of audit data by administrators (DBA)
	10.6:	Review logs and security events for all system components to identify anomalies or suspicious activity. *Note: Log harvesting, parsing, and alerting tools may be used to meet this Requirement.*		Oracle Audit Vault and Database Firewall provides out-of-box reports, customizable alerts, and an alert dashboard for monitoring audit data.
	10.7:	Retain audit trail history for at least one year, with a minimum of three months immediately available for analysis (for example, online, archived, or restorable from back-up)		Oracle Audit Vault and Database Firewall provides a scalable and secure audit warehouse for storing large volumes (Terabytes) of audit data.

Appendix A:

APPENDIX A: ADDITIONAL PCI DSS REQUIREMENTS FOR SHARED HOSTING PROVIDERS				
A:	**SHARED HOSTING PROVIDERS MUST PROTECT THE CARDHOLDER DATA ENVIRONMENT**			
	As referenced in Requirement 12.8 and 12.9, all service providers with access to cardholder data (including shared hosting providers) must adhere to the PCI DSS. In addition, Requirement 2.6 states that shared hosting providers must protect each entity's hosted environment and data. Therefore, shared hosting providers must additionally comply with the requirements in this Appendix.			
	A.1:		Protect each entity's (that is merchant, service provider, or other entity) hosted environment and data, per A.1.1 through A.1.4: A hosting provider must fulfill these requirements as well as all other relevant sections of the PCI DSS. *Note: even though a hosting provider may meet these requirements, the compliance of the entity that uses the hosting provider is not guaranteed. Each entity must comply with the PCI DSS and validate compliance as applicable.*	
		A.1.1	Ensure that each entity only runs processes that have access to that entity's cardholder data environment.	MySQL object privileges and user access controls provide basic security.
		A.1.2:	Restrict each entity's access and privileges to its own cardholder data environment only.	MySQL object privileges and user access controls provide basic security.
		A.1.3:	Ensure logging and audit trails are enabled and unique to each entity's cardholder data environment and consistent with PCI DSS Requirement 10.	Oracle Audit Vault and Database Firewall policies can be easily deployed to databases, enabling consistent auditing of access to cardholder data. Only the audit administrator may change audit policies and processes

		A.1.4:	Enable processes to provide for timely forensic investigation in the event of a compromise to any hosted merchant or service provider.	Oracle Audit Vault and Database Firewall provides out-of-box reports, customizable alerts and an alert dashboard for monitoring audit data. Customized reports can be created using various reporting tools. The Oracle Audit Vault and Database Firewall warehouse schema is published.

Conclusion

PCI-DSS represents perhaps the most promising effort at industry self-policing we have seen since the ubiquitous use of the public Internet and the widespread growth in Internet fraud began moving information security concerns to the fore. As criminals moved online to exploit billions of insecure consumer information records, the credit card companies moved with the times to create a highly prescriptive blueprint for securing cardholder data. This blueprint provides a strong foundation that I.T. organizations can build upon to create good data security programs.

Compliance and data security have proven difficult for many organizations, particularly those retailers whose security policies are not as mature as organizations in some other industries. However, the goals of PCI DSS are attainable, and the requirements of the standard can be fulfilled in a sustainable continuous program. The path forward requires a combination of sound security policy and the support of automated tools from leaders in data security like Oracle that enable compliance while simultaneously empowering organizations to improve their business practices.

Additional Resources

MySQL White Papers

MySQL Enterprise Edition
MySQL Enterprise Edition includes the most comprehensive set of advanced features and management tools to achieve the highest levels of scalability, security, reliability, and uptime.

MySQL: An Ideal Choice for The Cloud
Understand what makes MySQL an ideal choice for cloud-based database deployments, and why the leading Web database has already become ubiquitous in the cloud.

MySQL Enterprise Monitor
Explores the MySQL Enterprise Monitor in detail and explains how it can be leveraged as a "Virtual MySQL DBA" assistant to help over-extended DBAs proactively manage more MySQL servers with less time and effort.

Oracle Premier Support for MySQL: Rely on The Experts & Get Unique Benefits
Oracle Premier Support for MySQL offers benefits far beyond "insurance" -- value you can leverage even if you never experience problems.

Read more MySQL White Papers>>