MySQL and VMware: Encryption and Key Management for Developers
Overview

Whether you develop mostly on Linux or Windows, you can achieve a compliant implementation of encryption with the MySQL database and variations of MySQL like MariaDB. This white paper looks at some key decisions you will need to make about the encryption approach, and how we help our customers get encryption right.

Using the MySQL Built-in Encryption Primitives

MySQL supports a number of encryption and encoding operations directly from the SQL language. When encrypting a column you can use the ENCRYPT function, AES_ENCRYPT function, the older DES_ENCRYPT function, or the encoding or compression algorithms. If you want to use this approach to encryption and decryption, I would recommend that you use AES_ENCRYPT and AES_DECRYPT. For this, primitive MySQL uses the industry standard 128-bit AES algorithm, which is considered strong encryption and meets compliance regulations.

While the DES_ENCRYPT support is still a part of recognized industry standards, you will have a performance benefit when using the stronger 128-bit AES support and are not likely to run into the problem of a future deprecation of the Triple DES algorithm used by the MySQL DES_ENCRYPT method.

Here is an example of a MySQL insert statement that uses AES_ENCRYPT with a hex representation of the encryption key:

```sql
INSERT INTO t
VALUES (1,AES_ENCRYPT('text',UNHEX('F3229A0B371ED2D9441B830D21A390C3')));
```

Of course, hard-coding the encryption key is poor security practice and there is more about that below.

HINT: MySQL likes strings for encryption keys and Townsend Security’s Alliance Key Manager can return both Base64 and Hex encoded keys for use with MySQL statements.

Encryption at the Application Layer

Encrypting and decrypting directly in your SQL statements is not always possible or optimal. Don’t worry, you can also implement encryption in your application code if that makes more sense. Alliance Key Manager includes several language-specific software libraries for developers. For example, if you are a Java developer you can install and use our Java .jar files for full support for encryption key retrieval and on-device encryption. If you are a Windows C# developer you can add our Windows .NET Client to your Visual Studio project and have full support for key management. Both Java and C# have great support for encryption - you won’t need support from third parties for AES encryption - but you will need to implement encryption key management the right way.

What are some reasons you might want to do encryption at the application layer?

- Minimize the changes to SQL for different databases.
- Take advantage of the stronger 256-bit AES encryption method.
- Use an AES mode of encryption that uses an Initialization Vector such as Cipher Block Chaining (CBC) mode.
- Embed information in the encrypted field about the key used, the version of the key, and the IV.
- Create custom logic for encrypting larger blobs in the database.

Whatever the reason or combination of reasons, implementing encryption at the application layer is an easy choice to make with the Alliance Key Manager language SDKs.

Where are the Encryption Keys?

The single biggest challenge a developer will face when deploying encryption for MySQL is how to properly manage encryption keys. Not getting it right leaves the organization open to security failures, audit failures, and litigation. Here are some ways NOT TO STORE encryption keys:

- As a part of the SQL statement (see above).
- In the application code.
- In a file on the same server.
- In a file on a separate server.
- In a separate table in the MySQL database.

All of these approaches have been the cause of security audit failures for our customers. Don’t let this happen to you.
Where Do You Want to Do Encryption?

In some environments you may feel it too risky to retrieve an encryption key and use it in your application code or directly with SQL statements. Some types of environments that carry extra risk include:

- Internet-facing SaaS solutions
- Internet-facing Web services
- Internet-facing Web applications
- Unattended kiosk applications
- Internet-of-Things (IoT) devices

All of these environments represent the potential for higher risk. Instead of retrieving an encryption key and using it in your application or MySQL database where it may be exposed to loss, you can send the data to Alliance Key Manager and have it encrypted or decrypted inside the key server, and returned to you. The encryption key never leaves the key server, and the encrypted or decrypted data is protected by TLS strong encryption between your application and the key server.

How to Protect the Local Key Server Credentials

Storing encryption keys on a server that is designed for that purpose is the most important step you can take to protect those keys. You should also protect and monitor who has access to the key server credentials. With Alliance Key Manager, those credentials are a client-side certificate and private key. Here are ways to monitor access to those credentials:

**Windows:**
The client-side certificate and private key are stored in the Windows Certificate Store for the Alliance Key Manager .NET client software. Be sure to implement proper protections for certificates and keys in Windows Certificate Manager.

If you are using a scripting language like Python, Perl or Ruby and the credentials are stored in .PEM files, use native Windows auditing of these files. In Windows Explorer, select the credential files, right click and select Properties, select Security, then Advanced, then Auditing. Enable auditing for these files.

**Linux:**
Use native Linux file access controls to protect the .PEM files used to validate the connection to the key server. It is also recommended that you deploy a facility like <auditctl> and <auditd> to monitor access to the credentials.

VMware Deployment Considerations

A larger percentage of organizations in the US and globally use VMware or similar virtualization technology to lower infrastructure and system administrative costs. But not all encryption and key management solutions are friendly to VMware. If you are currently using VMware or plan to soon, here are some important considerations:

**Encryption Key Management:**
Encryption key management solutions are single-purpose security solutions that are usually deployed as a network-attached hardware server, or as a virtual machine. Be sure your key management solution can deploy as a virtual machine in the VMware environment. There should be no need to deploy expensive hardware outside of the VMware platform.

**HINT:** Alliance Key Manager is certified as VMware Ready for use as a VMware virtual machine in vSphere and vCloud environments. See the Coalfire/VMware Product Applicability Guide (PAG) for more information about how to deploy Alliance Key Manager to meet PCI DSS in a VMware context.

**Client Side Applications:**
Not all encryption applications and Software Development Kits (SDKs) are compatible with VMware. Be sure that your application or SDK explicitly supports the VMware platform. For example, if your Windows SDK is using the DPAPI function to protect encryption keys, it will not be compatible with many cloud or virtual platforms. All of the Alliance Key Manager applications and SDKs are fully supported on VMware and specific cloud platforms.

Key Rollover

It often comes as a surprise to developers that encryption keys have a life-time and that it is important to re-encrypt data periodically to maintain good security. This lifetime is called the “crypto-period”. The National Institute of Standards and Technology (NIST) has a good description of crypto-periods in Special Publication 800-57, Key Management Best Practices.

The practical impact to the developer is that you must plan on how key rollover activity can best be managed with
the minimum of fuss. Here are some practices that may be helpful:

**Store the Key Name in the Column or the Table**
All professional key management systems provide a user-friendly name for encryption keys. Consider adding a column to your MySQL table to hold the name of the key used to encrypt data. Knowing the key name used to encrypt data makes it easy to perform key rollover later.

**Store the Key Version in the Column or the Table**
Key managers also implement key versioning. That is, they will keep track of the current and past versions of the key and will assign those versions a user-friendly name. Be sure to store the key version name in the Column data itself, or add a new column to the table to hold this name.

**Store the Initialization Vector in the Column or the Table**
If you use an encryption algorithm that requires an Initialization Vector (IV) such as Cipher Block Chaining (CBC) or Counter mode (CTR) you must use a unique IV to encrypt and decrypt the data. The same value must be used on decryption that was used to encrypt the data. Be sure to store the IV in the encrypted column, or store it in a new column in the Table. The IV does not have to be encrypted itself, but the same value must be used on the decryption operation that was used for encryption.

If you are storing key names, key versions and IVs with the encrypted data you will find it much easier to perform key rollover when the time comes.

---

**Encryption Algorithm**

Always use a well known, standard form of encryption such as 256-bit AES. Using non-standard encryption algorithms can lower your security level and will certainly fail security audits like HIPAA, PCI-DSS and FISMA. It's not that AES is particularly better than some other algorithms that are available, but it is clearly the standard selected by NIST and written into many compliance regulations. Don't get caught short using a non-standard encryption algorithm - stick with AES!

**Encrypted Indexes**

Encrypted indexes can present a couple of challenges. First, they can slow the performance of your application if you need to periodically scan and decrypt the index in order to perform a SQL query. Additionally, if you implement the encryption at the application level you may find that encrypted indexes are not ordered in the way you expect. The ordering of the encrypted values may be quite different than the order of the plaintext, unencrypted values!

You can often mitigate the impact of an encrypted index by storing a partial extract of the data in a separate column, or doing a full decryption of an index to a temporary table and performing the query on the temporary index. Be aware of these potential issues as you deploy your encryption strategy.

---

**Summary**

Developers are the tip-of-the-spear when it comes to protecting their organizations from data breaches. When they are aware of the critical success factors for an encryption strategy they can dramatically improve the overall security posture of their companies.

At Townsend Security we provide developers with the tools they need to be successful and to get encryption right. Our key management solution, Alliance Key Manager, runs in all of the platform environments that developers need. The applications and software development kits that come with Alliance Key Manager run in VMware and everywhere else you might deploy the MySQL database.

---

**How Townsend Security Answers These Challenges**

Townsend Security's [Alliance Key Manager for VMware](https://townsendsecurity.com/products/alliance-key-manager-for-vmware) provides key answers to these challenges of working with VMware, securing information and the need to manage the keys and security of that information.

**Alliance Key Manager for VMware** provides a FIPS 140-2 compliant encryption key manager to VMware users who need to meet data privacy compliance regulations and security best practices.

**Alliance Key Manager for VMware** supports the following features:
- Deploys as VMware OVA
- FIPS 140-2 compliant
- OASIS KMIP compliant
- Supports VMware ESX/ESXi and vCloud Air environments

These key features, along with Townsend Security solution’s industry leading support team will work with you to show you how to address compliance requirements and how to assure that your information is safe, protected and that you have the right systems in place to provide for that protection going forward.
Contact us today to find out more about getting started, activating a trial of Alliance Key Manager for VMware or to further discuss exactly how compliance requirements, best practices and rock-solid security measures can come together to protect your systems and data.

Further Resources

Product Web Page:
Alliance Key Manager for VMware

eBook:
VMware Encryption - 9 Critical Components of a Defensible Encryption Strategy

Webinar:
Securing Data in VMware with Encryption and Key Management

Podcast:
Protecting Data with Encryption in VMware

About Townsend Security

Townsend Security creates data privacy solutions that help organizations meet evolving compliance requirements and mitigate the risk of data breaches and cyber-attacks. Over 3,000 companies worldwide trust Townsend Security’s NIST-compliant and FIPS 140-2 compliant solutions to meet the encryption and key management requirements in PCI DSS, HIPAA/HITECH, FISMA, GLBA/FFIEC, SOX, and other regulatory compliance requirements.

We invite you to learn more about us and view comments on the latest happenings in the security and encryption space by going to our blog.