InterSystems
White Paper

Using InterSystems Caché for Securely Storing Credit Card Data

InterSystems Caché

Andreas Dieckow
Principal Product Manager
InterSystems Corporation

Using InterSystems Caché for Securely Storing Credit Card Data

Introduction

In today's world, an ever-increasing number of purchases and payments are being made by credit card. Although merchants and service providers who accept credit cards have an obligation to protect customers' sensitive information, the software solutions they use may not support "best practices" for securing credit card information. To help combat this issue, a security standard for credit card information has been developed and is being widely adopted.

The Payment Card Industry (PCI) Data Security Standard (DSS) is a set of guidelines for securely handling credit card information. Among its provisions are recommendations for storing customer information in a database. This paper will outline how software vendors can take advantage of the InterSystems Caché® database – now and in the future – to comply with data storage guidelines within the PCI DSS.

Use cases for securing credit card information

The best way to ensure the security of information within a database is not to persist (store) it at all. But obviously, enterprises that deal with credit card data must save some information, such as the cardholders' names, Personal Account Numbers (PANs), expiration dates, and service codes. The PCI DSS recommends only persisting the minimum necessary amount of cardholder data. When cardholder data must be stored, the PCI DSS requires that (at a minimum) PANs be rendered unreadable in the database and in all journal logs.

Exactly how PANs are secured may differ according to the use case. In general, use cases fall into one of two categories: when a credit card is being used purely for verification, and when it is being used to pay for goods or services.

Using a credit card for verification

When an application is designed to accept a credit card as a form of identification (for example, when someone uses their card to retrieve a record of their airline reservation) it is not necessary for a useable clear-text PAN to be stored at all. According to the PCI DSS, either hashing or truncation may be used to store a version of the PAN that is sufficient for verification purposes.

- **Hashing**
  Information is transformed according to a complex algorithm and only the transformed, or hashed, version is stored. The hashing algorithm only works one way – it is impossible to uniquely determine the original information from the hashed version. For verification purposes, a hashed version of the PAN provided by the cardholder can be compared to the stored hash value.

- **Truncation**
  Only a portion of the information is stored. For verification purposes, the PAN provided by the cardholder is truncated in the same way and compared to the stored value. In general, truncation provides weaker security than hashing.

Using a credit card for payments

Applications that accept credit card payments must have access to a usable PAN in order to process a transaction. According to the PCI DSS, there are three acceptable ways of handling PAN information.

- **The PAN is not persisted (stored on disk) at all**
  This scenario might occur, for instance, when someone makes a credit card purchase from an online vendor and checks out as a "guest". The card holder must provide the complete PAN each time they make a purchase. The application will use the PAN in memory, but not persist it. (The PCI DSS includes guidelines for securing PANs and other sensitive information in transit, but that is beyond the scope of this paper.)

- **Truncation**
  Only a portion of the PAN is stored. The cardholder must provide the missing information that will allow the application to reassemble the PAN in memory. A useable PAN exists only in memory, not on disk.

- **Encryption**
  The PAN is transformed into cipher-text or clear-text according to a complex algorithm, using an encryption key, but only cipher-text is stored. Unlike hashing, encryption allows for two-way transformation. Using the encryption key, the application can decipher the PAN and use it (in memory) to process a credit card transaction.

The PCI DSS calls for using "strong" encryption and re-encrypting information periodically. Also, encryption keys must not be stored in, or tied to, user accounts.

How Caché currently enables secure data storage

Caché offers a strong, consistent, and high-performance security structure for applications, and is certified by Common Criteria.

Here is how the current release (Caché 2010) enables applications to securely store data such as PANs:

- **Hashing**
  Caché provides built-in access to several Secure Hash Algorithms (e.g. SHA-1) to hash data.

- **Truncation**
  Fully supported, and implemented as part of the application running inside Caché.

- **Database encryption**
  Caché implements the Advanced Encryption Standard (AES) algorithm.

With database encryption, the entire database is encrypted using one encryption key. Access to the key is managed by the system, so a user account (i.e., a process) does not hold the database encryption key.

All information, including indexes, stored in an encrypted database is protected.

- **Data element encryption**
  Caché allows for individual pieces of information to be encrypted by offering developers access to the encryption suite. Data element encryption is often preferred to store sensitive information, like PANs, because it allows (with the correct provisioning) the re-encryption of data elements, without interruptions to database access.

- **Auditing**
  Caché comes with a robust and tamper-resistant auditing system, which audits all changes to the security model. Application developers can use the same audit database by incorporating calls to the audit system in the application code.

Enhancements to the Caché Security Model

Because the PCI DSS is widely used, Caché includes several changes specifically designed to make it easier to build applications that comply with the standard. These enhancements, available in Caché 2011, are mainly concerned with the key management used by data element encryption.

Managed keys

With this enhancement, the encryption key material used for data element encryption is securely held by the system, by storing it in the same protected memory location as the database encryption key. Applications will refer to individual encryption keys using a unique KeyID, therefore eliminating direct access to the key material itself.

To make things easier for developers, when this new method of data element encryption is used, the KeyID is embedded in the resulting cipher-text. This enables the decryption process to automatically identify the key that was used to encrypt the data. The new managed key system supports several such keys in addition to the database encryption key. This will make it easy for application developers to satisfy re-encryption requirements in real time and with virtually no impact to the performance of the deployed application.

Conclusion

The PCI DSS is being adopted by merchants and service providers around the world who need to securely handle credit card information. Application providers need to make sure their solutions are compliant with this standard.

InterSystems Caché gives developers the ability to build applications that comply with the PCI DSS. Enhancements included in Caché 2011 will make that task even easier.

InterSystems Corporation

World Headquarters
One Memorial Drive
Cambridge, MA 02142-1356
Tel: +1.617.621.0600
Fax: +1.617.494.1631

InterSystems.com

InterSystems