EXADIGM

GSPE 16.01.015 G3

PA-DSS 3.1 Implementation Guide

Document Approval Sign-off

Position	Name	Date	Signature
President, CFO and General Manager of the POS Division	Tony Dabbene		
Director of International Business	Bill Gao		
Director of Relationship Management	Patricia Aguilera		
Director of Operations and QA.	Rosalie Krondak		

Copyright 2016 ExaDigm, Inc.

All Rights Reserved.

Printed in USA

IMPORTANT NOTICE

This document contains proprietary information of ExaDigm, Inc. The information contained herein is confidential and its use is bound by the conditions of any and all binding Non-Disclosure Agreements. Reproduction or further distribution of any information contained within this document is strictly forbidden unless prior written consent has been obtained from ExaDigm, Inc.

Revision History

Date	Version	Description	Author
07/03/2013	1.00	Initial	Rosalie Krondak
09/24/2013	1.01	Updated with PA-DSS requirements	Alex Grigoryev
10/3/2013	1.02	Additional procedures edited	Alex Grigoryev
10/15/2013	1.03	Formatting	Rosalie Krondak
12/11/2013	1.04	403 Labs requested updates	Rosalie Krondak
01/16/2014	1.05	403 Labs requested updates	Rosalie Krondak
12/02/2014	2.0	Update to 3.0 for NX2200e, NX1200, NP8110 and NP8210 GSPE	Bryan Schmidt
01/21/2015	2.1	403 Labs requested updates	Bryan Schmidt
05/15/2015	2.2	403 Labs requested updates	Bryan Schmidt
01/11/2016	3.0	Update to 3.1 for G3 GSPE	Bryan Schmidt

Table of Contents

© 2016 ExaDigm, Inc.

1.0 PA-DSS Requirements for Compliance

1.1 Implementation Guide Purpose

This document is provided as an Implementation Guide to instruct the end user and resellers/integrators on secure product implementation and to document the secure configuration specifics mentioned throughout the PCI PA-DSS requirements documentation. The document delineates vendor, reseller/integrator, and customer responsibilities for meeting all compliance requirements. It provides the details for how the customer and/or reseller/integrator should enable security settings within the customer's network. As an example, the Implementation Guide covers responsibilities and basic features of password security even though this is not controlled by the payment application, so that the customer and/or reseller/integrator clearly understand how to implement secure passwords for compliance.

It is highly recommended that the vendor, reseller/integrator, and customer level users familiarize themselves and adheres to PCI DSS standards available at https://www.pcisecuritystandards.org/security_standards/pci_dss.shtml

Special notes:

- Failure to adhere to PCI security compliance requirements may result in revenue loss and legal consequences.
- Adding, altering and/or changing IP protocols, Services, Security Protocols supplied by ExaDigm invalidates the PA-DSS approval granted on tested set of tools and libraries. Software application processing of financial transactions must use TLS v.1.2, or HTTPS libraries and tools provided by ExaDigm in order to secure the data in compliant fashion.
- All vendor, reseller/integrator, and customer level users are encouraged to register for security updates via an email sent to compliance@exadigm.com .
- If a vendor, reseller/integrator, and customer level users become aware of an existing vulnerability, they should inform ExaDigm by sending an e-mail to compliance@exadigm.com.

1.2 Implementation Guide Target Audience and Distribution

This document is intended for the end users, resellers and integrators who place ExaDigm products in service for processing credit card and debit card transactions, the ExaDigm product is deployed with ExaDigm Payment Applications or custom developed payment applications adhering to PCI PA-DSS requirements referred above.

The typical users, resellers and integrators include but are not limited to:

- ISO (Independent Sales Organizations)
- POS equipment distributors
- Payment Software Application developers
- Payment Systems Integrators
- Payment Processors
- Banks
- Merchants of all categories and payment industries accepting credit and debit cards

All categories of ExaDigm customers are encouraged to read the Implementation Guide at the event of purchasing an ExaDigm product. The document is available on the ExaDigm Inc. website www.exadigm.com.

As an alternative option the document can be delivered by e-mail, fax or postal service to customers who choose to receive a hard copy. The Customer Account Manager is responsible to deliver the Implementation Guide and new revisions of the Implementation Guide to each customer as requested.

1.3 Implementation Guide Maintenance Policy

This Implementation Guide is subject to annual review and maintenance updates, which address the changes implied by new revisions of the PCI DSS and PCI PA-DSS standards as well as any software updates related to improvement of the security features in ExaDigm products.

ExaDigm Security Officer is responsible to maintain the Implementation Guide and perform annual review with ExaDigm PCI Compliance Committee.

ExaDigm PCI Compliance Committee is required to review and approve the updated version of the Implementation Guide. The Customer Account Manager is responsible to notify customers of the updated copy as defined in the paragraph “1.2 Implementation Guide Target Audience and Distribution”.

1.4 Security Implementation Requirements

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
PA-DSS 1	**Do not retain full magnetic stripe, card verification code or value (CAV2, CID, CVC2, CVV2), or PIN block data**		
PA-DSS 1.1	Do not store sensitive authentication data after authorization (even if encrypted)		**Software Vendor:** Grape Secure Payment Engine does not store sensitive authentication data (full track data, CAV2/CVC2/CVV2/CID, PINs/PIN blocks) after authorization. The sensitive authentication data is cleared after authorization by using a c/c++ memset. **Customers & Resellers/Integrators:** N/A
PA-DSS 1.1.1	After authorization, do not store the full contents of any track from the magnetic stripe (located on the back of a card, equivalent data contained on a chip, or elsewhere). This data is alternatively called full track, track, track 1, track 2, and magnetic-stripe data.	Magnetic stripe data must not be stored in long-term storages (databases, files). At the end of transaction all temporary memory space, which kept these data must be erased.	**Software Vendor:** Grape Secure Payment Engine must not store any historical data containing full track data after finishing transaction. For host and terminal based applications this type of data is erased from memory right after the transaction. **Customers & Resellers/Integrators:** N/A
PA-DSS 1.1.2	After authorization, do not store the card verification value or code (three-digit or four-digit number printed on the front	Card verification value or code must not be stored in long-term storages (databases, files). At the end of transaction all temporary memory	**Software Vendor:** Grape Secure Payment Engine must not store card verification value or code after finishing transaction. For host based and terminal based applications this type of data is erasing from memory right after the transaction. The Payment Engine will

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
	or back of a payment card) used to verify card-not-present transactions.	space, which kept these data must be erased.	automatically erase without a trace sensitive authentication data stored by previous payment application versions because this data cannot be recognized by it. **Customers & Resellers/Integrators:** N/A
PA-DSS 1.1.3	After authorization, do not store the personal identification number (PIN) or the encrypted PIN block.	Personal identification number (PIN) or the encrypted PIN block must not be stored in long-term storages (databases, files). At the end of transaction all temporary memory space, which kept these data must be erased.	**Software Vendor:** Grape Secure Payment Engine doesn't have any access to original PIN value. Only encrypted PIN block is used. Payment application is used on PCI PTS certified system only. Payment Engine must not store encrypted PIN block after finishing transaction. For host and terminal based applications this type of data is erased from memory right after the transaction. **Customers & Resellers/Integrators:** N/A
PA-DSS 1.1.4	Delete sensitive authentication data stored by previous payment application versions	Historical data must be removed (magnetic stripe data, card validation codes, PINs, or PIN blocks stored by previous versions of the payment application).	**Software Vendor:** Payment Engine does not store any historical data containing cardholder and sensitive authentication data. ExaDigm's Grape Secure Payment Engine uses a Data Encryption Key randomly generated for each batch of transactions, and the key is changed after each settlement. Therefore no manual deletion is necessary. When new versions of the application are installed the Payment Application will automatically erase without a trace sensitive authentication data stored by previous versions which is absolutely necessary for PCI DSS compliance. **1.1.4:** All historical data is automatically deleted by software upgrade and installation procedure. To ensure all such data is deleted there is an option in the NXShell menu of the application which will reset all configurations and databases to initial state (cleaning up all data, resetting DEKs, etc). To perform this procedure manually turn off the terminal and then power on. When the logo appears listen for the

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			double beep and key in 3446. Select App Download from the System Menu, press enter at cable confirmation message. Select TMS from Main menu, then select Setup and then select Reset Config. Enter Admin password. Once process is performed reports with show No Trans or 0 transactions. Merchants must settle all transactions prior to clearing the batch to prevent loss of unsettled transactions. Historical data removal included in the upgrade process is absolutely necessary for PCI DSS compliance. See PCI DSS requirement 3.2 Configuration information is inputted into the terminal by manually entering the information in the Admin>App Setup>Merchant setup. Processor information is located under Host setup. To erase configuration information power off and then on after the logo appears listen for the double beep and then key 3446. Select App Download from the System Menu, select TMS, then select Setup and then Reset Config. This procedure will erase current parameter settings and restore the terminal to its default. ***Merchants must settle all transactions before activating this procedure to prevent data lost.*** Security assessments for the purpose to identify potential security weaknesses are outsourced to a certified lab. The lab will verify that no data is stored within the application. **Customers & Resellers/Integrators:** Before upgrading software applications, ensure that all transactions are settled to the remote system. All data present on terminal will be lost after upgrade. Refer to User Manual for a specific Payment Application for settlement process steps.
PA-DSS 1.1.5	Delete any sensitive authentication data (pre-authorization)	Historical data must be removed (magnetic stripe data, card validation codes,	**Software Vendor:** ExaDigm Grape Secure Payment Engine does not collect sensitive authentication data (pre-authorization) for troubleshooting

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
	gathered as a result of troubleshooting the payment application	PINs, or PIN blocks stored by previous versions of the payment application)	the payment application. ExaDigm recommends using Test Cards and Test Account data for troubleshooting purposes. For more details see *Technical Support Procedures.* **Customers & Resellers/Integrators:** Customers should avoid using real cardholder's information for troubleshooting purposes, Test Cards and Test Account data should be used for troubleshooting purposes. ▪ Collect these data only when needed to solve a specific problem ▪ Store these data only in specific, known locations with limited access ▪ Collect as little of these data as necessary to solve the specific problem ▪ Encrypt these data when stored ▪ Securely delete these data immediately after use
PA-DSS 2	**Protect stored cardholder data**		
PA-DSS 2.1	Delete cardholder data after customer-defined retention period	▪ Cardholder data must be deleted after it exceeds the customer-defined retention period ▪ All locations where payment application stores cardholder data should be deleted	**Software Vendor:** The ExaDigm Grape Secure Payment Engine is set to delete all cardholder information (all transactions) after exceeding customer defined retention period (default 14 days) or after the Settlement has been completed, whichever comes first. Application will warn user before deleting, giving a chance to settle the transactions to the Payment Processor. Cardholder data is stored in the SQLite DB. App 1: /opt/exadigm/apps/app1/var/db/app.sqlite App2: /opt/exadigm/apps/app2/var/db/app.sqlite No data is retained within back-ups or restore points as the terminals do not perform back-ups or restore points. The parameter to set the retention period is TVO_CHRETENTIONTIME in the terminal. The field is configurable to any number of days. When the limit is

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			reached a warning will appear on the terminal. Cardholder data is deleted by using the SQlite “DELETE” command which removes all data from the SQLite database. *G3 User Guide.* **Customers & Resellers/Integrators:** When the application issues a warning regarding exceeding customer defined retention period for storing cardholder data, user must immediately settle all financial transactions to the payment processor to avoid data and financial loss. All cardholder data must be deleted by settling the batch when it is no longer required for legal, regulatory or business purposes. The procedure to do so is: End of Day>Input User ID>Enter Password>Settlement>Print Report>Confirm Settlement.
PA-DSS 2.2	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 can see the full PAN.	Full PAN is never printed or displayed by the application.	**Software Vendor:** ExaDigm Grape Secure Payment Engine always used special functions to access masked PAN. Test scripts contain use cases to verify all display and printer output, which may contain masked PAN. **Customers & Resellers/Integrators:** There is a configuration option to switch between the first six and last four digits for PAN output if applicable. It is not possible to view the full PAN under any circumstances. The masked PAN is displayed on receipts, reports, when viewing transactions in the view transaction menu. The PAN is also masked when manually entering a transaction.
PA-DSS 2.3	Render PAN unreadable anywhere it is stored, (including data on portable digital media, backup media, and in logs)	PAN must be encrypted via strong cryptography with associated key management processes and procedures	**Software Vendor:** ExaDigm Grape Secure Payment Engine doesn’t store PANs on portable media and logs. 192-bit Triple DES CBC Mode Key is used to encrypt PAN data. ExaDigm Payment Application uses a Data Encryption Key randomly generated for one batch of transactions. After settlement completed all transactions are erased from

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			database. If a settlement is not preformed within the set retention time of 14 days (default) the database is removed. There is no way for the transactions to be stored for a longer period of time as they are deleted from the DB. PAN storage and encryption is not configurable or exportable. **Customers & Resellers/Integrators:** Before upgrading software applications, ensure that all transactions are settled to the remote system. All data present on terminal will be lost after upgrade. Refer to User Manual for a specific Payment Application for settlement process steps.
PA-DSS 2.4	Payment application must protect any keys used to secure cardholder data against disclosure and misuse.	All Data Encryption Keys must be encrypted by strong encryption algorithms. Key Encryption Key must be stored securely and separately from Data Encryption Keys	**Software Vendor:** N/A handled by the application. Key custodians are not necessary because there is no user key management it is performed by the application. The device uses key management provided by firmware for storing KEK. For more details on secure key storage and key management see *Secure Data Handling.* Data Encryption Keys are encrypted via Triple DES by Key Encryption Key stored encrypted in firmware protected area. DEKs and KEKs are stored in separate area. KEK is generated when the Payment Application started first time after installation or upgrade. Payment application doesn’t have access to protected key storage provided by this firmware. ExaDigm Payment Application uses a Data Encryption Key randomly generated for one batch of transactions. After settlement completed all transactions are erased from database. The application stores the key in the fewest number of locations possible. **Customers & Resellers/Integrators:** Before upgrading software applications, ensure that all transactions are settled to the remote system. All data present on terminal will be lost after upgrade due to security key change. Historical

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			data will not be affected.
PA-DSS 2.5	Payment application must implement key management processes and procedures for cryptographic keys used for encryption of cardholder data	Implement key management processes and procedures for cryptographic keys used for encryption of cardholder data.	**Software Vendor:** ExaDigm Grape Secure Payment Engine is running on PCI PTS compliant device and uses key management provided by firmware for storing KEK. For more details on secure key storage and key management see *Secure Data Handling.* KEK is generated when the Grape Secure Payment Engine starts the first time after installation or upgrade. Payment application doesn't have access to protected key storage provided by this firmware. ExaDigm Payment Application uses a Data Encryption Key randomly generated for one batch of transactions. After settlement completed all transactions are erased from database and DEK and KEK are regenerated. **Customers & Resellers/Integrators:** Before upgrading software applications, ensure that all transactions are settled to the remote system. All data present on terminal will be lost after upgrade due to security key change. Historical data will not be affected.
PA-DSS 2.6	Delete cryptographic key material or cryptograms stored by previous payment application versions	▪ Cryptographic material used for securing sensitive data must be removed ▪ Re-encrypting of historic data with new keys, if applicable	**Software Vendor:** For the end-user's PCI DSS compliance, it is absolutely necessary that cryptographic key material and/or cryptograms must be rendered irretrievable. The application facilitates this by the following. Cryptographic key material and/or cryptograms are irretrievable. The ExaDigm Payment application automatically changes the secure cryptographic keys after updating the application and after each settlement. Data stored in the terminal after settlement does not contain sensitive authentication or cardholder data and is not encrypted. After the application update the KEKs keys are also recreated rendering all previous cryptographic materials irretrievable. When the terminal is downloaded prior to installation and delivery with TMS all cryptographic materials are erased.

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			During this process a memory sweep of the system is performed, clearing it of all DEKs, sensitive data and recreating KEKs. All data is removed from the system during a software update and installation procedure. Configuration information can be entered manually through the Admin>App Setup>Merchant. Processor information is entered using Host Setup menu. To erase configuration information manually power off and then on after the logo appears listen for the double beep and then key 3446. Select App Download from the System Menu, select TMS, then select Setup and then Reset Config. This procedure will erase current parameter settings and restore the terminal to its default. ***Merchants must settle all transactions before activating this procedure to prevent data loss.*** Security assessments for the purpose to identify potential security weaknesses are outsourced to a certified lab. The lab will verify that no data is stored within the application. **Customers & Resellers/Integrators:** Before upgrading software applications, ensure that all transactions are settled to the remote system. All data present on terminal will be lost after upgrade due to security key change. Historical data will not be affected.
PA-DSS 3.1.1-3.1.11	Use unique user IDs and secure authentication for administrative access and access to cardholder data	▪ Do not use default administrative accounts for payment application logins. ▪ Assign secure authentication to default accounts (even if not used), and disable or do not use the accounts. ▪ Use secure	**Software Vendor:** ExaDigm hardware devices use a stripped down version of Embedded Linux that has no user interactive capabilities. Grape Secure Payment Engine is always running under a regular unprivileged system user account and it doesn't have any access to system critical areas of the underlying Operation System. A running ExaDigm Grape Secure Payment Engine has full access to its home directory only. The only accounts within the application

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
		authentication for the payment application and system whenever possible. ▪ How to create secure authentication to access the payment application, per PCI DSS Requirements	are for cashiers and for simple administration of the application. This simple administration account can only set up other cashier users and perform functions like manually set up an IP address. None of these accounts has the ability to access sensitive data. Upon entering administrative menu of a newly installed application the terminal will require the initial login User ID, User Name and User Password to be changed. The default User ID is manager1 and password is q123456. A unique user ID must be used. No duplicate ID will be allowed. The user ID can be alpha/numeric up to 20 characters. The password must contain a seven digit alpha-numeric password to ensure the strength against non-authorized usage. The password cannot be the same as the last four used. To setup users: Security manager provides the user interface to manage users. To access the user management area, the highest level (manager) credential is required to perform the following operations: ▪ Add User ▪ Edit User ○ Unlock User ○ Change Name ○ Change Pwd ○ Change Role ○ Enable/Disable ▪ Delete user ▪ Print users Managers can add a new user, delete or edit current user or print users list in user management menu. If the manager changes another users password, the first time the user logs in the terminal will prompt them to change the password again. **Unique User ID:** User ID must be unique when the new user is created. ▪ No blank user ID is allowed. Otherwise the payment application displays the warning "Input must be

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			minimum 1 character" and goes back to User ID prompt. ▪ After User ID is entered the application checks the User DB if there is a user with the identical User ID. ▪ If a user with an identical user ID is found, the application displays the warning "User already exists" and goes back to User ID prompt. ▪ If user ID is unique (not blank and not found in User DB) the application goes on with the other user data prompts such as user name, password and user role. ▪ After all necessary user data is entered, the application appends the new user to User DB. **Roles:** Users are assigned to a specific role when they are created. Currently 3 predefined levels of roles are available: ▪ Clerk ▪ Supervisor ▪ Manager Users' role can be changed from user management area in the edit user section. This change can only be done by the Manager. All Users can change their own password by going to Application Manager>Admin>User Manager>Change Pswd **User Passwords** 1 All user passwords must contain numeric <u>and</u> alpha characters. It must be a minimum of 7 and a maximum of 20 characters. If the requirement is not met the terminal will display an error message and the user will be forced to try again. 2 User passwords expire every 90 days. Upon expiration the terminal will display a message to the User to change their password. If password has expired the User will be prompted to enter a new password. 3 The previous four passwords cannot

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			be used. If a previously used password is entered the terminal will display an error message requesting the correct password. 4 Passwords are not stored in the system alone. Data with a combination of password and User ID is encrypted with SHA-256 algorithm and kept in the system. 5 A user can change their password by following the instructions below: 1. From App Manager 2. Go to Admin 3. Go to User Manager 4. Select Change Pswd 6 The security manager forces users to change any default password after the first successful login. 7 The security manager forces users to change any password changed in the user management menu after the first successful login. **User Rules** 1 Each user has a unique ID. No duplicate user IDs are allowed, the terminal will display message that the ID is already used. 2 After 3 failed login attempts, the user's account is locked. ▪ Lock duration is 30 minutes. ▪ Managers can unlock the user from the user management menu in edit user section or the user needs to wait 30 minutes before trying to log in again. 3 If terminal is in idle mode for more than 3 minutes, the terminal screen locks. In order to unlock it, the last logged in user's authentication is required. See more details on user account management in *Security Manager* section in *G3 User Guide.* **Customers & Resellers/Integrators:** N/A

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
PA-DSS 3.2	Use unique user IDs and secure authentication for access to PCs, servers, and databases with payment applications	Use unique user names and secure authentication to access any PCs, servers, and databases with payment applications and/or cardholder data, per PCI DSS Requirements 8.5.8 through 8.5.15.	**Software Vendor:** ExaDigm hardware devices use a stripped down version of Embedded Linux that has no user interactive capabilities. The only accounts within the application are for cashiers and for simple administration of the application. This simple administration account can only set up other cashier users and perform functions like manually set up an IP address. None of these accounts has the ability to access sensitive data. **Customers & Resellers/Integrators:** ExaDigm strongly advices that customers and resellers/integrators control access, via unique user ID and PCI DSS-compliant secure authentication, to any PCs, servers, and databases with payment applications and cardholder data.
PA-DSS 3.3.1-3.3.2	Render payment application passwords unreadable during transmission and storage, using strong cryptography based on approved standards.	Use strong cryptography to encrypt all application passwords.	**Software Vendor:** ExaDigm Grape Secure Payment Engine is uses SHA-256 algorithm to encrypt passwords before storing to the user database. Passwords are not transmitted and remain in a separate user database. Passwords are concatenated with a unique User ID and then encrypted. **Customers & Resellers/Integrators:** N/A
PA-DSS 4.1	At the completion of the installation process, the "out of the box" default installation of the payment application must log all user access (especially users with administrative privileges), and be able to link all activities to individual users.	User access logging must be always enabled.	**Software Vendor:** ExaDigm Grape Secure Payment Engine is always logging any access events (these cannot be configured by users). **Customers & Resellers/Integrators:** N/A
PA-DSS 4.2	Implement automated audit trails	▪ Set PCI DSS-compliant log settings, per PCI DSS Requirement 10. ▪ Logs must be enabled, and	**Software Vendor:** ExaDigm Grape Secure Payment Engine is always logging the following events (these cannot be configured by users): ▪ Individual access to cardholder data ▪ Actions taken by any individual with

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
		disabling the logs will result in non-compliance with PCI DSS.	administrative privileges ▪ Access to application audit trails managed by or within the application ▪ Invalid logical access attempts ▪ Use of, and changes to the payment application's identification and authentication mechanisms, and all changes, additions, deletions to application accounts with root or administrative privileges. ▪ Initialization of application audit logs ▪ Creation and deletion of system-level objects within or by the application **Customers & Resellers/Integrators:** N/A
PA-DSS 4.3	Record audit trails entries for each event	▪ Set PCI DSS-compliant log settings, per PCI DSS Requirement 10. ▪ Logs must include required fields.	**Software Vendor:** ExaDigm Grape Secure Payment Engine is always logging following fields (these cannot be configured by users): ▪ User name ▪ Event type ▪ Date and time stamp ▪ Success and failure indication (Event result code) ▪ Origination of event (application module identity) ▪ Names of affected data, components and resources. **Customers & Resellers/Integrators:** N/A
PA-DSS 4.4	Payment application must facilitate centralized logging.	A unified centralized logging must be used.	**Software Vendor:** ExaDigm Grape Secure Payment Engine is using unified logging module to log events. The logging files format is delimited text and it is the same for all system components. Logs can be extracted to an USB stick. File is comma delimited text log. A folder must be on the USB stick named *log* in order for the file to download the log to the stick. **Customers & Resellers/Integrators:** The procedure to do so is: Admin>App Setup>Input User ID>Input password>Security Setup>PCI Log

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			Setup>Retrieve PCI Log>prompt Insert USB stick, insert the cable adapter into the RS-232 port on the bottom of the G3 (under the EMV reader)>then insert the USB memory stick into the adapter>Once message “PCI Log Transferred Successfully” displays press ENTER. **PCI Log Legend:** Log Formatting MM/DD/YY, HH:MM:SS, ”User, Log Module, Event, Event Outcome, Details of Event *Example* 03/17/14,12:26:28,"m,USR,ADD_USER ,SUCCESS,User <T>, name[T], Manager" “03/17/14,12:26:28” - Date and time Event took place “m” - User accessing the section (user ID will only show if event is performed in a password protected area) “USR” – Log Module “ADD_USER” – Event: Add new user “SUCCESS” – Event Outcome Details of Event “User <T>” – User ID created “Name [T]” – Username created “Manager” – Role of user *Event Definitions* TE_VARO_COMMDEVOPT Communication method 1- Wired, 2- Wi-Fi, 3- CDMA, 5- Dial TTO_CREDIT_SALE Credit Sale TTO_CREDIT_RETURN Credit Return TTO_CREDIT_VOID Credit Void TTO_TCS_SETTLEMENT Terminal Settled ***Log Module Definitions:*** TRN – Transaction USR – User DSP – Display PRN – Print

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			RPR – Report DB – Database COM – Communication MSG – Messages format build/parse VAR – Variables, store/receive values CUST – All other objects
PA-DSS 5	**Develop secure payment applications**		
PA-DSS 5.1	The software vendor develops payment applications in accordance with PCI DSS and PA-DSS and based on industry best practices, and incorporates information security throughout the software development life cycle	Provide a detailed document describing how industry best practices are used in software development process Provide training for engineering team on industry best practices	**Software Vendor:** ExaDigm's development process is based on industry Best Practices. Secure Coding Practices – Appendix F. Software modification control – Appendix G. **Customers & Resellers/Integrators:** N/A
PA-DSS 5.1.1	Live PANs are not used for testing or development.	Provide special testing PANs for development and testing.	**Software Vendor:** ExaDigm always uses test data and accounts for development and testing. When delivering to customers the Grape Secure Payment Engine application doesn't contain any test data or accounts. The testing data and account are requested from the Payment Gateways when needed. **Customers & Resellers/Integrators:** N/A
PA-DSS 5.1.2	Removal of test data and accounts before release to customer.	Do not include any testing accounts and data into the software delivery package.	**Software Vendor:** Grape Secure Payment Engine application package is not including any testing accounts and data. Database included into the packages is empty. Default configuration files, which are included in package are fixed and reviewed before releases. These files are also a part of our version revision control system. **Customers & Resellers/Integrators:** N/A
PA-DSS 5.1.3	Removal of custom payment application accounts, user IDs, and passwords	Do not include any custom payment application accounts, user IDs, and passwords into releases.	**Software Vendor:** ExaDigm Grape Secure Payment Engine package is delivered with default account, user IDs and passwords. **Customers & Resellers/Integrators:**

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
	before payment applications are released to customers		Customers are responsible for creating all custom accounts, user IDs and change all default passwords at first application run. The payment application will ask for password change, when started first time.
PA-DSS 5.1.4	Review of payment application code prior to release to customers after any significant change, to identify any potential coding vulnerability	Make code review a part of Software development process.	**Software Vendor:** ExaDigm's development process includes code review stage as a standard part of the process. For Details see *Software Code Review Policies* **Customers & Resellers/Integrators:** N/A
PA-DSS 5.1.5	Secure source-control practices are implemented to verify integrity of source code during the development process	Include secure source control practices during the development process to maintain integrity.	**Software Vendor:** ExaDigm's development process includes secure source-control practices. For details see *Software Development Best Practices and Training Policies* **Customers & Resellers/Integrators:** N/A
PA-DSS 5.1.6-5.1.6.1	Payment applications are developed according to industry best practices for secure coding techniques	▪ Developing with the least privilege for the application environment. ▪ Developing with fail-safe defaults (all execution is by default denied unless specified within initial design). ▪ Developing for all access point considerations, including input variances such as multi-channel input to the application.	**Software Vendor:** ExaDigm's application is developed according to industry best practices for secure coding techniques. For details see Appendix F. **Customers & Resellers/Integrators:** N/A
PA-DSS 5.1.7	Provide training in secure development practices for application developers, as applicable for the developer's job function and	▪ Secure application design. ▪ Secure coding techniques to avoid common coding vulnerabilities. ▪ Managing	**Software Vendor:** ExaDigm's application is developed according to industry best practices for secure coding techniques. For details see Appendix F. **Customers & Resellers/Integrators:** N/A

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
	technology used.	sensitive data in memory. ▪ Code reviews. ▪ Security testing. ▪ Risk-assessment techniques.	
PA-DSS 5.2	Develop all payment applications to prevent common coding vulnerabilities in software-development processes.	Include guidelines to prevent common coding vulnerabilities as a part of the software development process.	**Software Vendor:** ExaDigm's development process includes guidelines to prevent common coding vulnerabilities. For details see Appendix F. **Customers & Resellers/Integrators:** N/A
PA-DSS 5.3	Software vendor must follow change control procedures for all product software configuration changes	▪ Provide change control procedures for all software changes. ▪ Include all software modification stages in the change control system.	**Software Vendor:** ExaDigm uses JIRA to manage software development process and save all software modification stages. For source code version control concurrent Versioning System (CVS) is used. See Appendix G for more details. **Customers & Resellers/Integrators:** N/A
PA-DSS 5.4.4	The payment application vendor must document and follow a software-versioning methodology as part of their system development lifecycle.	Versioning methodology must be in accordance with the PA-DSS Program Guide.	**Software Vendor:** ExaDigm follows a specific, required versioning methodology in accordance with the PA-DSS Program Guide. Each change to the application is reflected in the versioning. For versioning information see *Appendix G – Software Version Management System* and *Naming Convention for Application, Naming Convention for Kernel and Naming Convention for Core.* **Customers & Resellers/Integrators:** N/A
PA-DSS 6.1	Securely implement wireless technology	If wireless is used within payment environment, install a firewall per PCI DSS Requirement 1.3.8.	**Software Vendor:** ExaDigm requires that perimeter firewalls be installed between any wireless networks and systems that store cardholder data, and that these firewalls must deny, limit or otherwise control any traffic from the wireless environment into the cardholder data environment, if such traffic is necessary for business purposes. It is prohibited to use

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			ExaDigm terminals in DMZ networks or on the same network as any other system with unrestricted access to the Internet. **Customers & Resellers/Integrators:** All ExaDigm terminals utilizing Ethernet or Wi-Fi LAN must be deployed in a secure network environment only. Maintaining the secure network includes the proper setup and use of a Network Firewall. It is recommended that a security audit be carried out by a company specializing in network security. A complete list of security assessors can be found at https://www.pcisecuritystandards.org/pdfs/pci_qsa_list.pdf Merchants using ExaDigm terminal connected to IP networks must adhere to the latest PCI security standards for financial transactions (www.pcisecuritystandards.org). This website has a self-assessment questionnaire which should be filled out for each installation - https://www.pcisecuritystandards.org/saq/index.shtml ▪ Instructions if wireless is used: Change encryption keys at installation and whenever anyone with knowledge of the keys leaves the company or changes positions ▪ Change default SNMP community strings • Change default passwords and passphrases ▪ Update firmware to support strong encryption for authentication and transmission ▪ Identify and change any other security-related vendor defaults ▪ A firewall must be installed between any wireless networks and systems that store cardholder data ▪ Configure access for the terminal to port 443 for downloading from TMS ▪ Firewalls must be configured to

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			deny or control (if such traffic is necessary for business purposes) any traffic from the wireless environment into the cardholder data environment
PA-DSS 6.2-6.3	Secure transmissions of cardholder data over wireless networks	If payment application is implemented into a wireless environment, use PCI DSS-compliant wireless settings, per PCI DSS Requirement 4.1.1	**Software Vendor:** Implementations using Wi-Fi LAN must include WLAN security set at minimum "WPA Personal" using TKIP/PSK key management. It is recommended that network administrators and software developers utilize WPA2 with PSK or AES authentication scheme included in the "WPA_Supplicant" utility provided by ExaDigm. The WPA Supplicant is a software implementation of an IEEE 802.11i supplicant that is used to connect a terminal's WiFi device to a WiFi network. Once the program is running with the correct parameters (SSID, password/key, etc) it provides the WiFI interface and associated IP address information to the terminal. Since 802.11b implementation does not support WPA/WPA2 modem upgrade to 802.11g is required in order to implement maximum security configuration. Additional security measures include: ▪ Change default administrative account settings and passwords; use strong passwords ▪ Enable TLS security for WLAN router configuration console ▪ Change default service set identifier (SSID) ▪ Disabling SSID broadcasting ▪ Enabling MAC-address filtering ▪ Change SNMP community strings ▪ For new WLAN implementation WEP security is prohibited after March 31, 2009 ▪ For existing WLAN implementations WEP security is prohibited after June 30, 2010 **Customers & Resellers/Integrators:** User must apply recommended security changes for securing the Wireless network for payment applications per PCI DSS requirement 4.1.1. Instructions if wireless is used:

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			▪ Change encryption keys at installation and whenever anyone with knowledge of the keys leaves the company or changes positions ▪ Change default SNMP community strings ▪ Change default passwords and passphrases ▪ Update firmware to support strong encryption for authentication and transmission ▪ Identify and change any other security-related vendor defaults ▪ A firewall must be installed between any wireless networks and systems that store cardholder data ▪ Firewalls must be configured to deny or control (if such traffic is necessary for business purposes) any traffic from the wireless environment into the cardholder data environment
PA-DSS 7	**Test payment applications to address vulnerabilities**		
PA-DSS 7.1	Software vendors must establish a process to identify and assign a risk ranking to newly discovered security vulnerabilities and to test their payment applications for vulnerabilities.	Assign a role to perform identifying and assigning a risk ranking to newly discovered security vulnerabilities. Test application for vulnerabilities prior to release. Provide newly discovered security vulnerabilities as a part of Software Development process.	**Software Vendor:** ExaDigm has an assigned person, who is responsible to verify all news and reports on newly discovered security vulnerabilities. This person is responsible to report these vulnerabilities to project manager and then project manager issues a task for SQA to test it and in case this vulnerability is applied to ExaDigm Payment Application issues a task to fix the issue. Applications are tested for vulnerabilities prior to release. See appendix D and E for more details on secure protocols reviews. **Customers & Resellers/Integrators:** Must update the software as soon as possible after ExaDigm provides the update.
PA-DSS 7.2	Software vendors must establish a process for timely development and deployment of security patches	Provide development and deployment of security patches and upgrades as a part of software development	**Software Vendor:** ExaDigm's Project Manager is responsible to issue a security vulnerability fixing task. Depending on a risk ranking assigned to this vulnerability the task priority must

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
	and upgrades, which includes delivery of updates and patches in a secure manner with a known chain-of-trust, and maintenance of the integrity of patch and update code during delivery and deployment	process.	be set. **Customers & Resellers/Integrators:** They have to update software as soon as possible after ExaDigm provides the update.
PA-DSS 8	**Facilitate secure network implementation**		
PA-DSS 8.1	The payment application must be able to be implemented into a secure network environment. Application must not interfere with use of devices, applications, or configurations required for PCI DSS compliance.	Do not interfere with PCI DSS components.	**Software Vendor:** ExaDigm must provide a testing environment for the Grape Secure Payment Engine, which complies with PCI DSS Requirements 1,3,4,5, and 6. **Customers & Resellers/Integrators:** Establish and maintain secure environment to run the payment application per the PA-DSS Implementation Guide and PCI DSS Requirements 1,3,4,5, and 6.
PA-DSS 8.2	The payment application must only use or require use of necessary and secure services, protocols, daemons, components, and dependent software and hardware, including those provided by third parties, for any functionality of the payment application	Provide support for a set of networking protocols and services minimally needed for Payment Application Functionality.	**Software Vendor:** ExaDigm Grape Secure Payment Engine includes support for the following protocols: TCP, UDP, TLS, and ICMP. Services supported are cURL, PPPD and DNS. Application Libraries include Transaction Engine, ACL, libsdn.so., libservice.so. and libsutil.so.. System-level Libraries include ExaCore, LIBMATHS, libdbus.so. (D-Bus), libexpat.so. (XML), libsqlite3.so. (SQLite), libssl.so., libcrypto and libz.so. Wi-Fi Utilities include iwconfig, iwlist, pppd, wpa_cli, wpa_supplicant, libdirect.so., libfusion.so., libjpeg.so., libpng.so. and libts.so. Hardware-level Libraries include XPG and LIBDEV. Necessary Hardware includes touch screen, LCD, Keyboard, MSR, Printer, Smart Card, and NFC. Optional Hardware includes Wireless Modem, and Wi-Fi. Additional services include Database, Status bar Service, Comm. Service, Download Service, Secure Service and Crypto Service. Communication includes TLS port 443

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			for download and the required port as assigned by the processor. *The terminal cannot receive any incoming communication; it can only communicate with the processor through its Requirement Specification for TLS communication. For more details see Appendix A. **Customers & Resellers/Integrators:** N/A
PA-DSS 9	**Cardholder data must never be stored on a server connected to the Internet**		
PA-DSS 9.1	Store cardholder data only on servers not connected to the Internet	Do not store cardholder data on Internet-accessible systems (for example, web server and database server must not be on same server)	**Software Vendor:** It is prohibited to use ExaDigm terminals in DMZ networks or on the same network as any other system with unrestricted access to the Internet. Grape Secure Payment Engine does not require any external storage or on any other Internet system outside the boundaries of the secure storage embedded in the ExaDigm terminal. **Customers & Resellers/Integrators:** Establish and maintain payment applications so that cardholder data is not stored on Internet-accessible systems, per the PA-DSS Implementation Guide and PCI DSS Requirement 1.3.7
PA-DSS 10	**Facilitate secure remote access to payment application**		
PA-DSS 10.1	Allow remote access technologies with use of a two-factor authentication mechanisms such as RADIUS or TACACS with tokens, or VPN with individual certificates	When using remote access allow two-factor authentication mechanisms such as RADIUS or TACACS with tokens, or VPN with individual certificates	**Software Vendor:** ExaDigm terminal is a hardware terminal that has no interactive user shell and cannot be accessed remotely. There is no ability to access the device as an Operating System user neither locally, nor remotely. **Customers & Resellers/Integrators:** N/A
PA-DSS 10.2-10.2.2	Implement two-factor authentication for remote access to payment application	Use two-factor authentication (user ID and password and an additional authentication item such as a token) if the payment application may be accessed remotely.	**Software Vendor:** ExaDigm terminal is a hardware terminal that has no interactive user shell and cannot be accessed remotely. There is no ability to access the device as an Operating System user neither locally, nor remotely. **Customers & Resellers/Integrators:** N/A
PA-DSS 10.3	Securely implement	Implement and use	**Software Vendor:** ExaDigm terminal is

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
	remote access software	remote access software security features if remote access software is used to remotely access the payment application or payment environment.	a hardware terminal that has no interactive user shell and cannot be accessed remotely. There is no ability to access the device as an Operating System user neither locally, nor remotely. **Customers & Resellers/Integrators:** N/A
PA-DSS 11	**Encrypt sensitive traffic over public networks**		
PA-DSS 11.1	Secure transmissions of cardholder data over public networks.	Implement and use TLS for secure cardholder data transmission over public networks, in accordance with PCI DSS Requirement 4.1	**Software Vendor:** ExaDigm Grape Secure Payment Engine is compiled with TLS 1.2 or HTTPS protocol to protect the security of the cardholder data over public networks, cannot be disabled and is not configurable. **Customers & Resellers/Integrators:** Establish and maintain secure transmissions of cardholder data, per the PA-DSS Implementation Guide and PCI DSS Requirement 4.1.
PA-DSS 11.2	Encrypt cardholder data sent over end-user messaging technologies	Implement and use an encryption solution for if PANs can be sent with end-user messaging technologies.	**Software Vendor:** ExaDigm Grape Secure Payment Engine does not allow or facilitate the transmission of PANs or any other security sensitive data via e-mail or any other end-user messaging technology. **Customers & Resellers/Integrators:** For non-ExaDigm payment solutions make sure that those solutions encrypt all PANs sent with end-user messaging technologies, per the PA-DSS Implementation Guide and PCI DSS Requirement 4.2.
PA-DSS 12	**Encrypt all non-console administrative access**		
PA-DSS 12.1-12.2	Encrypt non-console administrative access	Implement and use SSH, VPN, or TLS for encryption of any non-console administrative access to payment application or servers in cardholder data environment.	**Software Vendor:** ExaDigm Grape Secure Payment Engine does not allow for non-console administrative access to the terminal resources and data. **Customers & Resellers/Integrators:** For non-ExaDigm payment solutions make sure that they encrypt all non-console administrative access, per the PA-DSS Implementation Guide and PCI DSS Requirement 2.3.
PA-DSS 13	**Maintain instructional documentation and training programs for customers, resellers, and integrators**		
PA-DSS 13.1	Develop, maintain, and disseminate a PA-DSS	Provide *PA-DSS Implementation Guide document*.	**Software Vendor:** ExaDigm provides *PA-DSS Implementation Guide* document.

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
	Implementation Guide(s) for customers, resellers, and integrators.		The customer release package contains all files/documents required by customer (updated if needed), such as ▪ Documentation (User Guides, QRG, QDG, TG, Release Notes, etc.) ▪ Release application loaded to TMS and FTP ▪ USB download files, if applicable ▪ Documentation of the impact of changes/additions made ▪ Secure Data Handling (Integrators only) ▪ Naming Conventions (Integrators only) ▪ Implementation Guide All necessary files should be located in the Engineering Release folder. The above mentioned files and/or documents are located in the FTP and the ExaDigm website. Documents may also be provided via e-mail upon request. Announcements (at least annually) regarding application changes or PA-DSS requirement updates will be sent via the distribution service to all customers, resellers and integrators as changes are required or made. ExaDigm, in an ongoing manner, updates both the application and associated documentation and then informs customers, resellers and integrators of these changes. This must be completed by the following procedure: ▪ When an application is changed/updated the PA-DSS Implementation Guide is updated with the changes and then disseminated to all customers, resellers, and integrators via the e-mail distribution service as well as on the ExaDigm website. The application will be available for download from TMS and is uploaded to the FTP. ▪ If a customer, reseller or integrator would like to

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			receive the PA-DSS Implementation Guide at any time they are directed to the ExaDigm website, FTP or e-mailed depending on preference. **Customers & Resellers/Integrators:** Have to use ExaDigm *PA-DSS Implementation Guide* and referred documents to understand how ExaDigm address all PA-DSS requirements and keep their internal processes in accordance with it.
PA-DSS 13.1.1	Provides relevant information specific to the application for customers, resellers, and integrators to use.	- Include application name and versions to which it applies - Provide details of all application dependencies required for the application to be PA-DSS compliant	**Software Vendor:** Grape Secure Payment Engine G3 Version: 16.01.015 **Customer & Resellers/Integrators:** The Grape Secure Payment Engine version is printed on the Print Config report.
PA-DSS 13.1.2	Addresses all requirements in this document wherever the *PA-DSS Implementation Guide* is referenced	*Address all issues from pa-dss_v3.pdf*	**Software Vendor:** ExaDigm provides *PA-DSS Implementation Guide* document, which includes all requirements from *pa-dss_v3-1.pdf*. **Customers & Resellers/Integrators:** Have to use ExaDigm *PA-DSS Implementation Guide* and referred documents to understand how ExaDigm address all PA-DSS requirements and keep their internal processes in accordance with it.
PA-DSS 13.1.3	Includes a review at least annually and updates to keep the documentation current with all major and minor software changes as well as with changes to the requirements in this Document.	- Provide updates of *PA-DSS Implementation Guide* after all Grape Secure Payment Engine changes. - Provide annual review and update *PA-DSS Implementation Guide* in accordance with changes in application and PA-DSS.	**Software Vendor:** When PCI Security Standards Council issues an update ExaDigm reviews the PA-DSS Implementation Guide to keep it compliant with the latest PA-DSS requirements. This document is also updated after any Grape Secure Payment Engine changes are released to ensure PA-DSS requirements are addressed by updated application. In addition communication is sent to all resellers and customers via the email distribution service. Reviews are conducted annually. **Customers & Resellers/Integrators:** Make sure the internal processes will be

PA-DSS Requirement	Requirement Topic	Implementation Steps	Implementation Responsibility
			updated according to new versions of PA-DSS Implementation Guide.
PA-DSS 14.3	Develop and implement training and communication programs to ensure payment application resellers and integrators know how to implement the payment application and related systems and networks according to the *PA-DSS Implementation Guide* and in a PCI DSS-compliant manner.	Provide training materials and communication programs.	**Software Vendor:** ExaDigm provides following training documents: ▪ *PA-DSS Implementation Guide* ▪ *G3 User Guide* ▪ *Secure Data Handling* ▪ *Secure Manager* ▪ *Naming Convention documents* ▪ *G3 Hardware specifications* ▪ *Key Injection Guide* **Customers & Resellers/Integrators:** Make sure the internal processes will be updated according to provided training materials.
PA-DSS 14.3.1	Update the training materials on an annual basis and whenever new payment application versions are released.	Provide training materials update procedure as a part of ExaDigm Software Development process.	**Software Vendor:** ExaDigm includes training materials update as a regular stage software development process. All training materials must be updated after any software update if necessary and on an annual basis. When released the new documents are available on ExaDigm web site www.exadigm.com. **Customers & Resellers/Integrators:** Make sure the internal processes will be updated according to new versions of training materials.

2.0 Other Security Implementation Guidelines

2.1 Password Strength and Password Management

There are three access levels to the Payment Application features – Clerk (cashier, server, waiter, etc.) Supervisor and Manager (Administrator).

For all payment applicative features, which require password authentication it is required that users follow these guidelines:

- Users must change the default username, user ID and passwords upon first login.
- ExaDigm requires passwords with a minimum length of 7 characters – there should be a mixture of alpha and numeric characters (at minimum 1 alpha for 6 numeric characters or 1 numeric for 6 alpha characters).
- It is recommended to avoid sequential and repeated numbers and characters (123456X and X111111 are examples of very weak passwords).
- It is required that these passwords are updated and changed frequently (at least once in every 90 days) or as recommended by a security specialist outlined in the PCI DSS approved vendor list.
- If the end user enters the wrong password more than 3 times, then the terminal application locks user access to all restricted functions, which require password entry. The access restriction (terminal) locking is set to 30 minutes. The lock can be reset only by the user with higher access level privileges or after locking timeout expiration upon successful password entry.

Appendix A – Security Conformance Review/Audit for Protocols

All Items must be signed off by the person who reviewed/updated the task. If anything was updated or changed as a result, the change must be documented in the comment section for that task,

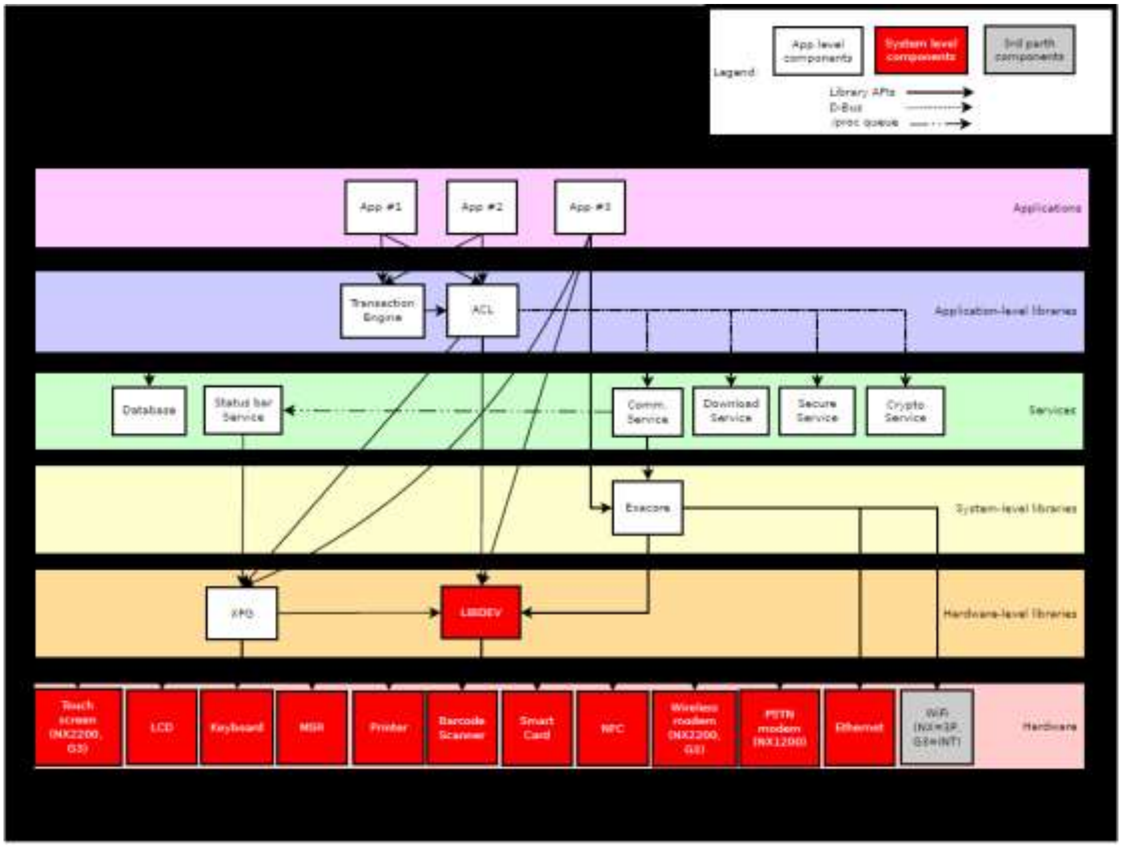
Reviewer must go to the listed site and verify current version. If the version has changed, or interim versions have been released, the reviewer must look at these changes, and if there are security weaknesses, a new version must be released specific to ExaDigm. ExaDigm will offer to its developers an interim Core release to address this vulnerability.

1. IP Protocols (UDP, TCP and ICMP) in Linux: http://www.kernel.org/. Current version: 2.6.32.68 ExaDigm is subscribed to the Linux mail list (http://vger.kernel.org/majordomo-info.html) via the email address compliance@exadigm.com. On an on-going basis, all relevant announcements and issues will be entered into entered into ExaDigm's JIRA tracking software for assignment and threat analysis. Annual audits will be conducted on list by viewing its archive. Any updates or issues will be logged into ExaDigm's JIRA tracking software for assignment and threat analysis. If there are any items found to be vulnerable, a plan will be put in place to address the issue in a timely manner per ExaDigm's PA-DSS policy. ExaDigm will at time of review run a vulnerability threat assessment on its OS. As this version of Linux OS is no longer actively supported, a full featured second party tool will be used for this assessment. Currently, ExaDigm uses Nessus (http://www.nessus.org/documentation/) for its vulnerability assessments. At the completion of testing, all issues found will be logged into ExaDigm's JIRA tracking software. If there are any items found to be vulnerable, a plan will be put in place to address the issue in a timely manner per ExaDigm's PA-DSS policy. Appendix D has details on the assessment. In addition, ExaDigm is subscribed to and monitors the National Vulnerability Database. Vulnerabilities are addressed in the Guidelines.

2. Security Protocols included in OpenSSL: http://www.openssl.org – Current version: OpenSSL FISP V2.0.10 ExaDigm is subscribed to the OpenSSL "announce" mail list via the email address compliance@exadigm.com (http://www.openssl.org/support/). On an on-going basis, all relevant announcements and issues will be entered into entered into ExaDigm's JIRA tracking software for assignment and threat analysis. Annual audits will be conducted on OpenSSL by viewing the mail archive at http://www.mail-archive.com/openssl-announce@openssl.org/maillist.html. Any updates or issues will be logged into ExaDigm;'s JIRA tracking software for assignment and threat analysis. If there are any items found to be vulnerable, a plan will be put in place to address the issue in a timely manner per ExaDigm's PA-DSS policy. Appendix D has details on the assessment. In addition, ExaDigm is subscribed to and monitors the National Vulnerability Database. Vulnerabilities are addressed in the Guidelines.

3. IP Services included in the platform:
   - PPPD http://ppp.samba.org/ppp/README.html - Current version 2.4.7. On an on-going basis, all relevant announcements and issues will be entered into entered into ExaDigm's JIRA tracking software for assignment and threat analysis. Annual audits will be conducted on PPPD. Any updates or issues will be logged into ExaDigm;'s JIRA tracking software for assignment and threat analysis. If there are any items found to be vulnerable, a plan will be put in place to address the issue in a timely manner per ExaDigm's PA-DSS policy. In addition, ExaDigm is subscribed to and monitors the National Vulnerability Database. Vulnerabilities are addressed in the Guidelines.
   - cURL http://curl.haxx.se – Current version 7.46.0 ExaDigm is subscribed to the email list at the listed Website. On an on-going basis, all relevant announcements and issues will be entered into entered into ExaDigm's JIRA tracking software for assignment and threat analysis. Annual audits will be conducted on DHCPCD by viewing its mail archive. Any updates or issues will be logged into ExaDigm's JIRA tracking software for assignment and threat analysis. If there are any items found to be vulnerable, a plan will be put in place to address the issue in a timely manner per ExaDigm's PA-DSS policy

- DNS – GNU C LIbrary: http://www.gnu.org/s/libc/ Current version: 2.5, compiled using GNU compiler version 4.1.2. ExaDigm is subscribed to the glibc mail list (http://sources.redhat.com/ml/libc-announce/) via the email address compliance@exadigm.com. On an on-going basis, all relevant announcements and issues will be entered into entered into ExaDigm's JIRA tracking software for assignment and threat analysis. Annual audits will be conducted on list by viewing its archive. Any updates or issues will be logged into ExaDigm's JIRA tracking software for assignment and threat analysis. If there are any items found to be vulnerable, a plan will be put in place to address the issue in a timely manner per ExaDigm's PA-DSS policy.



Reviewed by______________________________ Review Date__________________

Annual conformance audit signoff:

Signoff PCI-DSS conformance review

Name:

Date:

Title:

Signature ____

Comments:

Appendix B – TLS Implementation Guidelines

PTS Approved Cipher Suites – TLS V 1.2 Authentication

The POS device must support all 128-bit or stronger ciphers from the TLS 1.2 cipher suite list.

Only SHA based hashes are permitted. ***No MD5 hashes may be utilized.*** Only TLS V1.2 is allowed.

The following are the only ciphers that should be used. At minimum, one of the below ciphers should be used. It is a recommended best practice to include all three ciphers for compatibility. Adding other ciphers can compromise security and invalidate PTS.

Cipher – TLS V 1.2	Auth	Key	Encryption	Digest
TLS_RSA_WITH_3DES_EDE_CBC_SHA	RSA	RSA	3DES_EDE_CBC	SHA-256
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA	DSS	DHE	3DES_EDE_CBC	SHA-256
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA	RSA	DHE	3DES_EDE_CBC	SHA-256

Random number generator for use with OpenSSL:

OpenSSL by default uses a random number generator. The OpenSSL FIPS v 2.0.10 module has been FPIS validated, and controls all RNG functionality – including RNG seeding. For PTS compliance it is not recommended to change the OpenSSL FIPS functionality in any way.

Replay attack prevention:

To prevent replay attacks, each session must be unique. This can be accomplished with TLS by closing each session after the Financial transaction is complete. For Terminal management, the session is closed after the update operation. The OpenSSL function by default protects against replay attacks.

Each session should be unique. This is accomplished by appending unique data to the message digest. See sample code for example

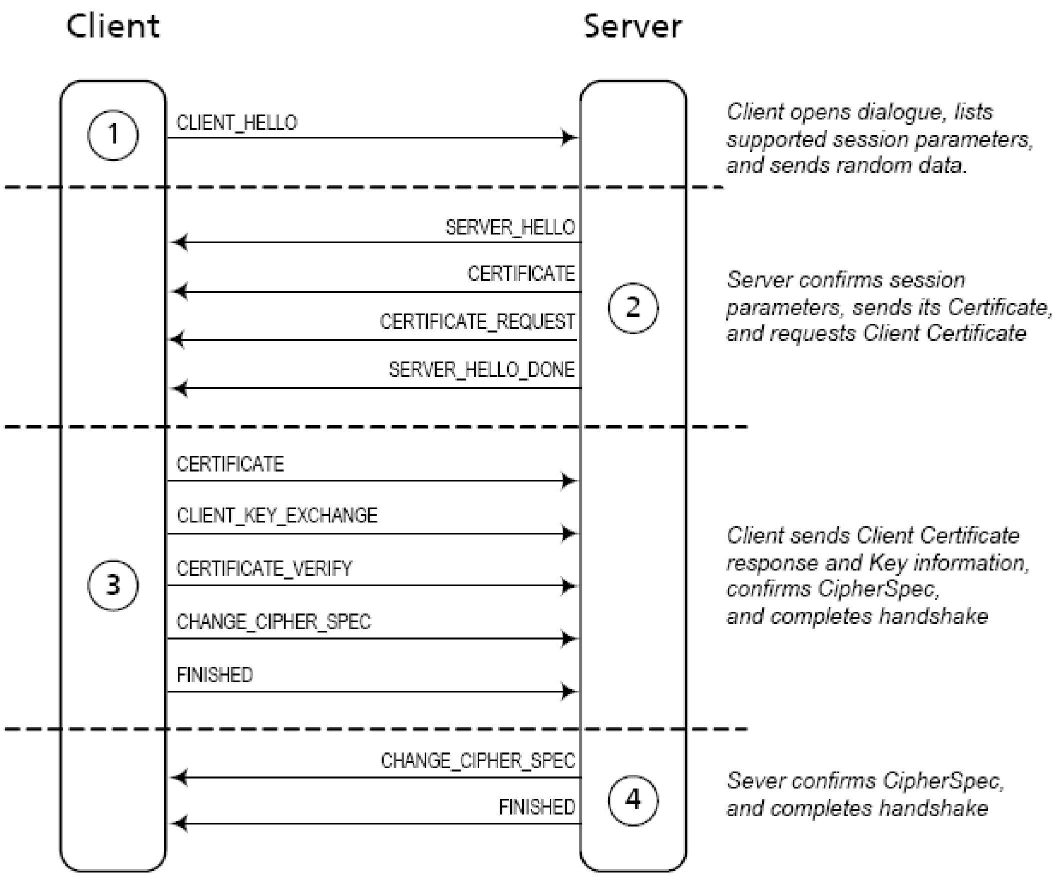
TLS session length

TLS sessions should be closed at the end of a financial transaction, or Terminal management update. TLS sessions should implement a reasonable timeout period – after timing out, the TLS session should be closed.

Certificate Compliance and use are covered in Appendix A

TLS security protocols if used securely offer protection by uniquely encrypting a data ‘tunnel’ between client and host applications. By following these steps, including certificate compliance (for server authentication), FIPS approved openssl, approved ciphers, replay attack prevention (unique message digest), a secure financial transaction can be achieved. Please follow all guidelines set forth in this manual.

The TLS handshake consists of 4 steps:


Client
Server
1
CLIENT_HELLO
Client opens dialogue, lists supported session parameters, and sends random data.
SERVER_HELLO
CERTIFICATE
CERTIFICATE_REQUEST
SERVER_HELLO_DONE
2
Server confirms session parameters, sends its Certificate, and requests Client Certificate
3
CERTIFICATE
CLIENT_KEY_EXCHANGE
CERTIFICATE_VERIFY
CHANGE_CIPHER_SPEC
FINISHED
Client sends Client Certificate response and Key information, confirms CipherSpec, and completes handshake
CHANGE_CIPHER_SPEC
FINISHED
4
Sever confirms CipherSpec, and completes handshake

Appendix C – Certificate Bundles – Handling and Distribution

ExaDigm has a local store for trusted certificates for use in authenticating a Host.
Programmers must allow certificate bundles to be upgradeable. While this can be accomplished through a full download of the terminal, it is a recommended best practice to allow the certificate to be updated through terminal management. This should be done by performing an update via a Terminal Management system such as ExaDigm's TMS. To perform this process the terminal will need to be sent to ExaDigm and then shipped out when the process is complete.

ExaDigm will from time to time provide updates due to vulnerabilities such as revoked certificates. Updates to the certificate bundle will be distributed to programmers in two manners:

- Email distribution (all developers subscribing to ExaDigm's Compliance@exadigm.com email list.
- Developer section on ExaDigm's Development Partner Portal. http://developers.exadigm.com/

The ExaDigm provided root certificate uses 2048 bit RSA for public key certificate. Root certificates are generated as part of the Mozilla open source project. More information on ExaDigm's provided certificates can be found at: http://curl.haxx.se/docs/caextract.html. The cURL site runs a pearl script directly to Mozilla's development site. The certificate bundle is updated daily. More information on the Mozilla project can be found at http://lxr.mozilla.org/seamonkey/source/security/nss/lib/ckfw/builtins/ .

Appendix D – IP Protocol Review (UDP, TCP and ICMP)

Vulnerability Scanning and Analysis

1. QA scans terminals with Nessus tool (3rd party security vulnerability scanner) as well as Wireshark periodically as well as before any application release. (See Vulnerability Testing Policy)
2. If there is/are new vulnerabilities, they are put into is the project tracking tool JIRA.
   JIRA tracks to all newly discovered security threats based on the result of Nessus.
3. Project manager assigns an engineer or more to this issue(s).
   - Engineer team formulates responses to security threats with vulnerability assessments.
   - For 3rd party libraries or module, it regularly checks with the community online or newsgroup to get the latest information or update.
   - The final code should be reviewed by management based on security procedures prior to release.
   - Security engineer signs the final kernel or rootfs images in secure room.
4. Only ExaDigm can release Kernel and Rootfs images since ExaDigm solely holds the kernel and rootfs certificate to sign with.
5. Security patch related to system side will be updated to ExaDigm's TMS.
6. QA verifies if these issues are fixed or not.

Appendix E – Security Protocol Review (Openssl)

ExaDigm currently only uses the openssl-fips 2.0.10 implementation of TLS security protocol.

Appendix F – Secure Coding Practices

Software developers programming software applications for ExaDigm platforms are required to adopt and implement into internal Software Development Life Cycle Secure Coding Practices which are compliant with PCI DSS and PCI PA-DSS requirements (PCI DSS 5.1, 6.3, PCI PA-DSS 5).

Applications Development using C/C++

ExaDigm have implemented the following secure coding practices for developing software applications using C/C++ programming languages as defined by CERT:

1. **Validate input.** Validate input from all non-trusted data sources. Proper input validation can eliminate the vast majority of software vulnerabilities. Be suspicious of most external data sources, including command line arguments, network interfaces, environmental variables, and user controlled files. This specification is for compliance with PA-DSS RSAP 3.1, section 5.2.
2. **Heed compiler warnings.** Compile code using the highest warning level available for your compiler and eliminate warnings by modifying the code.
3. **Architect and design for security policies**. Architect and design your software to implement and enforce security policies. For example, if your system requires different privileges at different times, consider dividing the system into distinct intercommunicating subsystems, each with an appropriately set privilege.
4. **Keep it simple.** Keep the design as simple and small as possible. Complex designs increase the likelihood that errors will be made in their implementation, configuration, and use. Additionally, the effort required to achieve an appropriate level of assurance increases dramatically as security mechanisms become more complex.

5. **Default Deny.** Base access decisions on permission rather than exclusion. This means that, by default, access is denied and the protection scheme identifies conditions under which access is permitted.

6. **Adhere to the principle of least privilege.** Every process should execute with the least set of privileges necessary to complete the job. Any elevated permission should be held for the shortest possible length of time. This approach reduces the opportunities under which an attacker can execute arbitrary code with elevated privileges.

7. **Sanitize data sent to other systems.** Sanitize all data passed to complex subsystems such as command shells, relational databases, and commercial off-the-shelf (COTS) components. Attackers may be able to invoke unused functionality in these components through the use of SQL, command, or other injection attacks. This is not necessarily an input validation problem because the complex subsystem being invoked does not understand the context in which the call is made. Because the calling process understands the context, it is responsible for sanitizing the data before invoking the subsystem.

8. **Practice defense in depth.** Manage risk with multiple defensive strategies so that if one layer of defense turns out to be inadequate, another layer of defense can prevent a security flaw from becoming an exploitable vulnerability and/or limit the consequences of a successful exploit. For example, combining secure programming techniques with secure runtime environments should reduce the likelihood that vulnerabilities remaining in the code at deployment time can be exploited in the operational environment.

9. **Use effective quality assurance techniques.** Good quality assurance techniques can be effective in identifying and eliminating vulnerabilities. Penetration testing, fuzz testing, and source code audits should all be incorporated as part of an effective quality assurance program. Independent security reviews can lead to more secure systems. For example, external reviewers bring an independent perspective in identifying and correcting invalid assumptions.

10. **Adopt a secure coding standard.** Develop and/or apply a secure coding standard for your target development language and platform.

11. **Define security requirements.** Identify and document security requirements early in the development life cycle and make sure that subsequent development artifacts are evaluated for compliance with those requirements. When security requirements are not defined, the security of the resulting system cannot be effectively evaluated.

12. **Model threats.** Use threat modeling to anticipate the threats to which the software will be subjected. Threat modeling involves identifying key assets, decomposing the application, identifying and categorizing the threats to each asset or component, rating the threats based on a risk ranking, and then developing threat mitigation strategies that are implemented in designs, code, and test cases.

Web-Applications Development

ExaDigm have implemented the following secure coding practices for developing software applications for Word Wide Web as defined by The Open Web Application Security Project (OWASP):

1. **Minimize attack surface area.** Every feature that is added to an application adds a certain amount of risk to the overall application. The aim of secure development is to reduce the overall risk by reducing the attack surface area.

   For example, a web application implements online help with a search function. The search function may be vulnerable to SQL injection attacks. If the help feature was limited to authorized users, the attack likelihood is reduced. If the help feature's search function was gated through centralized data validation routines, the ability to perform SQL injection is dramatically reduced. However, if the help feature was re-written to eliminate the search function (through a better user interface, for example), this almost eliminates the attack surface area, even if the help feature was available to the Internet at large.

2. **Establish Secure Defaults.** There are many ways to deliver an “out of the box” experience for users. However, by default, the experience should be secure and it should be up to the user to reduce their security if they are allowed.

   For example, password aging and complexity should be enabled by default. Users might be allowed to turn these two features off to simplify their use of the application and increase their risk.

3. **Principle of Least Privilege.** The principle of least privilege recommends that accounts have the least amount of privilege required to perform their business processes. This encompasses user rights, resource permissions such as CPU limits, memory, network, and file system permissions.

   For example, if a middleware server only requires access to the network, read access to a database table, and the ability to write to a log, this describes all the permissions that should be granted. Under no circumstances should the middleware be granted administrative privileges.

4. **Principle of Defense in depth.** The principle of defense in depth suggests that where one control would be reasonable, more controls that approach risks in different fashions are better. Controls, when used in depth, can make severe vulnerabilities extraordinarily difficult to exploit and thus unlikely to occur.

   With secure coding, this may take the form of tier-based validation, centralized auditing controls, and requiring users to be logged on all pages.

   For example, a flawed administrative interface is unlikely to be vulnerable to anonymous attack if it correctly gates access to production management networks, checks for administrative user authorization, and logs all access.

5. **Fail securely.** Applications regularly fail to process transactions for many reasons. How they fail can determine if an application is secure or not.

6. **Don’t trust services.** Many organizations utilize the processing capabilities of third party partners, who more than likely have security policies and procedures that differ from yours. It is unlikely that you can influence or control any external third party, whether they are home users or major suppliers or partners.

   Therefore, implicit trust of externally run systems is not warranted. All external systems should be treated in a similar fashion.

   For example, a loyalty program provider provides data that is used by Internet Banking, providing the number of reward points and a small list of potential redemption items. However, the data should be checked to ensure that it is safe to display to end users; also, the values of reward points should be verified as positive values and not improbably large.

7. **Separation of duties.** A key fraud control is separation of duties. For example, someone who requests a computer cannot also sign for it, nor should they directly receive the computer. This prevents the user from requesting many computers, and claiming they never arrived.

   Certain roles have levels of trust that differ from normal users. In particular, administrators differ from normal users. In general, administrators should not be users of the application.

   For example, an administrator should be able to turn the system on or off and set password policy, but shouldn’t be able to log on to the storefront as a super privileged user to “buy” goods on behalf of other users.

8. **Avoid security by obscurity.** Security through obscurity is a weak security control, and nearly always fails when it is the only control. This is not to say that keeping secrets is a bad idea, it simply means that the security of key systems should not be reliant upon keeping details hidden.

   For example, the security of an application should not rely upon knowledge of the source code being kept secret. The security should rely upon many other factors, including reasonable password policies, defense in depth, business transaction limits, solid network architecture, and fraud and audit controls.

A practical example is Linux. Linux's source code is widely available, and yet when properly secured, Linux is a hardy, secure and robust operating system.

9. **Keep security simple.** Attack surface area and simplicity go hand in hand. Certain software engineering fads prefer overly complex approaches to what would otherwise be relatively straightforward and simple code.

   Developers should avoid the use of double negatives and complex architectures when a simpler approach would be faster and simpler.

   For example, although it might be fashionable to have a slew of singleton entity beans running on a separate middleware server, it is more secure and faster to simply use global variables with an appropriate mutex mechanism to protect against race conditions.

10. **Fix security issues correctly.** Once a security issue has been identified, it is important to develop a test for it and to understand the root cause of the issue. When design patterns are used, it is likely that the security issue is widespread among all code bases; developing the right fix without introducing regressions is essential.

    For example, a user has found that they can see another user's balance by adjusting their cookie. The fix seems to be relatively straightforward, but as the cookie handling code is shared among all applications, a change to just one application will trickle through to all other applications. The fix must therefore be tested on all affected applications.

    Cover prevention of the common code vulnerabilities for all applications:

    - Code Injection flows (PA-DSS RSAP 3.1, 5.2.1)
    - Malicious file execution
    - Information leakage and improper error handling (PA-DSS RSAP 3.1, 5.2.5)
    - Broken authentication and session management
    - Insecure cryptographic storage (PA-DSS RSAP 3.1, 5.2.3)
    - Insecure communications (PA-DSS RSAP 3.1, 5.2.4)

    In addition, cover prevention of the common code vulnerabilities for all web-based applications:

    - Cross-site scripting (XSS) (PA-DSS RSAP 3.1, 5.2.7)
    - Insecure direct object references (PA-DSS RSAP 3.1, 5.2.8)
    - Cross-site request forgery (CSRF) (PA-DSS RSAP 3.1, 5.2.9)
    - Failure to restrict URL access (PA-DSS RSAP 3.1, 5.2.8)

    For public facing web applications, address new threat vulnerabilities on an on-going basis and ensure that all applications are protected against known attacks by reviewing public-facing web applications via manual or automated application vulnerability security assessment tools or methods, at least annually and after any changes. (PA-DSS RSAP 3.1, 7.1.1-2).

    Deployment guide must include the recommendation for safe deployment options including firewall installation in front of the public facing web applications.

Software Development Process

- "Using both incremental and iterative development", by Alistair Cockburn, Ph.D. http://www.crosstalkonline.org/storage/issue-archives/2008/200805/200805-Cockburn.pdf http://alistair.cockburn.us/Using+both+incremental+and+iterative+development
- "Incremental versus iterative development", by Alistair Cockburn, Ph.D. http://alistair.cockburn.us/Incremental+versus+iterative+development
- "Iterative and Incremental Development: A Brief History", by Craig Larman and Victor R. Basili http://www.craiglarman.com/wiki/downloads/misc/history-of-iterative-larman-and-basili-ieee-computer.pdf

Software Development Best Practices

- "Top 10 Secure Coding Practices", from the Software Engineering Institute at Carnegie Mellon University https://www.securecoding.cert.org/confluence/display/seccode/Top+10+Secure+Coding+Practices;jsessionid=3F479BA97E93FD365EB7989526CF6E9A
- "CERT C Coding Standard", from the Software Engineering Institute at Carnegie Mellon University https://www.securecoding.cert.org/confluence/display/seccode/CERT+C+Coding+Standard
- "Secure Coding Principles", from the Open Web Application Security Project https://www.owasp.org/index.php/Secure_Coding_Principles

Appendix G – Software Version Management System

All software source code files and corresponding documentation are to be preserved in the Software Version Management System Database. ExaDigm employs the CVS VM system.

Access to source code is restricted by user name and password. User names and passwords are assigned by the Access Administrator. This measure ensures that only authorized personnel have the ability to make changes to the source code base.

The Access Administrator is notified by the Human Resource manager regarding termination of employment and takes action to disable access by former employees.

Application Version Numbering

New version of the custom Application is written based on the latest released version of the Grape Payment Application Framework in accordance with *PA-DSS Program Guide*. Payment Applications have separate naming and versioning conventions, and Grape Payment Application Framework have a separate versioning convention.

Grape Payment Application Framework version has the format "XX.YY.ZZZ" and is assigned by the CTO according to the following rules:

- First two digits "XX" represent the "Major" version number, i.e. "01". Major version represents the version of the main branch. It is increased only, if changes were made to the application's external interfaces, such as Shared Objects API or inter-application API, if the new version is a result of a merge of multiple offspring branches, if a security flaw was found or if the key management technique was modified as required by PCI PA-DSS.
- Next two digits "YY" serve the purpose of tracking the branches and minor releases. Branch can be opened to start the parallel development based on the previously released version of Grape Payment Application Framework.
- "ZZZ" digits represent minor revision started due to bug fixing after software was published. I.e. "01.10.002" is a second minor revision of the branch version "01.12"
- Versions should not be changed during internal Development-Testing iterations between QA and Engineering. Minor version ZZZ is increased if the release is a result of correction of the defect reported by customer.

If new Application is written, it must be given a unique name according to ExaDigm Naming Convention described in the document "Application Naming Convention". Importing the application with the same name and version as one of existing applications will result in an error in order to not overwrite the previous application.

In addition to name and version Application is characterized by a minimum Core and Kernel version, which are required for the application to run properly. For example, the application may require minimum

Core version “0800105.G3.NXG300U0.15111502”, which means, that this application will not run properly on lower versions of Core. Such limitations should be written in Application Release Notes document.

Minimum Kernel version “02063101001.NXG300K0.15111501” is changed only if it is absolutely necessary for application development, i.e. there is no way to make the application backwards compatible with the lower Kernel versions.

For additional information refer to *Application Naming Convention.*