

INDUSTRY INTERNSHIP PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
AWARD OF THE DEGREE OF
BACHELOR OF TECHNOLOGY

UNDER COURSE
Information Technology

SUBMITTED BY

Name of Student :
Prateek Joshi

Roll No:
131416

Under the Guidance of
Dr. Pradeep Kumar Singh
Assistant Professor(Senior Grade)

SUBMITTED TO
Department of Computer Science & IT



DECLARATION

I hereby declare that the training report is an authentic record of my own work carried out as requirements of industrial training for the award of B.Tech degree in Information Technology from Jaypee University of Information Technology, HP, under the guidance of Dr. Pradeep Kumar Singh, from 6th February to June 2016. All the information furnished in this training report is based on my work in company and is genuine.

Name of Student: Prateek Joshi

Roll No: 131416

(Signature of Student)

Date:

CERTIFICATE

This is to certify that the declaration statement made by Prateek Joshi is correct to the best of my knowledge and belief. He has been pursuing this training under my guidance and supervision. No part of the work has ever been submitted for any other degree at any University. The training report is fit for the submission and partial fulfillment of the conditions for the award of B.Tech degree in Information Technology from Jaypee University of Information Technology, HP.

Dr. Pradeep Kumar Singh
Assistant Professor (Senior Grade)
Department of Computer Science
JUIT
Solan, HP.

Date:

CERTIFICATE

This is to certify that Prateek Joshi, student of B. Tech IT of Jaypee University of Information Technology is currently pursuing industrial training from February 6th, 2017. He is working in Crypto Management department on the product ***Hardware Security Module*** under the guidance of Mr. Suhail Shrivastava.

Signature of Manager

ACKNOWLEDGEMENT

I take this opportunity to thank **Mr. Suhail Shrivastava (Manager, Crypto Management)** who was with me to help me out in the entire major as well as the minor issues that I faced. He guided me on a lot of aspects of work throughout the training duration. I am highly grateful to **Mr. Shivam Garg (Tech Lead), Mr. Vishwa Prakash Pandey (Senior Software Engineer), Ms. Kanchan Mehta (Software Engineer) & Mr. Rahul Kumar (Associate Software Engineer)** for all the help and guidance extended by him at every stage during the training of Luna EFT. Special thanks to **Mr. Sudhakar Porwal (Senior Manager)** for his timely motivation and friendly talks which helped to understand the organization and its code of conduct.

I wish to place on record my gratitude towards **Gemalto, Noida** for providing me an opportunity to work on these projects of such an importance. My stay in the organization has been a great learning experience. This exposure has enriched me with a wide technical knowledge and has also introduced me to the attributes of a successful professional life.

I am also grateful to **Dr. Pradeep Kumar Singh** who was also there for help when I needed it. He remained calm and gave me the solution to all problems which I took to him. His guidance let me carry out the training with ease.

ABOUT GEMALTO: THE DATA PROTECTION COMPANY

Gemalto is a leading global provider of data protection. For over 30 years, Fortune 500 global corporations and government agencies have turned to Gemalto to secure and protect their most valuable data assets and intellectual property. Our data-centric approach focuses on the protection of high value information throughout its lifecycle, from the data center to the cloud. More than 25,000 customers across commercial enterprises and government agencies trust Gemalto's SafeNet Identity and Data Protection solutions to protect and control access to sensitive data, manage risk, ensure compliance, and secure virtual and cloud environments.

Safeguarding High Value Data

Gemalto's data-centric SafeNet security solutions have enabled our customers to adapt to the escalating internal and external threats to their high-value data, and rapidly evolve to address new business requirements and compliance mandates. Our products deliver persistent protection of sensitive data throughout the information lifecycle by:

- Protecting Identities of users and applications
- Securing Transactions of critical, high-performance digital processes
- Encrypting Data as it is created accessed, shared, stored, and moved
- Enabling Cloud-based Infrastructures through protection and control of data in virtual environments.
- Protecting the Value of Intellectual Property for Independent Software Vendors by providing software and technology vendors with the tools required to effectively and efficiently manage and enforce software licenses, and expand revenue and monetization opportunities in all environments, including the cloud.

Gemalto Technology

Expertise for Tomorrow's Security Needs Gemalto's employees are connected by a sustained force of innovation that has driven the development of many of the industry's cutting edge products that are currently securing the world's most valuable and sensitive asset including 550 cryptographic engineers. Their leadership has enabled many of our products to obtain the highest levels of security certifications and accreditation, including Federal Information Processing Standards (FIPS) and Common Criteria for Information Technology Security Evaluation. Gemalto also holds 100 distinct patents in the United States and over 70 foreign-issued patents. SafeNet is a member of technological world standard organizations, including the Institute of Electrical and Electronics Engineers (IEEE), the Internet Engineering Task Force (IETF), the 3rd Generation Partnership Project (3GPP), and the Open Mobile Alliance (OMA.) SafeNet is currently involved with the International Organization for Standardization (ISO) 2008 to update future technological principals.

Gemalto's Customers—Increasingly Global, Mobile, and in the Cloud

Gemalto serves more than 25,000 customers in over 100 countries, across both commercial and government agencies. Many of the world's leading global organizations rely on our SafeNet solutions to secure their high-value data throughout the information lifecycle— from the data center into the cloud. Those customers utilizing SafeNet solutions include Citigroup, Banamex, Bank of America, Gap, Netflix, Starbucks, Kaiser Permanente, Cisco, Dell, and Hewlett-Packard.

Table of Contents

<i>Declaration</i>	2
<i>Certificate</i>	3
<i>Certificate</i>	4
<i>Acknowledgement</i>	5
<i>About The Company</i>	6
<i>Table of Contents</i>	8

Unit 1: Introduction

Chapter 1: Overview

1.1 Abstract	12
1.2 HSM Overview	12
1.3 Product Overview	13
1.4 Project Overview	15

Unit 2: Project-1

Chapter 2: Requirement and Analysis

2.1 Software Requirements Specifications	16
2.1.1 Purpose	
2.1.2 Scope	
2.1.3 Feasibility Analysis	
2.1.3.1 Technical Feasibility	
2.1.3.2 Cost Feasibility	
2.1.3.3 Time Feasibility	
2.1.4 Environment	18
2.1.4.1 Input/ Output	
2.1.4.2 Hardware and Software Requirement	
2.1.4.3 System Security	
2.1.4.4 System Interaction	

2.2 Product Perspective	19
2.2.1 Interfaces	
2.2.2 Operations	
2.3 Product Functions	20
2.4 User Characteristics	21
2.5 Constraints	21

Chapter 3: Design Specification

3.1 Introduction	22
3.2 Development Strategies	22
3.3 System Architecture	24
3.4 Policies and Tactics	26

Chapter 4: My Role and Learning

4.1 Basic Cryptography	28
4.2 Technology Used/Studied	34
4.2.1 C Language	
4.2.2 Linux OS Basic	

Chapter 5: Development and Snapshots

5.1 Development Overview and Code Snippets	38
5.2 Overall Flow of Task	38
5.3 List of Tasks	39
5.2.1 Encipher API Hands On	
5.2.2 Re-Encrypt API Modification	
5.4 Snapshots	40

Chapter 6: Conclusions

6.1 Conclusions —————50

Chapter 7: References —————51

UNIT 1: INTRODUCTION

Chapter 1: Overview

1.1 Abstract

The industrial training is in reality a great platform to inculcate the work culture of any organization. Trainee gets a complete exposure to industry which forms the foundation for rest of the career as understanding of industry operation strengthens. Ability to comprehend the problem and solve it enhances great deal because professionals around you always motivate you and compel you to do things near perfection.

I learnt a lot in company and also had a chance to work on several products Gemalto is known for, as instance Payment HSM Luna EFT and Crypto Command Center. Both of these are flagship projects of Gemalto (earlier SafenetInc). HSMs provide reliable protection for applications, transactions and information assets by securing cryptographic keys.

1.2 HSM Overview

SafeNet Hardware Security Modules (HSMs) provide reliable protection for transactions, identities, and applications by securing cryptographic keys and provisioning encryption, decryption, authentication, and digital signing services. SafeNet HSMs are ranked #1 in the market worldwide. They provide the highest-performing, most secure, and easiest-to-integrate application and transaction security solution for enterprise and government organizations.

With a broad range of HSM offerings and a full range of API support, SafeNet HSMs enable application developers to easily integrate security into custom applications. In partnership with leading application solution providers, SafeNet has produced HSMs that offer end-to-end protection for organizations, helping them achieve regulatory

compliance, streamline business processes, reduce legal liabilities, and improve profitability.



1.1 Luna EFT

1.3 Product Overview

SafeNet offers the following type of HSMs:

- General Purpose HSMs, Embedded



1.2 General Purpose HSMs Embedded

SafeNet embedded HSMs are designed for tight integration between an application server and the HSM. Sensitive data is protected cost-effectively, and business processes are uninterrupted, making SafeNet embedded HSMs the highest performance solution in the market for the protection of cryptographic keys.

- General Purpose HSMs, Network Attached



1.3 General Purpose HSMs, Network Attached

When the hardware security module (HSM) needs to be shared between a number of application servers, SafeNet network-attached HSMs safeguard the cryptographic keys used to secure transactions, applications, and sensitive data.

- Payment HSMs



1.4 Payment HSMs

For over 25 years, SafeNet products have protected and secured transactions around the world. SafeNet HSMs provide powerful end-to-end security for online banking transactions and applications for credit, debit, and chip cards.

SafeNet products protect and secure transactions including Web-based PIN issuance and management, PIN mailers and generators, online banking transactions, m-payments, contact-less payments, and card issuance.

- Validations and Certifications



1.5 FIPS Standard

SafeNet Hardware Security Modules (HSMs) provide reliable protection against compromise for applications and information assets to ensure regulatory compliance, reduce the risk of legal liability, and improve profitability. SafeNet's robust FIPS and Common Criteria validated HSM solutions are tamper resistant and offer the highest level of security.

1.4 Project Overview

First project deals with modifying an API in payment HSM which re-encrypts the master key used for maintaining the confidentiality so that the critical data is not compromised. The master key is used to encrypt and decrypt the session keys and the other keys used in the transaction. I went through an extensive research about cryptography and various algorithms dealing with data confidentiality, integrity and authenticity. Apart from this understanding the architecture of the product and the guidelines to manage the code was a learning experience. The coding language used was C, required a basic level knowledge of Linux platform and virtual machines.

The other task deals with the Crypto Command Center and its integration with the Oracle database. Currently the CCC is integrated to work with Postgres database and the future requirement is to integrate it with Oracle. The project required an extensive research of the market to select the viable version of Oracle database to start the project with. Apart from it project needed an extensive knowledge of java programming language, Postgres and certainly Oracle.

UNIT 2:Project-1

Chapter 2: Requirement and Analysis

2.1 Software Requirements Specifications

2.1.1 Purpose

The purpose of this task was to modify an existing API and create a new one which could address the new requirement. The modification dealt with the change in encryption and decryption flow and the type of encryption technique used. Apart from modifying, creating extensive test case set to harden the API was also the purpose of this task.

2.1.2 Scope

This project was designed as an assessment task. It was a requirement by a customer and was given to me as an assessment.

2.1.3 Feasibility Analysis

2.1.3.1 Technical Feasibility

During our analysis about the project we found that all the resources both hardware & software like the OS environment, language to be used for developing the system etc. which are required to develop the project are available in the market & there is no resource which is not available required by the project. Since all the resources are easily available therefore by this analysis we arrive at the decision that the proposed system is technically feasible.

2.1.3.1 Cost Feasibility

The cost feasibility was analyzed using the basic **COCOMO** model. The development was considered as organic since project deals with developing a well understood application program. In organic type, the size of the development team is reasonably small, and the team members are experienced in developing similar types of projects. The following equations can be used to estimate the cost required to complete the task.

$$\text{Effort Applied (E)} = a_b(\text{KLOC})^{b_b} \text{ [man-months]}$$

$$\text{Development Time (D)} = c_b(\text{Effort Applied})^{d_b} \text{ [months]}$$

$$\text{People required (P)} = \text{Effort Applied} / \text{Development Time [count]}$$

Where,

$$a_b = 2.4$$

$$b_b = 1.05$$

$$c_b = 2.5$$

$$d_b = 0.38$$

So by this analysis it was found that the total cost was less than the revenue generated by the system and the number of people required to develop is two; one for developing the project and one for supervision as this an assessment task.

2.1.3.1 Time Feasibility

By COCOMO model analysis it was found that the total time required to develop the project is 1 month. So it was needed to develop the proposed system within 1 month.

All the resources for the development of the proposed system are available and there is no specific dependency which needs to be resolved so the development needs is not an issue. However as I am new to the product it took some time to understand the basic architecture and design of the product. Initial task was to acquire the knowledge

regarding the Payment HSM and the working of the API to be modified. Hence this analysis helped in estimating the time which will be required to completely understand the product and develop the required system.

2.1.4 Environment

2.1.4.1 Input/ Output

The system takes input as the command or string to call the API which is required by the user to perform the specific task. The input contains the function code and data which is to be processed by the API.

The output of the system will provide the required key encrypted as required and the response code for the call, may it be successful or some kind of failure response code.

2.1.4.2 Hardware and Software Requirement

The Payment HSM is used by the client using a utility software which is installed in host machine. The call to the API is made thorough this client. The specifications are listed as follows.

Platform : Linux and Windows OS

Tools : VMware/Virtual Box

Hardware : The client can run on any general hardware system.

Software : No additional software requirement these are provided by Safenet.

2.1.4.3 System Security

The API requires no additional security to be implemented for development, however dependencies needs to be resolved if any arises due to addition of the API into the system.

2.1.4.4 System Interaction

The project to be developed can make interaction with only those systems whose IP address is entered by the user or admin over the product no other system is able to interact with the system.

2.2 Product Perspective

2.2.1 Interfaces

The payment HSM for which the API is created is self-contained and requires no input from any other system except the test data or the processing data. All the operations are performed on the HSM internally.

User Interfaces

The system comprises of well-defined user interface to operate the payment HSM.

It contains several user screens:

1. Lush (Luna Shell)
2. Web based

Hardware Interfaces: These interfaces are used to operate the HSM

LCD Screen On HSM

Apart from these interfaces we require a USB Keyboard or Pin Entry Device to perform key entry in the HSM.

Communication Interfaces:

1. ASYNC – This process listens on the Serial Host port and processes incoming requests by passing them to the HSM function manager. It uses proprietary protocols.
2. ETHERNET–This process handles the raw Ethernet protocol.

3. TCP/IP – This process listens on the TCP port and processes incoming requests by passing them to the HSM function manager. It also routes the replies back to the originator. It uses proprietary protocols.

2.2.2 Operations

The user interface is provided with the list of operations which can be performed by the product. The enabling of the API designed and providing the necessary details to utilize it can be given using the UI of the product. The list of operations cannot be disclosed as per the company policy however generic operation includes key entry, data encryption, pin mailer etc.

2.3 Product Function

The project performs a wide variety of tasks however the API which I developed decrypts the data it receives and again encrypts it following a particular scheme which was required by the customer. The scheme cannot be disclosed.

2.4 User Characteristics

The Product provides a featured UI for user to perform several operations in admin mode:

1. Create Administrators.
2. Change Password.
3. Configure APIs or Host Function.
4. Add Host connections.
5. Add Keys.
6. Configure Different Properties of HSM.
7. Choose Type of Connection TCP/IP or https.
8. Enable the Sensitive Functions.

9. Delete the Keys if not required.

10. View Audit Logs.

In User Mode the available options are:

1. Configure HSM and APIs

2. View Audit Logs.

2.5 Constraints

There are various constraints to be kept in mind:

1. Client must have minimum of 64 MB RAM to support Client-Server architecture.
2. The connectivity can only be provided in terms of specified communication modes.

Chapter 3: Design Specification

3.1 Introduction

The API does not alter the design already used by the HSM. It is an addition to the existing function space. So the purpose of the “Design Specification” would be to explain the high level design used by the HSM which I am working on.

The Design activity is often divided into separate phases – System Design and Detailed Design. System Design is sometimes also called Top-level Design. This system design aims to identify and modules that should be in the system, the specification of these modules, and how they interact with each other to produce the desired result. At the end of the System Design all the major data structures, file formats, and the major modules in the system and their specification are decided.

3.2 Development Strategies

For the development of this API, we have adopted Agile Software development model approach. Agile software development is a group of software development methods based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development and delivery, a time-boxed iterative approach, and encourages rapid and flexible response to change.



3.1 Agile Development

While working on this task we followed agile development as we divided the task into modules and started doing altogether by achieving smaller goals on daily basis.

Agile Training

A training session on Agile Development was provided by Gemalto. It was a two day session held in company premises by Andrew Goddard. He has been leading Agile teams for over 8 years following the Scrum framework and other Agile/Lean techniques. He is a certified Scrum master. His specific interest is in working with clients and teams to focus on the value that is delivered through the software development process.

Agile methods emphasize face-to-face communication over written documents when the team is all in the same location. Most agile teams work in a single open office (called a bullpen), which facilitates such communication. Team size is typically small (5-9 people) to simplify team communication and team collaboration. Larger development efforts can be delivered by multiple teams working toward a common goal or on different parts of an effort. This might require a coordination of priorities across teams. When a team works in different locations, they maintain daily contact through videoconferencing, voice chat, e-mail, etc.

A Short Note - Agile Process

- 1.** The task is broken into smaller user stories by product owner or any member of the team for the product backlog.
- 2.** The stories are prioritized using some measure important in terms of business.
- 3.** The team is assigned some stories based on priority to be completed in the duration of a sprint.
- 4.** Sprint can be of one or two weeks, depends on the team.
- 5.** After the sprint is completed the team delivers a functional product which has some market value.

- At the end of each sprint the deliverables are reviewed and the stories for the next sprint get decided.

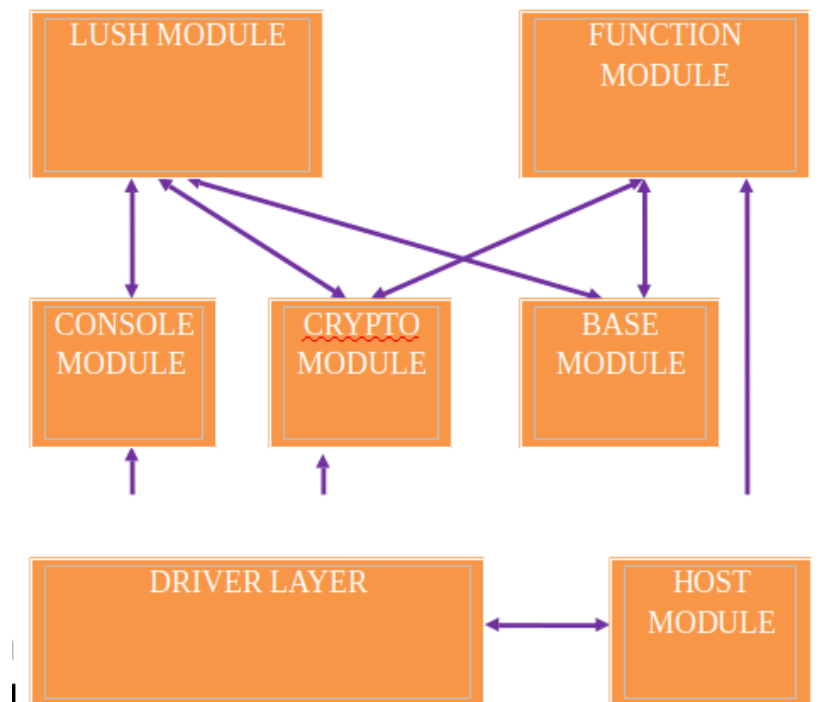
3.3 System Architecture

3.3.1 Purpose

The goal of this section is to describe the overall high level architecture of the HSM and its communication with the APIs and hosts.

3.3.2 Architecture

The following diagram explains the basic architecture of the Payment HSM –

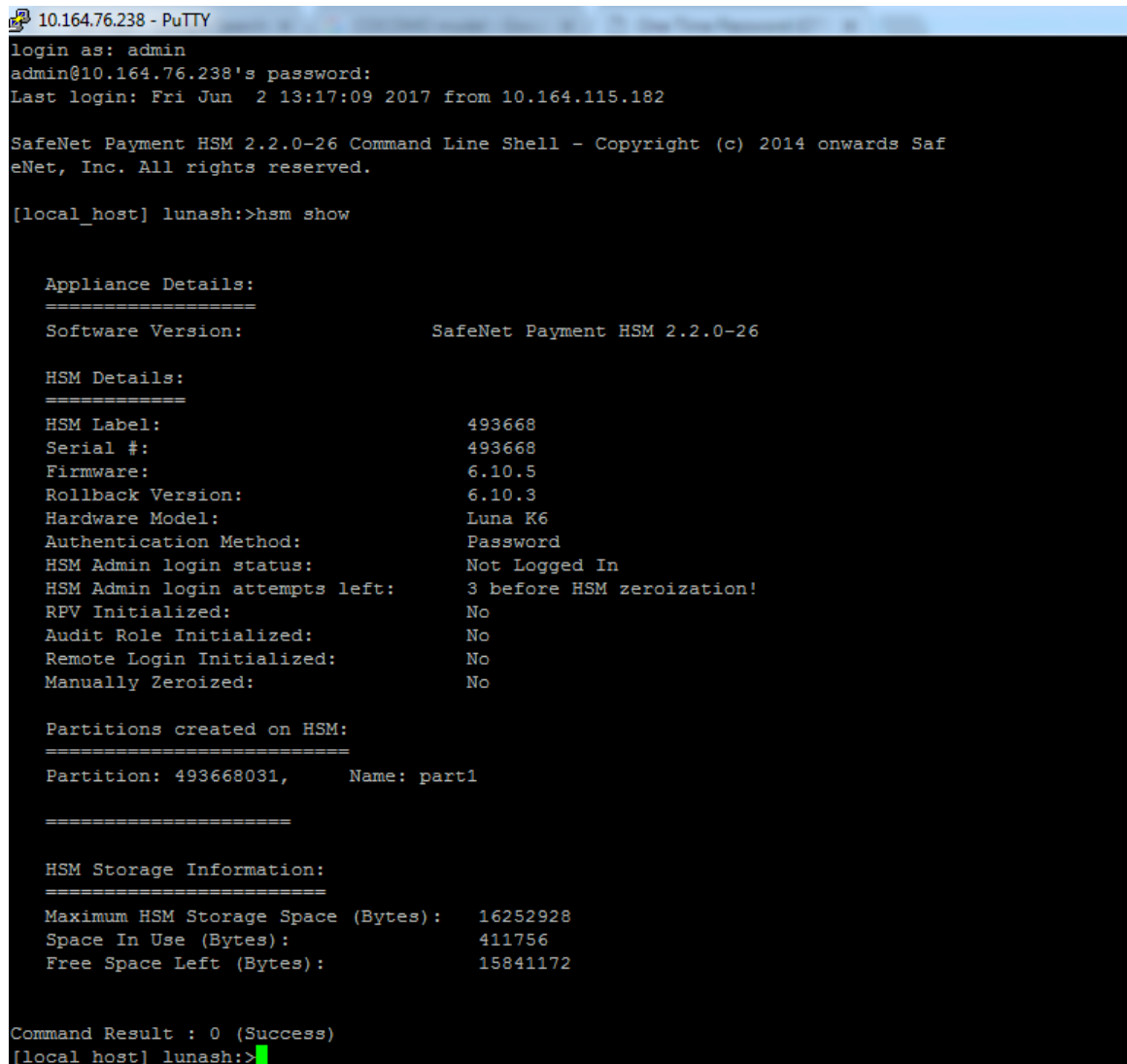


3.2 Software Architecture

3.3.2.1 Lush Module

The Lush stands for Luna Shell. It provides a custom console where you can do operations on HSM using some custom commands provided by the HSM API. It is quite similar to entering commands in Bash shell.

The implementation of this module is such that we can add new commands in the future without changing much in code. The lush module is written using C & C++. To use lush as an interface you need to connect to the HSM machine using SSH .



```
10.164.76.238 - PuTTY
login as: admin
admin@10.164.76.238's password:
Last login: Fri Jun  2 13:17:09 2017 from 10.164.115.182

SafeNet Payment HSM 2.2.0-26 Command Line Shell - Copyright (c) 2014 onwards SafeNet, Inc. All rights reserved.

[local_host] lunash:>hsm show

Appliance Details:
=====
Software Version:                SafeNet Payment HSM 2.2.0-26

HSM Details:
=====
HSM Label:                       493668
Serial #:                         493668
Firmware:                         6.10.5
Rollback Version:                 6.10.3
Hardware Model:                   Luna K6
Authentication Method:            Password
HSM Admin login status:           Not Logged In
HSM Admin login attempts left:    3 before HSM zeroization!
RPV Initialized:                   No
Audit Role Initialized:           No
Remote Login Initialized:         No
Manually Zeroized:                No

Partitions created on HSM:
=====
Partition: 493668031,      Name: part1
=====

HSM Storage Information:
=====
Maximum HSM Storage Space (Bytes): 16252928
Space In Use (Bytes):         411756
Free Space Left (Bytes):      15841172

Command Result : 0 (Success)
[local_host] lunash:>
```


3.3.2.2 Function Module

This module contains all the methods which control the working of the HSM. It handles all the host requests and redirects the calls to appropriate APIs. Important operations include creating the threads for smooth execution, loading and initializing the modules, initializing the communication between several components, setting up the entry and exit point of the requests, creating and managing the shared objects and many more. It also performs tasks like disabling/enabling functions under user control or automatically disabling functions based on usage or timeout. The module also counts function calls for statistical purposes.

3.3.2.3 Host Module

The Host module comprises of different modes of communication that user can use to make function request. Following are three different mode of communication as already described in previous sections

- Async Server
- Ethernet Server
- TCP/IP Server

3.3.2.4 Crypto Module

This module represents the software/hardware implementation of various symmetric and asymmetric cryptographic algorithms and Digest algorithms. Key generation algorithms are implemented here as well.

3.3.2.5 Base Module

This module contains core libraries which support the implementation of the various HSM functions.

3.3.2.6 Console Module

Console module consists of a list of modules that helps the HSM administrator to perform the console operations. The operations may include setting up the admin credentials of HSM, managing the keys, configuring hosts etc.

3.4 Policies & Tactics

Following is a list of Policies & Tactics that are used in the creation of this Software Design Specification document:

- **Choice of which specific product to use** (compiler, interpreter, database, library, etc. ...) –

C language, “gdb” compiler is used to develop the whole project. MS Excel sheet is used as input to the system (No database is used).

- **Coding guidelines and conventions** –

All the standard C language guidelines are used in developing the API.

- **Plans for testing the software** –

Testing is done manually and through automation in this project in which actual output is shown by proper functioning of the project.

- **Plans for maintaining the software** –

System can be easily maintained by the user as all the resources required by the system are easily available in the market.

- **Interfaces for end-users, software, hardware, and communications** –

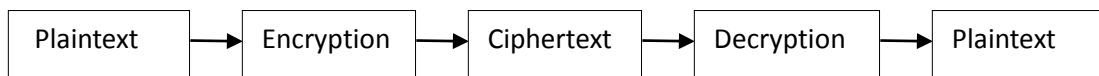
The user of this product is supposed to be fairly aware about the usage of the computers like software engineers. He should be able to work on Linux or Windows OS. The user of the system should also have in depth knowledge about HSMs like

Luna SA, Luna PCI etc. A person who has no knowledge of computers & HSMs will find it difficult to understand the system.

Chapter 4: My Role and Learning

4.1 Basic Cryptography

Cryptography is a method of storing and transmitting data in a form that only those its intended for can read and process. It is considered a science of protecting information by encoding it into an unreadable format. Cryptography is an effective way of protecting sensitive information as it is stored on media or transmitted through untrusted network communication paths.



4.1 Cryptography Flow

4.1.1 Cryptosystem

A cryptosystem encompasses all of the necessary components for encryption and decryption to take place. Pretty Good Privacy (PGP) is just one example of a cryptosystem.

A cryptosystem is made up of at least the following:

- Software
- Protocols
- Algorithms
- Keys

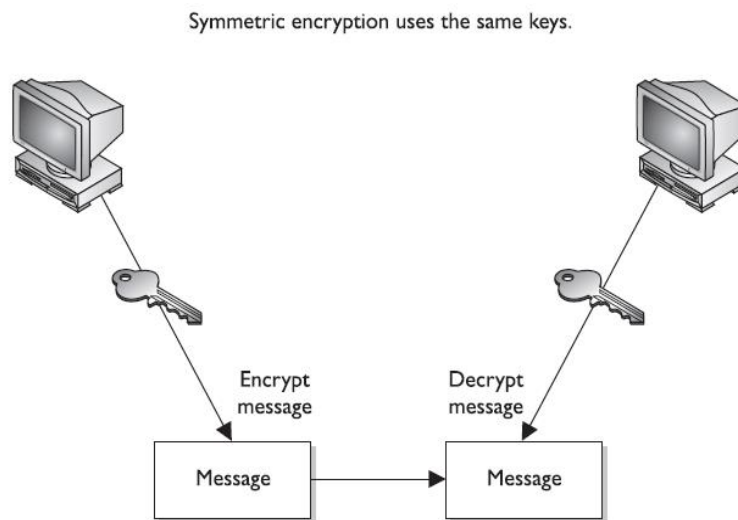
4.1.2 Services of Cryptosystems

- **Confidentiality** :Denies unauthorized parties access to information.
- **Authenticity** :Validates the source of the message, to ensure that the sender is properly identified.
- **Integrity** :Provides assurance that the message was not modified, accidentally or intentionally.

- **Nonrepudiation** :Establishes that a particular sender has sent the message so that they cannot deny having sent the message at a later date.

4.1.3 Symmetric Cryptography

In a cryptosystem that uses symmetric cryptography, the sender and receiver use two instances of the same key for encryption and decryption, as shown in Figure 4.2. So the key has dual functionality, in that it can carry out both encryption and decryption processes. Symmetric keys are also called secret keys, because this type of encryption relies on each user to keep the key a secret and properly protected. If an intruder were to get this key, the intruder could decrypt any intercepted message encrypted with this key.



4.2 Symmetric Cryptography

Strengths

- Much faster than asymmetric systems.
- Hard to break if using a large key size.

Weaknesses

- Requires a secure mechanism to deliver keys properly.

- Each pair of user needs a unique key, so as the number of individuals increases so does the number of keys, and key management can become overwhelming.
- Provides confidentiality but not authenticity or nonrepudiation.

Symmetric Cryptography Algorithms

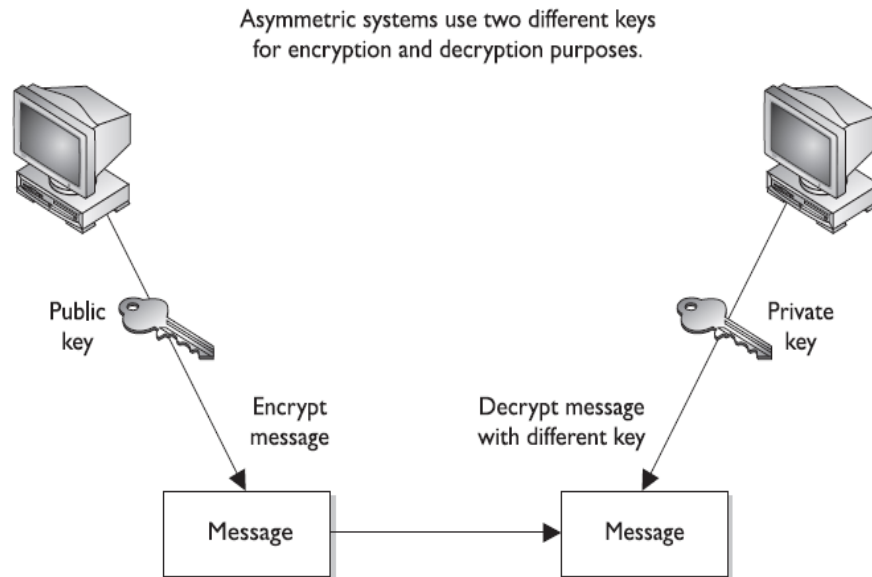
- Data Encryption Standard (DES)
- Triple-DES (3DES)
- Blowfish
- International Data Encryption Algorithm(IDEA)
- RC4, RC5, and RC6
- Advanced Encryption Standard (AES)

4.1.4 Asymmetric Cryptography

In symmetric key cryptography, a single secret key is used between entities, whereas in public key systems, each entity has different keys, or asymmetric keys. The two different asymmetric keys are mathematically related. If a message is encrypted by one key, the other key is required in order to decrypt the message. In a public key system, the pair of keys is made up of one public key and one private key. The public key can be known to everyone, and the private key must be known and used only by the owner. Many times, public keys are listed in directories and databases of e-mail addresses so that they are available to anyone who wants to use these keys to encrypt or decrypt data when communicating with a particular person. Figure 4.3 illustrates the use of the different keys.

The public and private keys of an asymmetric cryptosystem are mathematically related, but if someone gets another person's public key, she should not be able to figure out the corresponding private key. This means that if an evildoer gets a copy of Bob's public key,

it does not mean that she can now use some mathematical magic and find out Bob's private key. But if someone got Bob's private key, then there is big trouble no one other than the owner should have access to a private key.



4.3 Asymmetric Cryptography

Strengths

- Better key distribution than symmetric systems
- Better scalability than symmetric systems
- Can provide authentication and nonrepudiation

Weaknesses

- Works much more slowly than symmetric systems
- Mathematically intensive tasks

Asymmetric Cryptography Algorithm

- RSA
- Elliptic curve cryptosystem (ECC)
- Diffie-Hellman
- El Gamal
- Digital Signature Algorithm (DSA)

4.1.5 Data Encryption Standard

DES is a symmetric block encryption algorithm. When 64-bit blocks of plaintext go in, 64-bit blocks of ciphertext come out. It is also a symmetric algorithm, meaning the same key is used for encryption and decryption. It uses a 64-bit key: 56 bits make up the true key, and 8 bits are used for parity. When the DES algorithm is applied to data, it divides the message into blocks and operates on them one at a time. The blocks are put through 16 rounds of transposition and substitution functions. The order and type of transposition and substitution functions depend on the value of the key that is used with the algorithm. The result is 64-bit blocks of ciphertext.

4.1.6 DES Modes

- **Electronic Code Book (ECB)**-The simplest mode is the electronic codebook (ECB) mode, in which plaintext is handled one block at a time and each block of plaintext is encrypted using the same key.
- **Cipher Block Chaining (CBC)** -It combines the previous ciphertext block with the current message block before encrypting, hence making every cipher text dependent on previous ciphers.
- **Cipher Feedback (CFB)**-Instead of combining the previous cipher text to the plain text it combines some s bits plain text to the s bits of shift register contents. Initially an IV is used.

- **Output Feedback (OFB)**-The output feedback (OFB) mode is similar in structure to that of CFB, except that the output of the encryption function is fed back to the shift register in OFB, whereas in CFB the ciphertext unit is fed back to the shift register.
- **Counter Mode (CM)**-This mode uses a counter which increments the value of a register which is combined with the plaintext to get the cipher.

4.1.7 Triple-DES

Double-DES has a key length of 112 bits, but there is a specific attack against Double-

DES that reduces its work factor to about the same as DES. Thus, it is no more secure than DES. 3DES uses 48 rounds in its computation, which makes it highly resistant to differential cryptanalysis. However, because of the extra work that 3DES performs, there is a heavy performance hit. It can take up to three times longer than DES to perform encryption and decryption.

4.1.7 Message Integrity

It allows communicating parties to verify that received messages are authentic. When sensitive information is exchanged, the receiver must have the assurance that the message has come intact from the intended sender and is not modified inadvertently or otherwise.

4.1.8 Hash Functions

A hash function is a mathematical function that converts a numerical input value into another compressed numerical value. The input to the hash function is of arbitrary length but output is always of fixed length. Values returned by a hash function are called message digest or simply hash values.

It is in general always desired that the hash must be irreversible or computationally very difficult, means it must be pre image resistant. The hash must not collide for any two or more inputs, it should be unique.

4.1.9 Some Popular Hash Functions

Message Digest (MD)

The MD family comprises of hash functions MD2, MD4, MD5 and MD6. MD5 digests have been widely used in the software world to provide assurance about integrity of transferred file. For example, file servers often provide a pre-computed MD5 checksum for the files, so that a user can compare the checksum of the downloaded file to it.

Secure Hash Function (SHA)

Family of SHA comprise of four SHA algorithms; SHA-0, SHA-1, SHA-2, and SHA-3. Though from same family, there are structurally different.

The original version is SHA-0, a 160-bit hash function, was published by the National Institute of Standards and Technology (NIST) in 1993. It had few weaknesses and did not become very popular. Later in 1995, SHA-1 was designed to correct alleged weaknesses of SHA-0.

SHA-1 is the most widely used of the existing SHA hash functions. It is employed in several widely used applications and protocols including Secure Socket Layer (SSL) security.

4.2 Technologies Used/Studied

4.2.1 C Language

The language used to develop the API was C. It is the most popular programming language and has many advantages:

1. Modularity.
2. Portability.
3. Speed.
4. Flexibility.

- **Modularity:** Ability to breakdown a large module into manageable sub modules called as modularity that is an important feature of structured programming languages.

Advantages:

1. Projects can be completed in time.
 2. Debugging will be easier and faster.
- **Portability:** The ability to port i.e. to install the software in different platform is called portability. C language offers highest degree of portability i.e., percentage of changes to be made to the sources code is at minimum when the software is to be loaded in another platform.
 - **SPEED:** C is also called as middle level language because programs written in C language run at the speeds matching to that of the same programs written in assembly language so C language has both the merits of high level and middle level language and because of this feature it is mainly used in developing system software.

- **Flexibility:** C provides various levels of flexibilities. There are several kinds or types of declaration techniques for pointers, we can see the flexibility of For loop, there are several ways in which an inbuilt function can be used etc. Number of keywords helps the programmer to use according to need and hence again provides flexibility in terms of selecting the variable type , loops and many more.

4.2.2 Linux Basics

In order to work with the HSM codebase we had to deploy the source code in a virtual machine which running Ubuntu kernel. Hence it was of utter Importance that we have the grip over the basic Linux commands in order to work efficiently. I am listing some of the commands which are most generally used while working in the Linux environment-

- **pwd:**The pwd command reports the full path to the current directory.
- **cd:**The cd command is used to change the current directory in Linux and other unix like operating system.
- **mkdir:**The mkdir command is used to create new directories.
- **rmdir:**The rmdir command is used to remove empty directories in Linux and other Unix-like operating systems.
- **mv:**The mv command is used to rename and remove and move files and directories.
- **cat:**cat is one of the most frequently used command on Unix-like operating systems. It has three related functions with regard to text files: Displaying them, Combining copies of them and Creating new ones.
- **file:**The file command attempts to classify each file system object that is provided to it as an argument.
- **cp:**The cp command is used to copy files and directories. The copies become independent of originals.
- **rm:**The rm command is used to delete files and directories.

- **Ls:** The ls command is used for listing all the content of working directory.
- **chmod:** The chmod command is used to set access permissions on a file.
- **ps:** The ps command lists the currently running processes and their process identification numbers (PIDs).
- **uptime:** Shows the current time, how long the system has been running since it was booted up, how many user sessions are currently open and load averages.
- **uname:** Provides basic information about a system's software.
- **su:** Used to change a login session's owner without the owner having to first log out of that session.
- **whoami:** The whoami command returns the user name of the owner of the current login session.
- **man:** The man command is used to format and display the man pages. The man pages are a user manual that is by default built into most Linux distributions during installation. The man pages are usually viewed in a console or terminal window.
- **ifconfig:** ifconfig(interface configurator) command is used to initialize an interface, assign IP address to interface and enable or disable interface on demand. Can also be used to view IP address and Hardware/MAC address assign to interface and also MTU (Maximum transmission Unit) size.
- **ping:** PING(Packet Internet Groper) command is the best way to test connectivity between two nodes. Whether it is LAN or WAN. Ping use ICMP to communicate to other devices. In Linux ping command keep executing until you interrupt.
- **traceroute:** traceroute is network troubleshooting utility which shows number of hops taken to reach destination also determine packets traveling path.
- **route:** route command is used to show and manipulate ip routing table.

- **host:**host command is used to find name to IP or IP to name in IPv4 or IPv6 and also query DNS records.
- **alias:**The alias command makes it possible to launch any command or group of commands by entering a pre-set string.Multiple commands can be included in the same alias by inserting them within the same pair of quotation marks and separating them with semicolons.Aliases can even be created to call other aliases.We can make aliases permanent by entering them in the .bashrcfile in the root's home directory.

Chapter 5: Development & Snapshots

5.1 Development Overview & Code Snippets

The development task was accomplished using the C language. All the coding was done in the ‘Source Insight’ editor. It is a very helpful tool as it parses the whole project and is very helpful to search the references or variables across the different modules.

The source code cannot be shared as per the company policy.

5.2 Overall Flow of Task

Firstly I went through extensive exercise of understanding the basic knowledge of the product which includes-

1. Understanding the architecture of HSM.
2. Understanding how to deploy the code on a machine.
3. Learning basics about virtual machines and its environment setup.
4. Learning how to compile and run the source in debug mode.
5. Learning the process to test the API using the techniques already being used.
6. Creating test sheets manually for testing the API, learning to use various macros created to generate test cases in MS Excel.

Step by Step Description-

- **Learning about API** – Understanding the implementation of the API in order to modify it.
- **Coding** – Using Source Insight and C as language modified the code.

- **Compiling** – The code is compiled using a virtual machine and ‘gdb’ compiler.
- **Testing and Debugging** – The modified API is tested using ESM Tester or by using a command line executable ‘nttt’ it parses the input test file and fires it to the HSM. The ‘gdb’ compiler helps to debug the code as it can step through your source code line-by-line or even instruction by instruction.
 - I. Creating the test cases in excel.
 - II. Generating the test files readable by the HSM.
 - III. Firing the test cases to HSM which redirects it to the relevant API.

If the output is mismatched then we debug the code using the gdb compiler.

5.3 List of Tasks

5.3.1 Encipher API Hands On

After going through the training process for understanding the architecture of HSM, development environment and basic cryptography first task was assigned to me which comprised of understanding the Encipher API and doing some code changes in it as required. The code change included modifying the flow of encryption and decryption by adding/removing certain factors like variants and encryption modes. After generating the manual test case for the changed function I tested it for accuracy. The test was successful and task was accomplished.

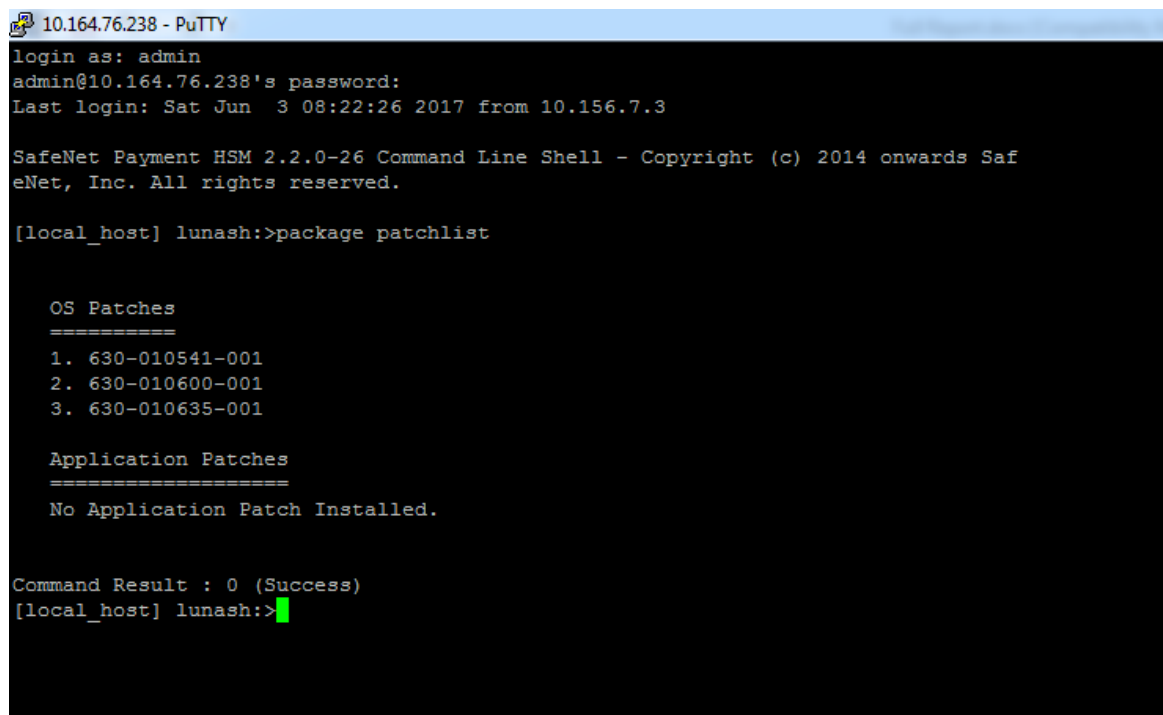
5.3.2 Re-Encrypt API Modification

This task was bit more challenging as it required a new function to be added inside the function space of the HSM. The new function contained some specific requirements which included changing the input and output data the function takes and returns respectively, modifying the mode of encryption and decryption, changing the variants used, updating the function space, enabling the function etc.

After the function was implemented the new test cases were generated and the function was tested. The task was successful and at the end it was reviewed by the project lead of our team.

5.3.3 Added lush command to list installed OS & Application patched on HSM

In this task, I was supposed to give customer a feature so that he can view the list of installed OS & Application patches on HSM. This feature was delivered as a Lush command “package patchlist”. The technology used to implement this feature were Shell script & C language.



```
10.164.76.238 - PuTTY
login as: admin
admin@10.164.76.238's password:
Last login: Sat Jun  3 08:22:26 2017 from 10.156.7.3

SafeNet Payment HSM 2.2.0-26 Command Line Shell - Copyright (c) 2014 onwards Saf
eNet, Inc. All rights reserved.

[local_host] lunash:>package patchlist

OS Patches
=====
1. 630-010541-001
2. 630-010600-001
3. 630-010635-001

Application Patches
=====
No Application Patch Installed.

Command Result : 0 (Success)
[local_host] lunash:>
```

5.3.4 Worked on Checksum calculation module

The checksum module was used to verify and check the integrity of the data. Checksum value is added corresponding to each data and file so that if the data is changed, so the checksum value. Some of the most common hash functions which are used to calculate checksum are MD5, SHA-1, SHA-2 & SHA-256.

5.3.5 Worked on Session timeout module

Timeout is used to re-authenticate the user after certain amount of interval or events. In Luna EFT, we have a fixed timeout period and a certain timeout after number of sensitive operations.

Below are the best practices to implement session timeout module:

- 1) Always set session timeout period as minimal as possible.
- 2) Always avoid infinite session timeout.
- 3) Always log the start and end of session in log file, so that in case of any issue you can identify the user responsible for the issue.

5.3.5 Removed all C code warnings from Luna-EFT codebase

I was assigned task to remove all C code warnings from the code base of Luna-EFT so that code compilations looks cleaner and to improve code quality. Initially there were around 4000 lines of wanings, after the completion of the task it was reduced to zero.

After removing all the warnings from the code base, the root directory Makefile of the codebase is protected with -Werror flag so that no further warnings can be introduced in the codebase. If further wanings are introduced then they will be treated as errors.

```
root@ubuntu: /mnt/git/source/s90/core/source/audit 12:54 AM
##### BEGIN BOILER PLATE #####
ifndef MKSRCDIR
include $(MKSRCBASE)/build/mk/target.mk
else
include $(MKSRCBASE)/build/mk/defaults.mk
##### END BOILER PLATE #####

# the library to build
MKLIB=libs90_audit

# Treating warning as an error
CFLAGS+= -Werror

MKMETA_LIBS+=crypto-libs/libetkms
# source to build the library with
MKSRC+=audit.c

# libraries to link with (automatically derives compiler and linker flags)
MKMETA_LIBS+=sys-libs/libos2compat \
s90/core/source/ktmmutil:libs90_ktmmutil \
s90/core/source/conutil:libs90_conutil \
s90/core/source/menus:libs90_menus \
s90/core/source/genutil:libs90_genutil \
s90/core/source/printer:libs90_printer \
s90/core/source/usb:libs90_usbcons

# files to be released
MKREL_INC+=audit.h
MKREL_FILES+=$(MKINSTALL_DIR)/core/$(MKLIB_SHARED)

##### BEGIN BOILER PLATE #####
"Makefile" 35L, 1112C
```

5.3.6 Bug fixes in the Luna-EFT

I was also regularly assigned tasks to remove bugs which were reported by the testing team. Fixing bugs led me understand the common pitfalls and other features of the product as well. After the fixing of the bug we were supposed to attach unit test result for the bug. After this the testing team verifies the bug to check if it is fixed and mark it as fixed. The whole process from reporting to fixing was made really easier using the JIRA tool.

5.3.7 Coverity fixes in Luna-EFT

Coverity is an open source tool which is used for static and dynamic code analysis. We used this tool to capture possible security bug and other issues so that we can fix them. Coverity generate a report after completing a scan of the code base. The report consists of list of possible bugs in the code with coverity ID, priority and other details. After taking appropriate action on the issue, you can mark it as fixed or as false positive.

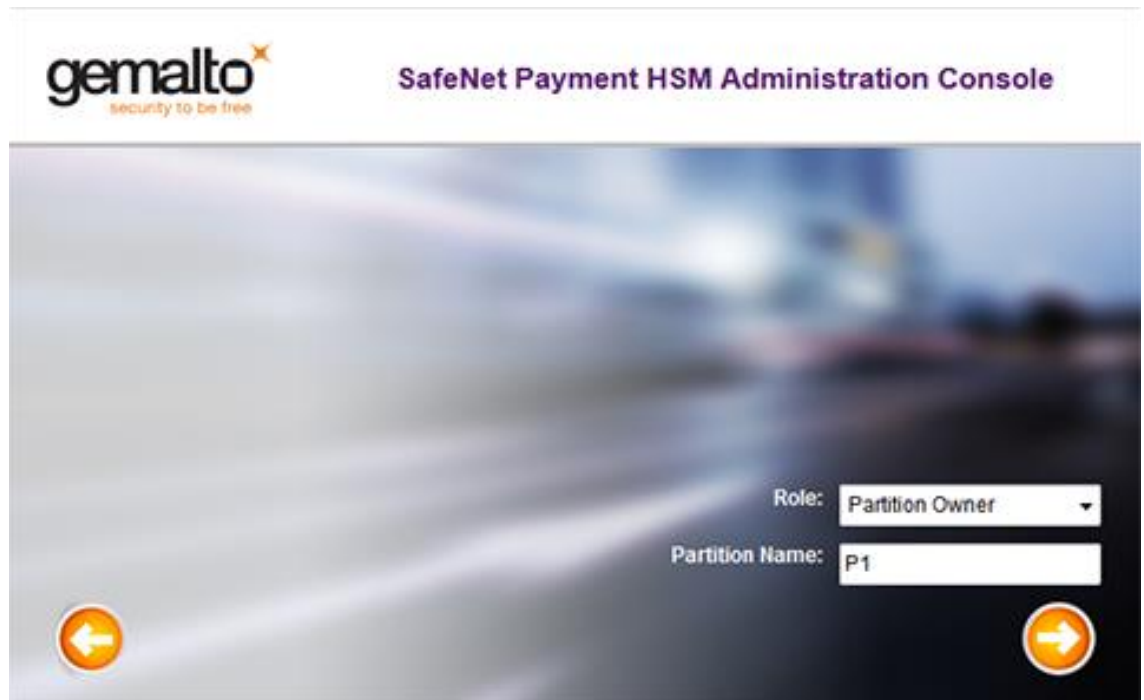
5.4 Snapshots

This section contains some of the permitted snapshots of HSM interface, coding snippets and some other snapshots.

- **Product & accessories of the product**



- Using Web console



- Lush Console

```

10.164.76.238 - PuTTY
login as: admin
admin@10.164.76.238's password:
Last login: Fri Jun 2 13:17:09 2017 from 10.164.115.182

SafeNet Payment HSM 2.2.0-26 Command Line Shell - Copyright (c) 2014 onwards Saf
eNet, Inc. All rights reserved.

[local_host] lunash:>hsm show

Appliance Details:
=====
Software Version:                SafeNet Payment HSM 2.2.0-26

HSM Details:
=====
HSM Label:                        493668
Serial #:                          493668
Firmware:                          6.10.5
Rollback Version:                  6.10.3
Hardware Model:                    Luna K6
Authentication Method:             Password
HSM Admin login status:           Not Logged In
HSM Admin login attempts left:    3 before HSM zeroization!
RPV Initialized:                   No
Audit Role Initialized:            No
Remote Login Initialized:          No
Manually Zeroized:                 No

Partitions created on HSM:
=====
Partition: 493668031,             Name: part1
=====

HSM Storage Information:
=====
Maximum HSM Storage Space (Bytes): 16252928
Space In Use (Bytes):             411756
Free Space Left (Bytes):           15841172

Command Result : 0 (Success)
[local_host] lunash:>

```

- **JIRA Console**

EFT 2.1.0 Sprint Report Switch report ▾

Mar 2016

Status Report * Issue add

Completed Issues

Key	Summary	Issue Type	Priority	Status
LUNAFT-15103 *	Support remote to test and from test to test interface	Story	High	CLOSED
LUNAFT-18953	Admin interface to activate single job for remote printing	Story	P3	CLOSED
LUNAFT-18984	Fix Ozone 0.11.4 support	Story	P1	CLOSED
LUNAFT-26523	Support Ozone on ESO	Story	High	CLOSED
LUNAFT-26741 *	Upgrade H2 database and test supporting with binary code	Story	Medium	CLOSED

Issues Removed From Sprint

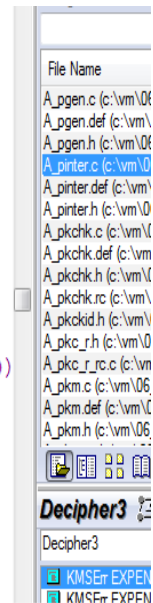
Key	Summary	Issue Type	Priority	Status
LUNAFT-17885	Stop running every process as privileged process	Story	Medium	CLOSED
LUNAFT-26524	Upgrade Release	Story	High	CLOSED

- **Source Code Snippet**

```

01027:     {
01028:         err = FN_DEA_DISABLED;
01029:         goto verrExit;
01030:     }
01031:
01032:     /*for(i=0; i < kekLen; i++) {
01033:         kekwrk[i] = kek[i] ^ ((keyLenKMAC == KMS_SINGLE_KEY || i % 2 == 0) ? 0x24
01034:             : keyLenKMAC == KMS_DBL_KEY ? 0xc0 : 0x30);
01035:     }*/
01036:     for(i=0; i < kekLen; i++) {
01037:         kekwrk[i] = kek[i];
01038:     }
01039:     if (kSpec == 0x21) {
01040:         if((err = LoadClearKey(dcp, kekwrk, KMS_DBL_KEY, KMS_DECRYPT)) != KMS_NO_ERR)
01041:             goto verrConvertExit;
01042:         //if((err = LoadIV(dcp, iv)) != KMS_NO_ERR)
01043:             //goto errConvertExit;
01044:         if((err = Decipher(dcp, kmac, Req45A0.eKEK_KMAC + prefixByteLen, keyLenKMAC, KMS_ECB_MODE | KMS_TRIPLE_MODE))
01045:             != KMS_NO_ERR)
01046:             goto verrConvertExit;
01047:     } else {
01048:         if((err = Decipher3(dcp, kmac, Req45A0.eKEK_KMAC + prefixByteLen,
01049:             keyLenKMAC, kekwrk, iv, KMS_ECB_MODE)) != KMS_NO_ERR) {
01050:             goto verrConvertExit;
01051:         }
01052:         memset(iv, '\0', sizeof(iv));           // Beware - residual IV returned here !
01053:     }
01054:

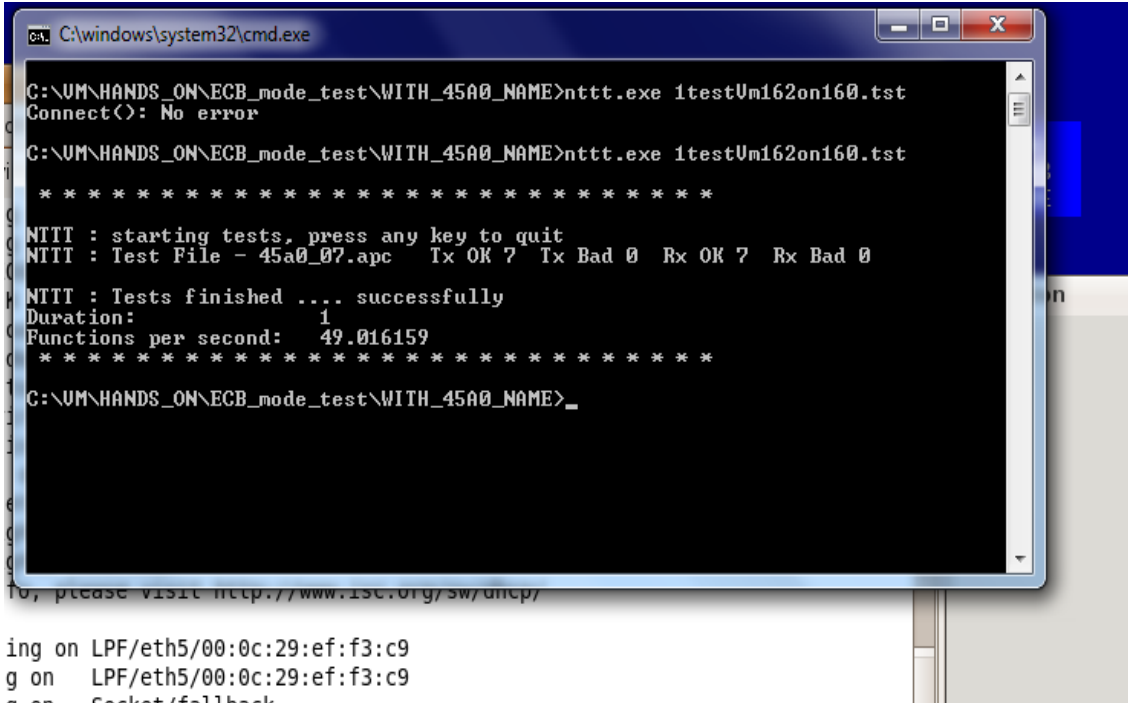
```



- **Test File**

```
1 |; Automatic function generation by createScriptFiles function
2
3 ; Function: EE0800
4
5 ; Script filename: EE0800_01.esm
6
7 ; Tester: r1kumar
8
9 ; This function enciphers the supplied data using a host-stored session key (DPK) supplied within a key-specifier.
10
11 ; Date = 15 Feb 2016
12
13 ; Purpose: Valid Field Content - Single Length ESM Stored DPK Index = 1, Mode = ECB, Data = 8 Bytes
14
15 write = 1
16 read = 2
17 nop = 3
18
19 .ccw write,burst
20 EEh, 08h, 00h, ;Func Code
21 00h, ;FM
22 02h, 01h, 01h, ;DPK Index
23 00h, ;Cipher Mode
24 00h, 00h, 00h, 00h, 00h, 00h, 00h, 00h, ;Input Chaining Value
25 08h, ;Data Var Length
26 30h, 30h, 30h, 30h, 30h, 30h, 30h, 30h, ;Data
27 .end
28
29 .ccw read, rcmp + burst + eoc
30 EEh, 08h, 00h, ;Func Code
31 00h, ;Return Code
32 00h, 00h, 00h, 00h, 00h, 00h, 00h, 00h, ;Output Chaining Value
33 08h, ;Data Var Length
34 7Dh, F6h, 12h, A4h, C3h, A1h, 56h, D6h, ;eDPK(Data)
35 .end
36
```


- **Testing API**



```
C:\windows\system32\cmd.exe
C:\UM\HANDS_ON\ECB_mode_test\WITH_45A0_NAME>nttt.exe 1testUm162on160.tst
Connect(): No error
C:\UM\HANDS_ON\ECB_mode_test\WITH_45A0_NAME>nttt.exe 1testUm162on160.tst
*****
NTTT : starting tests, press any key to quit
NTTT : Test File - 45a0_07.apc  Tx OK 7  Tx Bad 0  Rx OK 7  Rx Bad 0
NTTT : Tests finished .... successfully
Duration:          1
Functions per second: 49.016159
*****
C:\UM\HANDS_ON\ECB_mode_test\WITH_45A0_NAME>_

To, please visit http://www.isc.org/sw/uncp/
ing on LPF/eth5/00:0c:29:ef:f3:c9
g on LPF/eth5/00:0c:29:ef:f3:c9
n on Socket/fallback
```

Chapter 6: Conclusions

6.1 Conclusions

The tasks which I did during these four months were both dealing with the cutting edge technology being used today by the IT industry. The Luna EFT is the most trusted Payment HSM around the globe. Luna EFT is a high-performance, tamper-resistant solution that protects cryptographic keys and other sensitive information, such as customer PINs and cardholder data. By achieving PCI compliance, SafeNet's Luna EFT HSMs can not only assist them in meeting the basic compliance requirements but also leverage the technology to secure other key assets of their business. I got to understand the product and work on it.

I would conclude the report by saying that project work assigned to me has been completed to my satisfaction and I have added significant knowledge to myself.

Chapter 7: References

1. <https://www.scrumalliance.org/community/profile/agoddard>
2. <http://aaron.sanders.name/agile-team-members-roles-and-responsibilities/>
3. <https://docs.oracle.com/database/121/DBLIC/editions.htm#DBLIC109>
4. <http://www.vogella.com/tutorials/JavaPersistenceAPI/article.html>
5. <https://www.youtube.com/watch?v=kPBjIhpcZgE>
6. <http://ryanstutorials.net/linuxtutorial/>
7. https://docs.oracle.com/cd/B28359_01/license.111/b28287/editions.htm
8. <http://db-engines.com/en/system/Oracle%3BPostgreSQL>
9. <https://www.cs.cmu.edu/~pmerson/docs/OracleToPostgres.pdf>
10. <http://www.safenet-inc.com/about-safenet/>

