Payment Card Industry Compliance for Large Computing Systems

Applying Payment Card Industry Compliance Standards to Mainframe Environments

12th July, 2010

Version 1.1.2

atsec information security corporation
9130 Jollyville Road, Suite 260
Austin, TX 78759
Tel: +1 512 615 7300
Fax: +1 512 615 7301
www.atsec.com

Table of contents

Acknowledgements

atsec gratefully acknowledges contributions from several people. Their expertise and knowledge on this subject is invaluable.

From atsec: Jeff Jilg, Gerald Krummeck, Helmut Kurth, Sal La Pietra, Lou Losee, David Ochel, Fiona Pattinson, Andreas Siegert, Erik Wilson, Clemens Wittinger

From IBM®: Greg Boyd, John Dayka, Lennie J Dymoke-Bradshaw, Walter Farrell, John C Jones, Michael Jordan, William Penny

Copyright and Trademarks

Notes

This report has not been produced in association with the PCI SSC. The statements and opinions made in this report are those of the authors. Each Report on Compliance is written by Qualified Security Assessors (QSAs) properly accredited by the PCI SSC and it is the QSA’s responsibility to ensure that all the relevant interpretations and guidance issued by the PCI SSC are considered. The authors, reviewers and atsec do not accept any responsibility for the acceptance by the relevant stakeholders of any suggestions made in this report.

Release Notes

Version	Date	Notes
1.0	March 2010	First public version
1.1.1	2 June, 2010	Technical corrections
1.1.2	12th July, 2010	Addition of discussion on two-factor authentication

Foreword

Payment Card Industry Compliance for Large Computing Systems, by atsec in association with IBM and other leading Large Computing Systems (LCSs) experts, is a report that aims to address common questions regarding the compliance of LCS environments with the Payment Card Industry Security Standards Council's Data Security Standard. This report is aimed at addressing the need for guidance and information by Qualified Security Assessors (QSAs), merchants, and service providers whose cardholder data environment is largely based on LCS technology. It may also be of interest to acquirers and card brands.

Achieving and assessing Payment Card Industry (PCI) compliance in an LCS environment can be a challenge as the standard is more focused towards a distributed systems paradigm. A full understanding of the security features and advantages inherent to LCS systems enables the assessment of compliance to the standard but is a very broad and detailed topic.

Drawing from an extensive knowledge of mainframe and LCS security coupled with experience as an accredited QSA company, atsec's consultants have gained a thorough understanding of the security of these systems at every level, including operating systems, virtualization technology, applications, networking and communication, and mainframe environments. This experience has been gained through Common Criteria evaluation under US and European governments, cryptographic testing and analysis, and mainframe penetration testing for large financial customers on the following systems and applications:

- z/OS®
- z/VM®
- Processor Resource/Systems Manager™ (PR/SM)™
- DB2®
- Multiple Tivoli and third party vendor applications

atsec presents an analysis of the PCI standard in the context of an LCS environment and provides focused guidance to QSAs and their subject organizations on the PCI assessment of such environments and the resources available to support an assessment. This report provides the necessary insight into an LCS's security to QSAs and other PCI professionals.

Executive Summary

The context of an LCS brings with it various strengths and weaknesses that an organization using such a system must understand. In the context of a PCI DSS assessment the QSA assigned to assess compliance with the PCI DSS should also be aware of these parameters. In LCSs, utilizing a centralized architecture, the cited strengths often include better security, better manageability, and a more integrated model for dependability and fault tolerance. On the other hand the disadvantages include increased complexity, the often associated large organizations and unpredictability.

The users and operators of LCS are typically major players in the industry and include not just the larger merchants, but also banks operating as issuers or acquirers. They have typically been in the industry for a long time and understand the field well.

In the LCS environment, defining the scope, tracking down all the instances of card holder data is important. The LCS has often been an integral part of an organizations IT facilities for many years, and legacy data models, applications and configurations are often found and need to be dealt with in the context of the relatively new requirements of the PCI DSS.

In this version of the paper we concentrate on the native operating system, z/OS, while it is true that LCS virtualization is also heavily used, we have not discussed this topic in as much detail as we would like. We have introduced the topic of virtualization and explain the basics including LPAR and z/VM. We guide the reader to the key references throughout the paper to support our arguments, and that are useful for those involved with ensuring compliance.

We draw out the argument that the LCS has “security designed in” and that segregation of data, such as card holder data, is not only possible but can be relied upon with some assurance. We explain several of the mechanisms that achieve this, including of course features such as RACF. We also discuss some of the formal assurance measures that vendors have invested in to support these claims. We examine the file systems employed and the security features they provide.

We also discuss some of the communications features that the PCI DSS focuses on including discussion of the native TCP/IP stack and the Parallel Sysplex.

The cryptographic functionality which provides the basic encryption capabilities is an important topic for PCI DSS compliance, as is auditing and these too are explained in the text.

We also give some thought to topics such as vulnerability scanning and penetration testing which in an LCS environment are not typical “off the shelf” activities found in most of the industry.

Finally we have discussed each of the requirements of the PCI DSS in the context of LCS, discussing specific topics of note in this environment. We have provided some typically used and recognized compensating controls, although of course in this paper they are examples and may not apply in your specific LCS environment.

We hope to have the opportunity to continue to develop this paper and add key topics and more in depth discussion where appropriate. In order to do this we canvas feedback and involvement from the industry.

1 Introduction

In the “PCI Quick Reference Guide: Understanding the Payment Card Industry Data Security Standard version 1.2” [5], the Payment Card Industry (PCI) Security Standards Council (SSC) introduces the need for protecting cardholder data with the words:

“The twentieth century U.S. criminal Willie Sutton was said to rob banks because “that’s where the money is.” The same motivation in our digital age makes merchants the new target for financial fraud. Occasionally lax security by some merchants enables criminals to easily steal and use personal consumer financial information from payment card transactions and processing systems.

It is a serious problem. More than 234 million records with sensitive information have been breached since January 2005, according to Privacy Rights Clearinghouse.org. As a merchant, you are at the center of payment card transactions so it is imperative that you use standard security procedures and technologies to thwart theft of cardholder data.

Merchant-based vulnerabilities may appear almost anywhere in the card-processing ecosystem including point-of-sale devices; personal computers or servers; wireless hotspots or Web shopping applications; in paper-based storage systems; and unsecured transmission of cardholder data to service providers. Vulnerabilities may even extend to systems operated by service providers and acquirers, which are the financial institutions that initiate and maintain the relationships with merchants that accept payment cards. Compliance with the Payment Card Industry (PCI) Data Security Standard (DSS) helps to alleviate these vulnerabilities and protect cardholder data.”

Each card brand runs a security program that is organized independently of the PCI SSC however the PCI DSS is a resource specified by all of the major payment card brands as the key document for on-site assessments led by Qualified Security Assessors (QSAs) and for self-assessments. The standard gives twelve requirements covering many security technologies and business processes, and reflects some of the financial industry’s best practices for securing sensitive information. The intent of the card brand security programs and the supporting standards is to reduce the risk of cardholder data (CHD), and other critical information, being compromised.

1.1 Large Computing Systems

What do we mean by Large Computing Systems or LCSs? In this report, we tackle the case of centralized systems of which the System z® is typical. We do not attempt to discuss grid computing, service oriented architecture, and “cloud” architectures.

The commonly adopted distributed systems architecture presents advantages to the commercial market segment. With the rise of the Internet and greater connectivity facilitating online commercial transactions, the benefits presented by such architecture; resource sharing, openness, concurrency, scalability, and fault tolerance have encouraged wide adoption of a distributed architecture in many industries, including the payment card industry. The disadvantages of distributed architecture are also well known and typically cite the following issues; complexity, security, manageability, and unpredictability.

In LCSs, utilizing a centralized architecture, the cited strengths are better security, better manageability, and a more integrated model for dependability and fault tolerance.

The prevalence of distributed systems architecture and web based applications in the general payment processing population is reflected in the PCI DSS.

QSAs who become involved with assessing CHD environments outside the typical distributed systems paradigm reflected by the PCI DSS may find challenges in reasonably interpreting the DSS and assessing the efficacy of the sub-requirements in regard to the intent of the standard. In this report we argue that a QSA with a thorough understanding of the security mechanisms presented by an LCS would be able to address the intention of the PCI DSS requirements and also to ensure that the scoping of the cardholder data environment is made in the most efficient way.

This report is presented in three main sections:

- Chapters 1 and 2 contain a brief background of the Payment Card Industry and an introduction to the standards. We assume that QSAs already have an excellent knowledge of this topic, and indeed the assessed organizations' PCI assessment staff would also typically already have this knowledge. It is included here for the casual reader, and to refresh the audience on some of the key points before they embark on the discussion of PCI compliance in an LCS environment.
- Chapter 3 discusses the typical cardholder data environment in an LCS scenario. Chapters 7 and 8 discuss in more detail the requirements of the PCI DSS and any compensating controls that typically may need to be specified in this environment. We examine the case of a typical LCS environment, discussing the sub-requirements provided by the PCI DSS in the context of such a case, and describing other assurances that can be used to supplement the PCI DSS checklist. We explain the risks that are apparent in the typical distributed system paradigm and how these and other risks are mitigated by the controls and mechanisms built-in to a centralized computing paradigm.
- Chapters 4, 5, and 6 contain technical discussions of the features, technologies, and facilities offered by LCSs and that a QSA may draw from in assessing compliance with the PCI DSS.

Finally, we make recommendations for compensating controls in this environment that we hope can become generally accepted by the community and introduced into future versions of the standard.

This report refers largely to IBM's System z and IBM products, a vendor with whom atsec has a long history. Over the years we have developed great knowledge of their products and designs. We welcome input from other vendors of LCS products and applications whether running on System z or other LCS platforms.

2 Payment Card Industry Security

The following sections provide an overview of how security is enforced throughout the payment card industry. We introduce the main players in the industry, security programs mandated by the card brands, and describe the Payment Card Industry (PCI) Security Standards Council (SSC) and its PCI Data Security Standard (DSS), which has become a central element of the brand's security programs and is enforced throughout large parts of the industry.

The great rise in fraud associated with payment card processing perpetrated on the industry over the last few decades prompted each of the card brands to instigate security programs and standards applicable to their members with the goal of reducing the risks of loss. With recognition that the problem was not by any means restricted to one particular stakeholder, it became apparent that a common approach, adopted by the key players, would add much value to the efforts to reduce fraud in the industry.

2.1 The Payment Card Industry Security Standards Council

The Payment Card Industry (PCI) Security Standards Council (SSC) [51] was founded by five international payment card brands, American Express, Discover Financial Services, JCB International, MasterCard Worldwide, and Visa, Inc. in 2004. The council's mission is to provide common standards, education, and awareness of data security among merchants, service providers, and acquirers active in the PCI who store, process, or transmit cardholder data.

The various card brands that formed the PCI SSC specify, through contractual agreements, with their stakeholders the various security specifications applicable. Originally the standards supporting these operations were independent and developed as part of the card brand security programs, such as Visa's Account Information Security (AIS), MasterCard's Site Data Protection (SDP), American Express's Data Security Standards (DSS) , Discover Card's Discover Card Information Security and Compliance (DISC), and the JCB International Data Security Program (for more information, see table below).

In December 2004, the first version of one aligned standard, the Payment Card Industry Data Security Standard (PCI DSS), was released by the PCI SSC. The PCI DSS specifies security requirements for cardholder data environments – environments that contain systems which process, transmit, and/or or store cardholder data – both on a technical (architectural) and operational (security management) level. The standard is maintained over a two year cycle, with version 1.1 issued in 2006, version 1.2 issued in 2008 and a new version currently scheduled for release in 2010. In the years after the PCI DSS, other standards regarding technical requirements for payment applications and terminals were added to the PCI SSC portfolio.

Although the PCI DSS and other jointly developed standards are now jointly referred to by the different card brands, each brand still maintains its own agreements with stakeholders and runs their own security programs to actually enforce the compliance with these standards. The security programs and where to find them are listed in the table below.

Card Brand	Web site
The MasterCard Site Data Protection Program(SDP)	http://www.mastercard.com/us/sdp/index.html
Visa Cardholder Information Security Program (CISP)	http://usa.visa.com/merchants/risk_management/cisp_overview.html
American Express Data Security	https://www209.americanexpress.com/merchant/sin

Operating Policy (DSOP) Compliance Program	glevoice/dsw/FrontServlet?request_type=dsw&pg_nm=spinfo&ln=en&frm=US&tabbed=complienceRequirement
Discover Information Security & Compliance (DISC)	http://www.discovernetwork.com/fraudsecurity/disc.html
JCB	http://www.jcb-global.com/english/pci/

The PCI SSC will issue minor updates to the DSS standards as needed, at the time of writing the current version of the PCI DSS is 1.2.1. The council has also released supplemental documentation with the intention of clarifying the standard and shedding light on requirements that are often misunderstood or that require more detailed interpretation. These documents include information supplements, such as for the requirements "Requirement 11.3 Penetration Testing", "Requirement 6.6 Code Reviews and Application Firewalls Clarified", and the "PCI DSS Wireless Guidelines". An explanatory document "Navigating the PCI SSC - Understanding the Intent of the Requirements" has also been produced.

Further standards and programs have also been produced by the PCI SSC supporting the industry. For specifying the security requirements for payment application software intended to be sold as Commercial off the shelf (COTS), the Payment Application Data Security Standard applies. For guiding the efforts of Approved Scanning Vendors (ASV) as they perform the specified network vulnerability scans, the "PCI DSS Validation Requirements for Approved Scanning Vendors" apply. For Pin Transaction Security Devices such as Pin Pad devices, point of sale devices (POS), Hardware Security Modules (HSM), and unattended payment terminals (UPT), the requirements are now centrally specified in standards that are managed by the PCI SSC as well.

2.2 The Key Players in the Payment Card Industry

There are several roles typically identified in the payment card industry:

- Cardholders – who use payment cards (for example, credit cards and debit cards) to facilitate the purchase of goods and services
- Issuers - who are typically banks who provide the payment cards to the cardholders and assume the risk of extending credit to them
- Merchants – who accept payment via payment cards for the exchange of goods or services and generate payment transactions to settle the financial obligations created
- Acquirers – who are typically "Merchant banks", soliciting relationships with merchants and charge them for the provision of authorization, clearing and settlement services
- Payment or Card Brands, commonly referred to as "brands" – who run payment networks and define the general terms and conditions that apply to all participants in their payment scheme; for example American Express, Discover, JCB, MasterCard, Visa Inc., and Visa Europe
- Service Providers – who perform ancillary services within the industry that involve processing, storage, transmission, and switching of transaction and cardholder data; for example transaction processors, payment gateways, sales organizations performing recruitment of cardholders and merchants, credit reporting services, card personalization services, and managed firewall providers

It is typically the acquirers who have the job of ensuring that their merchants comply with the payment brands' security programs and hence the PCI DSS.

2.3 The Data Security Standard Described

The PCI DSS is a compliance standard, produced by the PCI SSC, with which all merchants, service providers, and acquirers are mandated to comply by the various payment card brands. The standard provides controls with the goal of protecting certain valuable information assets (for example, cardholder data identified by the card brands).

One key observation to make about the environment in which the standard operates, the payment card industry, is that is has high value assets to be protected, an environment a broad attack surface, with well provisioned attackers and a fast evolving attack technology [10]. This is the classic definition of a hostile environment.

The standard implies that the assets of value to be protected are the cardholder data and associated sensitive authentication data (see "PCI DSS Applicability Information" in the current standard [1]). Of these identified data assets, the other data elements are considered to be in scope only if the primary account number (PAN) is stored. Neither specific threats nor vulnerabilities are identified in the standard and the reader is obliged to reverse-engineer these.

The attack-surface, or the "cardholder data environment," is assumed to be the network segment containing the cardholder data and the interface of any software applications handling the assets. The sub-requirements (or controls) given in the DSS are strongly oriented towards the typical network designed with a distributed systems architecture. For example, requirement 2.2.1 mandates that only one primary function per server is implemented and the supporting documentation [3] confirms that this does not apply in mainframe environments.

The PCI DSS presents a common set of both security functional and process controls addressing broadly applicable threats and vulnerabilities to the identified assets, across the payment card industry.

The subject organization is also required to produce an annual formal risk assessment for the scope of the assessment at hand, "the card holder data environment." They may add further controls, which would not be assessed by a QSA, but they cannot reduce the requirements made on them by the DSS.

Several organizational process assurance requirements to be included in the organization's Security Policy are made. These include operational security procedures, human resource requirements, handling third parties, and incident response.

An appendix specifies additional requirements for shared hosting providers as the nature of shared hosting introduces a class of service provider that presents additional vulnerabilities to the card holder data stored in such an environment.

A related standard, the Payment Application DSS (PA-DSS), also produced by the PCI SSC and currently being enforced by Visa, specifies in-depth controls to ensure that commercial off-the-shelf payment applications are developed based on secure software development practices in accordance with the PCI DSS.

Various levels of assurance of compliance with the DSS are defined by the brands which are selected as applicable to the stakeholder based upon pre-defined criteria. These criteria include the annual number of transactions or whether a particular subject organization has already suffered a successful attack.

At the highest level of assurance, an independent QSA uses the methodology defined in the standard to assess compliance to the DSS. At the lowest level of assurance, a simple self assessment questionnaire is presented to monitor compliance. However in all cases full compliance with the PCI DSS is expected

2.3.1 Control Objectives: Requirements for PCI Compliance

The PCI DSS standard is built around twelve key requirements. These are reproduced below and are the basis for every assessment of compliance [1]. The audience of this report should already be very familiar with them.

Build and Maintain a Secure Network

1. Install and maintain a firewall configuration to protect cardholder data.
2. Do not use vendor-supplied defaults for system passwords and other security parameters.

Protect Cardholder Data

3. Protect stored cardholder data.
4. Encrypt transmission of cardholder data across open, public networks.

Maintain a Vulnerability Management Program

5. Use and regularly update anti-virus software.
6. Develop and maintain secure systems and applications.

Implement Strong Access Control Measures

7. Restrict access to cardholder data by business need-to-know.
8. Assign a unique ID to each person with computer access.
9. Restrict physical access to cardholder data.

Regularly Monitor and Test Networks

10. Track and monitor all access to network resources and cardholder data.
11. Regularly test security systems and processes.

Maintain an Information Security Policy

12. Maintain a policy that addresses information security.

2.3.2 PCI Terminology and the LCS Model

A few of the key terms to this report and their official definitions are taken from the PCI Glossary [4] and are reproduced below for ease of reference and to support our discussion.

Cardholder data environment: “Area of computer system network that possesses cardholder data or sensitive authentication data and those systems and segments that directly attach or support cardholder processing, storage, or transmission. Adequate network segmentation, which isolates systems that store, process, or transmit cardholder data from those that do not, may reduce the scope of the cardholder data environment and thus the scope of the PCI DSS assessment. A cardholder data environment is comprised of system components. See System Components.”

Server: “Computer that provides a service to other computers, such as processing communications, file storage, or accessing a printing facility. Servers include, but are not limited to web, database, application, authentication, DNS, mail, proxy, and NTP.”

Strong cryptography: “Cryptography based on industry-tested and accepted algorithms, along with strong key lengths and proper key-management practices. Cryptography is a method to protect data and includes both encryption (which is reversible) and hashing (which is not reversible, or “one way”). SHA-1 is an example of an industry-tested and accepted hashing algorithm. Examples of industry-tested and accepted standards and algorithms for encryption include AES (128 bits and higher), TDES (minimum double-length keys), RSA (1024 bits and higher), ECC (160 bits and higher), and ElGamal (1024 bits and higher). For more information, see NIST Special Publication 800-57 (http://csrc.nist.gov/publications/).

System components: “any network component, server, or application that is included in or connected to the cardholder data environment.” We note that all system components must comply with the PCI DSS.

Network segmentation is the method typically used to isolate the cardholder data environment from the remainder of the organization’s network. It is not a PCI DSS requirement to use network segmentation, but doing so can reduce the scope of the infrastructure that needs to be operated in compliance with the standard significantly, reducing the effort and the associated costs of a PCI DSS compliance assessment. Note that network segmentation implies, and is indeed very often achieved, through the use of technologies such as internal network firewalls and routers with access control lists, but it can also be used to describe other technologies that restrict access to a particular segment of a network.

As we have already illustrated, the PCI DSS standard is clearly oriented towards a distributed systems paradigm. The concept of “network” is interpreted at the level of “computer” connection, often interpreted by a QSA as the physical boundary of a machine. This works well in a distributed model where it is both possible, and sensible to specify for example that a single system performs only one function (PCI DSS requirement 2.2.1).

Throughout the PCI DSS, the definition of network and server is dependent on the term “computer” which is not a defined term. Another key term not defined is “system”. Both of these terms are hard to define, especially in a broad standard aimed at a whole population of technology environments.

The adoption of a distributed systems paradigm is a logical one for the PCI DSS. It is difficult for a standard to directly address all cases when those subject to it are from a large disparate population. This is why organized oversight schemes for any compliance standard provide mechanisms for interpretations of the standard(s) in less common cases, and the PCI DSS goes one step further in allowing the specification of “compensating controls” when an organization is not able to directly meet the requirement in question.

In all cases, it is imperative that the organization intending to be compliant, and the assessors validating such compliance, consider the intention of the standard, which is to “facilitate the broad adoption of consistent data security measures” that are “applicable to all System components.”

In this report we adopt the notion that the term “computer” does not necessarily define the physical boundary of a machine, but frequently is a logical definition of a secure processing unit with well-defined boundaries and that a “system” is the sum of the various components, including physical devices, virtual machines, software, and people performing the task considered (for example, the storing, processing and transmission of cardholder data).

2.4 The Benefits of the LCS Paradigm

In the following reproduced list, Mike Kahn in [31] summarizes some of the key points showing the benefits of a centralized solution. He states:

1. Widely-distributed systems and data stores usually bring many security implementation and compliance complications, when compared to fewer (more centralized) systems and data stores.
2. Using different security solutions (on different platforms) adds greatly to the complexity of achieving coverage (in general), in effectively managing the ongoing effort, and in auditing the processes and systems.
3. A centrally-controlled security solution, therefore, may be the most effective and the most efficient solution.
4. The security solution is the most important application in the enterprise. The highest availability is required for the security solution. Additionally, it is critical for all important business activities (for example, you can’t process credit card transactions without adequate security) plus you can’t run any applications if you can’t isolate applications from inappropriate users and also isolate confidential data from “wandering” applications and unapproved users.

3 The LCS Cardholder Data Environment

The System z environment is well suited to hosting and maintaining cardholder data (CHD) since System z is designed from the ground-up to provide a secure system and network. IBM is focused on the payment card industry, providing hardware, software, and services that provide an end-to-end solution for the processing of card payments [33]. The System z technologies provide an extensible solution that can be configured securely using a variety of storage and network technologies. The PCI DSS is a fairly rigorous standard requiring attention to appropriate configuration to secure the CHD environment. Since a common configuration in an LCS is clusters, this section includes Sysplex clusters and their impacts on PCI DSS compliance and related assessments.

Figure 3.1 represents a sample configuration which could be found at almost any organization's environment that is being assessed for PCI DSS compliance. The web server acts as an interface to the outside world for the applications accomplishing credit card transactions. The database server stores product data and often CHD as well. This typical configuration shows separation of the applications onto separate servers which are behind firewalls.

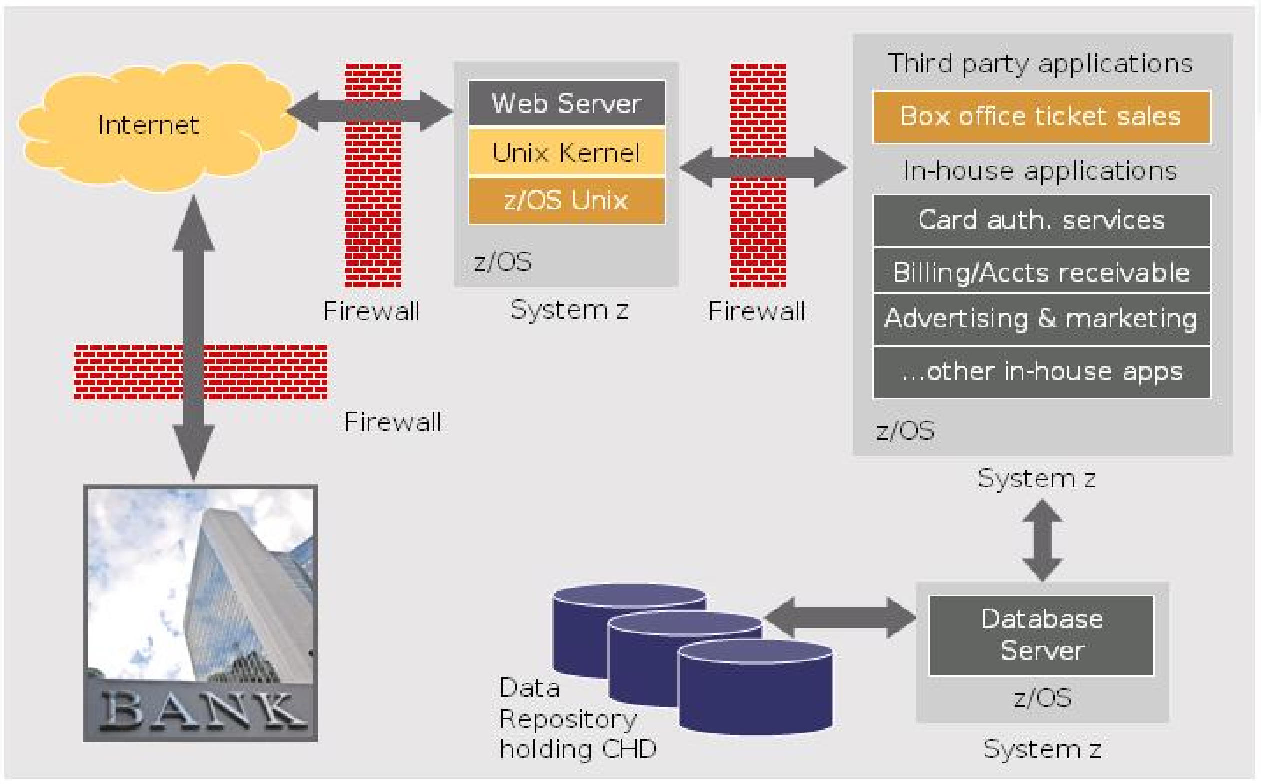

Figure 3.1 A typical PCI DSS configuration

This configuration meets the networking and firewall requirements specified in PCI DSS section 1. The firewalls cordon off traffic between the different server types, and the CHD repository is insulated from direct access by general internet users/consumers.

In an LCS, the configuration may be slightly different as well as the variety of options available in System z deployments which need to be evaluated for PCI DSS assessments. As an example, in System z it is common to create one or more clusters to create a workload sharing system that can easily be extended as demands and business grow. Parallel Sysplex® clusters are discussed in section 3.2.2. System z also supports multiple operating systems including different versions of z/OS, TPF, SUSE Linux, and Red Hat Linux. These operating systems are discussed in section 4.1.4. There are also different communications technologies available, for example, ESCON®, FICON®, Infiniband, and TCP. Some of the communications technologies such as Infiniband are typically dedicated to server-storage networks, so those technologies will be covered in the following chapters. Figure 3.1 will be revisited later in this chapter, focusing on some of the specifics for LCS environments.

With all the hardware and software variety, IBM has developed an infrastructure for flexible, yet secure, configurations intended to provide a framework for PCI DSS compliance [35] [24]. In the rest of this chapter, the CHD environment will be explored with respect to some of the underlying technologies involved in an LCS. Specifically, the management environment, the communications technologies, and Parallel Sysplex clustering will be discussed with relevance to PCI DSS implementations.

3.1 Development and Management of an LCS Environment

The LCS environment is well-defined and has evolved over many decades of investment in security-rooted initiatives, including Common Criteria certifications of System z at EAL4 [11]. With these certifications, IBM has developed a stringent product process for development and release of System z. The result of this effort is a highly tested set of products. To accompany the products, IBM has also developed a large set of training materials and courses intended to arm system administrators with the right tools to setup and maintain LCS environments [12]. From a PCI DSS perspective, QSAs assess system configuration and also the ability to maintain the system in a rigorous fashion. IBM's focus on delivering robust, certified LCS components enables an environment that can be used to establish a secure and manageable solution to host critical CHD environments.

An LCS environment is only as secure as its basic components. Naturally, System z provides support for the native z/OS operating system. One flexible option for defining and maintaining security policies with z/OS is Resource Access Control Facility (RACF®). For more information about RACF, see Section 6. Additionally, SMP/E on System z provides robust change control and configuration management capabilities.

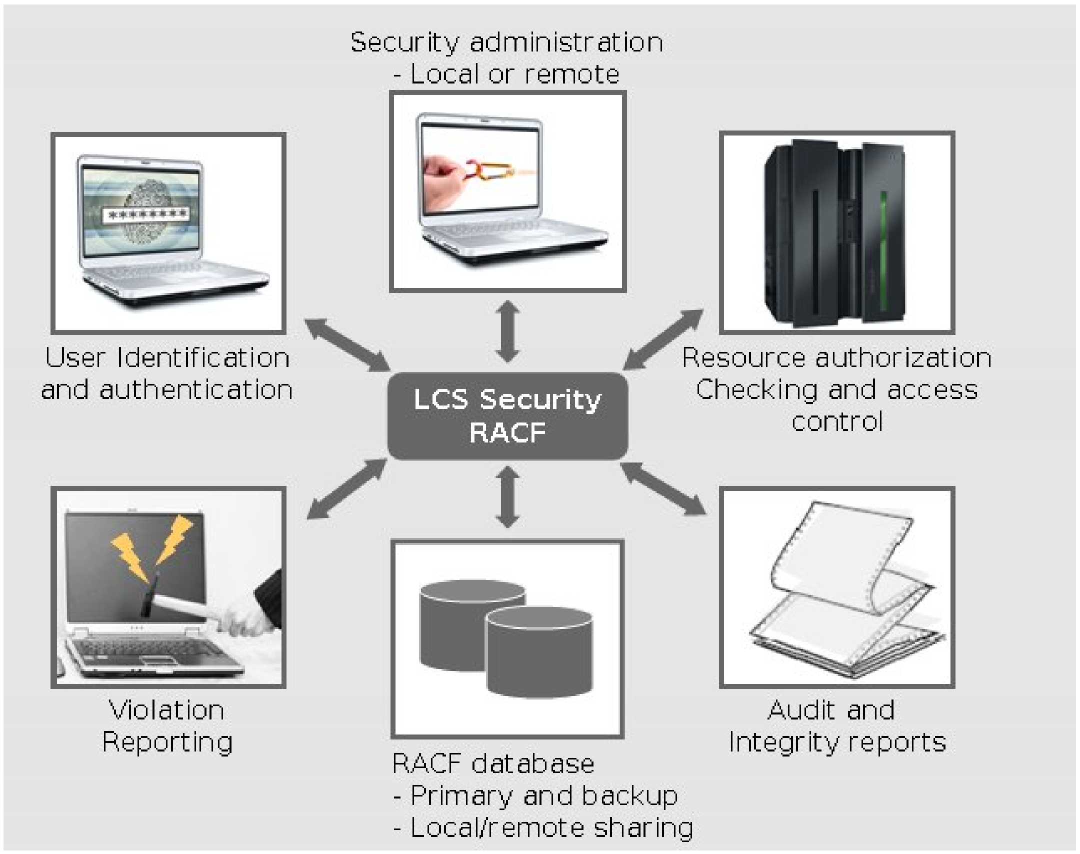

Figure 3.2 IBM's Resource Access Control Facility (RACF)

While RACF is an add-on feature, it or an alternative product is present in virtually all System z installations. Its primary function is to manage users, groups, and their access rights to resources and enforce those access rights when users attempt to access resources. This user to resource mapping can also be done on a group level. User access to a resource is authenticated against the access defined by management to that resource. The resources can include hardware, logical hard configurations (such as network and storage affiliation), and software. Since RACF is an enablement technology working at the operating system level, programmatic access via APIs are provided so that applications can access the technology directly [49] [17]. Transaction logs and reports are available to monitor events and attempts to access the resource. RACF holds information about users, groups, hardware and software resources, and access authority profiles. From a PCI DSS perspective, an LCS environment which is using RACF for access control is easy to audit. Because RACF has been around since 1973, it has been enabled with many of IBM products including CICS®, WebSphere Application Server, DB2, Lotus® Notes, and Novell Directory Services (NDS). As an example of this enablement, a user ID established during application login (for example, Lotus Notes) can be mapped to a RACF user ID with the proper authentication. Subsequently, this RACF user ID can be used to access other system resources such as content in data sets that rely on RACF authentication. Please refer to section 6.1 for more information about RACF and how its capabilities can be used to the advantage of security in an LCS environment.

Since many LCS environments are being maintained by a trusted set of trained systems administrators with separation of duties, one can develop a higher level of assurance that configuration changes can be properly implemented over time. For example, as IBM releases fully tested hardware or software patches, the maintenance process to move new code into production is more easily accomplished with seasoned administrators. They are more likely to comply with defined onsite testing policies prior to moving the patches into production.

Similarly, more seasoned and highly trained LCS administrators and managers will have segmented test/QA systems into separate networks or even separate physical locations to ensure robust change management practices. This separation between test and production environments is typically found in more mature environments, where conservative patch and upgrade management is the norm. In some environments it may be possible to find interns administering the environment, but this is typically not the case in an LCS environment. Since LCS is often used for mission critical production work, the requirement to retain well-trained systems managers is common.

Even further, such training on LCSs is typically extended to the software development team. IBM's training courses extend to the development team as well, and this training includes practices and recommendations on programming secure solutions [12] [49] [19]. Extensive training leads to more awareness of information security practices during application development. It also leads to better development of software lifecycle processes, which result in easier compliance with PCI DSS section 6 which focuses on secure system development and maintenance. By dovetailing development training and systems administration training together in one set of cohesive course material, IBM enables secure and maintainable deployment in LCS environments.

A well-developed LCS needs regular monitoring. LCS implementers can deploy complete audit and reporting of configuration changes using IBM Tivoli Security Information and Event Manager (TSIEM) [48]. This product is part of the IBM Tivoli Compliance Insight Manager (TCIM) suite.

TSIEM has a variety of features, including:

- Monitoring of privileged users activities on LCSs, applications, and databases
- Coalesces and formats native log data into intelligible formats
- Ability to create custom reports for compliance with security standards
- Integrates with other IBM Tivoli management solutions

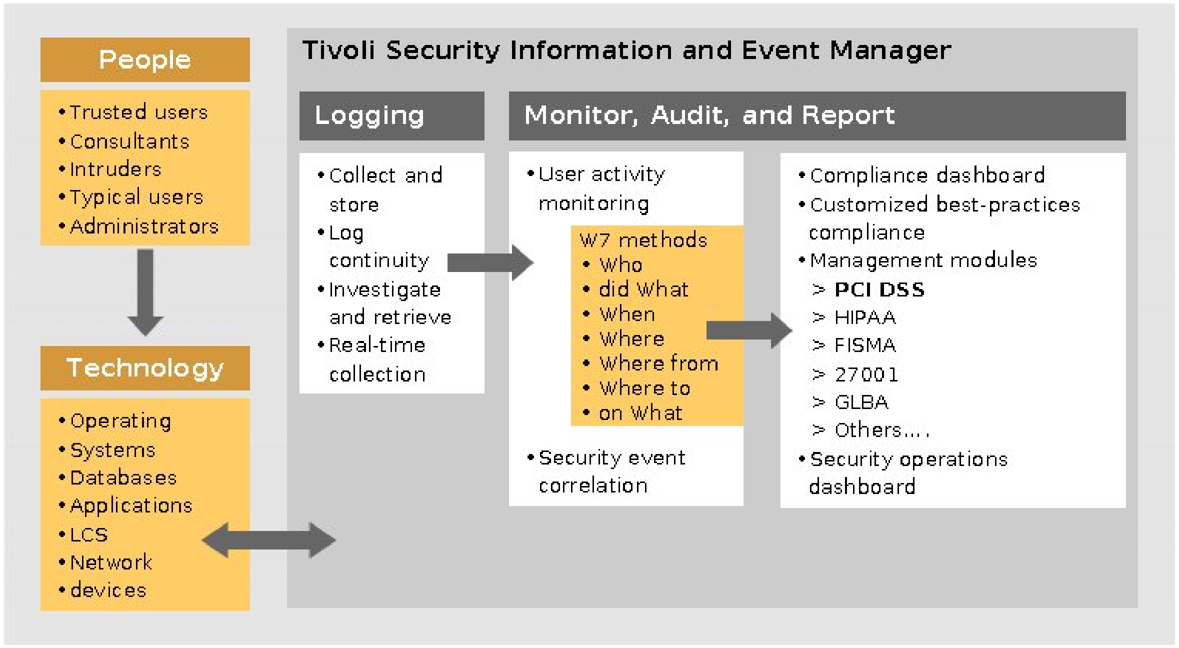

Figure 3.3 IBM Tivoli Security Information and Event Manager

An even more applicable solution is found in the plug-in modules available for TSIEM. These modules were developed to manage compliance activities associated with different IT standards. Of direct interest here is the IBM Tivoli PCI DSS Management Module (plug-in) which is focused on providing compliance in PCI DSS environments [20], enabling the capability to comply with PCI DSS requirements in section 10 concerned with auditing, audit trails, forensic capabilities, and log archives. The Tivoli plug-in module can be configured to accomplish auditing, alerting, compliance reports, and event archiving. It can audit privileged users' access to an LCS installation. It is much easier to use a tool to search in log files than to accomplish this task manually. TSIEM and the PCI DSS module can collect information over time and provide reports on those events. This allows for general analysis and also detailed forensic investigation on suspect activities. For the savvy LCS environment which has these products enabled, security of the general environment is increased. And compliance with PCI DSS becomes a much easier task to implement and maintain over time.

3.2 Communication Technology in an LCS Environment

A key underlying communication technology found in LCSs is TCP/IP. As discussed in the previous section, RACF can be used to set up and protect user/group to resource mappings. A RACF policy can also be applied to permit or deny users or groups of users' access to a TCP/IP stack, specific IP addresses or communication ports [14] enabling a much more granular network access control than typically employed by firewalls. Since this access control works at the operating system level, basic network access is controlled in a secure and easily maintainable fashion. Standard firewall functions like rule-based packet filtering are also part of an LCS. Since RACF enables logging and auditing, attempts to access the network can easily be reviewed.

At a control level, access to TCP/IP system administration commands is critical. The ability to start, stop, or modify network paths is important but should be provided on a restricted need-to-access basis. RACF controls are sufficiently granular to protect command level authority of TCP/IP stacks. RACF controls also include the ability to control access to IP Security (IPSec) and other network management activities. This is relevant to PCI DSS since access controls to the LCS configuration should be limited, and also maintainable. In particular, using strong access controls is dictated in PCI DSS requirement 7 so that privileges are only granted to those who need them. It bears repeating that RACF is auditable; so all changes to the network are recorded and available for investigation as needed.

In LCSs, security at the transport layer is commonly provided by Transport Layer Security (TLS) and Secure Sockets Layer (SSL). These protocols enable data flows to be encrypted after the identities of the client and server are established. From a PCI DSS perspective, it is useful to see proper configuration and maintenance of TCP/IP. In Figure 3.1, the representative LCS configuration, the proper ports for outgoing and incoming traffic must be secured, and access to that configuration maintained properly. Figure 3.4 shows the maintenance of the configuration for these stacks.

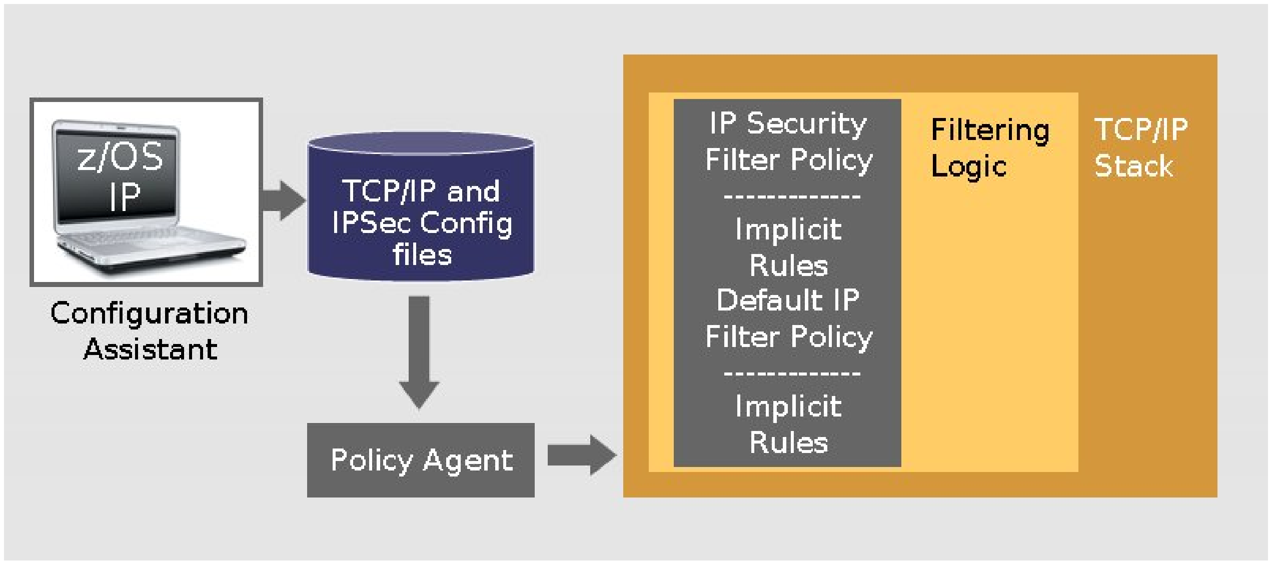

Figure 3.4 Policy Agent and communication stack

In Figure 3.4, the Configuration Assistant is used to maintain many of the configuration settings in the TCP/IP stack. The Policy Agent (PAgent) on z/OS is a component that applies and manages how policies from the configuration are applied to applications and users [14]. The Policy Agent essentially implements policy-based networking. The agent affects IP filtering, IP security, Network Address Translation (NAT), and Intrusion Detection Services (IDS).

z/OS also supports Application Transparent TLS (AT-TLS). AT-TLS is a protocol for transparently applying encryption between two applications using TCP-based protocols that themselves have no integrated encryption functionality. This additional protocol layer makes it easier for application developers to incorporate TLS. Using AT-TLS, application developers do not have to care about the SSL/TLS protocol functions to authenticate the communication between applications, establish a session key, and encrypt the communication traffic. The integration of the security provided by the SSL/TLS protocol is fully transparent to the application. While it is not shown in Figure 3.4, AT-TLS is also managed by the Policy Agent.

Because it handles a lot of responsibility in the system, the Policy Agent (and associated network configuration) in an LCS is a nice tool to centralize access control to critical resources. Since it integrates with RACF, this provides the potential to develop a secure facility for deploying and controlling the network stack. The Policy Agent in combination with RACF provides a facility for securely hosting a CHD environment once the proper policies and ports are defined. In a PCI DSS assessment, on an LCS configuration, it is invaluable to have the z/OS TCP/IP reference manual handy [14].

One of the components included with z/OS is the Communications Server, which is a technology for enabling application communications across a network. The z/OS Communications Server includes Intrusion Detection Services (IDS) to inspect inbound and outbound network traffic. The IDS performs scan detection, attack detection. The resultant traffic regulations are based on the established policies.

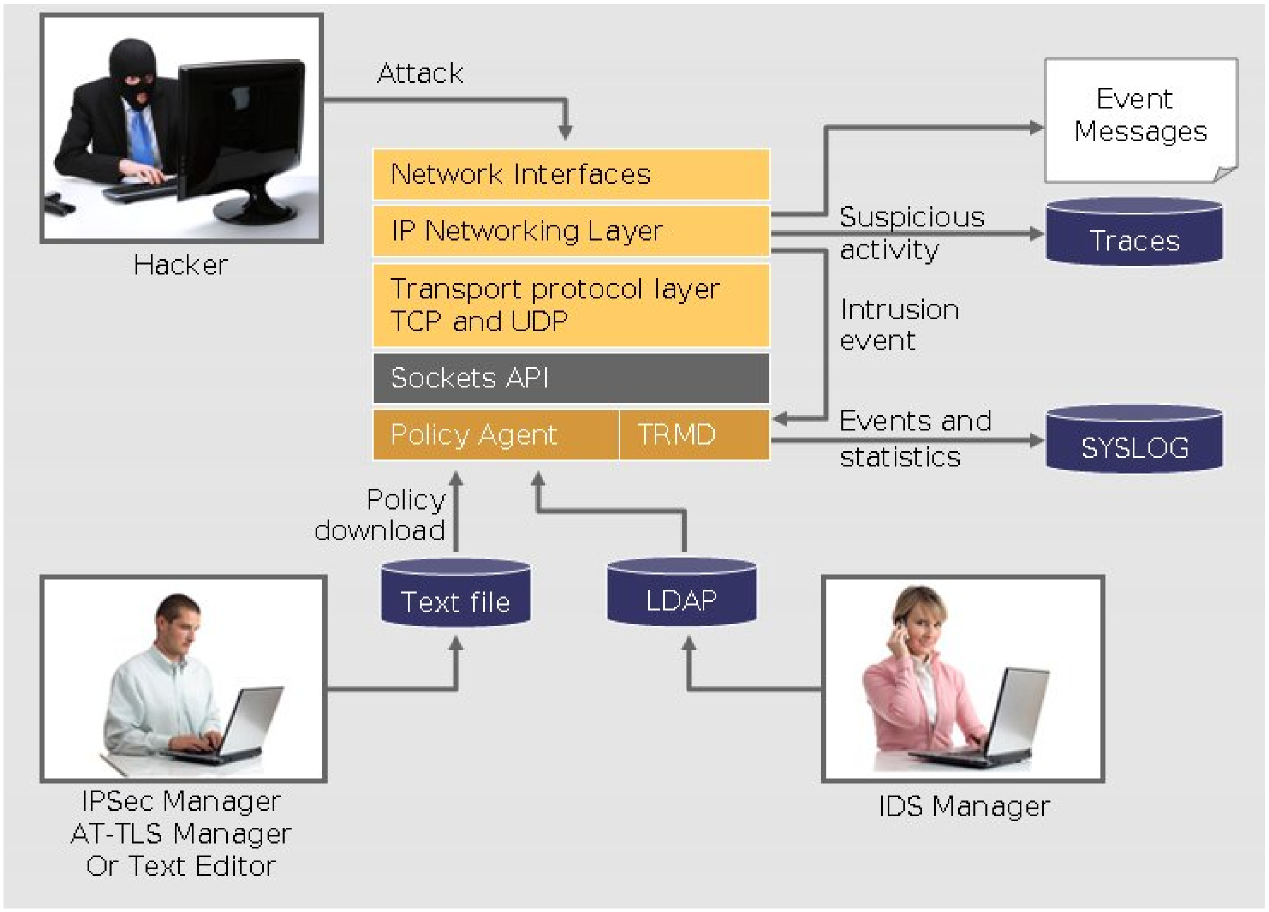

Figure 3.5 Z/OS Intrusion Detection Service (IDS)

The IDS policies can be stored in flat files to obviate the need for an LDAP server, or the policies can be stored on an LDAP server and migrated to the Policy Agent. In effect, the IDS policies become part of the Policy Agent since the Policy Agent is a centralized mechanism for managing network policy on z/OS. As with other IDS's, scan policies can be established to monitor both slow scans and fast scans. Attack categories are defined with the objective of excluding valid users trying to use the LCS. IDS policies are implemented using a flexible GUI, and that may be a useful tool for PCI DSS evaluators during assessment. The fact that z/OS comes bundled with an IDS highlights the fact that IBM is focused on security with their LCS. The bundled z/OS IDS furthers the argument for LCSs hosting a CHD environment.

The IDS notification methods allow events to be displayed on the system console and logged to syslogd. Since message flooding can occur, IDS configuration allows intervals to be setup for messages so that only one message every 5 minutes would be logged for example. Part of the review for PCI DSS assessment should include a review of IDS policies and notification methods.

3.2.1 Data Input Methods

When different input methods are considered, the most popular form of input is currently through the web. LCSs support a natively hosted web server as discussed in section 5.1.2. It is typically the case for the web server to use port 80 and port 443, where 443 is used for SSL traffic, so the web server configuration should be reviewed. The administration of the web server needs to be assessed to ensure access control and authentication is restricted to a limited number of individuals.

If dynamic web page content is used, then an application server is part of the scope of the PCI DSS assessment.

A good example of an application server in this configuration is IBM's WebSphere® Application Server for z/OS (aka WebSphere) which conforms to the Java 2 Enterprise Edition (J2EE) specification. This conformance allows applications developed on J2EE to be portable to other environments. WebSphere also enables connectivity to DB2, CICS, Lotus Domino®, and other applications [6]. In addition, WebSphere is integrated with RACF. When assessing an LCS environment with WebSphere, the RACF configuration associated with WebSphere needs to be reviewed, as well as a review of the integration with other applications. Application development using WebSphere or other application servers should comply with defined procedures. This is discussed in section 5.1.3.

In a Parallel Sysplex cluster, WebSphere can be configured with multiple copies to ensure high availability. The objective of this configuration is to allow one or more copies of WebSphere to always be available to accomplish transactions. If this configuration is established in the LCS environment, it is probable the WebSphere copies have the same integration configuration and data access configuration. a QSA's assessment should include a review of each WebSphere application instance to ensure this is the case in this particular configuration. The next section covers Parallel Sysplex in more depth.

z/OS supports SSL/TLS, VPN, and SSH. Per PCI DSS requirement 2.3, the QSA should review compliance with PCI DSS for web-based management and non-console administrative access. Associated access to the facility should also be reviewed with regards to PCI DSS requirements in section 2.3. For example, consoles in the CDE should be locked to prevent unauthorized access if an individual has gained access to the facility.

3.2.2 Distributed Systems Using Clustering Technology

In large commercial transaction oriented systems, the combination of attributes such as high availability, optimized use of processing resources, reconfiguration and addition of hardware (processing capabilities as well as storage), together with application transparency and centralized management, are becoming more and more important. This all needs to be provided without compromising the security and integrity of the data processed.

Single, centralized computing systems are usually not able to provide all those capabilities at the same time. High availability is often limited by several single points of failures like single power supplies, shared bus systems, or shared devices. Optimized use of processing resources, like multiple processors, often requires the application to balance the workload unless several parallel applications are started on the same system. Several parallel applications on the same system either need to be highly decoupled or require their own fine-grained method to serialize the use of shared resources.

Many server operating systems were originally designed without support for high availability and load balancing between multiple processor cores. Support for those features has later been added. Quite often the performance gain in those systems does not scale linearly with the additional processors added.

Distributed systems, as a combination of multiple single systems connected via fast communication links, may overcome this problem, but the network communication link between those systems often becomes a bottleneck, especially when the number of individual computer systems in the distributed system grows. Optimizing the distribution of workload between the individual systems is a hard issue to solve. Even harder is the maintenance of a harmonized configuration of all the elements of a distributed system. In addition, the synchronization of the timers of the individual systems as well as the system wide synchronization of access to resources poses other major problems. Therefore most commercial solutions for distributed systems implement only a subset of the attributes mentioned in the beginning of this section. Some provide high availability only; others are specifically designed for a dedicated application like a specific database and therefore can only be used by general applications built on top of that database. Some provide high availability for storage only.

In order to address all the attributes already mentioned, one often has to use a combination of different products from different vendors resulting in major difficulties to find the right way for their integration and management. Identifying the cause of a failure in such a distributed system can become a nightmare and any re-configuration has the risk that one of the many products used may be misconfigured resulting in potentially critical effects for the operation of the whole distributed system.

3.2.3 z/OS Parallel Sysplex in a PCI Environment

Only a few architectures provide a combination of all the attributes mentioned in the beginning of this section, with the System z mainframe providing probably the most integrated and elaborated solution. A System z mainframe capability called a “Parallel Sysplex” allows combining multiple System z processor complexes to operate virtually as a single system, sharing disk space, and distributing the workload over all processor complexes. In such a configuration, the individual processor complexes are interconnected via a “coupling facility”, which is much more than just a high speed communication link between the processor complexes. The coupling facility is a separate system that handles the synchronization of the timers of the individual processor complexes, the management of common caches for all processor complexes, handling of messages between processor complexes, and the Sysplex-wide synchronization of access to resources. Several components of z/OS can make use of those facilities to distribute the workload, centralize system logging, synchronize configurations, and handle failover conditions in the case of failures or maintenance within elements of the Sysplex. IBM’s transaction management system, CICS, and IBM’s relational database management system, DB2, are two examples of components that make explicit use of the Sysplex capabilities to enhance performance and availability. Also the security subsystem RACF uses Sysplex capabilities to propagate changes to the RACF configuration as well as any changes to the user database or access control information automatically and immediately to all members of the Sysplex.

This is very transparent to applications executing in such an environment. A system programmer can monitor the use of processor capabilities and resources and adjust the parameters of the workload manager and other components to optimize the workload distribution and resource usage. He can also centrally configure and manage most of the system’s parameters or react to problems. With the ability to have a “geographically dispersed Parallel Sysplex™” (GDPS®), an organization may even ensure continuous operation in the case of a major restructuring/maintenance or a major disaster that happens to shutdown one complete site. Even in this case, the data is still available and the applications will continue running. The overall throughput may be affected due to the missing element of the Sysplex cluster. Restarting this element later can be done without affecting applications running within the Sysplex.

In IBM’s Parallel Sysplex, a cluster is one or more operating systems (images) coupled together to act as a single unit. The individual operating systems that are clustered, can be located on the same physical CEC, or they can be located on different physical CECs in a Sysplex cluster. The rationale for coupling operating systems images together is to provide more reliability and availability. Once a Parallel Sysplex is established, the environment no longer has the potential for a single point of failure. Failover technology allows one OS-image to take over for a failing OS-image. An application can be setup to failover (as needed) to a healthy operating system image if the application is experiencing hardware/OS issues. This is called cloning, and the failover of one or more applications can be accomplished automatically if desired.

A typical Parallel Sysplex is shown in Figure 3.6.

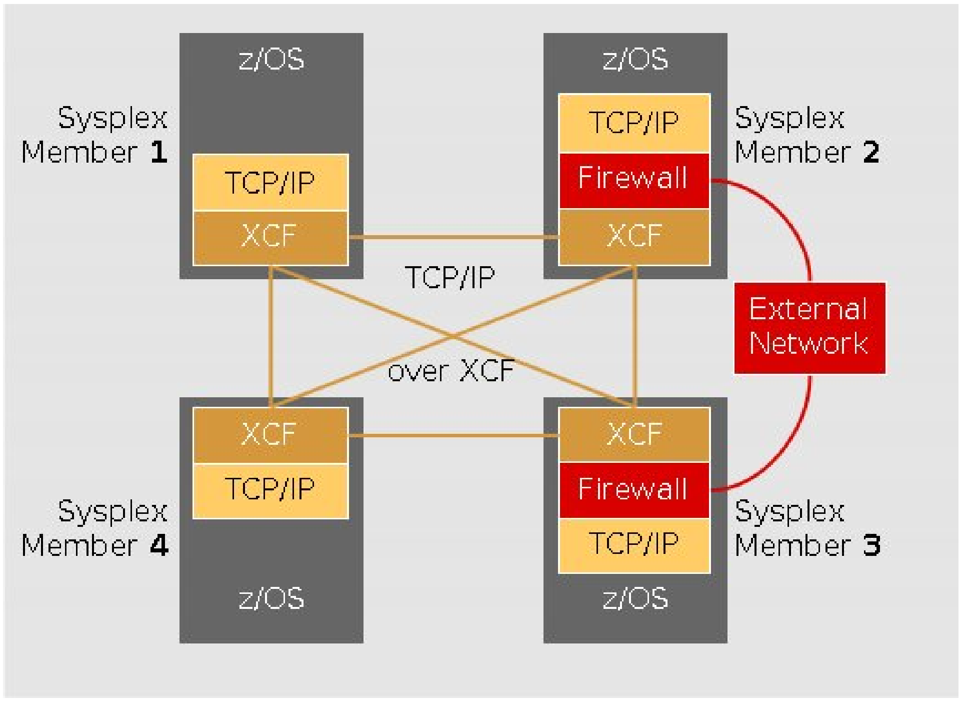

Figure 3.6 IO over XCF in a System z Parallel Sysplex

XCF is IBM’s proprietary protocol Cross Coupling Facility (XCF). XCF is the z/OS component which connects each of the four servers together to form the Sysplex (cluster). Applications within each of the operating systems communicate with each other through XCF. When issues occur on one of the Sysplex members, XCF status can be used to drive policies such as application restarts, or application transition from one Sysplex member to another Sysplex member. Since XCF provides the connectivity between the systems, if XCF has a problem on one or more Sysplex members, then part or all of the Parallel Sysplex is effectively disabled (as a Sysplex). With regard to security it needs to be kept in mind that XCF is a mechanism to ensure data consistency and computational operations within a Sysplex cluster. It is not a generalized data communication channel – unless explicitly instructed and instrumented so (see below). Since XCF is designed for sysplex communications, application development should be restricted to use this channel only for sysplex related traffic.

In Figure 3.6, the TCP/IP communications are occurring over the proprietary XCF protocol, and the communications are within the same CEC or datacenter. To simplify the figure, the hardware technology used for connecting XCF instances is not shown. IBM has different hardware options available for physical coupling facilities including InterSystem Channel-3 (ISC-3) and Parallel Sysplex using Infiniband (PSIFB) which both physically use fiber for connection termination.

This XCF network communication is considered to be an external data bus. In Figure 3.6, the firewalls that are indicated on servers #2 and #3 can be used to communicate with other networks. There is a distance limitation to this configuration so Sysplex clusters of this type are usually housed within the same physical room or building.

The data, applications, and configuration information in the cluster gets shared within the Sysplex. Many LCS environments enable RACF to centrally define and enforce a security policy into their Sysplex in order to simplify permissions and access controls. When RACF is configured, the master node in the Sysplex propagates the constraints and commands to designated peers in the Sysplex. This means that one (1) centrally defined and managed policy is active on every node in the Sysplex. Every data object in the Sysplex cluster is under control of this policy; every user and every program is subject to this policy.

There are some LCS configurations where even more stringent failover requirements exist. In these cases, the physical servers are geographically separated into separate buildings or cities. IBM's technology for this physically separate clustering for System z is called geographically dispersed Parallel Sysplex (GDPS) [9]. IBM offers a spectrum of solutions in this space to provide flexible configuration options. Initial setup and configuration is typically a joint effort between IBM and the LCS implementer. Actual installation only occurs after extensive multi-site datacenter and networking considerations have been planned. In some cases, networking is established with private encrypted networks. In other cases, IPsec networks are setup with dedicated bandwidth allocations to handle potential failovers.

IBM has dedicated a book to focus on Sysplex configuration, with extensive discussion on security configuration and how to properly exploit the Sysplex technology with other related System z products [47]. From a PCI DSS perspective, the IBM Parallel Sysplex technologies provide a reliable and available solution in LCS environments.

3.2.4 Change and Maintenance

It is easy for LCS administrators to keep abreast of hardware and software issues with IBM products via the Red Alerts subscription service [22]. This service allows for subscribers to receive notifications via email about important support issues, including security issues. The alerts come in different severity categories, and the website also maintains online information showing archived issues for hardware, operating systems, systems management software, and applications developed by IBM for LCSs. This is relevant to PCI DSS section 6.1 which requires subscription to hardware/software issue alert services. This also makes it easy for LCS administrators to comply with that requirement.

Once required changes are identified, they need to get tested and put into production. IBM's Hardware Management Console (HMC) is a secure means of introducing hardware or logical partition configuration changes to an LCS environment [8] [16]. Each HMC is delivered as a single purpose, closed system upon which no other applications may be loaded. This is pertinent to PCI DSS since secure systems management is required. Since many LCSs are high-uptime configurations, IBM developed a solution to inject updates without disrupting operations. The HMC is a standalone laptop system which cannot have other applications installed on it – so it is dedicated to change management and support applications. Since an HMC can be directly connected to the servers it administers, it can be used without networking. In Figure 3.1, this means an HMC can be physically connected to different servers in the diagram. This is pertinent to PCI DSS since secure systems management is required. Since many LCSs are high-uptime configurations, IBM has developed a solution to inject updates without disrupting operations.

In order for the HMC administrator to effect changes, good security policy is required to enable or disable their access to the affected hardware or software that is being updated. It is also important that if a single HMC is used, that it does not create a communications path between CHD environments and non-CHD environments.

A single HMC can be configured to support multiple System z servers and associated equipment. The primary objectives of the HMC are:

- Configuration management such as server or partition configuration
- Change management for hardware/partition configuration changes
- Problem management using debug data and hardware error codes
- Operations management on server and partition controls
- Service management including concurrent repair procedures

One purpose of the HMC is to insulate System z from the internet and other intranets during system maintenance. The result is a change control environment that can be securely configured, inline with the PCI DSS specifications (section 6) on maintaining secure systems.

4 LCS Architecture

Much in this report has been said about the higher level concepts of software running on System z computers. We now step back, and look at what is provided by the underlying hardware and lower-level devices.

The inner core of the System z computers, the CPU, conforms to the z/Architecture®. The z/Architecture is a well defined set of CPU instructions and functional properties which is provided uniformly throughout all System z computers. It provides the basic groundwork for implementing secure systems such as:

- Different processor modes, privileged and unprivileged, to separate certain privileged CPU instructions from unauthorized use
- Memory separation and virtual memory provisions, including several modes of address translations
- Provisions for efficient I/O
- Provisions for complete virtualization

The outer core of System z computers is a software layer that enables various levels of virtualization, partitioning, and separation of one physical entity (one CEC). This layer is called the Processor Resource/System Manager (PR/SM) and is the base hypervisor provided by every System z computer.

Further partitioning virtualization and separation can be provided by means of using the z/VM operating system.

The following sections will provide more insight in these topics.

4.1 Virtualization

Virtualization can happen at different layers in a system and it can have different scopes. In System z systems, virtualization is based on core CPU features (the “SIE” instruction [30]), firmware (the PR/SM hypervisor), as well as operating systems that support or implement virtualization technology (z/VM). Virtualization is an intrinsic feature of a System z computer. System z computers can provide full virtualization of computing resources such as CPU, memory, and peripheral devices such as disks or network interfaces.

An operating system or an application running on a System z computer is usually not aware it is running in a virtualized environment. However, each device it accesses, each memory access it performs, or each CPU instruction it issues, is in fact under full control of the underlying hypervisor. The virtualization provided by System z computer is full-platform virtualization (or Type-1), which means that it is possible and even supported to start other hypervisors in a virtualized environment. For example, a configuration where a z/OS system is running as a guest on an instance of z/VM which in turn is running on a partition provided by PR/SM is very common in LCS environments. The z/OS system does not have to be aware of this type of configuration – in fact it does not care about it. Both PR/SM and z/VM are so called Type-1 hypervisors. In the case of PR/SM, the performance loss when executing in a partition is zero compared to a system running natively. In the case of z/VM the typical loss of performance is negligible for most workloads.

The resources assigned to a partition or guest are under full control of the underlying hypervisor. That means that the hypervisor enforces the implemented guest or partition configuration. A device assigned to one partition can not be accessed by another partition unless the hypervisor permits it. That is even if the device is physically connected to the same box – maybe even to the same cable trunk. The same applies to memory – a partition or guest is not allowed to access another partition's memory unless the hypervisor permits it or the guest permits it by its own means (for example. application protocols, networking).

The virtualized environments can be configured to provide complete separation between the guests or partitions. Conversely, they can also be configured to share devices between different partitions. These basic properties have been subject to formal evaluation at different depths. For example, the PR/SM hypervisor has successfully passed several CC evaluations at an EAL5, and z/VM systems have successfully passed several evaluations at EAL4 (see also, section 6.3.1). To summarize this section: "If it's not shared, it's not there."

Figure 4.1 illustrates how physical devices and networks are linked to the System z hardware. Immediately above the System z hardware is the PR/SM hypervisor layer which controls access to the physical hardware and attached devices. Various operating systems such as z/OS and z/VM can execute within the Logical Partitions (LPARs) provided by PR/SM. Additionally, z/VM is capable of running multiple guest operating systems. All these isolation features, and the event loop, facilitate PCI and DSS compliance availability.

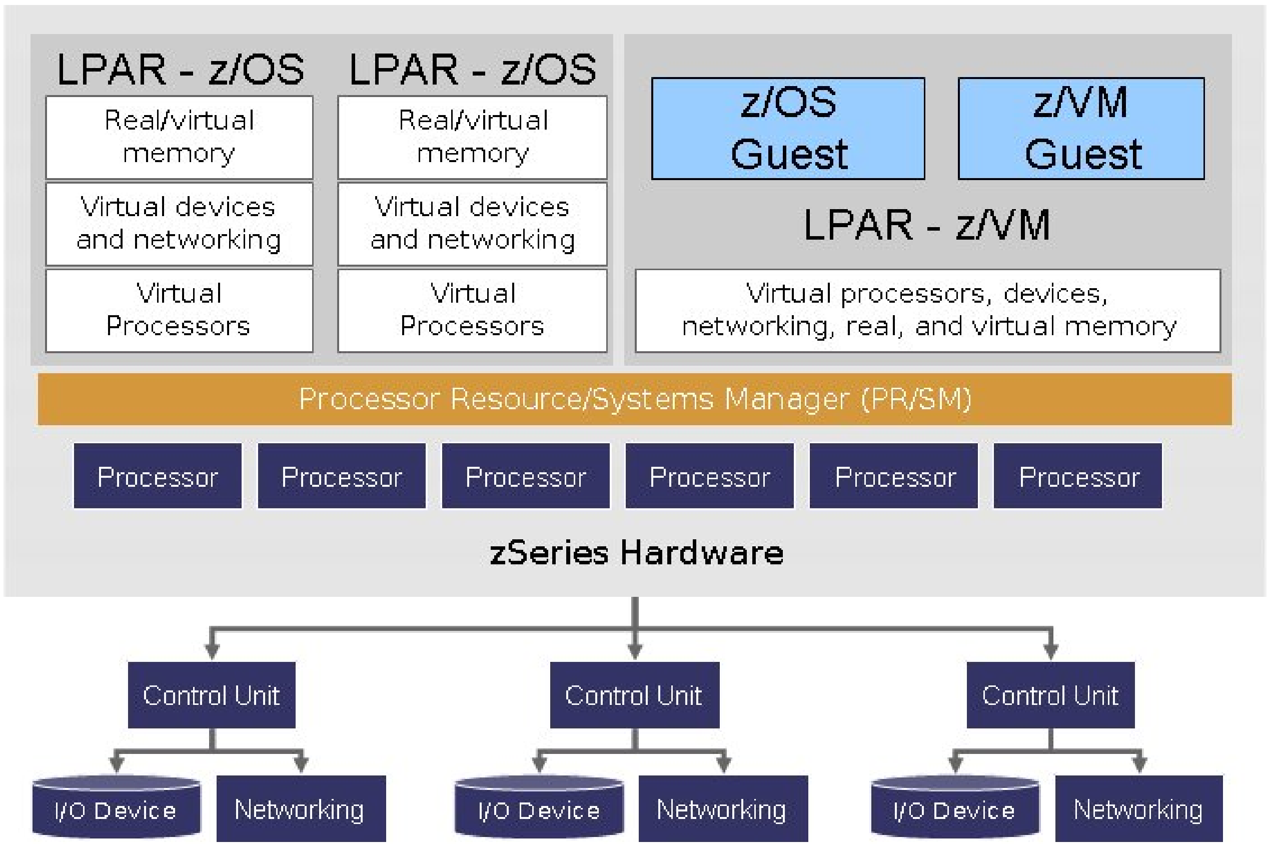

Figure 4.1 System z® Virtualization Architecture

4.1.1 Logical Partitions

A logical partition is a collection of computing resources creating a computing environment conforming to the z/Architecture. Any operating system able to run on z/Architecture platforms is able to run inside a logical partition. This section will briefly introduce the basic features of a logical partition.

A "logical partition" is a structure usually provided by the PR/SM hypervisor (see section 4.1.2). A z/VM (see section 4.1.3) system could provide the "same" environment by means of its own virtualization features. However, z/VM usually uses the term “guests”. The difference between them is mostly in their management. Functionally, they are the same.

The resources assigned to a logical partition are under the control of the underlying hypervisor. The resources governed are:

- The CPU
- The memory
- Devices, for example, Direct Access Storage Devices (DASD or disk), network interfaces, printers, terminals, channel definitions

In addition, the hypervisor can grant certain partitions certain authorizations. These authorizations allow systems running within the partition to access properties of the hypervisor. For example, it is possible for operating systems to access performance data from the hypervisor. However, that is only possible if the hypervisor has granted this authority to the partition running the operating system; and the operating system knows how to access the data.

What has been said previously is also true here, if a resource is not assigned to a partition, the partition has no way of accessing it by the means provided by the hypervisor, of course it might access the resource by higher level means (application protocol between different partitions on the same box) but that is out of scope for the hypervisor.

In general, a logical partition (or a “guest”) is a logical computer system conforming to the z/Architecture. It can only access devices and functions to which it has assigned and granted access.

4.1.2 PR/SM

Each physical system processor complex has two service elements attached that is used by the IBM engineer to control and configure the system for the customer. This small computer, is a closed system, called the Service Element (SE). The SE is an integral part of a System z processor complex. The SE communicates with the hardware master console (HMC) which is used by the customer’s operation staff.

The SE also plays an integral part when it comes to enforcing logical partitions on a System z machine. The HMC is the operations staff’s main interface for the Processor Resource/System Manager (PR/SM), the hypervisor built into every model of present System z computers.

This virtualization functionality is provided by PR/SM. PR/SM is configured and with the Hardware Management Console (HMC) using a specialized software application that allows you to configure and assign resources to a logical partition.

The HMC, working with the SE, provides an access controlled environment to the system’s configuration. The environment provides different roles for the management. So it is possible for the operations department to assign different tasks with different authorizations to the operations personnel.

Using the HMC is the only way to access this configuration. Systems running within the partitions cannot modify this configuration; unless, as already described above, they have been granted specific permissions.

4.1.3 z/VM

The forerunner of the z/VM operating was one of the first commercially available operating systems that used virtualization techniques. The technology z/VM has its roots in concepts that were developed 40 years ago and continue to mature to this day.

With regard to the computing environment it presents, there is no difference to an LPAR provided by PR/SM. Both conform to the z/Architecture. However, when it comes to the actual virtualization of devices and system resources such as CPUs and memory, z/VM is able to virtualize more than PR/SM.

For example, when z/VM implements virtual memory management, it might assign more memory than it has logically available to itself to one of its guests. It might also assign a part of a disk, a DASD, as a complete new exclusive DASD to a guest.

z/VMs virtualization features are much more fine grained than the ones provided by PR/SM.

Similar to PR/SM, z/VM systems have been successfully evaluated using the Common Criteria several times. The separation capabilities of z/VMs virtualization technology have always been the central focus of the evaluations.

4.1.4 Operating Systems

In System z computers, the environment provided by the PR/SM hypervisor is able to run any operating system that can run on a z/Architecture system. For example:

- z/VM
- z/OS
- Linux
- z/VSE
- z/TPF

Each operating system is subject to the resource assignment policy defined in the underlying hypervisor.

4.2 Summary of the Benefits of LCS

The following points summarize the obvious benefits of using an LCS in an environment where virtualization, efficient resource usage, security, and separation are issues of concern or planning:

- System z systems provide many ways and mechanisms to virtualize computing resources
- Every computing resource (CPU, Memory, Devices) can be virtualized on a System z System
- The computing environments provided by the hypervisors (PR/SM and z/VM) are completely separated from other computing environments provided by that hypervisor.

  The hypervisor is in complete control of the actual computing resources and only allows access to assigned resources.
- The separation capabilities of the hypervisors have been subjects to thorough security evaluations (see also section 6.3.1):
  - PR/SM - Common Criteria EAL5
  - z/VM – Common Criteria EAL4

5 Data Organization and Processing

The ability to store and retrieve data from files in memory, disk, cache, and offline storage requires a robust set of technologies to interact with the various devices. LCS environments are transactional, which requires technologies such as CICS to provide applications with interfaces to securely drive transactions from thousands of concurrent users. This section provides a brief overview of these technologies, highlighting their relevance to PCI DSS secure configurations.

5.1 File Systems

For LCS environments, PCI DSS implementers and QSAs have a few technologies with which to be concerned. LCS file systems are available for both native z/OS hosted applications and z/OS UNIX applications. Both sets of technologies have a rich history of development and as a result there are a library of IBM Redbooks® and manuals which provide insight into various configurations. The file systems on System z are also integrated with native operating system security, which can be extended with the addition of RACF. RACF and its role in maintaining system security and integrity is further explained in section 6.1. With regard to files (or data sets as they are called in the mainframe environment), it needs to be kept in mind that access to file system objects is fully governed by the policy defined to RACF. Access to a file system object is only possible if the RACF policy grants the access (the default is “no”). This applies to all supported file systems.

5.1.1 z/OS Native File Systems (Data Sets)

Nearly all interactions with computers end up with I/O to a storage device such as a hard disk. In z/OS, data sets are used to manage records stored in a data set. There are four types of datasets supported natively in z/OS: sequential, partitioned, partition extended, and VSAM [18] [26]. Of these dataset types, the following are the access methods used in z/OS:

- QSAM Queued Sequential Access Method (heavily used)
- BSAM Basic Sequential Access Method (special cases)
- BDAM Basic Direct Access Method (nearly obsolete)
- BPAM Basic Partitioned Access Method (for libraries)
- VSAM Virtual Sequential Access Method (for more complex applications)

Of these access methods, VSAM enables random access to a logical record in a dataset. Access using this method must be enabled in the application for the record lookup to be available. Since the LCS environment has its roots in the beginnings of the computer industry, data and file access includes methods targeted for data tape storage and retrieval. The objective in supporting both legacy and modern access methods is flexibility. In all cases, the application accessing the data must enforce the security constraints that are associated with the data; otherwise the application cannot access the data. A direct example is shown in Figure 3.1 which shows VSAM in an LCS environment.

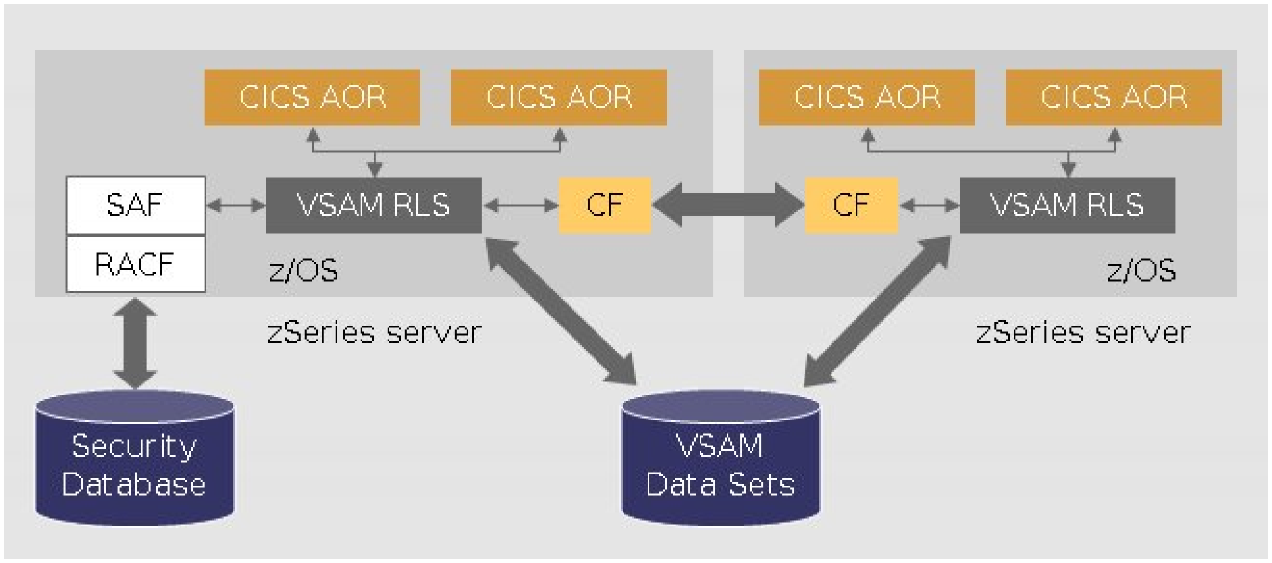

Figure 5.1 CICS and VSAM integration in an LCS Sysplex environment

Customer Information Control System (CICS) is a transaction processing system allowing many users and/or applications to conduct transactions concurrently. Since a transaction record typically results in data that is written to a file, CICS is integrated with the z/OS file system. CICS applications have the flexibility of different data set access methods including VSAM and BDAM. The integration between CICS and VSAM includes methods for authorization and protection of datasets. The authorization methods extend to RACF.

In Figure 5.1, a Parallel Sysplex cluster connects two System z servers running CICS enabled applications. The applications use VSAM to access shared datasets (files) on a group of disks. Record Level Sharing (RLS) is a VSAM function to enable data sharing and synchronization of access on a storage device. As mentioned above, authentication to the VSAM data is required. In this case, the authentication is configured to use RACF.

For PCI DSS, z/OS native file system access is accomplished in a manner that uses standard LCS technologies. This makes the security configuration easy to audit.

Since there are logging options available for the standard LCS file system technologies, this enables forensic analysis if a security event occurs. For VSAM, both read and write data integrity is ensured with RLS. CHD environments using VSAM would require the Sysplex usage constrained to the associated VSAM data sets. RACF can be used to forensically determine all user IDs that are associated with unique VSAM datasets. VSAM RLS logging can be setup in the coupling facility and those logs can also be used for forensic analysis as well. In a distributed system, logging would have to be synchronized across the different application servers. In VSAM RLS, since a coupling facility can be established this centralizes the logging for easier forensic analysis if it is ever needed.

5.1.2 z/OS UNIX File Systems

The objective of z/OS UNIX on z/OS for System z is to provide UNIX system services to host and manage UNIX-based applications [18]. There are mission critical applications which need z/OS UNIX to run, and these include TCP/IP, z/OS web server, LDAP, and Java (JDK/JRE).

The purpose of a file system is to enable a mechanism for accessing and storing files, while abiding by the access privileges associated with those files or folders. As shown in Figure 5.2, z/OS UNIX can flexibly host any or all of several file system types.

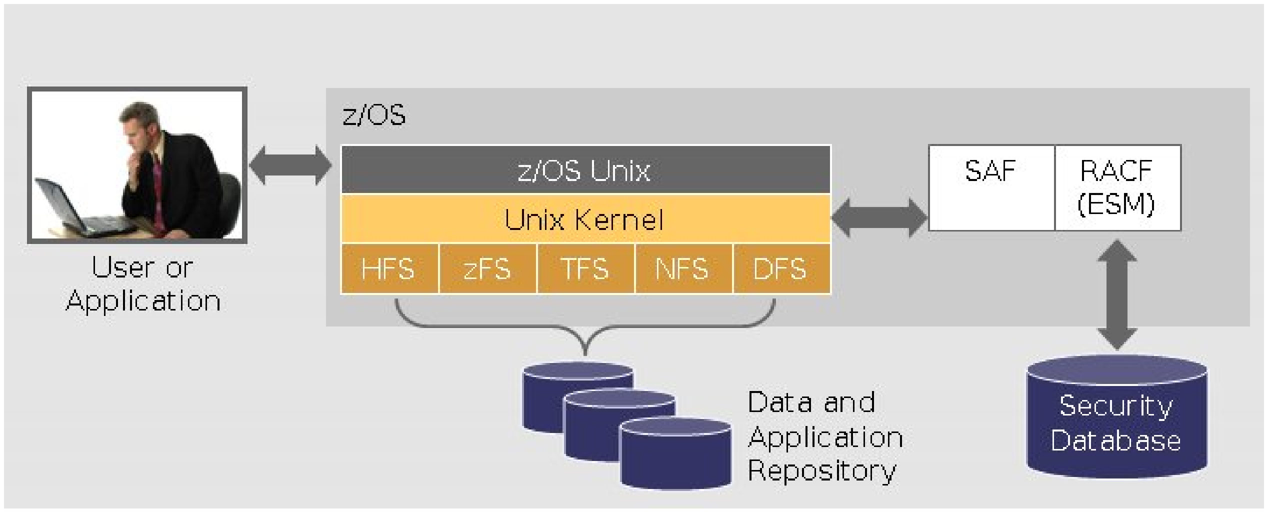

Figure 5.2 z/OS UNIX file systems

The z/OS UNIX file system types include:

- HFS (Hierarchical File System) for direct access storage device (DASD) support
- zFS (z File System) supporting ACLs in addition to permission bits, multilevel security and/or multiple logical file systems
- TFS (Temporary File System) for in-memory-only file system mapping to deliver extremely high performance
- NFS (Network File System) as found in traditional UNIX to access remote file systems via a network
- DFS (Distributed File System) to join local file systems of several different physical machines together to become available to all DFS client machines

z/OS UNIX file systems abide by the standard UNIX user ID (UID) and group ID (GID) allocation and privilege scheme. Delegation of authority can be as encompassing or granular as needed. Since this UID/GID model has been around for decades, it is a mature and secure standard for providing and enforcing restrictions to files and folders. z/OS UNIX implementation of UID and GID permissions is accomplished through RACF. RACF has been enhanced to support the semantics of the standard UNIX permission bits in conjunction with access control lists. Additional file attributes and user privileges beyond those defined in other UNIX implementations exist that allow a finer grained access control. Since RACF is used to determine access also to UNIX file system and inter-process communication objects, access to those objects can also be audited using the RACF audit capabilities.

The z/OS UNIX kernel requires an External Security Monitor (ESM) to resolve file system and directory access [29]. The most well known ESM implementations are RACF, eTrust CA-ACF2 Security from Computer Associates, and eTrust CA-Top Secret Security from Computer Associates.

The net result of this z/OS UNIX file system implementation is that the appropriate configuration of a secure solution is possible for CHD environments, and this in turn leads to a valid and certifiable PCI DSS configuration.

Since the UNIX file systems are compliant with UNIX POSIX standards, they also enable portability between different UNIX implementations. The reliance of the UNIX file system on an ESM such as RACF ties the security to well established LCS protocols and policies, leading to more rigorous implementations.

5.1.3 Data Storage Facilities

The advantage of z/OS is a fully elaborated storage system with hierarchical storage, backup procedures, dedicated storage management, and other procedures to access storage. This provides much higher reliability than other server systems. The feature in z/OS to manage data space is the Data Facility Storage Management Subsystem (DFSMS). The DFSMS component automates storage management so the application programmer does not have to directly control data set allocation and usage characteristics. Since z/OS UNIX is integrated with native LCS technology, the DFSMS features can be utilized in z/OS UNIX. In a DFSMS environment, RACF complements the data storage management. A PCI DSS assessment should include a review of the DFSMS policies to determine if RACF default data classes are in place, and whether Automatic Class Selection (ACS) routines are used to override RACF defaults.

If the TFS file system (as mentioned in the previous section) is used, it is typically applied to temporary files in the /tmp directory. TFS security is similar to other LCS file systems. In a PCI DSS assessment, it is useful to ask if TFS is in use and if the security policies are equally applied to files in that temporary repository.

When considering databases on an LCS, the two major contenders are DB2 and IMS™ DB. Since the content of databases may be sensitive, in some LCSs environments one or more database administrators (DBAs) may be assigned to manage the database(s). In a PCI DSS assessment, DBA access policies should be reviewed to ensure CHD environment integrity is preserved.

5.2 Payment Application Software

At some point, all transactions associated with payment cards conducted on computers must flow from their point of origin to the bank. All payment applications must comply with the PCI DSS requirements, along with the compliance of the whole PCI DSS hosted environment. The payment applications which are developed by vendors and which are sold or distributed to another party must – as currently required by Visa - be assessed to be compliant with the Payment Application Data Security Standard (PA-DSS) requirements as well. The PA-DSS requirements were developed to assess payment applications in order to ensure that they facilitate PCI DSS compliance in the operational environment. The PA-DSS requirements were derived from the PCI DSS requirements, where the PA-DSS requirements are focused on payment application reviews the PCI DSS requirements apply to all system components in a CHD environment.

For a distinction between these two types of applications, consider an extension of Figure 3.1 where some applications were developed in-house and the company has expanded operations.

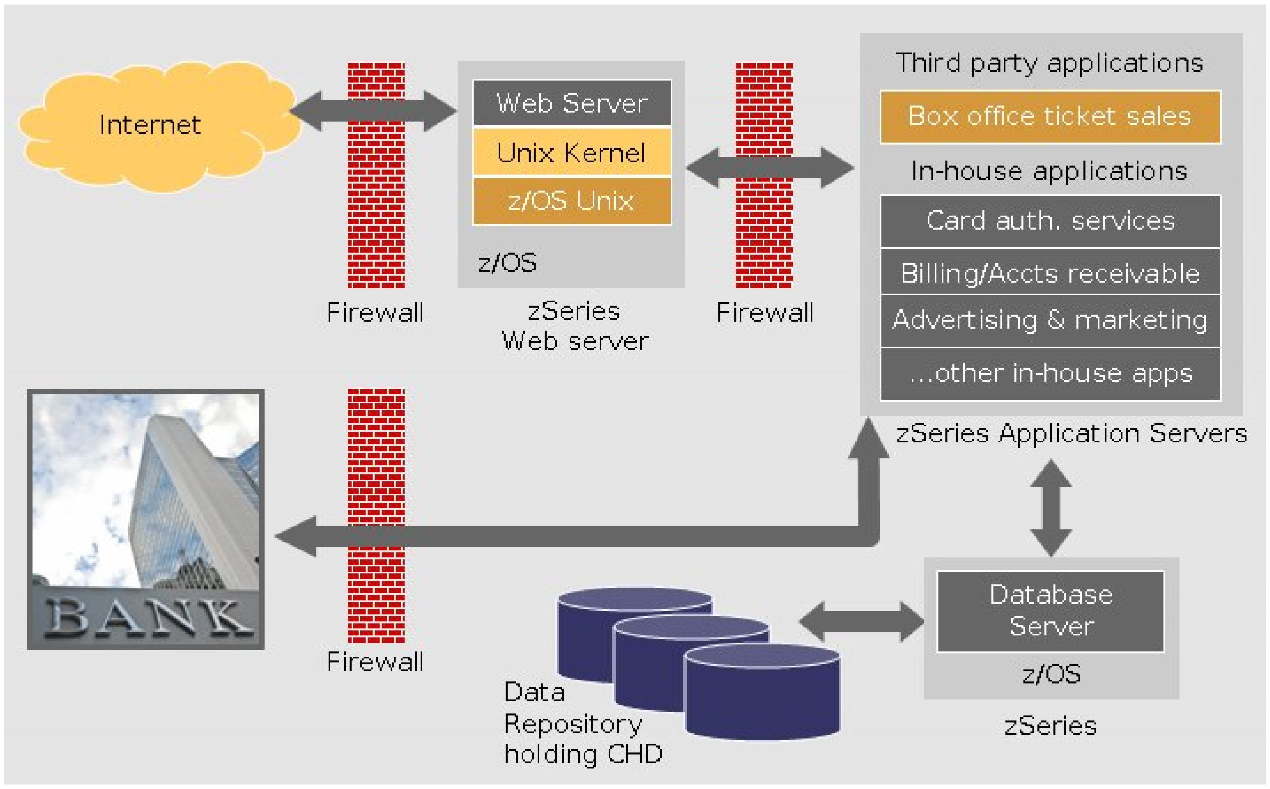

Figure 5.3 Environment with mixed payment application types

In this case, the environment hosts a web server to interact with the vendors on credit card services such as end-consumer authentication. The application servers host applications, which were developed in-house, to accomplish different functions such as generate monthly billing/receivables, or generate targeted advertising based on consumer preferences and purchases. As the business expanded, an externally developed application was added to handle box office ticket sales. The box office application can interact with personnel handling ticket sales at movie theaters, sports arenas, and similar venues.

The two application types reside in the same environment. The third party application can be configured to interact with the in-house applications as needed by the business. *In both cases, the applications must be compliant with PCI security standards.* The environment must be regularly reviewed to ensure that configuration changes do not create security problems. As discussed in section 3.1, the IBM Tivoli PCI DSS Management Module enables the ability to audit access and changes. Additionally, the administrators for the environment are required to monitor potential security issues from existing infrastructure, and from the externally developed application. For the IBM products, LCS administrators can keep aware of changes affecting IBM infrastructure as discussed in section 3.2.4. The external application vendor is also responsible for notifying customers (such as the owner of this environment) for security issues affecting the external application.

In the environment represented by Figure 5.3, both in-house applications and the third party applications must comply with PCI DSS. If a company wishes to use a third party application which meets these standards, the PCI website can be used as a tool to search for these applications:

https://www.pcisecuritystandards.org/security_standards/vpa/vpa_approval_list.html.

In LCSs, the two application types reside in the same environment. The third party application can be configured to interact with the in-house applications as needed by the business.

In both cases, the applications must pass PCI security standards.

Consider the PCI DSS requirement 6.3, which is concerned with software application development. Some of the requirement covers process, which is reviewed during PCI DSS assessment. PCI DSS 6.3.1 requires validation of input, error handling, and role-based access control (RBAC). In an LCS environment with RACF, application authentication must comply with RACF definitions. A PCI DSS review of LCSs should include questions about labeling and RACF in the application development processes.

If RACF is in the production environment, QSAs should check to ensure that RACF logging is enabled so that changes to the environment get recorded and can be forensically analyzed if needed. The logging should also be applied for superusers as well. While it is possible to directly review reformatted audit logs, a tool such as Tivoli Decision Support is a great way to summarize the security violations for regular review.

PCI DSS requirement 6.5 covers Open Web Application Security Project (OWASP http://www.owasp.org), which can be pertinent to LCSs if web applications are part of the scope. If this is pertinent, then a review of the associated application development process is required. Training records are also part of the PCI DSS assessment, and in this case the LCS assessment may not be different from a non-LCS assessment.

5.3 Summary of the Benefits of LCSs

The rich history of development by IBM on LCSs has generated a robust set of technologies which can interact with various devices including memory, disk, cache, and offline storage. Many LCS environments are highly transactional, and IBM has developed products such as CICS, IMS DB, and DB2 to drive and store transactions in a consistent and secure fashion. These products are also integrated with Parallel Sysplex cluster technology, providing the ability to create highly available LCS environments.

When reviewing the security functionality of LCSs, one item is constantly reoccurring: RACF. In an LCS environment, RACF can and should be used to model all security properties of the environment. When viewed from a PCI QSA standpoint, all security relevant policy configuration can be audited in one place. As RACF provides an authorization framework that is persuasive throughout LCSs a consistent security policy ensuring proper protection of CHD should be easily visible.

When using RACF, all operations regulated by RACF can be conveniently audited so that all audit and logging requirements of PCI DSS can be satisfied easily. Note that RACF is used here as the ESM. When using another ESM, similar principles apply.

Other inherent advantages of the LCSs are easy separation of production and test environments controlled by appropriate change management as well as by well trained operators and administrators.

LCS environments can use native z/OS data sets or z/OS UNIX file systems. For z/OS file systems there are at least five different *access methods* enabling access to both legacy storage and high speed disks. There are at least 5 different z/OS *file systems* available which flexibly support different application environments, and provide portability to and from other environments. Since the z/OS UNIX kernel requires an ESM for authentication, a secure configuration can be established. The z/OS DFSMS component automates storage management, and is enabled for use with an ESM like RACF.

The LCS environment can host different application types, with some built on IBM’s WebSphere Application Server for z/OS. PCI DSS assessments can use IBM Tivoli PCI DSS Management Module to audit application and configuration access and changes. If Tivoli Decision Support is in place, it can be used to review audit logs for security violations. The richness of tools available in an LCS environment allows a company to establish a secure and highly available solution set.

6 Other Aspects of LCS Security: RACF

The following sections describe other aspects of LCS security.

6.1 Security Services

Many operating systems have their security services distributed throughout the overall system. In the case of IBM’s System z, security services like access control, user authentication, user and group management, access control list management, privilege management, and auditing of security critical events are centralized in one system component, the Security Authorization Facility (SAF). IBM has documented and published the interfaces to this component, which allows third party vendors to supply an External Security Monitor as an alternative to the security monitor (RACF) that IBM provides. Examples of other external security monitors are CA-ACF2® or CA-Top Secret®.

This chapter will describe only the capabilities of the IBM supplied security monitor called Resource Access Control Facility (RACF). The two other products mentioned above provide similar functionality.

As previously stated, RACF authenticates users that want to connect to the system, controls access to all resources within the system, and allows generating detailed audit records for security related events. This is supported by an extensive set of management functions for configuring RACF itself, managing user, groups and their privileges, managing the resources to be controlled by RACF, managing the access control lists for those resources, and managing of the events that need to be recorded in the audit log.

6.1.1 Centralized Access Control Model

Unlike most other access control systems that allow management of access to a resource owner and specifically authorized administrators, RACF has been designed with a more centralized model for access control management. Large computing systems usually have to manage a large number of storage volumes, which include disks and tapes, with a huge number of individual files stored on them. Managing access control to all those files on an individual basis is impossible. RACF, therefore, allows for different and flexible ways to protect a large number of files or other protected resources that have identical or similar protection needs by combining them such that they are all protected using the same access control lists.

In conjunction with the possibility to define a hierarchy of user groups and the ability to delegate the management of just those groups to individual group administrators, RACF allows you to build an installation specific hierarchy of user groups and a hierarchy of protected objects where the management of each element of those hierarchies can be delegated.

This strategy allows for large number of users and large numbers of protected objects to still be manageable on a single system.

In addition to this flexibility, RACF allows applications to define their own classes of resources and to protect individual resources in those classes. An application can call RACF and ask if a specific user has the requested type of access to a resource it wants to protect. The advantage of this approach is that the whole framework for managing access control lists for resources that RACF provides can now be used also for application defined resources.

In order to provide a better understanding of the access control functions of RACF, here is a short description how they are implemented:

RACF manages user, group, and resource profiles where the security relevant attributes of users and groups are stored in user respective group profiles. The security relevant attributes of resources, including the access control lists, are stored in resource profiles. RACF provides a set of commands as well as library functions for use by applications that allow creating, deleting, and modifying those profiles. Whenever a user calls such a management function, RACF checks that the user has the required authority to perform the requested function.

6.1.2 Access Types

RACF knows a set of different types of access, which are organized hierarchically; (a hierarchically higher type of access automatically implies all lower access types). The access types (in hierarchical ascending order) are:

- None
- Execute
- Read
- Update
- Control
- Alter

The semantics of each access type is determined by the resource manager. In the most important case of z/OS data sets, this is:

- None: no access to the data set
- Execute: only used for programs, user can execute the program, but not read or copy the program
- Read: user can read the data set (which allows him to copy the data set)
- Update: user can write to the data set
- Control: for VSAM data sets: the user is allowed to use the CONTROL option. For non-VSAM data sets, Control is equivalent to Update.
- Alter: allows a user to create and delete a data set protected by that profile. In case of a discrete profile, the user is also allowed to modify the access control list for the data set (using the RACF PERMIT command)

6.1.3 Resource Classes

As stated before, RACF manages an open set of “resource classes”, the most important ones are:

- DATASET: class protecting access to z/OS data sets
- CONSOLE: class protecting access to consoles
- DASDVOL: class protecting access to volumes
- DEVICES: class protecting allocation of devices
- OPERCMDS: class protecting the use of operator commands
- PROGRAM: class protecting the execution of programs
- TAPEVOL: class protecting access to tape volumes
- FACILITY or XFACILIT: classes managing access to a number of privileges

- SERVAUTH: class managing access to a number of communication related resources and privileges
- UNIXPRIV: class managing access to a number of privileges for z/OS UNIX system services

This is just a small subset of the resources classes that z/OS itself uses to protect access to resources it manages. In addition, RACF is also used by a number of IBM and third party products to protect the resources they define, for example, IBM’s transaction monitor system CICS or IBM’s database management system DB2.

6.1.4 Label-based Access Control

In addition to the discretionary access control, RACF also supports label-based mandatory access control. This feature can be activated when an installation requires this additional level of access control. Label-based access control can be used to prohibit information flow between data with incompatible labels, thus allowing an installation to isolate specific data from unauthorized access by assigning a specific label to the data sets or files containing such data. The labeled security access protection then prevents any access to such data by users not assigned to the label, even if discretionary access permissions would allow such access. Labeled security also prohibits such data to be copied from a protected data set or file into one that is not assigned to the label of the originating data set. As a result, accidentally or deliberately copying critical data from one data set or file into another one does not result in an information flow that may breach security.

Within each resource class, an administrative user authorized for managing resources in the class can define resource profiles that are associated with resources to be protected. RACF differentiates between “discrete” profiles, which protect exactly one resource in a class and “generic” profiles, which can protect a whole set of resources within a class.

Whenever a “resource manager” (which is the program in control of a resource) wants to check a user’s authority to a resource, it calls RACF and asks: “Is user X authorized for access type A to resource Y in the resource class Z?” RACF will then check:

- Is the resource class known to RACF? If not, terminate and tell the caller that the resource class is unknown and RACF can therefore not determine if the user has access or not.
- Does the user or his active groups have privileges that allow him/her to access the resource in general? If yes, tell the resource manager to grant access.
- If labeled security is active, check if the current security label of the user and the security label of the resource are compatible for the type of access requested. Of they are not, access is rejected.
- Does a discrete profile exist for the resource. If yes, check in the access control list of this profile if the user has access or not and return the result. This also checks for access rights assigned on a group level.
- Is the resource protected by a generic profile? If there is more than one generic profile, RACF takes the most specific one. If this is the case, check in the access control list of this profile if the user has access or not and return the result. This also checks for access rights assigned on a group level.

If RACF can not find a clear “yes” or “no” answer, it indicates to the calling resource manager that it can not answer the question and leaves it up to the resource manager to decide if access is granted or not.

While the semantics are clear for access to data sets or volumes, RACF is also used within z/OS to define specific privileges for users or applications. For example, the privileges to perform specific storage management operations, like backup of a volume, are also managed by RACF. A user can be given the privilege to start a backup job by giving him access to a specific resource in a resource class used by the system's storage management. When the backup job is started, the backup program asks RACF if the calling user has READ access to a profile storage management associates with the privilege of performing a backup. The backup program will only continue when RACF answers that the user has been given this privilege (by granting him READ access to the profile).

6.1.5 Identity Mapping

In addition to the functions mentioned above, RACF can also be used to restrict a user's ability to call specific programs or to allow access to a data set only when the user has invoked a defined program and access is via this program. In UNIX-type systems, similar functions can be achieved using the setuid feature, but with the side effect that many systems then include the modified user identity in their audit records, making it hard if not impossible to trace the activity to the user that initiated it.

RACF always will include the identity of the originating user in the audit trail, regardless of which chain of commands and programs a user has invoked before the request is made. This provides a level of accountability that cannot be easily achieved in other systems.

RACF is designed to be open also for applications that want to protect access to their resources. To do this, an installation has to define a new resource class (adding it to the RACF class descriptor table and activating it using the RACF SETROPTS command), activate the class (using the RACF RLIST command), define generic or discrete profiles for resources in the class (using the RACF RDEFINE command), add access control lists to the profiles (using the RACF PERMIT command), and use the RACF programming interfaces within the application to check if a user has access to a resource. The use of all those commands requires specific privileges and authorities, which are checked by RACF.

6.1.6 User Identification

So far, we have mentioned RACF's capability to define and manage user, group and resource profiles and control access to resources. RACF is also the component within z/OS that is invoked when a user needs to be authenticated, regardless if the user accesses the system via a batch job, IBM's time sharing facility TSO, a UNIX shell, any remote access program like FTP, a UNIX r-command, a terminal emulation protocol like telnet or TN3270, or any z/OS specific program allowing remote access of users. All these call RACF for user authentication, allowing all these programs to use the various user authentication methods provided by RACF (passwords, passphrases, digital certificates, Kerberos tickets and more). RACF is also where the definition of restrictions when a user is allowed to connect to the system and via which communication links he is allowed to connect to the system.

To support specific applications that are not connected to a human user, RACF allows the definition of 'pseudo users'. Those are user profiles that can not be used to authenticate a user but can only be used for specific predefined batch jobs that can only be started by an authorized operator. Using the possibility to assign access rights and privileges to such pseudo users allows an installation to exactly define the resources an individual application executing with the identity of a pseudo user can access or use.

6.1.7 Security Related Audit Events

RACF also controls the security related audit events that are recorded in the overall z/OS audit log. Criteria defining the events to be recorded can be specified in the resources profiles as well as in user profiles. This enables an installation to exactly define the set of security related events to be recorded, for example, enable more detailed auditing for critical resources or for users that an installation requires to be monitored in more detail. An example is a critical data set used mainly by a specific application executing under the identity of a pseudo user. As long as this application alters information in this data set, only minimal auditing is required. On the other hand, if an administrative user alters information in this data set, all actions of such a user on the data set may be required to be audited. RACF allows addressing those requirements by specifying the details of the events to be recorded in the user and resource profiles.

6.1.8 RACF Remote Sharing Facility

Another unique feature of RACF is the RACF Remote Sharing Facility (RRSF). This function allows the RACF data to be synchronized within a network of z/OS systems. This not only allows for users within such a network to have a single user ID and use the same authentication information, it also allows to have the resource profiles being synchronized, resulting in the same overall access control policy and management of user privileges within the network of z/OS systems. This provides an even larger flexibility beyond a closely coupled Parallel Sysplex cluster, where synchronization of the RACF database is even easier.

In summary, RACF provides a powerful tool for managing users, groups, privileges, and access to a large number of different types of resources using a single management framework. It also manages the type of security related events to be audited allowing specifying those events on a user and resource basis. It can be extended to protect even application defined resources and can be synchronized over a network of z/OS systems thus providing not only single-sign-on capabilities but also an access control and audit policy harmonized within the whole network.

6.2 Cryptography

Cryptographic services, while becoming more and more important for the protection of data, are often still not an integrated part of many operating systems. While a number of libraries implementing cryptographic functions exist that can be linked with an application, their use still requires the cryptographic keys used to be protected by the application itself.

Software implementations of cryptographic functions often are also slow, especially when implementing algorithms for public key cryptography. They depend on the application itself to protect the cryptographic keys used. A higher level of security requires the support of specific cryptographic hardware that not only can speed up cryptographic operations but also provide a significantly improved protection of cryptographic key material. While such cryptographic coprocessors can be integrated into many different computer systems, none of them provides the integration of such coprocessors into the overall hardware and operating system to the extent that IBM's System z computer systems do.

A System z computer system may support a number of dedicated cryptographic coprocessors that can be used by applications via the interfaces provided by the Integrated Cryptographic Service Facility (ICSF) of z/OS. ICSF provides a set of standard interfaces to manage and use cryptographic keys and functions, which themselves then use the cryptographic coprocessors attached to the system. Keys can either be stored in ICSF managed key stores or (in the case of the private key of a private/public key pair) can be generated within the coprocessor and never leave the coprocessor.

Instead of specifying the key to be used, a caller of ICSF's cryptographic services just specifies a key identifier, which ICSF translates into the key or passes on to the coprocessor to be translated into the real key. RACF is also used to protect both access to keys, key identifier, and individual cryptographic functions using resources classes dedicated to ICSF. This allows for a level of security in the use of cryptographic functions not seen in other systems. Managing the coprocessor keys can be secured even more by requiring the use of a Trusted Key Entry (TKE) station for highly critical key management activities. This, together with the very high physical security of the coprocessors (they are FIPS 140-2 level 4 certified, the highest level for physical security defined in the FIPS 140-2 standard), takes the security of cryptographic operations to an even higher level.

While the retained key option of the cryptographic coprocessors is helpful and increases the confidence in the confidentiality of the retained private key, the user must be aware of certain functionality to use the function provided safely:

- If multiple cryptographic coprocessor cards are available onto which work can be scheduled, it is not possible to maintain the same set of private keys across all the cards. That is, the private key is retained within a single card and not the set of all cards.
- If the card maintaining the private key has a defect requiring replacement or if the card executes its key zeroization function as a result of an assumed attack, the stored private keys would be lost.

However, with consideration of the above points, the retained key option of the cryptographic coprocessor card is useful if used with care and with the knowledge that the retained private keys may be lost under certain conditions.

Many components of z/OS themselves use the services of ICSF and the cryptographic coprocessors. One example is the z/OS Public Key Infrastructure (PKI) services component, which provides the complete functions of a registration authority as well as a certification authority for a public key infrastructure. Another example is the System SSL component that provides the functions for implementing the SSL and TLS protocol. RACF uses those functions to generate and manage the digital certificates it optionally uses for the authentication of z/OS users.

In addition, System z processors have specific instructions for AES (with 128, 192, or 256 bit key length), 3DES, SHA-1, SHA-256, SHA-512, and a 3DES based keyed message authentication code. Those instructions allow an application to use high-speed symmetric cryptography directly without the need to link a software library.

6.3 Security Assurance Certifications in LCSs

Security has been a focus for many decades in the LCS environment. One way to independently verify this statement is to look at the various assurance certifications that have been awarded. In this section of the report, we focus upon certifications affecting the operating system and base architecture functions rather than applications.

The number of high-assurance certifications awarded in the LCS environment supports the claim that LCSs are highly secure. The number of re-certifications shows the commitment to maintaining that assurance, and in many cases, improving it. A brief look at the Common Criteria certifications shows that the evaluation assurance level is increasing, and in some cases is already achieving certification using formal methods (EAL 5).

6.3.1 Common Criteria

The evaluation of technical components and products against internationally-accepted, standardized criteria allows companies to objectively demonstrate the reliability of security functionality.

The Common Criteria (CC) and the internationally-recognized ISO standard (ISO/IEC 15408) are used by governments and other organizations to assess security and assurance of information technology products. The CC standard provides a uniform way of expressing security requirements and defines a set of rigorous criteria by which a product’s security aspects (for example, development environment, security functionality, and handling of security vulnerabilities) can be meaningfully evaluated.

LCS vendors, including operating system and application vendors, have invested greatly in such independent assurance giving confidence to others in the security posture of their LCS.

In several cases, Common Criteria certification is completed very soon after the release announcement.

The QSA should be aware that an LCS running a Common Criteria evaluated and certified OS or function must be configured according to the configuration guide relevant to that particular certification to be considered valid. It is important that a QSA check the configuration of the relevant parts of the LCS against these configuration guides as part of the PCI DSS Requirement 1.

At EAL4 and above, a vulnerability analysis is made and security functional testing is performed by the lab based on the vulnerability analysis.

At this level the evaluation by the laboratory also includes a thorough analysis of the development documentation from the architectural level to source code, mapping the design at each level of abstraction to the implementation.

A review of the development processes and the security of the development environment is also undertaken.

The table below shows a list of some of the evaluations that are relevant to LCSs. An EAL or evaluation assurance level ranges on a scale of 1-7, where EAL 4 is the highest level typically seen for evaluations in the commercial arena. Note that IBM publishes a full list of their certifications on the IBM web site [11] and several databases and applications have also been Common Criteria certified.

Some Common Criteria evaluations for LCS operating systems and applications	
IBM z/OS V1R10 EAL 4+	IBM PR/SM LPAR EAL 5
IBM z/OS V1R9 EAL 4+	IBM PR/SM z10 EC EAL 5
IBM z/OS V1R8 EAL 4+	IBM PR/SM z10 EC/BC EAL 5
IBM z/OS V1R7 EAL 4+	IBM PR/SM z9® 109 EAL 5
IBM z/OS V1R6 EAL 3+	IBM PR/SM z990 EAL 4
IBM z/VM 5.3 EAL 4+	IBM PR/SM z990 EAL 5
IBM z/VM 5.1 EAL 3+	IBM PR/SM z990/890 EAL 4
IBM LPAR EAL 4+	IBM PR/SM z990/890 EAL 5
	Vanguard Enforcer 7.1 EAL3+
	Computer Associates Top Secret for z/OS r14 (Currently in evaluation at EAL 4)
	Computer Associates ACF2 for z/OS r14 (Currently in evaluation at EAL 4)

6.3.2 FIPS 140-2

FIPS 140-2 is a specification published by NIST. It is a conformance standard mandatory for any cryptography used in the U.S. Federal government. There are several cryptographic modules that have been certified as having been validated as compliant with the standard by NIST that are typically included in LCS architecture. A full list of FIPS 140-2 certificates can be found on the NIST web site [40].

Cryptographic modules are assigned an overall security level which guides the assessor in understanding the level of assurance to be gained. Higher security level numbers indicate greater assurance. Software cryptographic modules may be security level 1 or security level

2, while those hardware modules, tested for compliance with the physical security requirements of FIPS 140-2 are described by security levels from 1 to 4.

It is important to note that the FIPS Approved and NIST recommended list of algorithms specified in the FIPS 140-2 annexes and implementation guidance may be different from those given by the PCI SSC in their definition of “Strong Cryptography” and at other places in the PCI Glossary. A brief comparison is given in the table below but this is a complex subject as NIST specify versions of standards, modes, and parameters in association with FIPS 140-2.

PCI Glossary	FIPS Approved/NIST recommended
AES (FIPS 197 with 128 bits or higher)	FIPS 197 with key lengths of 128, 192, 256 bits
ECC (160 bits and higher)	Elliptic Curve Digital Signature Algorithm (ECDSA) specified in FIPS 186-2
ElGamal (1024 bits and higher)	NIST does not specify ElGamal with FIPS 140-2
Hashing	See SHA-1 and SHA-2
MAC	NIST no longer allows the use of MAC with FIPS 140-2
SHA-1, SHA-2	SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512. are Approved
TDES	Special Publication 800-67: Three-key Triple DES and Two-key Triple DES are currently allowed by NIST Not all modes are approved
RSA	Signature schemes with appendix FIPS 186-2, ANSI X9.31-1998, PKCS #1 v2.1
NIST SP 800-57: Recommendations for Key Management	Not directly specified in association with FIPS 140-2 although the implementation guidance refers to it heavily.

For QSAs, the certified module gives the assurances noted in the list below:

- A FIPS 140-2 certified cryptographic module assures that all the FIPS Approved and NIST recommended algorithms are certified as implemented correctly, but note that algorithm validation can be obtained independently of FIPS 140-2 and if this claim is made QSAs can check on the NIST web pages for the algorithm validation program [38]. NIST statistics have shown that in as many as 25% of cases programming and other errors mean that the specified standard is not implemented correctly potentially making the use of that implementation much weaker [41].
- FIPS 140-2 specifies many requirements for key management. A correctly configured and installed module will implement the key management techniques conformant with the standard and specified in the associated Security Policy
- Configuration standards for PCI “system components” that are cryptographic modules are given in the associated security policy that are publicly available from the NIST web page giving cryptographic modules [40].

- Cryptographic modules are assigned an overall Security Level which guides the assessor in understanding the level of assurance to be gained. For software cryptographic modules two levels are generally assigned (1 and 2), while those hardware modules, tested for compliance with the physical security requirements of FIPS 140-2 are described by values between 1 and 4.

FIPS 140-2 certificates issued or in process for LCS architecture	
Cert: 35 - IBM 4758 PCI Cryptographic Coprocessor (Miniboot Layers 0 and 1) Cert: 40 - IBM S/390® CMOS Cryptographic Coprocessor Cert: 81 - IBM 4758 PCI Cryptographic Coprocessor (Miniboot Layers 0 and 1) Cert: 116 - IBM 4758-002 PCI Cryptographic Coprocessor (Miniboot Layers 0 and 1) Cert: 117 - IBM 4758-023 PCI Cryptographic Coprocessor (Miniboot Layers 0 and 1) Cert: 118 - IBM eServer™ zSeries 900 CMOS Cryptographic Coprocessor Cert: 121 - IBM 4758-001 PCI Cryptographic Coprocessor with CP/Q++ (Layer 2) Cert: 122 - IBM 4758-013 PCI Cryptographic Coprocessor with CP/Q++ (Layer 2) Cert: 345 - Security Module with CP/Q++ Cert: 346 - Security Module with CP/Q++ Cert: 350 - IBM® Crypto for C (ICC) Cert: 354 - IBM CryptoLite in Java Cert: 356 - IBM® CryptoLite in C Cert: 363 - JCOP21id 32K Cert: 376 - IBM Java JCE 140-2 Cryptographic Module 384 - IBM® Crypto for C (ICC) 406 - IBM® SSLite for Java 409 - IBM® Java JSSE FIPS 140-2 Cryptographic Module	497 - IBM Java JCE 140-2 Cryptographic Module 524 - IBM eServer Cryptographic Coprocessor Security Module 661 - IBM eServer Cryptographic Coprocessor Security Module 750 - IBM CryptoLite for C 775 - IBM® Crypto for C 899 - IBM CryptoLite for C 910 - IBM CryptoLite for Java 940 - IBM System Storage™ TS1120 Tape Drive - Machine Type 3592, Model E05 1081 - IBM Java JCE 140-2 Cryptographic Module 1152 – IBM System Storage LTO Ultrium 4 Tape Drive --- IBM Server Cryptographic Coprocessor Security Module (Currently in the validation process with NIST) IBM ® z/OS ® Version 1 Release 10 System SSL Cryptographic Module (Currently in the validation process with NIST) IBM System Storage TS1130 Tape Drive - Machine Type 3592, Model E06 (Currently in the validation process with NIST) IBM ® Crypto for C (Currently in the validation process with NIST)

Worthy of note is that the IBM 4758 PCI Cryptographic Co-Processor was the world's first product to be certified at NIST FIPS 140-1 Level 4. It has also been approved by German ZKA for operation as a security module in electronic cash networks.

6.4 Organizational Security

There are several assurance requirements specified by the PCI DSS. Depending on the business of the organization and the services and products that it offers, further legislation and regulation apart from the PCI DSS will very often be applicable. Furthermore, an organization that can afford to operate an LCS is very likely to operate on an international or global basis. Information Security Governance too is a topic that cannot be ignored, and issues such as privacy, copyright, and intellectual property rights must be considered in context.

We do not attempt to list every regulation or piece of legislation, but some examples of such legislation include:

In the US:

- Sarbanes Oxley (SOX)
- FISMA
- HSPD #12
- NSTISSP No. 11
- DoD directives and instructions
- HIPAA
- BIS cryptography export legislation
- State laws (California, Florida, Texas, Illinois…)

In Europe:

- Basel II
- EuroSox (8th directive 84/253/EEC)
- Electronic Signature Directive (1999/93/EC)
- In Germany: KonTraG (Gesetz zur Kontrolle und Transparenz im Unternehmensbereich), a law to improve corporate governance
- In Germany: MaRisk, minimum requirements for risk management for banks and assurance companies, issued by BaFin (German Federal Financial Supervisory Authority)

In Asia:

- Electronic Signature Law of the People's Republic of China, 2004
- Regulation of commercial encryption codes, Directive No. 273, 2000

Typically, an organization faced with such a complex regulatory requirements will draw from a standard such as ISO/IEC 27001 to provide a framework to build a management system that can deal with all these requirements.

Why is this important for a QSA? It is unusual for an organization operating an LCS not to have a developed management system with defined policies, procedures, work instructions, and record management. If ISO/IEC 27001 is implemented, the QSA will find that a risk management process is already established and the major controls from ISO/IEC 27002 matching the PCI DSS requirements are implemented and audited by the organization.

Appropriate measurements of the efficacy of the system and the implemented controls should be made.

If the organization does need to comply with other legislation and regulations, then the QSA may expect to find that other assessments using different assessment or audit schemes have been made (for example, ISO/IEC 27001, SAS/70, CobIT). Experienced QSAs will be used to the existence of reports from other assessments and be able to asses their applicability and relevance to the PCI DSS assessment. Similarly, by meeting the PCI DSS requirements, the requirements for other assessments and schemes may be met.

The message for organizations implementing compliance is to integrate the various requirements into a single integrated information security management system. This allows for efficiency and re-use and also should highlight any conflicting requirements and allow the organization to assess and manage such situations.

6.5 Penetration Testing

The most prevalent attacker cited in most threat models for LCS environments is the insider. With LCSs, threats involving insider attackers are a very serious issue. LCSs take time to configure properly, and are very complex.

.z/OS penetration tests are deployed to assess the technical security of a single system, a large complex network, or a specific application from an attacker’s point of view. Even a well planned infrastructure design does not prevent the technical implementation from containing vulnerabilities. Those vulnerabilities can only be reliably detected by penetration testing, where extensive knowledge and experience are used to search for erroneous configurations and flaws in the programming.

Regular penetration tests are an appreciated measure to guarantee a current overview of your company's security. Deficiencies in organizational processes for intrusion detection and reaction can be identified. Penetration testing shows whether a company's security policy is a living document or just another piece of paperwork.

Penetration testing in the LCS environment, in addition to more typical penetration testing activities such as web application assessment, should analyze the common interfaces used to transition from user state to system state, looking for integrity exposures that would allow an unauthorized user to exploit the interface in a way to gain system level authorities. The goal is to produce PoC exploits for z/OS Supervisor Calls (SVC) interfaces and z/OS Program Call (PC) interfaces. The SVCs analyzed include user and third party vendor supplied SVCs, along with IBM supplied SVCs that have been either “front-ended” or “hooked”. Other IBM SVCs may not need to be analyzed if they have already been analyzed during the z/OS Common Criteria evaluations and you are running in a Common Criteria evaluated configuration.

The tester should use automated tools to determine a definitive list of SVCs and PCs defined to the system. The list of PCs is then trimmed to eliminate those PCs which require the caller to be in supervisor state or with a Program Key Mask of 0-7, since these are already implicitly trusted. Priority is then given to analyzing PCs that are available globally to all users on the system.

The tester should use both manual and automated means to probe, test, and analyze the user/vendor, “hooked”, and “front-ended” SVCs, and the trimmed list of PCs for insecure aspects. These insecure aspects might allow unauthorized access to information processed by the system, escalation of privileges, exploitation of vulnerabilities, and circumvention of security functions. The analysis will concentrate on errors in parameter validation, parameter protection, “time of check” to “time of use” of parameters, and storage into unverified locations.

Finally, the tester develops methodologies for exploiting the potential vulnerabilities discovered. Code and/or procedural based exploits are developed to demonstrate the vulnerabilities found to facilitate interactions with the vendor responsible for the code containing the vulnerabilities. Code based exploits may utilize Assembler, REXX, CLIST, and common utilities to demonstrate vulnerabilities. The development of exploits may be time consuming and is only performed by prior agreement.

6.6 Auditing Facilities

z/OS provides a central auditing facility called System Management Facility (SMF). SMF provides a general framework for the generation, storage, protection, and management of any kind of logged data. z/OS is capable of generating a large number of different entries, called SMF records, in those logs. These entries are mainly for accountability, security, and performance measurement.

To distinguish the different SMF records generated by different components of a z/OS system, a large number of different record types exist with specific types assigned to specific components. This allows you to easily extract SMF records generated by a specific component.

Each SMF record type has a specific format that defines the information stored in the record. All records contain at least the date and time they have been generated as well as the identity of the user that was responsible for the action that caused the record to be created. As stated before, RACF is capable of generating specific SMF records for security critical events, for example, the execution of RACF commands, access attempts to protected resources, and attempts of users to authenticate to the system. A RACF auditor can define which events are actually audited. This can be defined based on the resource itself (for example, defining that all access attempts to a critical resource need to be audited while access attempts to less critical resources are not audited), the type of access requested to a resource in combination with the result (for instance SUCCESS(READ), FAIL(READ) for confidential material or SUCCESS(UPDATE), FAIL(READ) for data with integrity value to the system). Auditing can also be based on the identity of a user (for example, auditing more events for specific users) or on the specific attributes granted to a user that permits access to a resource, as well as several other criteria. This allows you to define the events to be audited based on the specific needs of an installation or the requirements of specific standards while also keeping the number of audit records to the minimum level required.

SMF also provides the functions to ensure that audit records are saved when the size of the audit trail exceeds a defined threshold, functions to sort and search the audit records and to generate reports. There are also functions to combine the audit records of all elements of a Parallel Sysplex into a single audit trail.

7 PCI Requirements in an LCS Environment

This chapter describes in more detail the considerations that a QSA can make in an LCS environment. First, we discuss the relevant scoping issues and then go on to describe each of the PCI DSS requirements and the associated LCS considerations. Finally, we give a very brief introduction to assessment tools and techniques available for LCSs.

7.1 Scoping and Scope Reduction

As already mentioned in the introduction, the scope of a PCI DSS assessment is a very important issue for consideration by the QSA. On the one hand, it is important to keep the scope as small as possible. A small scope will naturally reduce costs and effort by both the QSA and the subject organization. By combining the narrowest valid scope with the controls specified by the PCI DSS and the organization under assessment, the attack surface can be reduced thus making the environment more secure. On the other hand, it is vital for the QSA to accurately define the cardholder data environment as that part of the network that stores, processes, or transmits cardholder or sensitive authentication data. The PCI DSS requirements are applicable to all network components, servers, or applications that are within the defined part of the network.

In the PCI DSS, the notion is presented that the cardholder data environment can be reduced in size by properly configuring network security devices, such as routers and firewalls that reduce the traffic to that necessary for processing cardholder data through carefully defined and managed configurations.

It is evident from the previous chapters that LCS environments present very complex computing environments. The complexity partly arises from the fact that so many different logical computing entities may reside in one physical box. However, having the same physical housing does not mean that logically the systems within that housing have anything to do with each other, especially when assessed under PCI constraints.

Scoping is the most important task an assessor needs to perform in order to apply the PCI requirements. The requirements do apply to every system within scope. In an LCS environment, that scope is almost always a logical scope. The scope is enforced by logical separation mechanisms that have no representation in the physical world. Therefore, an assessor must clarify the scope of the system first. Everything else comes after that. For scoping, it needs to be kept in mind that, despite all their complexity, LCS environments are not complicated. In fact, the following key points are all that need to be examined for scoping:

1. Virtual machine/LPAR configuration and setup
2. Network and firewall configuration
3. RACF configuration and policy

Each of these mechanisms presents a strong logical and effective separation mechanism which can be employed to achieve levels of logical separation. The following list serves as a guideline for performing the actual scoping of the system:

- Examine the VM/LPAR configuration and determine which logical systems are defined and their functions. Next, determine which logical devices are relevant to the cardholder data environment (if they contain cardholder data, or services that support the cardholder data environment). Clarify with operations if and how these devices are shared between different system images or guests. If cardholder data is stored on shared devices, it needs to be assessed whether or not the accessing system operate under the same security policy (see below for RACF).

- Examine the network setup of the image under examination. Determine which applications have access to the network and how. Examine the relative network exposure of PCI cardholder data. Are the applications processing cardholder data connected to the network and how? Just because the system image has general network connectivity does not mean that the actual processing applications do also. Determine how systems connected to the same physical network are in fact part of the same logical network. Examine the firewall policies of the network stacks of the systems under examination. Even if the systems are part of the same logical network, the firewall configuration might still be used for separation of systems, applications and users.

- Examine the RACF (or other security product) configuration and determine the RACF protection policy for PCI cardholder data. Keep in mind that the policy can apply to many systems. If data is held on shared devices, other systems in a Sysplex can access this data. Verify that RACF is configured for enforcing a single policy within the Sysplex. Only after it is clarified how systems actually share data, can the scope be defined and the assessment can commence.

With confidence in the separation mechanisms discussed in this report the QSA should be able to specify a scope for assessment that focuses only on those system components that are relevant to storing, processing, and transmitting cardholder data.

7.2 PCI DSS Control Objectives in Detail

In this section of the report we discuss the PCI requirements in more detail, focusing only where requirements may be addressed with special context to the LCS by a QSA. It is beyond the scope of the report to address specific details of every system components but we do try to introduce some general concepts that can be extrapolated by QSAs to the LCS at hand.

The following subsections of the report list the major requirements of the PCI DSS in more detail. They provide a general discussion of the requirement in the context of LCSs and where appropriate, give more detail in reference to specific sub-requirements. For requirements and sub-requirements where we do not believe there is anything special to say about LCSs or their environments, we have presented the section in grey. In some cases we use an asterisk, “*”, to indicate that not all of the PCI DSS sub-requirements have been listed in this report. Where we thought it useful to do so, we have added reference to the Testing procedure text in addition to the PCI DSS requirement. In some cases, where we did not feel it was appropriate to our discussion, the PCI DSS requirement is not fully quoted. Where this occurred, we used ellipsis, “…”, to indicate that case.

7.2.1 Build and Maintain a Secure Network

1. Install and maintain a firewall configuration to protect cardholder data.

The intent of this requirement is to protect systems within the cardholder data environment from unauthorized access from untrusted networks whichever path is taken to obtain access. The standard identifies that firewalls are a key protection mechanism in achieving this for any network. This also applies to LCS environments, the QSA should be aware of z/OS firewall capabilities. However the QSA should also consider the configuration and separation capabilities provided by z/OS for virtual machines and defined subsystems. These system components are a vital part of the picture.

An important concept when reading the standard is to realize that although the term firewall is used, any technology providing appropriate restriction of unwanted traffic is applicable. The PCI SSC updated the standards to more explicitly include routers, but other technologies used can be included in the assurance discussion.

1.1 *Establish firewall and router configuration standards*

Include configuration standards for virtual machines, PR/SM, Logical Partitions, and subsystems (address spaces).

Any relevant Common Criteria configuration guides may be referred to.

Reference [13] Chapter 19 which describes the IP filtering capabilities of z/OS and [14] Chapter 7, IP Filtering, and chapter 14 IP Defensive Filtering

1.1.1 *Approving/testing all network connections and changes to the firewall and router configurations*

The QSA of an LCS should note that the z/OS communications server provides a full-grown packet filter, including the possibility to dynamically change rules, for example blocking IP address ranges which have been identified as hostile for a certain amount of time.

1.1.2 *Current network diagram with all connections to cardholder data and wireless networks.*

We would recommend to merchants and service-providers using LCSs that it is appropriate to include system components such as PR/SM, logical partitions, and subsystems (address spaces) on their network diagram to facilitate the work of the QSA in verifying the protection mechanisms employed.

For an LCS network diagram to be helpful for the QSA’s understanding of the processing environment, consider including the following items:

- Logical system boundaries (LPARs, VM guests, Sysplexes)
- Separation of network traffic within systems (different TCP/IP stacks, VIPAs, SNA communication)
- Separation achieved by other means (for example, mandatory separation using MLS mechanisms)
- Protected network connections (IPSec connections, AT-TLS, TLS/SSL-protected traffic)
- Shared devices and storage

In addition to the network diagram, high-level documentation of the policy configured for the network policy servers and agents will be helpful. Such documentation should include information about the way the policy is (centrally) defined and distributed between the logical systems, thus describing how the complexity is managed.

Note that since it is possible to use a partition or TCP/IP stack to communicate with another partition or VM, it means that the DMZ could be within the same machine. However, the same rules apply to this design.

1.1.3 *Requirements for a firewall at each Internet connection and between any demilitarized zone (DMZ) and the internal network zone*

Depending on the installation’s security policy, network traffic may be filtered by dedicated firewalls outside the LCS, and/or each system may set up its own packet filters and supporting network policy rules.

The QSA needs to understand that rather than ticking off boxes for each connection running though a firewall, each connection must be examined for appropriate separation, using the information provided for 1.1.2 and 1.1.5.

1.1.4 *Groups, roles, and responsibilities for management of network components*

No special considerations in an LCS environment

1.1.5 *Documentation and business justification for use of all services, protocols, and ports allowed, including documentation of security features implemented for those protocols considered to be insecure.*

Verify that firewall and router configuration standards include a documented list of services, protocols, and ports necessary for business

As with other systems, the QSA needs to assess the security of services by looking at the whole set of security measures applied to them. For example, the FTP service mentioned above may be secured by a variety of means in z/OS including:

- Using a protected channel (tunnel) for the connection such as IPSec, SSL/TLS, or AT-TLS (Application Transparent TLS) will result in all packets being sent encrypted without the FTP server being aware of this protection. See chapter 20 in [13], and chapter 12 in [14] for more details.
- Restricting access to the FTP service to selected systems and users, using strong authentication (PassTickets, Kerberos, digital certificates) to verify user identities
- For those who still use SNA (Systems Network Architecture), note that SNA allows session level encryption and should be used for transmitting CHD.

1.1.5.b *Identify insecure services, protocols, and ports allowed; and verify they are necessary and that security features are documented and implemented by examining firewall and router configuration standards and settings for each service. An example of an insecure service, protocol, or port is FTP, which passes user credentials in clear-text.*

If any insecure services, protocols and ports are identified as needed by the business then the QSA should ensure that access is restricted through RACF.

1.1.6 *Review firewall and router rule sets at least every six months*

No special considerations in an LCS environment

1.2 *Build a firewall configuration that restricts connections between untrusted networks and any system components in the cardholder data environment.*

1.2.1 *Restrict inbound, and outbound traffic to that which is necessary for the CDE.*

This requirement can be met by appropriately setting the packet filter rules for each communication stack. The QSA needs to review each individual configuration, unless the policies have been defined centrally (see below in 1.2.2).

Note that RACF can also be used to control access to the IP stacks, addresses and ports and hence restrict inbound and outbound traffic if necessary.

If routing capabilities are required for the system, then refer to chapter 15 in [14] and chapter 6, “Routing” in [13].

1.2.2 *Secure and synchronize router configuration files.*

A central configuration interface is provided through the policy agent, and its graphical implementation. For more informationrefer to “the IBM configuration assistant of z/OS Communication Server” (especially look at Figure 4. in [14]).

Apart from routers, this requirement extends in LCS environments to the network configuration managed by the LCS itself. Per-system configuration of the z/OS Communication Server’s TCP/IP stacks is found in the PROFILE.TCPIP and TCPIP.DATA datasets, which need to be assessed by the QSA.

Using the Communication Server Policy Agent, installations can greatly ease

	the QSA’s assessment by having a central Policy Definition Point (PDP), from where the policies (including AT-TLS definitions and packet filter configurations) can be distributed to the different communication stacks.
1.2.3	*Perimeter firewalls between wireless networks and CDE* No special considerations in an LCS environment.
1.3 1.3.1-1.3.5	*Prohibit direct public access between the Internet and any system component in the cardholder data environment.* Note that the DMZ may be in different partitions or virtual machines and that network policies should be applied.
1.3.6	*Implement stateful inspection, also known as dynamic packet filtering. (That is, only ”established” connections are allowed into the network.)* This requirement may be satisfied by appropriate rules for the packet filters (see 1.2.1 and 1.2.2 above). This requirement is related to the TCP protocol. Reference Chapter 17 of [13] where IP filtering, including checks for the logical direction (in, out) and the initiator (inbound, outbound) of the connection, is discussed.
1.3.7	*Place the network in an internal network zone, segregated from the DMZ.* Note that different network interfaces in one physical machine may end in different network zones depending on the logical machine they are attached to. The network plan from 1.1.2 should provide the necessary information.
1.3.8	*Implement IP masquerading to prevent internal addresses from being translated and revealed on the Internet, using RFC 1918 address space. Use network address translation (NAT) technologies—for example, port address translation (PAT).* This is a requirement for the external firewalls. We do not expect that the LCS will be directly connected to the internet without an additional dedicated firewall in between.
1.4.*	*Personal firewalls on PCs* No special considerations in an LCS environment

2. Do not use vendor-supplied defaults for system passwords and other security parameters.

2.1.*	*Attempt to log on with default passwords to system components and servers* No special considerations in an LCS environment
2.1.1	*Wireless networks: default keys, passwords, SNMP strings, etc. changed.* No special considerations in an LCS environment
2.2	*Develop configuration standards for all system components. Assure that these standards address all known security vulnerabilities and are consistent with industry-accepted system hardening standards.* *For example, SysAdmin Audit Network Security (SANS), National Institute of Standards Technology (NIST), and Center for Internet Security (CIS).* In addition to the organizations mentioned above, the Defense Information Systems Agency produces a series of Security Technical Implementation Guides which include: • OS/390 MVS Logical Partition STIG V2R2 • Virtual Machine STIG V2R2

- zOS STIG Ver 6 Rel 1.1 Memo 080609

2.2.1, 2.2.b *Implement only one primary function per server.*

See Navigating PCI DSS [3]*:* which states that “This requirement is meant for servers (usually UNIX, Linux, or Windows based), but not mainframe systems.”

Logical partitioning and virtualization allows separating critical functionality from less secure systems. Transaction processing systems should not be used for general dialog access, programming or testing.

We note that the virtualization SIG is currently considering this issue.

2.2.2, 2.2.b *Disable all unnecessary and insecure services and protocols (services and protocols not directly needed to perform the device’s specified function).*

Services can be disabled (if not needed at all) and services required by specific applications can be restricted (using RACF) to be used by those applications only and only on the TCP/IP stack and using the IP addresses where they are required. A QSA should inspect the TCP/IP configuration to see which services are enabled and check the RACF access control lists for the TCP/IP stacks, IP addresses, and ports.

2.2.3, 2.2.b *Configure system security parameters to prevent misuse*

Check the RACF options for secure settings (especially the following parameters should be set with the RACF SETROPTS command):

- NOADSP
- ERASE(ALL)[1]
- GENERIC[2]
- JES(BATCHALLRACF)
- PROTECTALL(FAILURES)
- TAPEDSN[3]
- Check the SETROPTS parameter for:
  - Auditing
  - Password rules
  - User inactivity period

Check the access control lists especially for write access to authorized

[1] Use of the ALL operand on the ERASE option causes all data sets (including temporary data sets) to be erased when deleted. This may have a severe performance impact on the system. Depending on you environment, you might consider using the ERASE option without an operand in conjunction with the erase indicator in generic and discrete RACF DATASET profiles. See the *Security Server RACF Security Administrator’s Guide* and the *Security Server RACF Command Language Reference* for more information.

[2] For guidelines on use of the GENERIC operand see the “Activating Generic Profile Checking and Generic Command Processing” section in the *Security Server RACF Security Administrator’s Guide*.

[3] If you are using IBM tape processing. For other vendors, refer to the vendor’s product documentation.

	libraries. Check user with specific RACF privileges (SPECIAL, AUDITOR, OPERATIONS, CLAUTH). Check the relevant members of SYS1.PARMLIB for the system configuration. *This list can be extended almost infinitely.*
	Include common security parameters in configuration standards for virtual machines, partitions and subsystems
2.2.4, 2.2.b	*Remove all unnecessary functionality, such as scripts, drivers, features and subsystems, file servers and unnecessary web servers* To answer this fully requires knowledge of the configuration. Generally the QSA will need to check the system configuration (mainly SYS1.PARMLIB) for services that are not required.
2.2.a, 2.2.c	*Configuration standards for all components include hardening, implemented* No special considerations in an LCS environment
2.3.*	*Encrypt all non-console administrative access. Use technologies such as SSH, VPN, or SSL/TLS for web-based management and other non-console administrative access.* All the services mentioned are available for z/OS. With AT-TLS there is also the capability to tunnel non-secured TCP-based protocols. Check the IP configuration for the IPSec, AT-TLS and the configuration of each protocol allowed if they are configured to automatically use encrypted and authenticated communication.
2.4.*	*Shared hosting providers must protect each entity’s hosted environment and cardholder data. These providers must meet specific requirements.* Is there an argument that the main system is a shared hosting provider? Some of the “customers” have CHD. Some do not. This can probably work in a virtualization scenario. Can we make it work also in the “traditional” system? Shared z/OS instances should not be allowed, but the separation mandated in Appendix “A” of the PCI DSS can be easily achieved.

7.2.2 Protect Cardholder Data

3. Protect stored cardholder data.

3.1.*	*Keep cardholder data storage to a minimum. Develop a data retention and disposal policy. Limit storage amount and retention time to that which is required for business, legal, and/or regulatory purposes, as documented in the data retention policy. …* No special considerations in an LCS environment
3.2.*	*Do not store sensitive authentication data after authorization (even if encrypted). …* No special considerations in an LCS environment
3.3*	*Mask PAN when displayed (the first six and last four digits are the maximum number of digits to be displayed).* No special considerations in an LCS environment
3.4.*	*Render PAN, at minimum unreadable anywhere it is stored (including on portable digital media, backup media, in logs) by using any of the following approaches: …* No special considerations in an LCS environment
3.4.1	*If disk encryption is used (rather than file- or column-level database encryption), logical access must be managed independently of native*

operating system access control mechanisms (for example, by not using local user account databases). Decryption keys must not be tied to user accounts.

3.4.1.a *If disk encryption is used, verify that logical access to encrypted file systems is implemented via a mechanism that is separate from the native operating systems mechanism (for example, not using local user account databases).*

Backup data can be automatically encrypted either by the data set management functions of z/OS or (in the case of specific tape drives) by the hardware. Removable devices can be configured to be automatically encrypted.

3.5.* *Protect cryptographic keys used for encryption of cardholder data against both disclosure and misuse:*

ICSF provides key encryption and access control to keys.

Access control is supported by RACF.

3.6.* *Fully document and implement all key-management processes and procedures for cryptographic keys used for encryption of cardholder data, including the following: …*

No special considerations in an LCS environment

3.6.6 *Split knowledge and establishment of dual control of cryptographic keys*

Verify that key-management procedures are implemented to require split knowledge and dual control of keys (for example, requiring two or three people, each knowing only their own part of the key, to reconstruct the whole key).

In general there are no special considerations in an LCS environment

In some cases for very high assurance the QSA may come across the Trusted Key Entry Station [28].

4. Encrypt transmission of cardholder data across open, public networks.

4.1 *Use strong cryptography and security protocols such as SSL/TLS or IPSEC to safeguard sensitive cardholder data during transmission over open, public networks.*

Examples of open, public networks that are in scope of the PCI DSS are:

- *The Internet*
- *Wireless technologies*
- *Global System for Mobile communications (GSM)*
- *General Packet Radio Service (GPRS)*

z/OS provides IPSec, SSH, SSL/TLS which can be used to protect cardholder data when transmitted over public network.

The Sysplex generally uses trusted channels and generally cannot be interpreted as an open public network. In cases where non-secured networks are used for communication between CPCs of a Parallel Sysplex, separate high-speed encryption boxes need to be used in order to provide the communication throughput required.

4.1.1 *Ensure wireless networks transmitting cardholder data or connected to the cardholder data environment, use industry best practices (for example, IEEE 802.11i) to implement strong encryption for authentication and transmission. …*

No special considerations in an LCS environment

4.2 *Never send unencrypted PANs by end-user messaging technologies (for example, e-mail, instant messaging, chat)*

The QSA should identify and consider any “messaging“ facilities, such as SndUsrMsg, that may be enabled on the system. The QSA should also check for any of the plethora of applications available for z/OS that allow for end-user messaging.

7.2.3 Maintain a Vulnerability Management Program

5. Use and regularly update anti-virus software on all systems commonly affected by malware.

5.1.* *Deploy anti-virus software on all systems commonly affected by malicious software (particularly personal computers and servers)*

It is generally accepted that LCSs operating systems such as z/OS are not greatly afflicted by malware. Currently the PCI SCC accept this and have mentioned it in Navigating the PCI DSS [3] for requirement 5.1 that mainframes typically are exempt from requirement 5.1.

Viruses of course can and have been written for LCSs but as well as being less common the typically well-managed environment inhibits their spread. The key point for this requirement is that LCS are not commonly affected. Note that this determination is based on the current situation, and should a QSA determine that LCS are “commonly affected” by malware then this requirement may be interpreted by the QSA as being applicable.

However, the prevalence of in-house programming and scripting observed in LCS environments allow for a greater chance of Trojans and malware from the insider and so the vulnerability assessments specified by requirement 6.2 are very important in this environment.

5.2 *Ensure that all anti-virus mechanisms are current, actively running, and capable of generating audit logs.*

See Navigating the PCI DSS [3]

6. Develop and maintain secure systems and applications.

As mentioned in Section 5.2, very few payment applications for LCSs appear on the list of approved payment applications given by the PCI DSS [6]. In an LCS environment, many of the payment applications are custom written and often legacy applications are based on software written before the PCI DSS was established. As a result, QSAs should pay particular attention to the development requirements in an LCS environment.

6.1.* *Ensure that all system components and software have the latest vendor-supplied security patches installed. Install critical security patches within one month of release.*

Note: An organization may consider applying a risk-based approach to prioritize their patch installations. For example, by prioritizing critical infrastructure (for example, public-facing devices and systems, databases) higher than less-critical internal devices, to ensure high-priority systems and devices are addressed within one month, and addressing less critical devices and systems within three months.

IBM releases APARs and PTFs on a regular basis. Once received by an organization these are usually tested in a testbed environment before they are applied to a production system. Because these are generally applied on a defined schedule and via a managed test environment a one month period may not be realistic for LCSs.

	OS and Application “Patches” are managed through the SMP/E System Modification Program/Extended feature.
	We also note that in a similar way that malware is not a significant problem in LCSs, the systems are very rarely a subject for zero day exploits.
6.2.*	*Establish a process to identify newly discovered security vulnerabilities (for example, subscribe to alert services freely available on the Internet). Update configuration standards as required by PCI DSS Requirement 2.2 to address new vulnerability issues.*
	z/OS security problems are usually not listed on the standard public web sites related to vulnerabilities (for example, cve.mitre.org, www.secunia.org, nvd.nist.gov).
	The RACF discussion list (RACF-L@LISTSERV.UGA.EDU) provides useful discussion on security topics of z/OS and RACF.
6.3	*Develop software applications in accordance with PCI DSS (for example, secure authentication and logging) and based on industry best practices, and incorporate information security throughout the software development life cycle.*
	Most organizations operating LCSs are also established developers (or involved with customization of the business applications they use). This includes not just the compiled programs but very often extensive job control scripts, batching facilities etc. These should be considered by the QSA as they can often introduce vulnerabilities to the CHD processing environment.
6.3.a	*Obtain and examine written software development processes to verify that the processes are based on industry standards, security is included throughout the life cycle, and software applications are developed in accordance with PCI DSS.*
	In the experience of the authors, most LCS organizations can be expected to have an established and defined software development life cycle.
6.3.1*	*Testing of all security patches, and system and software configuration changes before deployment, including but not limited to the following: …*
	No special considerations in an LCS environment
6.3.2	*Separate development/test and production environments*
	Testing can be done in a separate partition or a separate virtual machine. Cardholder data should not be accessible by those partitions or virtual machines.
6.3.3	*Separation of duties between development/test and production environments*
	No special considerations in an LCS environment
6.3.4	*Production data (live PANs) are not used for testing or development*
	No special considerations in an LCS environment
6.3.5	*Removal of test data and accounts before production systems become active*
	No special considerations in an LCS environment
6.3.6	*Removal of custom application accounts, user IDs, and passwords before applications become active or are released to customers*
	No special considerations in an LCS environment
6.3.7a and b	*Review of custom code prior to release to production or customers in order*

	to identify any potential coding vulnerability Note: This requirement for code reviews applies to all custom code (both internal and public-facing), as part of the system development life cycle required by PCI DSS Requirement 6.3. LCS web applications may not be the only type of application developed, following the intent of the PCI DSS this requirement should be extended to examine any standards for these applications. The QSA may come across use of languages such as COBOL, that are not typically seen in other environments, and JCL. Microsoft and other coding guidelines are available for common languages such as Java and C++. See also Requirement 6.5
6.4.*	*Change control for system components per DSS requirements* No special considerations in an LCS environment
6.5.*	*Develop all web applications (internal and external, and including web administrative access to application) based on secure coding guidelines such as the Open Web Application Security Project Guide. Cover prevention of common coding vulnerabilities in software development processes, to include…* No special considerations in an LCS environment
6.6	*For public-facing web applications, address new threats and vulnerabilities on an ongoing basis and ensure these applications are protected against known attacks by either of the following methods:* ● *Reviewing public-facing web applications via manual or automated application vulnerability security assessment tools or methods, at least annually and after any changes* ● *Installing a web-application firewall in front of public-facing web applications* There are few LCS related vulnerabilities mentioned in centralized vulnerability databases such as CVE. However the authors would recommend that QSAs check that their clients monitor some of the reputable LCS specific user forums such as those specific to RACF. Since few automated tools will include such vulnerabilities manual methods will need to be established to address any vulnerabilities noted.

7.2.4 Implement Strong Access Control Measures

7. Restrict access to cardholder data by business need-to-know.

With the connectivity and single image capabilities of z/OS configured as a Sysplex it is imperative that a QSA have a good understanding of the separation and partitioning ability of z/OS. This provides the ability to restrict the data to those with a need to know without impacting the other users of the system.

7.1	*Limit access to system components and cardholder data to only those individuals whose job requires such access. Access limitations must include the following:*
7.1.1-7.1.3	*Various organizational privileged access control requirements.* No special considerations in an LCS environment
7.1.4	*Implementation of an automated access control system* RACF or an alternative External Security Monitor should be installed and used. The hierarchical groups structure as well as the generic profile capabilities of RACF should be used to simplify the management of the

access control policy.

7.2 *Establish an access control system for systems components with multiple users that restricts access based on a user's need to know, and is set to "deny all" unless specifically allowed.*

Specify UACC(NONE) in the RACF profiles protecting all data sets with critical cardholder data. Ensure that all data sets are RACF protected.

7.2.1 *Coverage of all system components*

No special considerations in an LCS environment

7.2.2 *Assignment of privileges to individuals based on job classification and function.*

No special considerations in an LCS environment

7.2.3 *Default "deny-all" setting*

See above: Specify UACC(NONE) in all critical profiles.

8. Assign a unique ID to each person with computer access.

8.1. *Assign all users a unique ID before allowing them to access system components or cardholder data.*

Enforced by RACF

8.2. *In addition to assigning a unique ID, employ at least one of the following methods to authenticate all users:*

- *Password or passphrase*
- *Two-factor authentication (for example, token devices, smart cards, biometrics, or public keys)*

RACF allows for passwords, passphrases, and public key certificates.

There are various options for two-factor authentication including IBM's "Personal Communications for Windows" [50] in the chapter "Configuring and Using Security for Personal Communications" explains how the terminal emulation package can be configured to use the IBM Global Security Kit (GS/Kit) as well as Microsoft CryptoAPI (MSCAPI) to use an X/509 based certificate scheme using digital certificates that act as electronic ID cards. Some important notes are given about changing default passwords as well as advice to not use the relatively insecure password stash file. Smartcard support is discussed through the use of GS/Kit and is well explained in the Administrator's manual.

8.3 *Incorporate two-factor authentication for remote access (network-level access originating from outside the network) to the network by employees, administrators, and third parties. Use technologies such as remote authentication and dial-in service (RADIUS); terminal access controller access control system (TACACS) with tokens; or VPN (based on SSL/TLS or IPSEC) with individual certificates.*

z/OS allows for certificate based network entity authentication with IPSec or SSL/TLS.

Several two factor authentication solutions are available in the LCS environment including that available through terminal emulation as discussed under requirement 8.2 above.

8.4 *Render all passwords unreadable during transmission and storage on all system components using strong cryptography*

Passwords in z/OS are stored in encrypted form only and the system can be configured to never transmit them in clear. Network protocols that use password or passphrase based authentication should be secured with

	SSL/TLS (potentially by tunneling them using AT-TLS).
8.5*	*Ensure proper user authentication and password management for non-consumer users and administrators on all system components as follows: …* No special considerations in an LCS environment
8.5.16	*Authenticate all access to any database containing cardholder data. This includes access by applications, administrators, and all other users.* All applications that allow different users to connect, use RACF to authenticate the user. RACF also allows customer developed applications to use RACF to authenticate users.

9. Restrict physical access to cardholder data.

As a general point, an LCS is a significant investment for an organization and this investment is made because of extensive processing required by the business. It is very unusual to find an LCS environment that does not operate a mature datacenter environment. The QSA might expect to find other certifications pertaining such as ISO/IEC 27001 and SAS/70 that cover some of the same controls as specified by the PCI DSS.

9.1.*	*Use appropriate facility entry controls to limit and monitor physical access to systems in the cardholder data environment.* No special considerations in an LCS environment
9.2.*	*Develop procedures to help all personnel easily distinguish between employees and visitors, especially in areas where cardholder data is accessible.* *For purposes of this requirement, “employee” refers to full-time and part-time employees, temporary employees and personnel, and contractors and consultants who are “resident” on the entity’s site. A “visitor” is defined as a vendor, guest of an employee, service personnel, or anyone who needs to enter the facility for a short duration, usually not more than one day.* No special considerations in an LCS environment
9.3.*	*Make sure all visitors are handled as follows: …* No special considerations in an LCS environment
9.4.	*Use a visitor log to maintain a physical audit trail of visitor activity. Document the visitor’s name, the firm represented, and the employee authorizing physical access on the log. Retain this log for a minimum of three months, unless otherwise restricted by law.* No special considerations in an LCS environment
9.5.	*Store media back-ups in a secure location, preferably an off-site facility, such as an alternate or back-up site, or a commercial storage facility. Review the location’s security at least annually.* No special considerations in an LCS environment
9.6.	*Physically secure all paper and electronic media that contain cardholder data.* No special considerations in an LCS environment
9.7.*	*Maintain strict control over the internal or external distribution of any kind of media that contains cardholder data, including the following: …* No special considerations in an LCS environment
9.8.	*Ensure management approves any and all media containing cardholder data that is moved from a secured area (especially when media is distributed to individuals).* No special considerations in an LCS environment
9.9.*	*Maintain strict control over the storage and accessibility of media that contains cardholder data.* No special considerations in an LCS environment
9.10*	*Destroy media containing cardholder data when it is no longer needed for business or legal reasons as follows: …* No special considerations in an LCS environment

7.2.5 Regularly Monitor and Test Networks

10. Track and monitor all access to network resources and cardholder data.

Auditing, the generation of audit trails, their management and security has long been a feature of LCSs.

10.1	*Establish a process for linking all access to system components (especially access done with administrative privileges such as root) to each individual user.*
	10.1.a Verify through observation and interviewing the system administrator, that audit trails are enabled and active for system components.
	RACF reporting tools are available that list the access rights of users.
10.2*	*Implement automated audit trails for all system components to reconstruct the following events:*
10.3*	*Record at least the following audit trail entries for all system components for each event:*
10.4*	*Synchronize all critical system clocks and times.*
	This is performed automatically in a Parallel Sysplex by means of the sysplex time. In addition, NTP servers can be used as a clock reference – the sysplex can be configured to use NTP as the clock reference, consequently.
10.5*	*Secure audit trails so they cannot be altered*
	RACF can be utilized to ensure that audit trails cannot be altered.
10.5.4	*Write logs for external-facing technologies onto a log server on the internal LAN.*
	A separate partition must be used in an LCS environment giving the same (or more) protection than this PCI DSS requirement.
10.5.5	*Use file-integrity monitoring or change-detection software on logs to ensure that existing log data cannot be changed without generating alerts (although new data being added should not cause an alert).*
	Facilities for this are sparse in the LCS. You may be able to substitute a compensating control.
10.6	*Review logs for all system components at least daily. Log reviews must include those servers that perform security functions like intrusion-detection system (IDS) and authentication, authorization, and accounting protocol (AAA) servers (for example, RADIUS).*
	Note: Log harvesting, parsing, and alerting tools may be used to meet compliance with Requirement 10.6
	Specialized LCS tools are available, but no special requirements in an LCS environment.
10.7	*Retain audit trail history for at least one year, with a minimum of three months immediately available for analysis (for example, online, archived, or restorable from back-up).*
	No special requirements in an LCS environment

11. Regularly test security systems and processes.

11.1. *Test for the presence of wireless access points by using a wireless analyzer at least quarterly or deploying a wireless IDS/IPS to identify all wireless devices in use.*

Some of the common commercial network vulnerability scanning tools and services may not include LCS vulnerabilities. It is important that the organization review any vulnerability scanning tools employed in order to ensure that they can detect LCS vulnerabilities. The QSA should ensure that any tools used do include LCS vulnerabilities and are of a sufficient quality to meet the intent of the PCI DSS

11.2. *Run internal and external network vulnerability scans at least quarterly and after any significant change in the network (such as new system component installations, changes in network topology, firewall rule modifications, product upgrades). …*

When employing the services of an ASV it is imperative that the organization ensures that the ASV is aware that an LCS is the focus of the scan. Many ASV vulnerability databases may not include LCS vulnerabilities. This should be discussed with them as part of contract negotiations.

11.3.* *Perform external and internal penetration testing at least once a year and after any significant infrastructure or application upgrade or modification (such as an operating system upgrade, a sub-network added to the environment, or a web server added to the environment). These penetration tests must include the following: …*

Please see section 6.5 of this report for a discussion on penetration testing in an LCS environment.

11.4. *Use intrusion-detection systems, and/or intrusion-prevention systems to monitor all traffic in the cardholder data environment and alert personnel to suspected compromises. Keep all intrusion-detection and prevention engines up-to-date.*

IDS component of the z/OS Communication server can be utilized in support of this requirement.

11.5.* *Deploy file-integrity monitoring software to alert personnel to unauthorized modification of critical system files, configuration files, or content files; and configure the software to perform critical file comparisons at least weekly. Note: For file-integrity monitoring purposes, critical files are usually those that do not regularly change, but the modification of which could indicate a system compromise or risk of compromise. File-integrity monitoring products usually come pre-configured with critical files for the related operating system. Other critical files, such as those for custom applications, must be evaluated and defined by the entity (that is, the merchant or service provider).*

Refer to Section 8. for discussion of a suggested compensating control.

7.2.6 Maintain an Information Security Policy

12.Maintain a policy that addresses information security.

As a general point, an LCS is a significant investment for an organization and this investment is made because of extensive processing required by the business. It is very unusual to find an LCS environment that does not already have well developed operational procedures which include security policy.

12.1*	*Establish, publish, maintain, and disseminate a security policy* For a complex LCS environment, the organization may consider following framework standards such as ISO/IEC 27001 and ISO 20000 as best practices for establishing security policy and controls.
12.2.	*Develop daily operational security procedures that are consistent with requirements in this specification (for example, user account maintenance procedures, and log review procedures).* No special requirements in an LCS environment: It is very unusual to find an LCS environment that does not already have well developed operational procedures. Typically operations teams exist with well-defined processes and procedures.
12.3.*	*Develop usage policies for critical employee-facing technologies (for example, remote-access technologies, wireless technologies, removable electronic media, laptops, personal data/digital assistants (PDAs), e-mail usage and Internet usage) to define proper use of these technologies for all employees and contractors.* No special requirements in an LCS environment
12.4*	*Ensure that the security policy and procedures clearly define information security responsibilities for all employees and contractors* No special requirements in an LCS environment
12.5.*	*Assign to an individual or team the following information security management responsibilities* No special requirements in an LCS environment
12.6.*	*Implement a formal security awareness program to make all employees aware of the importance of cardholder data security.* No special requirements in an LCS environment
12.7.0	*Screen potential employees (see definition of "employee" at 9.2 above) prior to hire to minimize the risk of attacks from internal sources. ...* No special requirements in an LCS environment
12.8.*	*If cardholder data is shared with service providers, maintain and implement policies and procedures to manage service providers,* No special requirements in an LCS environment
12.9.*	*Implement an incident response plan. Be prepared to respond immediately to a system breach.* No special requirements in an LCS environment

7.3 Assessment Tools and Techniques

A great many tools supporting QSAs in their assessment are available. We do not propose to mention everything available, nor do we specifically recommend particular tools and techniques as a plethora of tools are available on the market. However, the QSA might consider some of the following items of interest in supporting their endeavors:

- Relevant Redbooks, for example “Enhanced Auditing using the RACF SMF Data Unload Utility. [15]
- Use of the IBM Health Checker for z/OS, to help maintain configuration and installation sensitive resources health check
- Tivoli’s zSecure Suite augments the monitoring capabilities of z/OS and RACF

8 LCS Environment: Compensating Controls

The following sections describe the LCS environment compensating controls. Compensating controls are used to provide the appropriate security measures to fulfill the meaning of a PCI requirement when the direct implementation is not feasible. The goal is to have a control in place that provides at least the same level of security. If appropriate, the requirement can be satisfied by using security mechanisms that are already used to fulfill other PCI requirements.. But the fulfillment of other requirements can not be used as an argument for fulfilling the requirement for which the compensating control is needed.

The compensating controls listed here do provide a way to satisfy the meaning of PCI requirements in cases where the direct fulfillment is not possible.

In this section we present some ideas for compensating controls that might be commonly used in an LCS environment. We do not intend to recommend that it is appropriate that these are used in every case, or even at all. This is the purview of the QSA.

8.1 Requirement 2.2.1 Implement only one primary function per server

PCI DSS V1.2 Requirement	**Testing Procedures**
2.2.1 Implement only one primary function per server	2.2.1 For a sample of system components, verify that only one primary function is implemented per server. For example, web servers, database servers, and DNS should be implemented on separate servers.

	Explanation
Constraints	In an LCS, very often with a great many virtual machines, subsystems etc, it is usual practice and the reason for specifying an LCS to implement more than one primary function per server.
Objective	According to the PCI SSC this is “To reduce the possibility of access to Card Holder Data through mis-configured or vulnerable applications and functions co-residing on a single server. This is intended to ensure your organization's system configuration standards and related processes address server functions that need to have different security levels, or that may introduce security weaknesses to other functions on the same server. For example: 1. A database, which needs to have strong security measures in place, would be at risk sharing a server with a web application, which needs to be open and directly face the Internet. 2. Failure to apply a patch to a seemingly minor function could result in a compromise that impacts other, more important functions (such as a database) on the same server.”
Identified Risk	On a properly configured LCS the segregation mechanisms

	involved are adequate to ensure separation and reduce vulnerabilities, the risk is minimal.
Validation of Compensating Controls	Refer to the PCI SSC document “Navigating the PCI DSS” which states that “This requirement is meant for servers (usually Unix, Linux, or Windows based), but not mainframe systems.” Under the discussion of requirement 2.2.1
Maintenance	Not applicable.

8.2 Requirement 6.1 Install Critical Security Patches Within One Month of Release

PCI DSS V1.2 Requirement	**Testing Procedures**
6.1 Ensure that all system components and software have the latest vendor-supplied security patches installed. Install critical security patches within one month of release. *Note: An organization may consider applying a risk-based approach to prioritize their patch installations. A company can prioritize critical infrastructure (for example, public-facing devices and systems, databases) higher than less-critical internal devices to ensure high-priority systems and devices are addressed within one month, and less critical devices and systems are addressed within three months.*	**6.1.a** For a sample of system components and related software, compare the list of security patches installed on each system to the most recent vendor security patch list to verify that current vendor patches are installed. **6.1.b** Examine policies related to security patch installation to verify they require installation of all critical new security patches within one month.

	Explanation
Constraints	In an LCS, it may not be impractical to include Program Temporary Fixes (PTFs) testing and installation within the time period specified by the PCI DSS requirements. The resources required for adequate testing in the time period specified may be cost-prohibitive to the LCS organization. Additionally, applying PTFs that have not been tested in the target LCS environment presents significant risks to the organizations operations and potentially to the security of the cardholder data. In some cases, a Recommended Service Update (RSU) containing lower priority PTFs as well as HIPERs may be issued up to three months after a single PTF has been made known to the subject organization. PTFs issues in RSUs have undergone more extensive testing by IBM. When considering the time taken for testing by the organization, the whole patch process may often exceed the one or three months quoted by the requirement. The logistics of patching in an LCS environment may be such that it is reasonable that the process extend beyond the timescales given in the PCI DSS requirements.
Objective	The objective of this requirement is to ensure that properly tested patches addressing vulnerabilities to the cardholder data environment are applied promptly, with the aim of

	reducing the risks of loss of cardholder data. The PCI SSC in version 1.2 added some flexibility to this requirement to allow for an organizations specific circumstances.
Identified Risk	The identified risk is that the period, during which known vulnerabilities remain uncompensated for, is extended beyond the PCI SSC requirement of one - three months, potentially providing a longer window of opportunity for attackers.
Validation of Compensating Controls	The QSA should assess the patching process including the testing performed both by IBM before release of the PTF, and the subject organization testing in the proposed operational environment for both HIPER and RSU cases. In particular the process of reviewing patches should include an assessment of criticality of the patch to the LCS under review. The QSA should assess if the periods achieved by the organization before a patch is applied are reasonable to ensure that adequate testing has been performed. Consideration of the risks involved including that the general security posture of an LCS is high should be performed. IBM issue HIPER PTFs on a weekly basis that includes high risk problems including those that may affect the integrity of the data, RSUs are issued typically on a monthly basis.
Maintenance	The QSA should discuss the process and controls in place to maintain compensating controls, perhaps describing the worst-case scenario.

8.3 Requirement 11.5 File-Integrity Checking

PCI DSS V1.2 Requirement	
11.5 Deploy file-integrity monitoring software to alert personnel to unauthorized modification of critical system files, configuration files, or content files; and configure the software to perform critical file comparisons at least weekly. ***Note****: For file-integrity monitoring purposes, critical files are usually those that do not regularly change, but the modification of which could indicate a system compromise or risk of compromise. File-integrity monitoring products usually come pre-configured with critical files for the related operating system. Other critical files, such as those for custom applications, must be evaluated and defined by the entity (that is, the merchant or service provider).*	**11.5** Verify the use of file-integrity monitoring products within the cardholder data environment by observing system settings and monitored files, as well as reviewing results from monitoring activities. Examples of files that should be monitored: • System executables • Application executables • Configuration and parameter files • Centrally stored, historical or archived, log and audit files

	Explanation
Constraints	File integrity checking software in the style of typical comparison databases like Tripwire or Aide do not exist on

	mainframes. In addition, those tools do not operate in real-time, so alerts can be had only after the fact.
Objective	The objective of the original control is to detect modifications to critical system files (executables, configuration information). For z/OS, this means that all system data sets, libraries and load modules, parmlibs (configuration data), and if used, any critical UNIX System Services files stored in zFS or HFS file systems need to be monitored for changes.
Identified Risk	There is no additional risk in using the compensating control. It actually reduces the risk compared to the original requirement, as alerts can be generated in real time, reducing the time between modification and detection.
Definition of Compensating Controls	Use the audit function of z/OS to monitor write access to all relevant system data sets and USS files mentioned above. What is monitored needs to be defined according to the local installation and the naming conventions used. All restricted items are candidates for monitoring. Users with the attribute SPECIAL or OPERATIONS need to be included in the list of audited entities (using the SETROPTS command). Note: This is also required to satisfy PCI requirements 10.
Validation of Compensating Controls	As modifications of critical system files in a production environment are not feasible, a different way to verify the working of the compensating control is needed. One way to check that the audit system monitors critical files is to check the audit records generated during the last patch cycle or the last system configuration changes. For all of these, the appropriate audit records need to be shown.
Maintenance	Maintenance Is performed by updating the audit source selection in case system maintenance has changed the critical system files that need to be monitored.

9 Summary/Conclusion

This report has looked at large computing systems and their generalized environments in the context of compliance with the Payment Card Industry Data Security Standard.

We have used our experience with LCSs and especially the IBM System z and our expertise in the intricacies of the security features of z/OS (for example, z/VM, partitioning, and RACF) to guide and educate the readers in interpreting the PCI DSS in the complex world of LCSs.

Our conclusion is that the PCI DSS can embrace LCSs with very few issues so long as the assessor, the QSA, is knowledgeable and skilled enough to understand the details of the systems under assessment as well as the intent of the PCI DSS.

We have illustrated that the threats vulnerabilities and attacks applicable in an LCS environment are not so very different to any other environment, but that the processes and technologies available in an LCS environment can mitigate these very well.

We have presented the case that the separations for virtual machines, logical partitions and other features such as RACF, meet a variety of stringent information security standards and are often independently assessed at high levels of assurance. This fact can give QSAs confidence that the intent and the requirements of PCI can be met in an LCS environment.

We hope that we have provided the necessary pointers and information to both the QSA and the organizations undergoing assessment to provide a framework for approaching the assessment of an LCS. We look forward to receiving feedback about this report from interested parties so that it can evolve and be improved in future editions.

Glossary

ACS	Automatic Class Selection
ADSP	Automatic Data Set Protection
AT-TLS	Application Transparent TLS
ASV	Approved Scanning Vendor
BDAM	Basic Direct Access Method (becoming obsolete)
BPAM	Basic Partitioned Access Method (for libraries)
BSAM	Basic Sequential Access Method (for special cases)
CC	Common Criteria
CEK	Central Electronic Complex
CHD	Cardholder Data
COTS	Commercial off the shelf
CPC	Central Processor Complex
CPU	Central Processing Unit
DASD	Disk Auxiliary Storage Device
DBA	Database Administrator
DFS	Distributed File System
DNS	Domain Name Server
DSS	Data Security Standard
EAL	Evaluation Assurance Level
ESM	External Security Manager
FIPS	Federal Information Processing Standard
GDPS	Geographically Dispersed Parallel Sysplex
GID	Group ID
HMC	Hardware Management Console
HSM	Hardware Security Module
IBM	International Business Machines Corporation
ICF	Integrated Coupling Facility
ICSF	Integrated Cryptographic Service Facility
IDS	Intrusion Detection Services
IFL	Integrated Facility for Linux
IPSEC	IP Security
ISC-3	InterSystem Channel-3
J2EE	Java 2 Enterprise Edition
JDK	Java Development Kitr

JRE	Java Runtime Environment
KEK	Key Encrypting Key
LCS	Large Computing System
LPAR	Logical Partition
NAT	Network Address Translation
NDS	Novell Directory Services
NFS	Network File System
NIST	National Institute of Standards and Technology
NTP	Network Time Protocol
OS	Operating System
PA	Payment Application
PC	Program Call
PCI	Payment Card Industry
PCI DSS	Payment Card Industry Data Security Standard
PCI SSC	Payment Card Industry Security Standards Council
PIN	Personal Identification Number
PKI	Public Key Infrastructure
POS	Point of Sale
PR/SM	Processor Resource/System Manager
PSIFB	Parallel Sysplex Using Infiniband
PU	Processing Unit
QSAM	Queued Sequential Access Method (heavily used)
RACF	Resource Access Control Facility
RLS	Record Level Sharing
RRSF	RACF Remote Sharing Facility
SAF	Security Authorization Facility
SAN	Storage Area Network
SIG	Special Interest Group
SPI	Serial Peripheral Interface
SSL	Secure Sockets Layer
SVC	Supervisor Calls
TCIM	IBM Tivoli Compliance Insight Manager
TCP	Transmission Control Protocol
TFS	Temporary File System
TKE	Trusted Key Entry

TLS	Transport Layer Security
TSIEM	IBM Tivoli Security Information and Event Manager
UID	User ID
UPT	Unattended Payment Terminal
U.S.	United States of America
VLAN	Virtual LAN
VM	Virtual Machine
VPN	Virtual Private Network
VSAM	Virtual Sequential Access Method (used for more complex applications)
XCF	Cross Coupling Facility
z/FS	System z File System
z/OS	IBM System z Operating System

Bibliography & References

PCI SSC Documents

[1] PCI SSC. 2009, Payment Card Industry (PCI) Data Security Standard: Requirements and Security Assessment Procedures July 2009 Version 1.2.1 https://www.pcisecuritystandards.org/security_standards/download.html?id=pci_dss_v1-2.pdf

[2] PCI SSC. 2009, Payment Card Industry (PCI) Payment Application Data Security Standard: Requirements and Security Assessment Procedures July 2009 Version 1.2.1 https://www.pcisecuritystandards.org/security_standards/pci_pa_dss.shtml

[3] PCI SSC. 2008, Navigating PCI DSS October, 2008 https://www.pcisecuritystandards.org/pdfs/pci_dss_saq_navigating_dss.pdf

[4] PCI SSC. 2008, Payment Card Industry (PCI) Data Security Standard (DSS) and Payment Application Data Security Standard (PA-DSS): Glossary of Terms, Abbreviations, and Acronyms October, 2008 https://www.pcisecuritystandards.org/pdfs/pci_dss_glossary.pdf

[5] PCI SSC, 2009, PCI Quick Reference Guide: Understanding the Payment Card IndustryData Security Standard version 1.2, https://www.pcisecuritystandards.org/pdfs/pci_ssc_quick_guide.pdf

[6] PCI SSC List of Validated Payment Applications. Available from https://www.pcisecuritystandards.org/security_standards/vpa/vpa_approval_list.html

Supporting

[7] DISA, Security Technical Implementation Guides, Available from http://iase.disa.mil/stigs/stig/index.html

[8] Fremstad, P., Fries, W., Gasparovic, M., Hamid, P., Hatfield, B., Jorna, D., Nogal, F., Stenfors, K.-E. & White, B. 2009, IBM System z10™ Enterprise Class Technical Guide, IBM Corp. http://www.redbooks.ibm.com/abstracts/sg247516.html?Open

[9] GDPS Family – An Introduction to Concepts and Capabilities. http://www.redbooks.ibm.com/redbooks/pdfs/sg246374.pdf

[10] Gutmann, P.2009 , Some thoughts on Threat Modeling, [Online], University of Auckland, Available from: http://www.cs.auckland.ac.nz/~pgut001/pubs/threat_modelling.pdf

[11] IBM Corp: Rogers, P., Feio, R., Lundgren, O., Pleus, Rita & Singh, K. 2008, ABCs of z/OS System Programming Volume 6: Introduction to security, RACF®, Digital certificates and PKI, Kerberos, cryptography and z9™ integrated cryptography, LDAP, and Enterprise Identity Mapping (EIM)., IBM Corp.

[12] IBM Corp: IBM Academic Initiative, System z course materials, [Online], IBM Corp., Available from: http://www-03.ibm.com/systems/z/advantages/charter/skills_coursematerials.html

[13] IBM Corp: z/OS V1R11 Communications Server IP Configuration Guide. Available from http://publibz.boulder.ibm.com/epubs/pdf/f1a1b320.pdf

[14] IBM Corp: z.OS V1R10 Communications Server TC/IP Implementation Volume 4: Security and Policy-based networking http://www.redbooks.ibm.com/redbooks/pdfs/sg247699.pdf

[15] IBM Corp: Enhanced Auditing Using the RACF SMF Data Unload Utility Available from http://www.redbooks.ibm.com/abstracts/gg244453.html?Open

[16] IBM Corp: IBM Course: zSeries Hardware Management Console (HMC) Operations Available from: http://www-03.ibm.com/services/learning/ites.wss/us/en?pageType=course_description&courseCode=ES240

[17] IBM Corp: Ebbers, M., Kettner, J., O'Brian, W. & Ogden, B. 2009, Introduction to the New Mainframe: z/OS Basics, Available from http://www.redbooks.ibm.com/abstracts/sg246366.html

[18] IBM Corp: Introduction to the New Mainframe Security.pdf http://www.redbooks.ibm.com/abstracts/sg246776.html?Open

[19] IBM Corp: Kappeler, P., Braney, J., Beda, P., Buzzetti, M., Granados, S., Pederson, E. M., Ng, K., Pnghena, M., Powers, E., Schmidt, M. & Schultz, R. 2008, Java Security on z/OS - The Complete View. Available from http://www.redbooks.ibm.com/abstracts/sg247610.html?Open

[20] IBM Corp: 2010, Security Evaluations for IBM Products - z series, [Online], IBM Corp., Available from: http://www-03.ibm.com/security/standards/st_evaluations.shtml#zSeries

[21] IBM Corp: IBM Tivoli Security Information and Event Manager Compliance Management Modules, [Online], IBM Corp., Available from: ftp://ftp.software.ibm.com/common/ssi/pm/sp/n/tid10408usen/TID10408USEN_HR.PDF.

[22] IBM Corp: Security Server (RACF) Introduction, IBM Corp. GC28-1912-05. http://publibz.boulder.ibm.com/epubs/pdf/ich1a510.pdf

[23] IBM Red Alerts subscription service, [Online], IBM Corp., Available from: http://www14.software.ibm.com/webapp/set2/sas/f/redAlerts/.

[24] IBM Corp: SNA Network Implementation Guide, Available from http://publib.boulder.ibm.com/infocenter/zos/v1r9/index.jsp?topic=/com.ibm.zos.r9.cs3/cs3.htm

[25] IBM Corp: Kapeller, P., Rodolfi, G., Donceel, K., Nuttal, M., Hollamby, I. & Darees, J. M. 2006, Sysplex eBusiness Securityz/OS V1R7 Update Available from http://www.redbooks.ibm.com/redbooks/pdfs/sg247150.pdf

[26] IBM Corp: VSAM Demystified http://www.redbooks.ibm.com/abstracts/SG246105.html?Open

[27] IBM Corp: z/OS Mainframe Security and Audit Management using IBM Tivoli zSecure Available from http://www.redbooks.ibm.com/abstracts/sg247633.html?Open

[28] IBM Corp: zSeries Trusted Key Entry (TKE) Version 4.2 Update, available from http://www.redbooks.ibm.com/redbooks/pdfs/sg246499.pdf

[29] IBM Corp: z/OS UNIX Security Fundamentals http://www.redbooks.ibm.com/abstracts/redp4193.html?Open

[30] IBM Corp: IBM Systems Journal, Volume 39, number 1, pp34 “ESA/390 interpretive-execution architecture, foundation for VM/ESA, available from http://domino.research.ibm.com/tchjr/journalindex.nsf/a3807c5b4823c53f85256561006324be/cae3cf60a5eee39585256bfa00685c52?OpenDocument

[31] Kahn, M. 2007, 'The Brave New World of PCI DSS- Why You May Need to Think Differently about Your Systems Architecture(s)', The Clipper Group: Captain's Log. www.clipper.com

[32] IBM Corp: Kyne, F., Dhondy, N., Raften, D., Ratte,Mark & Sale, G. 2009, GDPS Family - An Introduction to Concepts and Capabilities, IBM Corp. http://www.redbooks.ibm.com/redbooks/pdfs/sg246374.pdf

[33] Litner, H. 2007, 'IBM Announces Industry's First End-to-End Solution for PCI Compliance', www.ibm.com 1 November 2007.

[34] McFarland, A. 2007, 'Cloaking Cardholder Data With PCI Compliance', The Clipper Group: Navigator. www.clipper.com

[35] McFarland, A. 2007, 'PCI DSS Compliance and System z - A Combination That Makes Sense', The Clipper Group: Navigator. ftp://ftp.software.ibm.com/systems/z/pdf/TCG2007094LI.pdf

[36] Moore, M. 2007, 'Credit-Card Security with Mainframes', in IBM Systems Magazine, IBM Corp. http://www.ibmsystemsmag.com/mainframe/novemberdecember07/features/18240p1.aspx?ht=mary%20moore%20mary%20moore

[37] NIST Computer Security Division: Cryptographic Algorithm Validation Program http://csrc.nist.gov/groups/STM/cavp/index.html

[38] NIST Computer Security Division: Algorithm Validation Lists http://csrc.nist.gov/groups/STM/cavp/validation.html

[39] NIST Computer Security Division: Cryptographic Module Validation Program http://csrc.nist.gov/groups/STM/cmvp/

[40] NIST Computer Security Division: Cryptographic Module Validation Lists http://csrc.nist.gov/groups/STM/cmvp/validation.html

[41] NIST ITL Cryptographic Algorithm Validation Program http://www.nist.gov/itl/csd/sma/cavp.cfm

[42] Radding, A. 2003, 'Mainframe SAN', z/Journal, vol. 2003, no. April/May, pp. 8-12. http://www.zjournal.com/pdfIssue/pdfArticle/radding1.pdf

[43] Schesvold, J. 2007, 'Protecting Confidential Data With Payment Card Industry Compliance', in IBM Systems Magazine, IBM Corp. http://www.ibmsystemsmag.com/mainframe/novemberdecember07/technicalcorner/18751p4.aspx

[44] Schesvold, J. 2008, 'Application Integration With PCI', IBM Systems Magazine. http://www.ibmsystemsmag.com/mainframe/julyaugust08/technicalcorner/21047p1.aspx

[45] Schesvold, J. 2008, 'Protect Confidential Information With PCI Compliance', IBM Systems Magazine. http://www.ibmsystemsmag.com/mainframe/marchapril08/technicalcorner/19610p1.aspx

[46] Schesvold, J. 2008, 'Cloaking Cardholder Data With PCI Compliance', IBM Systems Magazine. http://www.ibmsystemsmag.com/mainframe/julyaugust08/technicalcorner/21047p1.aspx

[47] Sysplex eBusiness Security z/OS V1R7 Update http://www.redbooks.ibm.com/redbooks/pdfs/sg247150.pdf

[48] Tivoli Security Information and Event Manager, [Online], IBM Corp., Available from: http://www-2000.ibm.com/software/tivoli/products/security-info-event-mgr/index.html

[49] White, B., Ebbers, M., de Souza Braga Jr, V., Chen, W., Dente, G., Ferreira, O. L., Giudici, M., Porterie, J., Reichenberg, M. & Wijaya, A. 2009, ABCs of z/OS System Programming Volume 6: Introduction to security, RACF®, Digital certificates and PKI, Kerberos, cryptography and z9™ IBM z/OS V1R10 Communications Server TCP/IP

Implementation Volume 4:Security and Policy-Based Networking, IBM Corp. http://www.redbooks.ibm.com/abstracts/sg247516.html?Open

[50] Personal Communications for Windows, Version 5.9: Administrator's Guide and Reference.

http://publib.boulder.ibm.com/infocenter/pcomhelp/v5r9/index.jsp?topic=/com.ibm.pcomm.doc/books/html/admin_guide13.htm

Organizations

[51] PCI SSC 'Payment Card Industry Security Standards Council' http://www.pcisecuritystandards.org/