

Who's watching your back?

Developers are from Mars, Compliance Auditors are from Venus

Neelay S Shah
neelay.shah@foundstone.com

Rudolph Araujo
rudolph.araujo@foundstone.com

Foundstone Inc., A Division Of McAfee

November 25, 2010



Goal

- ▶ Provide guidance and best practices for software development from a regulatory compliance standpoint

Agenda

- ▶ Background
 - Need for regulatory compliance
 - Intent of regulatory compliance
 - Approach of regulatory compliance
 - Disconnect between the regulatory compliance and the developer community
- ▶ Technology Recommendations
- ▶ People Recommendations
- ▶ Process Recommendations
- ▶ Questions

Background

- ▶ The need for regulatory compliance
 - (In)Famous cases – Data Breaches
 - Heartland Payment Systems – Payment card data breach
 - Millions of payment card details compromised
 - Telstra – Personal account information data breach
 - Over 200,000 customers affected
 - Massachusetts community hospitals – Health information data breach
 - Over tens of thousands of patients impacted
 - Veteran Affairs Data Theft
 - Personal details of over 26.5 millions stolen
- ▶ Intent of regulatory compliance
 - Safeguard the sensitive data
 - Ensure confidentiality and integrity of the data

Background

- ▶ Reason for disconnect between the regulatory compliance and the development community
 - Regulations most often indicate the end goal however they do not talk anything about how to achieve this end goal
 - Language and context of the regulatory compliance tend to be directed towards “legal” community rather than “technical” community
- ▶ Impact of non-compliance
 - Fines, Legal punishment, Public embarrassment, Lack of customer confidence
 - Etc...
- ▶ Approach of the regulatory compliance
 - Physical Security
 - Infrastructure Security
 - Operational Security
 - **Application Development Security**

Background

► Example regulatory compliance

Name	Purpose / Scope
Payment Card Industry (PCI)	Applies to systems dealing with payment card data
European Union Data Protection Directive	Applies to systems collecting, processing and releasing personal data
Health Insurance Portability and Accountability Act (HIPAA)	Applies to systems dealing with personally identifiable health information
Sarbanes-Oxley (SOX)	Applies to all public companies and is concerned with the confidentiality and the integrity of financial reporting
Basel II	Applies to systems that deal with personal financial information

Technology Recommendations

- ▶ Map the regulatory compliance requirements into developer technical categories
 - Data protection in storage and transit
 - Authentication
 - Authorization
 - User and session management
 - Data validation and exception handling
 - Auditing and logging
 - Configuration management

Data Protection in Storage and Transit

ONLY store/transmit sensitive data if you cannot do without it

Do NOT use homegrown/in-house developed cryptographic algorithms

Use well known (tried and tested) libraries for cryptography functions:

Use strong one way hashes to ensure integrity of sensitive data in transit

Whenever possible, use strong one way hashes rather than reversible encryption algorithms to ensure confidentiality

Mask sensitive data within the application user interface

Data Protection in Storage and Transit

Use the OS / framework provided key stores to store the keys

Consider using a 2 key approach. Data encryption key (DEK) in accordance with a key encryption key (KEK).

Provide support for split knowledge and dual control

Use a cryptographic random number generator (RNG) to generate random salt and keys

Support key revocation and key rotation (at least annually)

Data Protection in Storage and Transit

Only support SSLv3/TLSv1

Only support strong ciphers (> 128 bits)

Server certificate must be issued by a well known trusted certification authority

Client application must validate the server certificate

Authentication

Existing single sign-on solutions such as Active Directory

Two factor authentication schemes

Do NOT use sensitive data as an user identifier

Strong password controls

Authorization

Perform authorization checks for every request

Check if the subject (for e.g. user) has the privilege to perform the action (CRUD – create, read, update and delete) on the object (for e.g. data element such as an account)

Do not use hidden fields for user / role identifiers

Do not use cookies for role identifiers

User and Session Management

Design and implement the password recovery scheme securely

Set initial passwords to a completely random value and mark it as expired

Support disabling and re-enabling of user accounts

Use framework issued session identifiers

Email notifications to the user about security sensitive events

Support configuring the user session inactivity timeout with a maximum of 15 minutes

Data Validation

Type, Length, Range and Format checks

Parameterized queries to prevent SQL Injection

Bind queries to prevent LDAP injection,

Safe APIs to prevent command injection

Parameterized XPATH queries to prevent XPATH injection

HTML output encoding to prevent cross site scripting attacks

Error and Exception Handling

Define a custom error page which returns a generic sanitized error message

Handle all known exceptions

Use specific exception handlers before generic “catch-all” handlers

Auditing and Logging

The audit log file should be able to answer the question "Who did what to whom and when".

The following actions must be logged:

- All attempts (failed or successful) to access (add, read, update, delete) objects
- All administrative actions
- All login attempts (failed or successful) to the application

In addition, the following metadata information must be logged for every event to ensure traceability:

- The date and time when the action was performed
- The source of the action e.g. the IP address
- The subject (user) requesting the action
- The result of the action - Success or Failure

The following data must NOT be logged

- Sensitive data such as credit card numbers, sensitive authentication data and application user passwords

Configuration Management

Support run-time configuration of the audit log

- Log location – Database, File etc.
- Log level
- Log maximum size
- Log archival strategy – Log rotation, Log wrapping etc.

Do NOT use default credentials for any third party / COTS component

For all new user / service accounts created as part of the application installer / runtime

- Do NOT use default / fixed credentials

Recommended Best Practices

► Cryptographic algorithms

Type	Name
Symmetric cryptography	AES-512
Asymmetric cryptography	RSA-2048
One way hash / MAC	SHA-512, HMAC-SHA512

► Password controls

Password Length	At least 8 characters
Password Complexity	1 upper case, 1 lower case, 1 numeral and 1 special character
Password Expiry	90 days at a maximum
Password History	Disallow at least the most recently used 4 passwords
Account Lockout	5 consecutive failed attempts

The Macro View

- ▶ What we focused on so far were the technical aspects
- ▶ Most regulatory compliance laws will also focus and impact the software development lifecycle
- ▶ Let's see those next

People and Process Recommendations

Train the development team in the fields of software security

Establish secure coding guidelines

Monitor all third party libraries and components being used for vulnerabilities

Perform prompt security code reviews

Perform prompt security testing of every change


Do NOT use production data in the testing environment

References

1. Defining the Developer's Role in achieving PCI-DSS compliance (Coming soon) - www.softwaremag.com
2. Regulatory Compliance Demystified: An Introduction to Compliance for Developers - <http://msdn.microsoft.com/en-us/library/aa480484.aspx>
3. Compliance: What Architects Must Know - <http://msdn.microsoft.com/en-us/library/bb421526.aspx>

Questions





Who's watching your back?

Developers are from Mars, Compliance Auditors are from Venus

Neelay S Shah
neelay.shah@foundstone.com

Rudolph Araujo
rudolph.araujo@foundstone.com

Foundstone Inc., A Division Of McAfee

November 25, 2010

