|  |  |
| --- | --- |
|  | THE UNIVERSITY OF BRITISH COLUMBIA (*UBC Athletics & Recreation*) |

|  |  |
| --- | --- |
| **Requirement 3: Protect stored cardholder data** | |
| **Date of Issue:**  () |

**PURPOSE**

To document the procedures to meet PCI compliance requirements(*3.4.1-3.6.8*)

**POLICY**

These procedures relate to university policy 106 (Access to and Security of Administrative Information), and section 5 (PCI-DSS Requirements) of UBC’s Information Security Manual.

**PERSONNEL INVOLVED**

(Manager - Computer Systems, Coordinator - Computer Systems)

**CONTACT INFORMATION**

For procedure enquiries, please contact (*Ben Jan, Manager – Computer Systems, 604.822.0444, webteam@rec.ubc.ca*).

**PROCEDURE and INFORMATION**

1. 3.4.1 does not apply as column-level database encryption is used instead of disk encryption.
2. 3.6.2 – 3.6.3 are handled by the Class software – see supporting documentation below
3. 3.5, 3.6.6, 3.6.8 do not apply as encryption keys are not displayed to users.
4. 3.6 – see supporting documentation below
5. 3.6.4 - 3.6.7 - Encryption keys will be changed annually or as needed by either the Manager – Computer Systems or Coordinator – Computer Systems.
   1. How to – see Implementing\_Class\_in\_a\_PCI\_compliant\_environment\_7.0X.pdf (Page 22)

**REVISION HISTORY**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date** | **Details of Change** | **Changed By** | **Approved By** | **Title** |
| (*Insert Date*) | (*Initial Version*) | (*Name*) | (*Name*) | (*Approver’s Job Title*) |

Supporting documentation, received from [Charissa.Wall@activenetwork.com](mailto:Charissa.Wall@activenetwork.com) 7/15/2010

Maintaining Encryption Keys  
  
The current encryption key and vector are not displayed to the user, the process is just used to generate a new key and vector. There is no minimum length for the key or vector as long as neither is blank. The key and vector (just as the user entered them) are encrypted using Internal Encryption and are then stored in the CLASS\_ENCRYPTION\_KEY table in the database. Each key record has a unique id so we can determine which key to use to decrypt the data when necessary.  
  
Internal Encryption / Decryption  
  
Class maintains an internal encryption key (192 bit) and vector (64 bit) as constants within the source code. Encryption and description of data being stored in the database is done using these keys through our Standard Encryption Process.  
  
Standard Encryption Process  
  
The data to encrypt and the applicable encryption key and vector are passed in to a standardized encryption procedure. All encryption and decryption is done using Microsoft advapi32.dll, specifying the "Microsoft Enhanced Cryptographic Provider v1.0".  
  
An MD5 hash is used on the encryption key passed in. A DES3 key is derived from the hashed key. The vector is added to DES3 key (if the vector is < 8 bytes it is first padded out to 8 bytes with null chars). Binary data returned from the encryption process is converted to a Base 64 string for storage in the database. Decryption is done exactly the same way as encryption, but in reverse.  
  
Encrypting Credit Card Data  
  
Active encryption key and vector are read from the CLASS\_ENCRYPTION\_KEY table in the database, and are decrypted using Internal Decryption. Credit card information is encrypted using this key and vector using the Standard Encryption Process. Credit card information is stored in the database in encrypted format (until it is no longer needed), along with the id of the encryption key used.  
  
Decrypting Credit Card Data  
  
Encrypted credit card information is read from the database, along with the id of the encryption key record. The appropriate encryption key and vector are read from the CLASS\_ENCRYPTION\_KEY table in the database, and are decrypted using Internal Decryption. Credit card information is decrypted using this key and vector using the Standard Encryption Process.



Figure 1 - Implementing\_Class\_in\_a\_PCI\_compliant\_environment\_7.0X.pdf