Getting-Ready For Secure Software Standard Security Assessment

A Vendor and a Penetration Tester Perspective

Foregenix, October 2024

Flavio Bonfiglio Sorans

Managing Principal
Head of Practice Software Security, Foregenix

FOREGENIX

Carlos Marquez

Senior Information Security Consultant, Foregenix

FOREGENIX

PCI Security Standards Council

Agenda

Getting Ready For Secure Software Standard Security Assessment

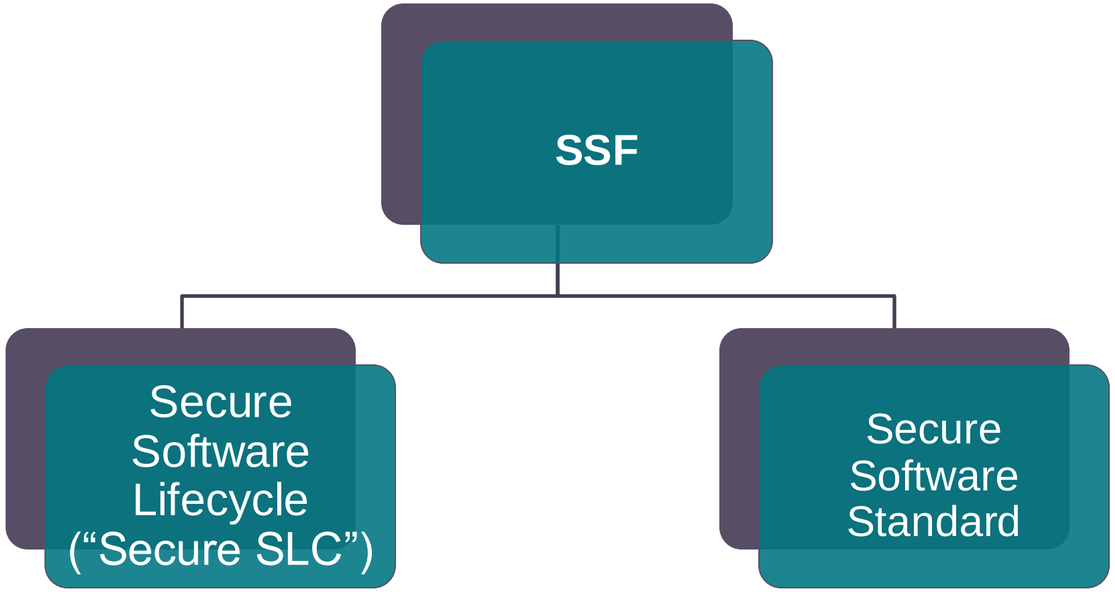
- Overview of the PCI Secure Software Standard
- **Traditional application penetration test**

  vs. **PCI Secure Software Standard testing**
- What should be prepared **before** the technical assessment?
- Security tests and common findings
- Final remarks

PCI Security Standards Council

Overview of the PCI Secure Software Standard

PCI Software Security Framework



Critical Assets

PCI SSF Key concepts

These three elements are explicitly considered **critical assets** in the standard

Sensitive Data

Sensitive Functions

Sensitive Resources

PCI Security Standards Council

Sensitive Data*

PCI SSF Key concepts

"**Any data** that requires protection from **unauthorized disclosure** (confidentiality) or **modification** (integrity)."

- Cardholder data (CHD)
- Sensitive authentication data (SAD)
- Tokens
- Cryptographic key material
- Authentication credentials
- Internal system information

*Per PCI Glossary

PCI Security Standards Council

Sensitive Functions*

PCI SSF Key concepts

Any functionality of the payment software that…

- alters other software **functionality** or **configuration**
- processes **Sensitive Data**
- provides **Security Features**
- interacts **with Sensitive Resources**

*Per PCI Glossary

Sensitive Resources*

PCI SSF Key concepts

- Shared files
- Registry keys
- Environmental settings
- Communication channels
- Cache
- Shared libraries
- System interfaces
- Web services

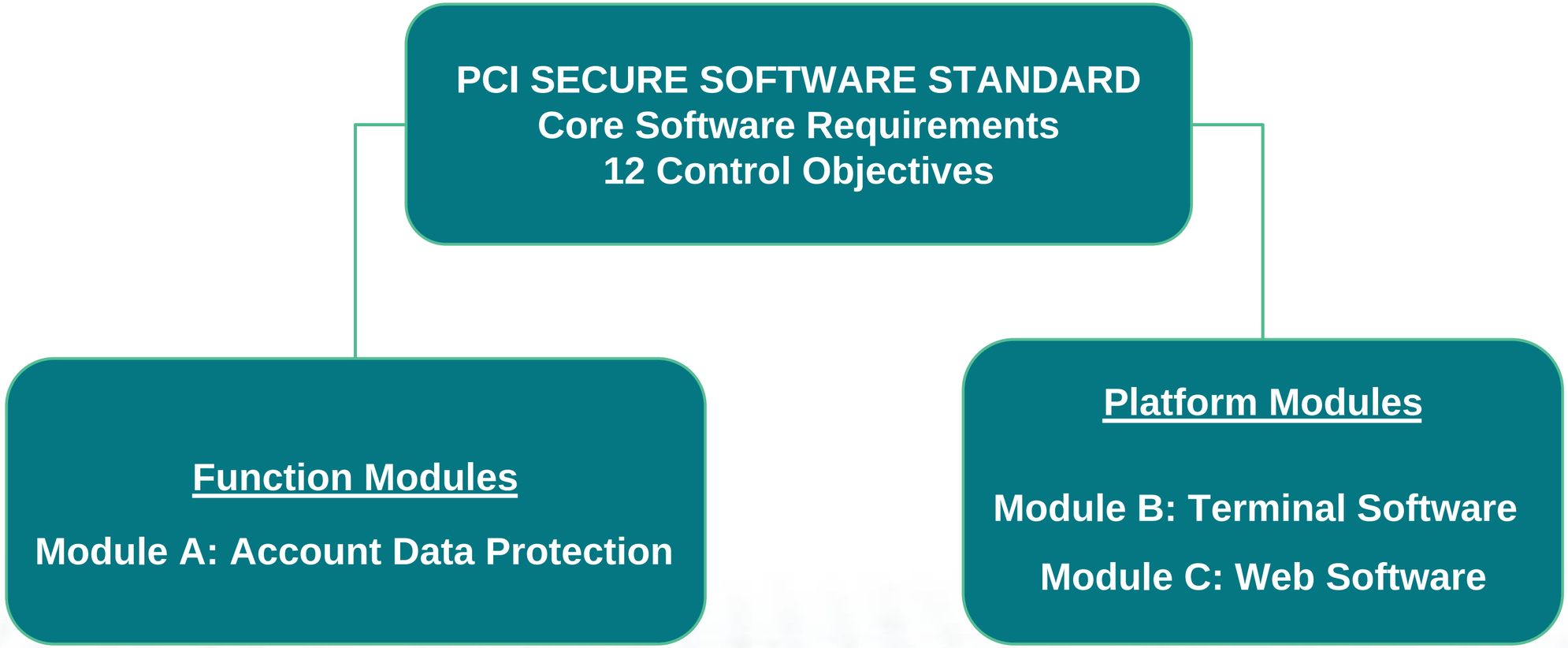
Sensitive Resources may also constitute **“Sensitive Data”** and may require protection from unauthorized disclosure or modification.

*Per PCI Glossary

PCI Security Standards Council

Introduction to the PCI SSF

Standard architecture: Modular requirements



PCI Security Standards Council

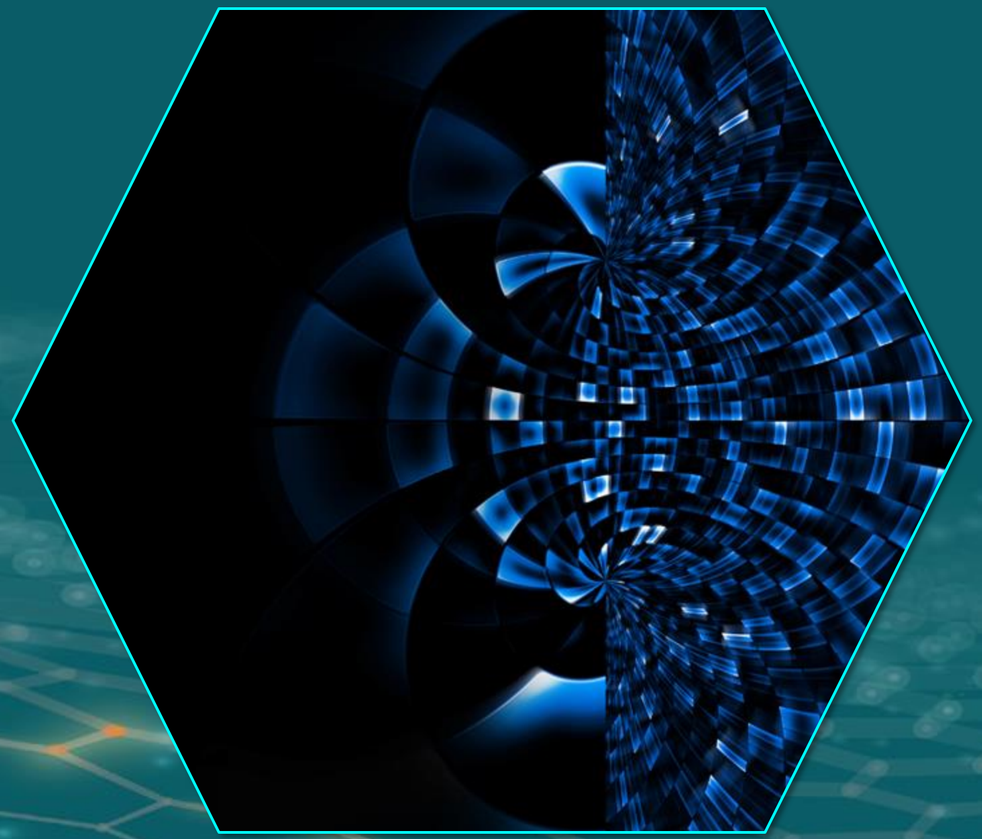
PCI Secure Software Standard v1.2.1

Core Requirements

Core Requirements must be assessed to all payment software

- **Module A** is required for payment software that handles Account Data
- **Module B** is required for payment software running in PTS devices
- **Module C** is required for payment software that conducts **electronic payment transactions** over the Internet.

PCI Security Standards Council



Traditional application penetration test vs. PCI Secure Software Standard testing

PCI Security Standards Council

Key Differences

Application Penetration test

- Blackbox /Graybox
- Limited Scope (Usually tested an already deployed solution)
- No access to source code or backend components
- Rules of the engagement
- Technical review

Security Tests for Secure Software Standard

- White Box.
  Access to the servers, databases and components
- Access to Source code to check sensitive functions and coding practices
- Access to secure test review results (e.g. SAST, DAST, components analysis, vulnerability scans, penetration tests)
- Software installation
  (Installation process and configuration)
- Sensitive data handling
  (Development cycle + deployment)
- Forensic review:
  memory dumps and disk images

PCI Security Standards Council

What should be prepared before the technical assessment?

Each stakeholder may have different priorities

Vendor - What to prepare

Software Owner

- Document Data Flows of sensitive data
- Understand how to securely install the application (Privileges, Secret management)
- Difference between Mutable and Immutable objects
- How secrets are protected (Database passwords, Secret for private keys or vaults, API keys)
- Clear understanding of authorisation and authentication details

Software consumer - What to check

Software Owner

- Certified environment (e.g. OSs, DBs, components)
- Technical installation process and requirements
- Understand the architecture and communications (Data/Management plane)
- **Operation responsibilities** (e.g. Cloud responsibility model, operations, integrity controls, activity tracking logs and monitoring, threat intelligence…)

Secure Software Assessor

Auditor

- Testing scenarios
  - Identify main data flows
  - Credentials, connectivity
  - Identify attack scenarios
- In scope components
- Front end components, Application components, Backend components (code, installers, documentation)

Security tests and common findings

Assessing the PCI Secure Software Standard testing

PCI Security Standards Council

Security Tests

Phases

1. Information Gathering
2. Source Code Review
3. Installation process tests
4. “Production” tests
5. Post-mortem tests / Forensic

Security Tests

1. Information gathering

Input:

- List of Sensitive data, function and resources [CO1]
- Configuration options: files, scripts, internal functions, menu and options, that could impact sensitive data. [CO1]
- Data flows → Key in the process [CO1]
- Application architecture, users (OS, DB, app), firewall rules
- Software Vendor Implementation Guidance [CO12]

Assessor tasks:

- Identify attack scenarios → From the architecture and data flows
- Define required tools/methods to complete tests

Security Tests

2. Secure Code Review

- **Immutable objects** used for sensitive data
  (e.g., String and BigInteger in Java, C#, .net) [CO3]
- Secure memory deletion of sensitive data in sensitive functions
- Secure coding practices: e.g. input validation, error management
- Application follows the guidance for the third-party components
  (e.g. if HSM in use then review/verify the configuration for FIPS 140-2 or 140-3)
- Hardcoded credentials or keys
- Encryption used for sensitive data, including passwords storage
- Use of RNG algorithms to create keys
- Secure connection to other components (DB, HSM, …)

Common findings

2. Secure Code Review

- Use of static entries
- No secure memory deletion of sensitive data:
  - Deletion is left to the Garbage Collector
- Insecure connection to other elements:
  - Connection to the Database using plain text passwords in configuration files
  - Microservices files/APIs

Security Tests

3. Installation Process

- Default users or certificates are required to be changed
- Users and privileges to run the processes
- Scripts and assigned permissions
- Installed components and third-party modules
- Environment variables, registry, others (passwords?)
- Configuration files and the impact in the application or data
- Processes listening before and after the installation
- Network/Firewall permissions

Common findings

3. Installation Process

- Default certificates
- Process requiring to run as a high privileged user
- Creating accounts or keys in bash leaving the password in the history
- Third-party modules installed with default configurations:
  - JBOSS by default (admin/admin)
  - Additional ports, not in use

1. In the folder where the dockerfile resides /home/dockerfiles/WARP_MP/WARP.ear, create a file called build.sh and run.sh. chmod both scripts to 775

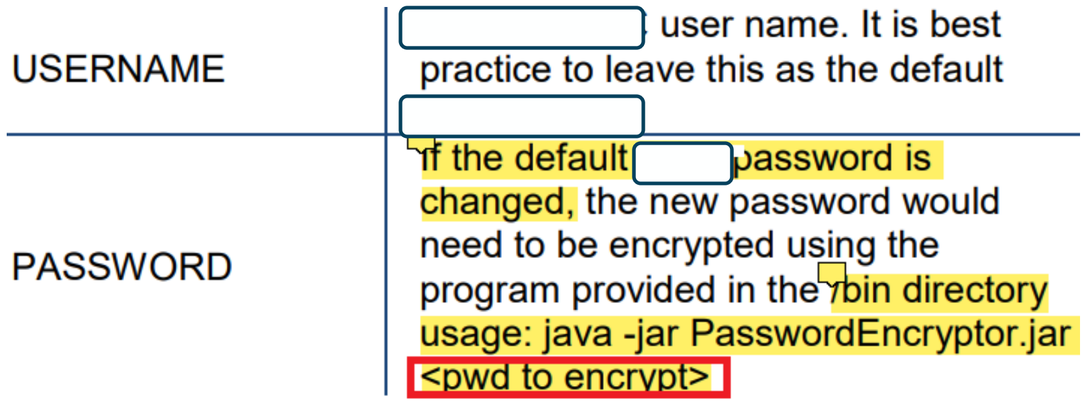
```
touch build.sh run.sh
chmod 775 *.sh
```

TCP	0.0.0.0:5000	0.0.0.0:0	LISTENING	6608
TCP	0.0.0.0:5040	0.0.0.0:0	LISTENING	9668
TCP	0.0.0.0:49664	0.0.0.0:0	LISTENING	1276
TCP	0.0.0.0:49665	0.0.0.0:0	LISTENING	1100
TCP	0.0.0.0:49666	0.0.0.0:0	LISTENING	1884
TCP	0.0.0.0:49667	0.0.0.0:0	LISTENING	3052
TCP	0.0.0.0:49672	0.0.0.0:0	LISTENING	4488
TCP	0.0.0.0:49678	0.0.0.0:0	LISTENING	1204
TCP	0.0.0.0:63807	0.0.0.0:0	LISTENING	4744
TCP	127.0.0.1:10391	0.0.0.0:0	LISTENING	14688

Common findings

3. Installation Process

- Creating accounts or keys in bash leaving the password in clear in the history
- Crontab required under root user calling dangerous scripts (to run zip or backups)



Security Tests

4. “Production” Tests

- Identify any APIs or interfaces exposed by default [CO 2.1.a]
- Authentication for interfaces relying in third parties [CO 2.1.b]
- Transmission of sensitive data relying in third parties [CO 2.1.c]
- Identify known vulnerabilities for the interfaces exposed [CO 2.1.d]
- Check that mitigations are identified by the vendor and guidance is provided in the software vendor threat information.
- information
- Identify third-party modules exposing functions [CO 2.1.e]
- TLS implementation + Transport (e.g. HTTP)

Security Tests

5. Post-mortem tests / Forensic test

- Identify sensitive data in memory: Memory dumps analysis
- Search for account data
- Authentication material
- Sensitive data in logs

Secure Software Standard Security Assessment

Final remarks

- Gathering the correct information of the tested payment application **prior** to the assessment is a key success factor for the validation process.

Based on real assessments' observations, we suggest:

- **Secure integration with 3rd party software** requires special attention.
- Proper **secret management** practices and **cleanup process after installation** are good practices.

PCI Security Standards Council

Security Standards Council®