SAFENET LUNA NETWORK HARDWARE SECURITY MODULE

Project report submitted in partial fulfillment of the requirement for the Degree of Bachelor of Technology

in

Information Technology

By

Ranjan Kumar (141443)

Under the supervision of

Manjari Sharma

(HSM GP Integration,Gemalto)

to

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat,
Solan-173234, Himachal Pradesh

DECLARATION

I hereby declare that the work presented in this report entitled **“SAFENET LUNA NETWORK HARDWARE SECURITY MODEL”** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Information Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat, Solan is an authentic record of my own work carried out over a period from February to May 2018 under the supervision **Manjari Sharma,**Senior Project Lead (HSM General Purpose integration) in Gemalto.
The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Name:Ranjan Kumar
Roll No: 141443

(Signature Of Student)
Dated**:** …../ May /2018

CERTIFICATE

This is to certify that Ranjan Kumar, student of B. Tech in Information Technology of Jaypee University of Information Technology is currently pursuing industrial training in Gemalto from February 2018. He is working in HSM department on the product ***Hardware Security Module*** under the guidance of Manjari Sharma,Senior Project Lead(HSM GP Integration).

Name: Sudhakar Porwal
Designation: Sr. Manager Engineering
Department Name: HSM
Dated: …../May/2018

ACKNOWLEDGEMENT

I consider it a great privilege and honor to have had the opportunity to undergo the industrial training in Gemalto SafeNet Inc. I would like to offer my heartiest thanks to **Manjari Sharma**, Senior Project Lead,HSM GP Integration,Gemalto for being the mentor throughout my industrial training. I am greatly indebted to **Sudhakar Porwal** (Sr. Manager, Engineering), **Md. Arif**(Technical Lead) **Somya Bansal** (Senior Software Engineer), **Deepak Gupta** (Software Engineer),**Chanpreet Grover**(Software Engineer),**Sahil Agarwal**(Software Engineer) , **Priyanka Batra** (Software Engineer),**Harshit Jain**(Software Engineer) at Gemalto SafeNet Inc. for their most valuable suggestions, constant encouragement, and guidance during the period of this training.

At last, I am greatly thankful to all my seniors and colleagues at Gemalto SafeNet Inc. for extending their constant cooperation which went a long way towards the completion of this training. I am also grateful to **Dr. Vivek Sehgal** who was also there for help when I needed it. His guidance let me carry out the training with ease.

Date: …../May/2018

Ranjan Kumar (141443)

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

This project titled “**SAFENET LUNA NETWORK HARDWARE SECURITY MODULE”** emphases on hardware security modules created to safeguard critical cryptographic keys and to speed up sensitive cryptographic actions across an extensive range of security applications.

SafeNet HSMs fall into three classes:

•**SafeNet PCI HSM** is a card-type HSM that installs into the PCI slot(s) of a host computer. Several SafeNet PCI HSMs can coexist in single host system. Each SafeNet PCI HSM provisions one HSM partition.

•**SafeNet USB HSM** is a desktop HSM unit that links locally to a host computer via USB interface. Multiple SafeNet USB HSMs can be linked via USB connection. To each SafeNet USB HSM supports 1 HSM partition.

•**SafeNet Network HSM** is a self-contained, network attached HSM appliance, comprising an HSM card like to SafeNet PCI HSM, and normally resides in an equipment rack in a server room and is log on remotely via secure administrative and client links. Apiece SafeNet Network HSM supports many HSM partitions, the quantity governed by bought licensesThis project titled **SAFENET LUNA HSM** emphases on hardware security modules created to safeguard critical cryptographic keys and to speed up sensitive cryptographic actions across an extensive range of security applications.

CHAPTER – 1
INTRODUCTION

1.1 Introduction

A well planned, systematically executed industrial internship is helpful in inculcating a good work culture. It provides a linkage between students and the industry in order to develop awareness of the industrial approach to problem solving based on broad understanding of operations of the industrial organizations. The following report describes the activities carried out during 4 months, full-time internship at Gemalto SafeNet Inc. The document contains information about the organization and the responsibilities performed throughout the period from February to May 2018. The objective of this report is to reflect upon the experiences collected during the internship from the perspective of a B.Tech CSE student. The first part of the report offers an overview of the organization, followed by the project overview. Following, it proceeds to describe in some detail the most relevant details of the. Finally, the report wraps up with conclusions from the experience.

1.2 Company Profile

SafeNet, Incorporation is an information security company with headquarters established in Belcamp, Maryland, United States. The firm is now the fifth largest seller in the security market and third largest supplier of information security solutions in the world with returns approximately $500M. Gemalto proclaimed On 8 August 2014,that it had signed a definitive agreement to acquire 100% of the share capital of SafeNet. This is notably one of the leading suppliers of encryption technology to the United States Government. Now, this is part of Gemalto and falls under the brand name of Gemalto.Safenet provides enterprise authentication, data encryption, key management and software monetization. SafeNet is a leading global provider of data protection.Companies like Fortune 500 global corporations and many government agencies have turned to SafeNet to secure and protect their most critical data assets and intellectual property for over 25 years,. SafeNet focuses on the protection of high value and critical information throughout its life span, takinf from the data center to the cloud. More than 25,000 customers across commercial enterprises and government agencies trust SafeNet to protect and control access to sensitive data, manage risk, ensure amenableness, and secure virtual and cloud environments.

SafeNet is the only company which provides all the components needed to secure and manage digital identities, thereby ensuring lesser costs in terms of both deployment and ongoing maintenance. In addition, since all elements of the solution are provided by a single vendor, any risk of interoperability failure is minimized.

CHAPTER – 2

PROFILE OF THE PROJECT

2.1 HSM Overview

An Safenet LUNA HSM is a Hardware Security Module. It has cryptographic, storage, and access-control functions, that allow cryptographic operations to be performed, and segregated within a secure physical hardware boundary, while offloading such functions from the general-purpose pathways of the host or client. Here are basic elements common to SafeNet HSMs.

2.1.1 Volatile and non-Volatile Data Storage

SafeNet HSM can store two type of data: volatile and non-volatile data.

- Non-volatile data contains identification parameters and data objects (such as certificates and keys) that are stored for use in lon term. The objects stored exist on the HSM until we delete them by ourselves . Also Non-volatile objects are stored in encrypted form.
- Volatile data are usually data which is lost when it is not used.When the HSM loses power, or when a session closes then, the volatile data is lost. Volatile data contains decrypted copies of non-volatile data.

Generally, The Cryptographic Keys and objects are stowed under multiple layers of encryption and are decrypted inside the physical bounds of the HSM, only into volatile/session storage, and only while being used. Some events that remove power to the HSM, instantly erases volatile objects.

2.1.2 Initialization

SafeNet Hardware Security Model should be initialized before use, for the first time (It is also required to initialize HSM when multiple login attempt fails and Security Officer (SO) account is locked).

The Initialization Of Hardware Security Module creates up a complete set of different HSM parameters.This includes identification and authentication of HSM Administrator i.e Security Officer (SO) and application Partition Crypto Officer(CO) and Crypto User(CU) who can access and create and use HSM Partition objects and also setting up the domain for the HSM Appliance .

Like keys, certificates, encrypted data, etc.Safenet provides the lunacm utility program on most of the supported platforms like windows,linux,sparc,AIX etc as some application like PKI and other cryptographic product vendors don't have the capability to initialize Safenet HSM.

When a SafeNet Luna HSM is initialized,no any other person can have accees to it unless the password is not provided which unlocks the partition.It has also the capability to reiniatialise the HSM which destroys all the data on the HSM.

2.1.3 Authentication methods

SafeNet Luna HSMs come with factory configured to be either:

- Password authenticated – In this we use the password to access the HSM uses typed text strings to access the HSM and authenticate to all roles on the HSM.Its advantage is greater convenience.
- PED authenticated – It uses physical tokens which is called PED Keys, inserted into a PIN Entry Device, or PED, to access the HSM and authenticate all roles on the HSM .Its advantage is greater security.

Safenet Luna HSM in the operating mode can't be changed from Password-authentication method to PED-authentication method or from PED-authentication method Password-authentication method But the only exception is with the SafeNet Luna Backup HSM, which sets its mode automatically at the time of backup, to match the authentication scheme of the Safenet HSM being backed up. The Backup HSM can do Backup and Restore only, and it has no ability to perform cryptographic operations.

CHAPTER – 3

PRODUCT OVERVIEW

3.1 About Safenet Luna HSM

SafeNet Luna Network HSM is an Hardware Security Module which is attached to Ethernet designed to store and protect critical cryptographic keys.It is used to accelerate sensitive cryptographic operations across a wide range of security applications. SafeNet Luna HSM has many features which enhances the security, connectivity, and ease-of-administration in dedicated and shared security applications.

SafeNet Network HSM falls under one of two model families, according to the level of authentication and access control. SafeNet Network HSM is factory configured to operate as either:

- Password Authenticated version(PW)- which is equivalent to FIPS 140-2 level 2, using passwords only for authentication and access control.
- Pin Entry Device (a Trusted Path) Authenticated version- which is equivalent to FIPS 140-2 level 3, which requires SafeNet Luna Pin entry Device and PED Keys for authentication and access control.

3.2 Physical Appearance

The standard appliance is the 1U-high, rack-mount device:

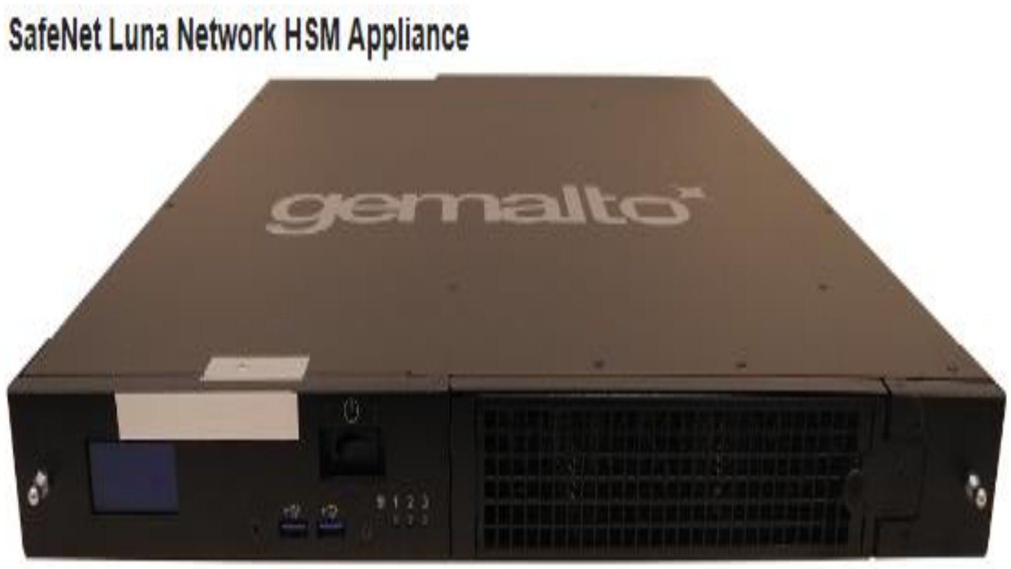


Fig. 3.1 HSM Appliance

3.2.1 Front View

First, the front; this illustration shows the appliance with its snap-on decorative bezel removed.

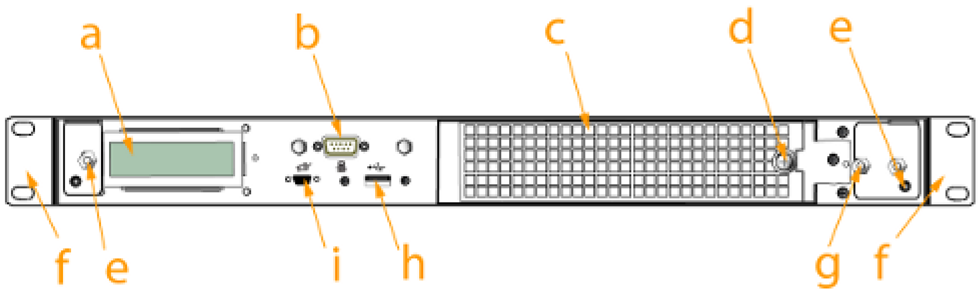
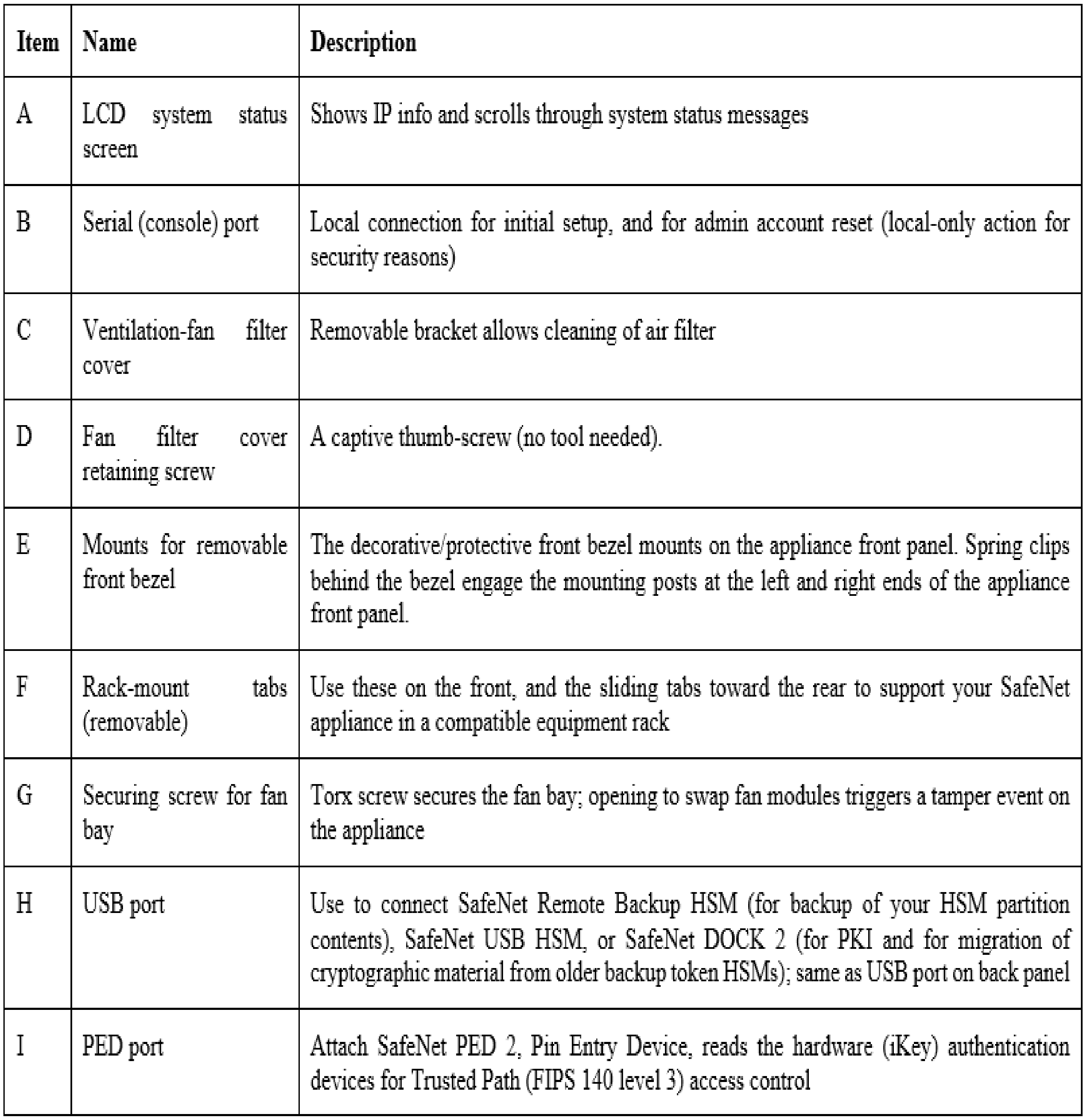


Fig. 3.2 HSM Appliance Front View



Item	Name	Description
A	LCD system status screen	Shows IP info and scrolls through system status messages
B	Serial (console) port	Local connection for initial setup, and for admin account reset (local-only action for security reasons)
C	Ventilation-fan filter cover	Removable bracket allows cleaning of air filter
D	Fan filter cover retaining screw	A captive thumb-screw (no tool needed).
E	Mounts for removable front bezel	The decorative/protective front bezel mounts on the appliance front panel. Spring clips behind the bezel engage the mounting posts at the left and right ends of the appliance front panel.
F	Rack-mount tabs (removable)	Use these on the front, and the sliding tabs toward the rear to support your SafeNet appliance in a compatible equipment rack
G	Securing screw for fan bay	Torx screw secures the fan bay; opening to swap fan modules triggers a tamper event on the appliance
H	USB port	Use to connect SafeNet Remote Backup HSM (for backup of your HSM partition contents), SafeNet USB HSM, or SafeNet DOCK 2 (for PKI and for migration of cryptographic material from older backup token HSMs); same as USB port on back panel
I	PED port	Attach SafeNet PED 2, Pin Entry Device, reads the hardware (iKey) authentication devices for Trusted Path (FIPS 140 level 3) access control

Table 3.1 Description Of HSM Appliance Front View

3.2.2 Rear View

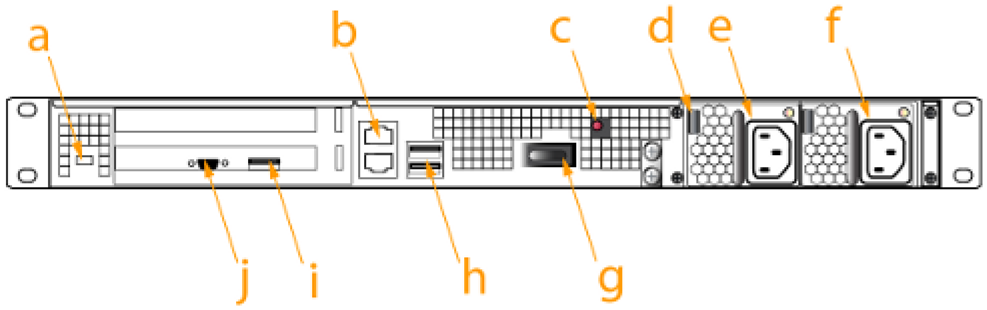


Fig. 3.2 HSM Appliance Rear View

Table 3.2 Description Of HSM Appliance Rear View

Item	Name	Description
a	Kensington Security Slot	Attach an industry-standard locking cable for additional physical security.
b	Ethernet ports	For network connection of your SafeNet appliance.
c	Decommissioning button	Recessed for safety; renders HSM contents unusable.
d	Power supply release tab	Press tab to release the catch, and slide the power supply out.
e	Removable power supply	One of two redundant power supplies.
f	Second removable power supply	The other of two redundant power supplies.
g	Start/stop switch	Use to stop the system if the command-line shutdown is not available; use to restart the system if it has been switched off.
h	USB ports	Use to connect SafeNet Remote Backup HSM (for backup of your HSM partition contents), SafeNet USB HSM, or SafeNet DOCK 2 (for PKI and for migration of cryptographic material from older backup token HSMs); same as USB port on front panel.
i, j	Unused ports	These ports are not used for SafeNet Network HSM; we recommend that you do not remove the covers that were installed at the factory.

3.3 Technical Specifications

3.3.1 Cryptographic Api's

- PKCS#11 v2.01
- Microsoft CAPI v2.0,
- Java JCA/JCE CSP
- OpenSSL

3.3.2 Cryptographic Hardware

Validation

- FIPS 140-2 Level 3 validated

  — certificate number 375
- FIPS 140-2 Level 2 validated

  — certificate number 436

3.3.3 Cryptographic Functions

- True hardware accelerated random number generation
- Symmetric and asymmetric key pair generation
- Encryption and decryption
- RSA
- Digital signing

3.3.4 Cryptographic Performance

Over 1200 1024-bit RSA cryptographic operations per second

3.3.5 Cryptographic Algorithms

- Asymmetric Key with Diffie Hellman (1024-4096 bit)
- RSA (512-4096 bit) and (PKCS#1 v1.5, OAEP PKCS#1v2.0)
- Digital Signing via RSA (1024-4096-bit)
- DSA (512-1024-bit), (PKCS#1 v1.5) and Symmetric Keys through 3DES (double and triple key lengths)
- AES
- RC2, RC4, RC5, CAST-128
- Hash Digest is SHA-1, SHA-2 (160, 256, and 512), and MD-5
- Message Authentication Codes (MAC) are HMAC-MD5, HMAC-SHA-1, SSL3-MD5-MAC, and SSL3-SHA-1-MAC

3.3.6 Physical Characteristics

Connectivity

- 2x 10/100 Ethernet, CAT5, UTP
- Luna PED authentication port
- Local serial console port
- Luna Token PC-Card slot

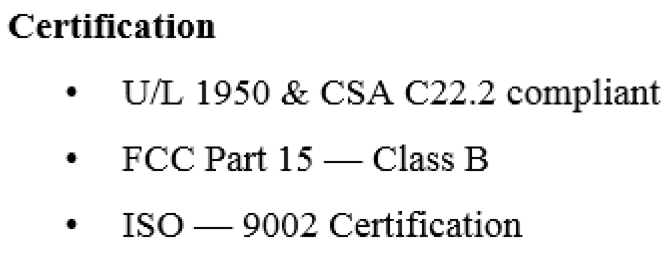
Dimensions

- 2U full-length 19" rack mount chassis (ANSI/EIA-310-D compliant)
- 19.0" x 20.6" x 3.45" (482.6 mm x 523.2mm x 87.7mm)
- 35lb (15.9kg)

Removable Storage

- PC Card Type II Slot, 5V (+/- 0.25V)
- Temperature
- Operating 0°C to 40°C,
- Storage -20°C to +65°C

3.3.7 Regulatory Standards



Certification

- U/L 1950 & CSA C22.2 compliant
- FCC Part 15 — Class B
- ISO — 9002 Certification

CHAPTER – 4

PRODUCT CONFIGURATION

To ensure a trouble-free configuration, the following steps are performed in the order indicated:

1. Planning Configuration
2. Configure Network Settings
3. Initialize the HSM
4. Set the HSM Policies
5. Create Application Partitions
6. Set the Partition Policies for Legacy Partitions
7. Create a Network Trust Link between the Client and the Appliance
8. Enable the Client to Access a Partition
9. Configure PPSO Application Partitions
10. Set the Partition Policies for PPSO Partitions

4.1 Planning Configuration

In SafeNet Network HSM,the roles is mainly divided into two categories:

- roles to access the appliance that contains the HSM and that provides the network connectivity; these are accessed through SSH or local serial connection, via the LunaSH or "lunash" command line
- roles that access the HSM include:
  1. 'HSM Administrator' or 'Security Officer' (SO) responsible for initialization of the HSM, setting and changing of global Policies (based on the HSM's Capabilities), creation and deletion of application partitions.
  2. ‘Application partition Security Officer' (SO) responsible for creating other roles in the partition, resetting passwords, setting and changing partition-level Policies (based on the HSM's and the partition's Capabilities).
  3. ‘Application partition Crypto Officer' [Mandatory], responsible for creating the Crypto User role, and for creating and modifying cryptographic objects in the HSM partition.

4. ‘Application partition Crypto User', responsible for using cryptographic objects (encrypt/decrypt, sign/verify...) in the HSM partition

4.2 Configure Network Settings

HSM Appliance Network Parameters

- the static IP address assigned to this device
- the hostname for the HSM appliance (registered with network DNS)
- domain name
- default gateway IP address
- DNS Name Server IP address
- search Domain name(s)
- device subnet mask
- Ethernet device (use eth0, which is the uppermost network jack on the HSM ppliance back panel, closest to the power supply)

4.3 Initialize The HSM

Safenet Luna Network HSM is initialized to configure and set up the necessary identities, ownership and authentication on the HSM. The initialization is mandatory before using the HSM.For this,the **hsm init command** is used.

For an HSM with Password Authentication, there is need to provide a label, password, and cloning domain.

Type the hsm init command at the prompt, supplying a text label for the new HSM.

```
lunash:> hsm -init -label myLuna
> Please enter a password for the security officer
> ********
Please re-enter password to confirm:
> ********
Please enter the cloning domain to use for initializing this HSM :
> ********
Please re-enter domain to confirm:
> ********
```

```
CAUTION:  Are you sure you wish to re-initialize this HSM?
All partitions and data will be erased.
Type 'proceed' to initialize the HSM, or 'quit' to quit now.
>proceed
'hsm - init' successful.
```

When activity is complete, the system displays a “success” message.

4.4 Set The HSM Policies

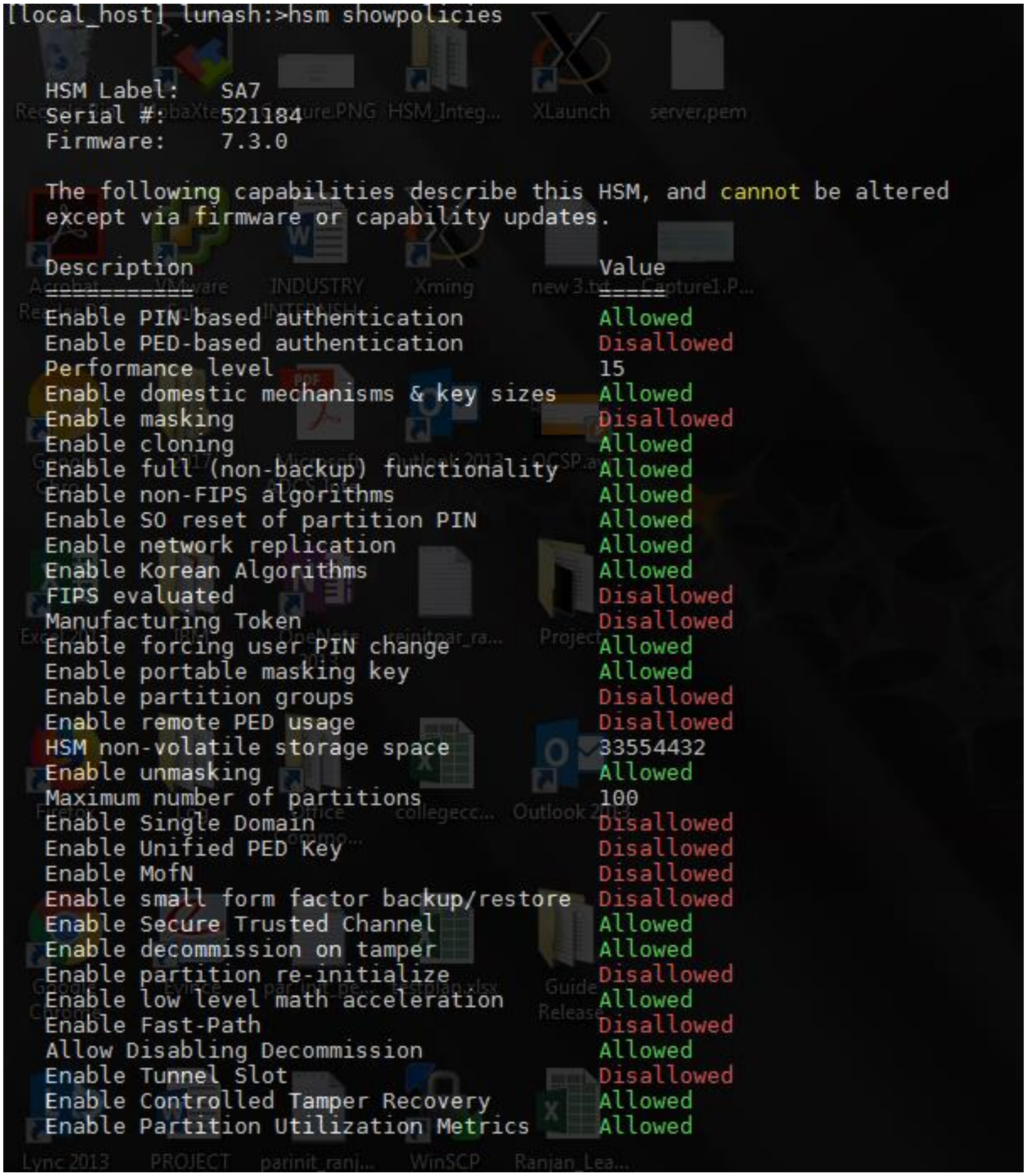
Type the hsm showPolicies command, to display the current policy set for the HSM.

In order to change HSM policies, the HSM SO must first login.

```
lunash:> hsm login
```

To modify a policy setting to comply with operational requirements, type:

```
lunash:> hsm changePolicy -policy <policyCode> -value <policyValue>
```

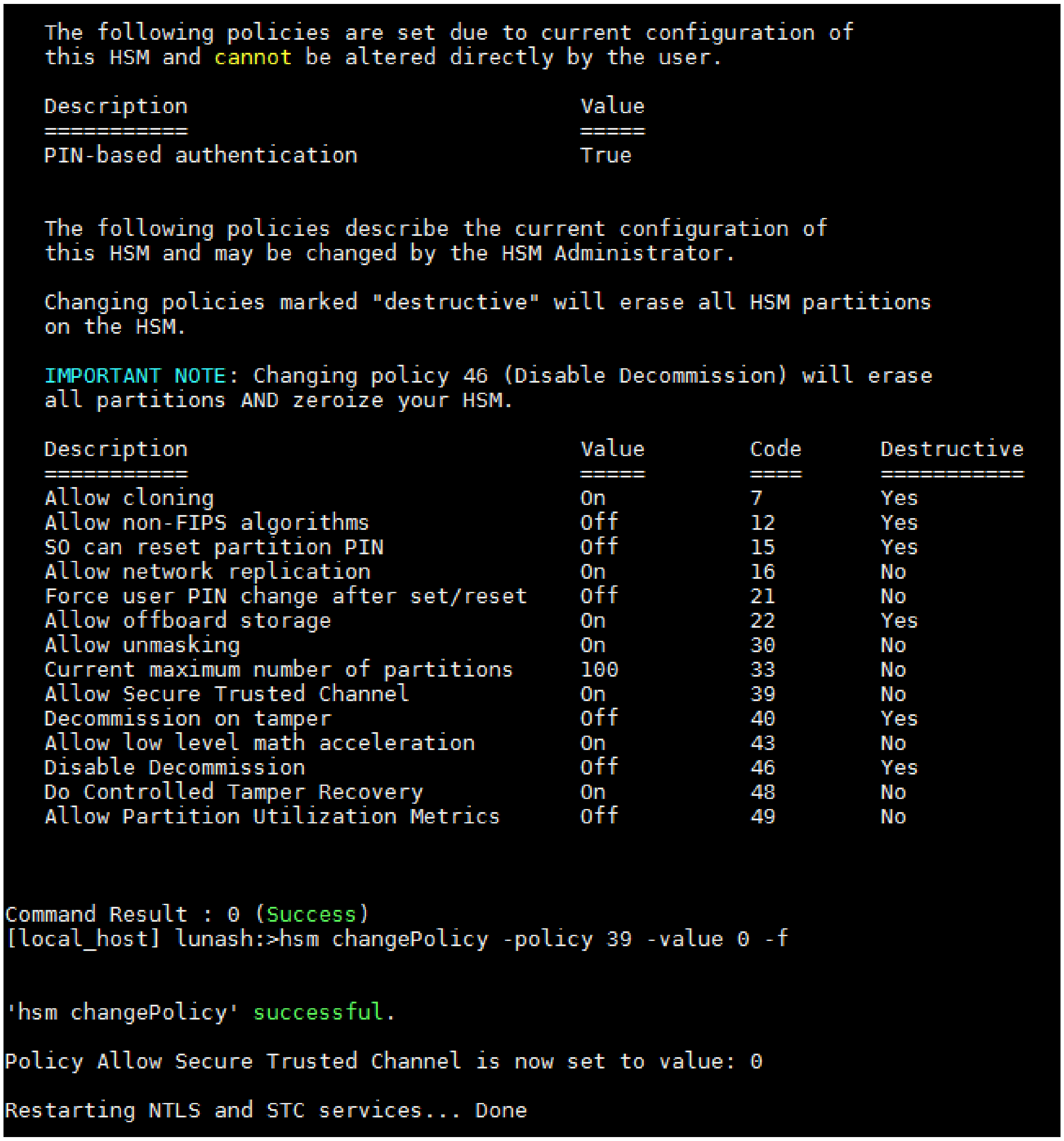


```
[local_host] lunash:>hsm showpolicies


   HSM Label:    SA7
   Serial #:     521184
   Firmware:     7.3.0

   The following capabilities describe this HSM, and cannot be altered
   except via firmware or capability updates.

   Description                                    Value
   ===========                                    =====
   Enable PIN-based authentication                Allowed
   Enable PED-based authentication                Disallowed
   Performance level                              15
   Enable domestic mechanisms & key sizes         Allowed
   Enable masking                                 Disallowed
   Enable cloning                                 Allowed
   Enable full (non-backup) functionality         Allowed
   Enable non-FIPS algorithms                     Allowed
   Enable SO reset of partition PIN               Allowed
   Enable network replication                     Allowed
   Enable Korean Algorithms                       Allowed
   FIPS evaluated                                 Disallowed
   Manufacturing Token                            Disallowed
   Enable forcing user PIN change                 Allowed
   Enable portable masking key                    Allowed
   Enable partition groups                        Disallowed
   Enable remote PED usage                        Disallowed
   HSM non-volatile storage space                 33554432
   Enable unmasking                               Allowed
   Maximum number of partitions                   100
   Enable Single Domain                           Disallowed
   Enable Unified PED Key                         Disallowed
   Enable MofN                                    Disallowed
   Enable small form factor backup/restore        Disallowed
   Enable Secure Trusted Channel                  Allowed
   Enable decommission on tamper                  Allowed
   Enable partition re-initialize                 Disallowed
   Enable low level math acceleration             Allowed
   Enable Fast-Path                               Disallowed
   Allow Disabling Decommission                   Allowed
   Enable Tunnel Slot                             Disallowed
   Enable Controlled Tamper Recovery              Allowed
   Enable Partition Utilization Metrics           Allowed
```

Fig.4.1 HSM Showpolicies

Above is the Output of the HSM ShowPolicies command when executed in Lunash.

Below is the illustration of HSM ChangePoliciy command when done on lunash.



```
   The following policies are set due to current configuration of
   this HSM and cannot be altered directly by the user.

   Description                               Value
   ===========                               =====
   PIN-based authentication                  True


   The following policies describe the current configuration of
   this HSM and may be changed by the HSM Administrator.

   Changing policies marked "destructive" will erase all HSM partitions
   on the HSM.

   IMPORTANT NOTE: Changing policy 46 (Disable Decommission) will erase
   all partitions AND zeroize your HSM.

   Description                               Value        Code      Destructive
   ===========                               =====        ====      ===========
   Allow cloning                             On           7         Yes
   Allow non-FIPS algorithms                 Off          12        Yes
   SO can reset partition PIN                Off          15        Yes
   Allow network replication                 On           16        No
   Force user PIN change after set/reset     Off          21        No
   Allow offboard storage                    On           22        Yes
   Allow unmasking                           On           30        No
   Current maximum number of partitions      100          33        No
   Allow Secure Trusted Channel              On           39        No
   Decommission on tamper                    Off          40        Yes
   Allow low level math acceleration         On           43        No
   Disable Decommission                      Off          46        Yes
   Do Controlled Tamper Recovery             On           48        No
   Allow Partition Utilization Metrics       Off          49        No



Command Result : 0 (Success)
[local_host] lunash:>hsm changePolicy -policy 39 -value 0 -f


'hsm changePolicy' successful.

Policy Allow Secure Trusted Channel is now set to value: 0

Restarting NTLS and STC services... Done
```

Fig.4.2 HSM ChangePolicy

4.5 Create Application Partitions

Choose Partition Type

The options are:

- Legacy-style application partitions are owned and administered by the HSM SO, who retains complete control.
- PPSO-style application partitions each have their own SO, independent of the HSM SO, and all control except partition creation and deletion resides with the Per-Partition SO

4.6 Set The Partition Policy Settings

First, display the policies (default) of the created legacy-style application Partition. In order to run the partition showPolicies command, you do not need to be logged into the HSM Partition. But to change policy of the Hardware Security Module or any individual Partition, login as HSM SO or Partition SO is mandatory.

To display the current partition policy settings

1. Open a LunaSH session on the appliance.
2. Enter the following command to display current partition capability and policy settings. Capabilities are factory settings. Policies are the means of modifying the adjustable capabilities:

   lunash:> partition showPolicies -partition <partition name>

Changing the Partition Policy Settings

After checking the Policy settings, we can now change the Partition Policy for a given Partition, if it is required.

To change a partition policy

1. Open a LunaSH session on the appliance.
2. Enter the following command to change a Partition Policy:

lunash:>partition changepolicy -partition <name of HSM Partition> -policy <policy_code> -value <new_ policy_value>

4.7 Create Network Trust Link Between The Client And The Appliance

The first step in preparing your clients to use the cryptographic resources provided by the HSM appliance is to create a secure network trust link (NTL) between the client and the appliance. After you create the NTL link between the client and the appliance, you can configure links to individual partitions on the appliance using NTL or Secure Trusted Channel (STC)
NTL is discussed in detail in upcoming topics.

4.8 Enable The Client To Access A Partition

After creating the network trust link between the client and the appliance, enable the client to access a specific partition on the appliance. Configure the client to access a partition using an NTL or STC connection, as follows:
NTL client-partition links: Assign the partition to a specific client using the LunaSH client assignpartition command. This allows the client to create NTL connections to the partition to perform cryptographic operations
STC client-partition links: Enable Secure Trusted Channel (STC) on the client and partition. This disables the NTL connection to the partition, and replaces it with an STC connection.

4.8.1 To Assign A Client To A Partition

1. Launch LunaSH and log in as the HSM SO.
2. Enter the following command to assign a client to a partition:
3. client assignPartition **-**client <clientname> **-**partition <partition name>
4. Enter the following command to verify that the HSM Partition is assigned to the client.
5. client show -client <clientname>

4.9 Configure PPSO Application Partitions

The configuration tasks needed to be performed depend on whether the partition is password-authenticated or PED-authenticated as follows:

Authentication Tasks

Password

1. Partition Security Officer is Iniatialized.
2. Crypto Officer Roles on a PW-Authorized PPSO Partition is initialized.
3. Finally,the Crypto User Role on a PW- Authorized PPSO Partition is initialized.

PED

1. Initialize the Partition SO and Crypto Officer Roles on a PED- Authorized PPSO Partition
2. Initialize the Crypto User Role on a PED- Authorized PPSO Partition
3. Activate a PED- Authorized PPSO Partition for the Crypto Officer Role or Activate a PED- Authorized PPSO Partition for the Crypto User Role.

4.10 Set The Partition Policies For PPSO Partitions

First, display the policies (default) of the created legacy-style application Partition.There is no login required into the HSM to run the partition showPolicies command.But to change the policy of HSM or any single partition ,login is mandatory.

To display the current partition policy settings

1. Open a Lunacm session.
2. Enter the following command to display current partition capability and policy settings. Capabilities are factory settings. Policies are the means of modifying the adjustable capabilities:

   partition showpolicies -partition <partition_name>

Having viewed the Policy settings, you can now modify a Partition Policy for a given Partition, if required.

To change a partition policy

1. Open a Lunacm session.
2. Enter the following command to change a Partition Policy:

   partition changepolicy -policy <policy_id> -value <policy_value>

4.11 HSM Partitions

HSM Partitions are independent logical HSMs that reside within the SafeNet HSM inside, or attached to, host computer or appliance. Each Partition inside the HSM has its own data, access controls, security policies, and separate administration access independent from other HSM partitions. Depending on the product, the HSM can contain multiple HSM partitions, and each partition can be associated with one or more Clients. Each HSM Partition has a special administrative account or role, who manages it.

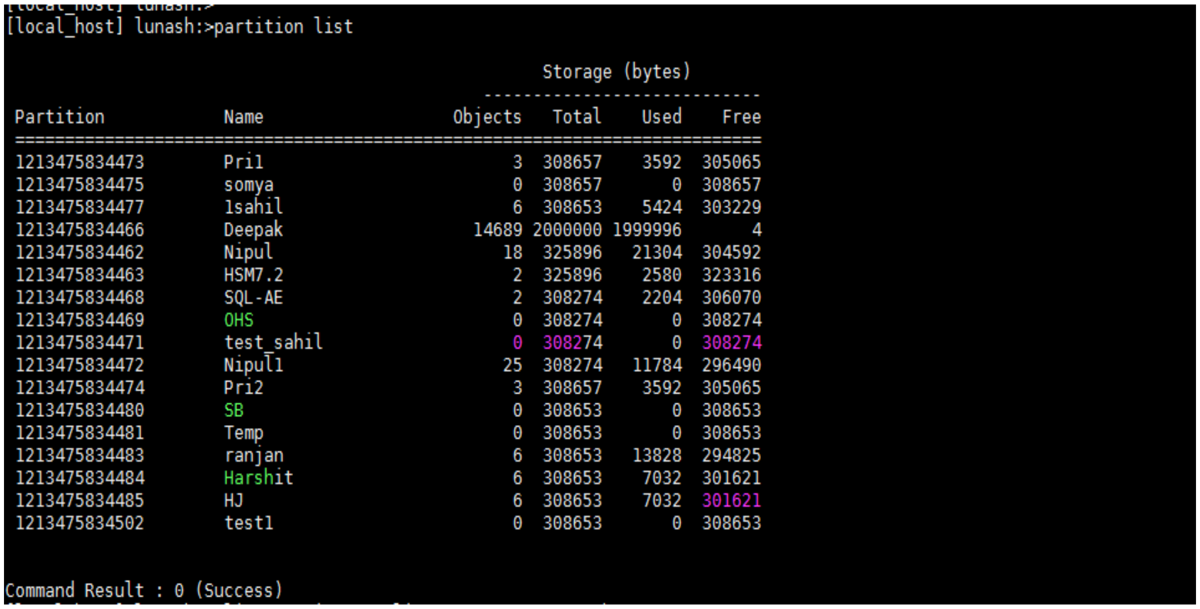
HSMs with firmware 6.22.0 or newer can have three types of partitions:

- HSM administrative partition, administered by the HSM SO
- Legacy-style application partition(s) administered at a high level by the HSM SO, but administered and operated at an operational level by the User or Crypto Officer role (with optional Crypto User).
- PPSO application partitions (requires that the PPSO capability is installed) that are created by the HSM SO, but are thereafter owned by their own local SOs, and administered and operated at an operational level by the Crypto Officer role (with optional Crypto User).

We can think of HSM Partitions as **'safe deposit boxes'** which reside within the K6/K7 Cryptographic Engine's 'vault'. The vault is itself has a high level of security for all the contents inside.Morover, each safe deposit box has its own security and access controls.The bank managers can have access to the vault but they still cannot open the individual safe deposit boxes because the safe deposit can only be opened by the owner who has the key.

A legacy application partition was/is owned by the HSM SO, who assigns a User or Crypto Officer to handle day-to-day management of partition contents, creation, use, and destruction of keys and objects, and so on. PPSO application partitions (where HSM firmware is version 6.22.0 or newer, and the PPSO capability is applied) have their own partition SO, distinct from the HSM SO. The HSM SO initializes the HSM, sets HSM-wide policies, creates an empty application partition, and hands off complete control to whoever is to become the partition SO. Thereafter, the HSM has no oversight and can do nothing with the partition except to delete it, if that is ever required. The Partition SO then initializes the partition creating a Crypto Officer

Depending upon the configuration, each SafeNet Network HSM can contain a number of HSM Partitions. Data can be stored in each partition, the no of objects that can be stored depends upon the size of each partition.. You can use the partition re-size command to modify the sizes of individual partitions until all memory on the HSM is allotted. So, you can increase the size of any partitions by shrinking others. Each partition can be assigned to a different client.Also a single HSM partition can be shared by multiple clients.



```
[local_host] lunash:>
[local_host] lunash:>partition list

                                                     Storage (bytes)
                                               ---------------------------
 Partition            Name                  Objects   Total    Used    Free
 ===========================================================================
 1213475834473        Pri1                        3  308657    3592  305065
 1213475834475        somya                       0  308657       0  308657
 1213475834477        1sahil                      6  308653    5424  303229
 1213475834466        Deepak                  14689 2000000 1999996       4
 1213475834462        Nipul                      18  325896   21304  304592
 1213475834463        HSM7.2                      2  325896    2580  323316
 1213475834468        SQL-AE                      2  308274    2204  306070
 1213475834469        OHS                         0  308274       0  308274
 1213475834471        test_sahil                  0  308274       0  308274
 1213475834472        Nipul1                     25  308274   11784  296490
 1213475834474        Pri2                        3  308657    3592  305065
 1213475834480        SB                          0  308653       0  308653
 1213475834481        Temp                        0  308653       0  308653
 1213475834483        ranjan                      6  308653   13828  294825
 1213475834484        Harshit                     6  308653    7032  301621
 1213475834485        HJ                          6  308653    7032  301621
 1213475834502        test1                       0  308653       0  308653


Command Result : 0 (Success)
```

Fig.4.3 Partition List

4.12 Network Trust Links

Network Trust Links (NTL) are secure, authenticated network connections between the SafeNet Network HSM and Clients. NTLs use two-way digital certificate authentication and TLS data encryption to protect sensitive data as it is transmitted between HSM Partitions on the SafeNet Network HSM and Clients. NTLs consist of the following parts:

- Network Trust Link Service (NTLS). The NTL server daemon runs on the SafeNet Network HSM appliance and manages the NTL connections to the appliance. NTL uses port 1792 on the SafeNet Network HSM appliance.
- Network Trust Link Agent (NTLA). The NTL agent runs on a SafeNet HSM client workstation and manages the NTL connections to the workstation. The NTL agent is included in the SafeNet HSM client software.
- The NTL itself is an encrypted and secure communications channel between the Clients' NTLA and the HSM appliance's NTLS.

Network Trust Links use digital certificates to verify the identities of connecting clients. During the initial HSM appliance configuration, the appliance administrator generated a unique certificate that identifies the HSM appliance. Similarly, each Client has to generate its own certificate which identifies it uniquely. Both the Client and the HSM appliance use these certificates to verify each other's identity before an NTL is created between them.

4.12.1 Creating A Network Trust Link

To create a Network Trust Link the Client and HSM appliance must first exchange each other certificates.So, Once their certificates have been exchanged, the Client registers the SafeNet Network HSM's certificate in a trust list, and the SafeNet Network HSM appliance, in turn, registers the Client's certificate in its list of clients.After the certificates have been exchanged and registered at each end, the NTL is setup and ready to use.

"Ready to use" means that an application at the client host (such as lunacm or your crypto-using application) can see the registered SafeNet Network HSM application partition(s) as slot(s) in the client slot list, can select such registered partitions by slot number, and can then perform cryptographic operations in those slots after providing appropriate partition authentication (Crypto Officer, Crypto User).

To create a network trust link

1. Prepare the client workstation:
    a. Install the SafeNet HSM client software.
    b. Install an SSH client to provide secure shell access to the SafeNet appliance for certificate exchange and registration. The PuTTY SSH client (putty.exe) is included in the SafeNet HSM client for Windows.
    c. Ensure that the client workstation has network access to the SafeNet Network HSM appliance. The appliance auto-negotiates network bandwidth up to Gigabit Ethernet speeds.
2. Open a SafeNet HSM client session:
    a. Open a command prompt or terminal window.
    b. Go to the SafeNet HSM client installation directory:

Windows	*C:\Program Files\SafeNet\LunaClient*
Linux/AIX	*/usr/safenet/lunaclient/bin*
Solaris/HP-UX	*/opt/safenet/lunaclient/bin*

3. Use *pscp* (Windows) or *scp* (Linux/UNIX) to import the HSM Appliance Server Certificate (*server.pem*) from the SafeNet Network HSM appliance to the SafeNet HSM client workstation. You require the SafeNet Network HSM appliance admin password to complete this step:

    Windows

    Syntax: *pscp [options] <user>@<host>:<source_filename> <target_filename>*

    Example: To copy the server certificate from host myHSM to the current (.) directory, keeping the same name:

```
pscp admin@myHSM:server.pem .
admin@myHSM's password:  *****
server.pem     | 1 kB |   1.1 kB/s | ETA: 00:00:00 | 100%
```

Linux/UNIX

Syntax:*scp [options] <user>@<host>:<source_filename> <target_filename>*

Example: To copy the server certificate from host IP 192.168.0.123 to the current (.) directory, keeping the same name:

```
scp admin@192.168.0.123:server.pem .
admin@192.168.0.123’s password:
server.pem      | 1 kB |   1.1 kB/s | ETA: 00:00:00 | 100%
```

4. Register the HSM Server Certificate with the client, using the vtl addserver command.

   Examples:

   The following command copies the server.pem file that was downloaded in the previous step, from <*luna_install_dir*> to <*luna_install_dir*>*/cert/server*, and registers the myLunaSA server certificate (<*luna_install_dir*>*/ cert/server/server.pem*), with the client:

   ```
   bash-2.05# ./vtl addServer –n myLunaSA –c server.pem
   New server myLunaSA successfully added to the server list.
   ```

   As shown, the server certificate from any SafeNet appliance arrives as the default named file server.pem. The vtl addserver command places a copy of the imported *server.pem* file in the *./cert/server folder*, (re-)naming the new file with the hostname (or IP) that you supply with –n in the command. In one example, above, the new copy would be 192.168.0.123Cert.pem. In the other example, the new cert file would be *myLunaSACert.pem*. Additionally, the command updates the *CAFile.pem* at that location, adding the new, named SafeNet Network HSM server certificate to the list of certs that the client recognizes.

   The downloaded *server.pem* file, from the earlier step, is now redundant and can be deleted, or it will be replaced the next time you download a server certificate from a SafeNet appliance.

5. Create a certificate and private key for the client, using the vtl createcert command. See VTL in the Utilities Reference Guide for full command syntax:

   ```
   vtl createcert –n <Luna_client_hostname_or_IP>
   ```

Example: The following command creates a certificate and private key for the client named myLunaClient:

```
bash-2.05# ./vtl createCert –n myClient1
Private Key created and written to:/usr/safenet/lunaclient/bin/cert/client/ `
myClientKey.pem
Certificate created and written to: /usr/safenet/lunaclient/bin/cert/client/myClient.pem
```

6. Export the client certificate to the HSM appliance, using pscp (Windows) or scp (Linux/UNIX). You require the SafeNet Network HSM appliance admin password to complete this step:

   Windows

   Syntax : *pscp [options] <source_filename> <user>@<host> :[<target_filename>]*

   Example:To copy the client certificate (*myLunaClient.pem*) to the myLunaSA appliance, keeping the same name:

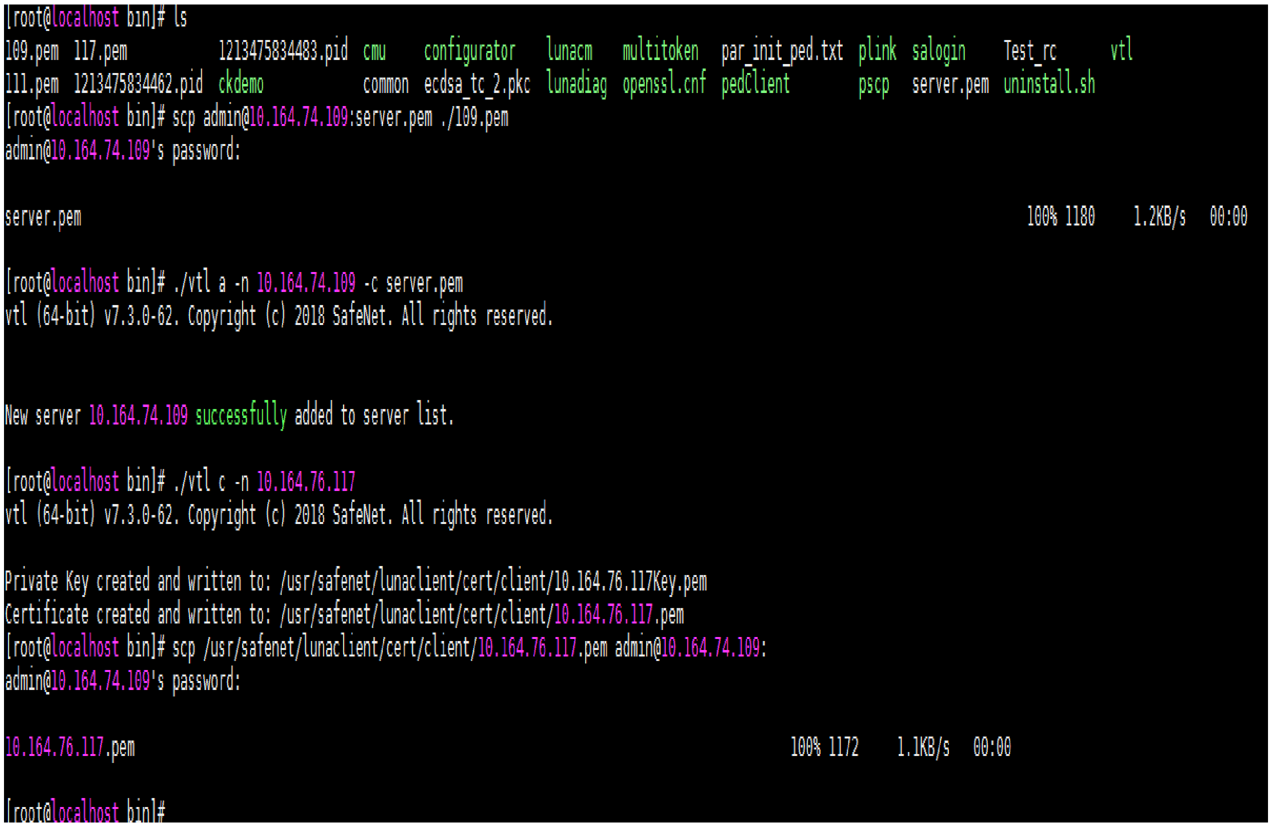
   ```
   pscp myLunaClient.pem admin@myLunaSA:
   admin@myLunaSA's password : ********
   myLunaClient.pem   | 1 kB |   1.1 kB/s | ETA : 00 :00 :00 | 100%
   ```

   Linux/UNIX

   Syntax : *scp [options] <source_filename> <user>@<host> :[<target_filename>]*

   Example: To copy the client certificate (*myLunaClient.pem*) to the SafeNet Network HSM appliance with IP 192.168.0.123, keeping the same name:

   ```
   scp myLunaClient.pem admin@192.168.0.123:
   admin@192.168.0.123's password: ********
   myLunaClient.pem   | 1 kB |   1.1 kB/s | ETA: 00:00:00 | 100%
   ```



```
[root@localhost bin]# ls
109.pem  117.pem          1213475834483.pid  cmu     configurator    lunacm    multitoken   par_init_ped.txt  plink  salogin     Test_rc       vtl
111.pem  1213475834462.pid  ckdemo           common  ecdsa_tc_2.pkc  lunadiag  openssl.cnf  pedClient         pscp   server.pem  uninstall.sh
[root@localhost bin]# scp admin@10.164.74.109:server.pem ./109.pem
admin@10.164.74.109's password:

server.pem                                                                                                100% 1180    1.2KB/s   00:00

[root@localhost bin]# ./vtl a -n 10.164.74.109 -c server.pem
vtl (64-bit) v7.3.0-62. Copyright (c) 2018 SafeNet. All rights reserved.


New server 10.164.74.109 successfully added to server list.

[root@localhost bin]# ./vtl c -n 10.164.76.117
vtl (64-bit) v7.3.0-62. Copyright (c) 2018 SafeNet. All rights reserved.

Private Key created and written to: /usr/safenet/lunaclient/cert/client/10.164.76.117Key.pem
Certificate created and written to: /usr/safenet/lunaclient/cert/client/10.164.76.117.pem
[root@localhost bin]# scp /usr/safenet/lunaclient/cert/client/10.164.76.117.pem admin@10.164.74.109:
admin@10.164.74.109's password:

10.164.76.117.pem                                                                   100% 1172    1.1KB/s   00:00

[root@localhost bin]#
```

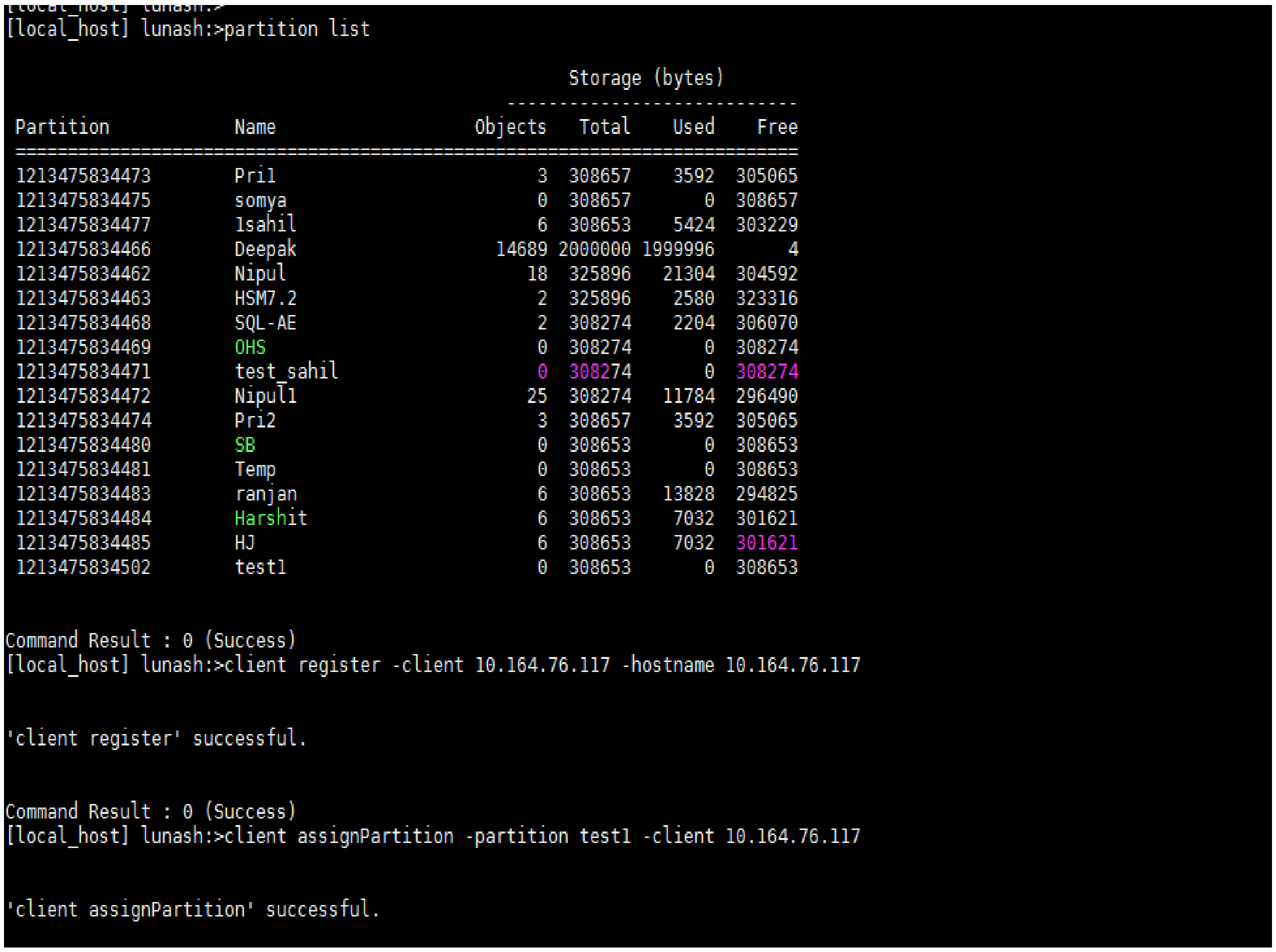
Fig. 4.4 Server Add and Client Certificate Create

7. Register the client certificate with the HSM appliance using the LunaSH client register command. You need an admin or operator-level account on the SafeNet Network HSM appliance to complete this step.
    a. Use an SSH client to connect to the SafeNet Network HSM appliance and login using an admin or operator-level account.
    b. Use the LunaSH client register command to register the client by IP address

       client register *-client <client_name> -ip <client_IP_address>*

       The *<client_name>*, above can be any string that allows you to easily identify this client.
8. Restart the Network Trust Link service. After registering a client, with a hostname certificate, or after registering a client with an IP certificate and then mapping the client hostname to its IP, stop and start the NTL service, to ensure that the new client is included.

   lunash:>service restart ntls



```
[local_host] lunash:>partition list

                                                         Storage (bytes)
                                                  ---------------------------
 Partition            Name                   Objects   Total    Used    Free
 ===========================================================================
 1213475834473        Pri1                         3  308657    3592  305065
 1213475834475        somya                        0  308657       0  308657
 1213475834477        1sahil                       6  308653    5424  303229
 1213475834466        Deepak                   14689 2000000 1999996       4
 1213475834462        Nipul                       18  325896   21304  304592
 1213475834463        HSM7.2                       2  325896    2580  323316
 1213475834468        SQL-AE                       2  308274    2204  306070
 1213475834469        OHS                          0  308274       0  308274
 1213475834471        test_sahil                   0  308274       0  308274
 1213475834472        Nipul1                      25  308274   11784  296490
 1213475834474        Pri2                         3  308657    3592  305065
 1213475834480        SB                           0  308653       0  308653
 1213475834481        Temp                         0  308653       0  308653
 1213475834483        ranjan                       6  308653   13828  294825
 1213475834484        Harshit                      6  308653    7032  301621
 1213475834485        HJ                           6  308653    7032  301621
 1213475834502        test1                        0  308653       0  308653


Command Result : 0 (Success)
[local_host] lunash:>client register -client 10.164.76.117 -hostname 10.164.76.117


'client register' successful.


Command Result : 0 (Success)
[local_host] lunash:>client assignPartition -partition test1 -client 10.164.76.117


'client assignPartition' successful.
```

Fig. 4.5 Client Registeration

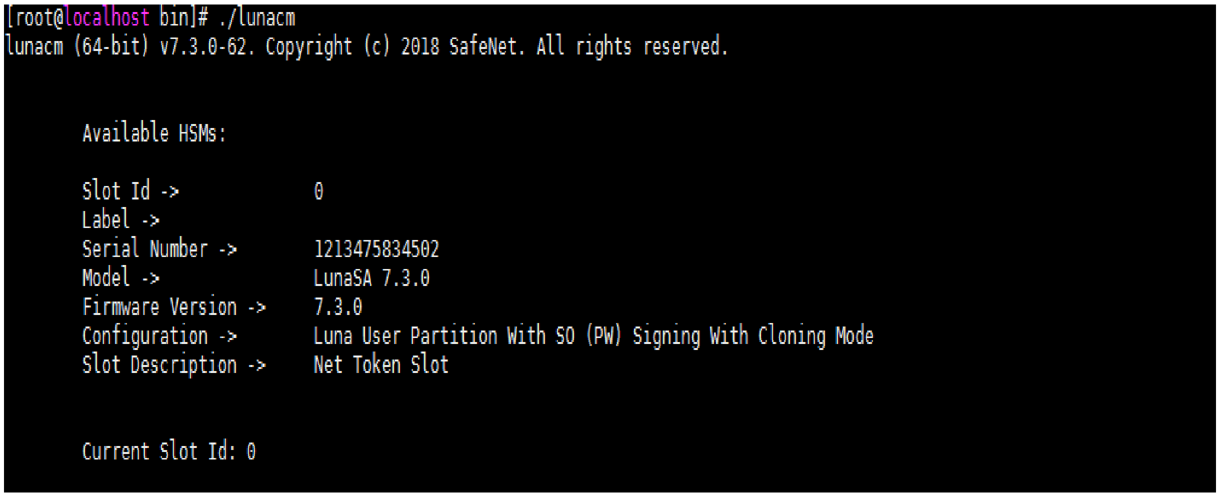
9.TCPKeepAlive is a TCP stack option, available at the LunaClient, and at the SafeNet Network HSM appliance. For SafeNet purposes, it is controlled via an entry in the *Chrystoki.conf /crystoki.ini* file on the LunaClient, and in an equivalent file on SafeNet Network HSM. For SafeNet HSM 6.1 and newer, a fresh client software installation includes an entry *"TCPKeepAlive=1"* in the *"LunaSA Client"* section of the configuration file *Chrystoki.conf* (Linux/UNIX) or *crystoki.ini* (Windows). Config files and certificates are normally preserved through an uninstall, unless you explicitly delete them.

As such, if you update (install) LunaClient software where you previously had an older LunaClient that did not have a TCPKeepAlive entry, one is added and set to "1" (enabled), by default. In the case of update, if TCPKeepAlive is already defined in the configuration file, then your existing setting (enabled or disabled) is preserved.

On the SafeNet Network HSM appliance, where you do not have direct access to the file system, the TCPKeepAlive setting is controlled by the *lunash:> ntls TCPKeepAlive* set command. The settings at the appliance and the client are independent. This allows a level of assurance, in case (for example) a firewall setting blocks in one direction.

9. Register the partition with client.

NTLS is created successfully.



```
[root@localhost bin]# ./lunacm
lunacm (64-bit) v7.3.0-62. Copyright (c) 2018 SafeNet. All rights reserved.


        Available HSMs:

        Slot Id ->              0
        Label ->
        Serial Number ->        1213475834502
        Model ->                LunaSA 7.3.0
        Firmware Version ->     7.3.0
        Configuration ->        Luna User Partition With SO (PW) Signing With Cloning Mode
        Slot Description ->     Net Token Slot


        Current Slot Id: 0
```

Fig. 4.6 Lunacm

4.13 Secure Trusted Channel Link

If a higher level of security is required for network links than is offered by NTL, such as in cloud environments, or in situations where message integrity is paramount, Secure Trusted Channel (STC) is used to provide very secure client-partition links. STC offers the following features to ensure the security and integrity of your client-partition communications:

- Privacy of all communicated data through the use of symmetric encryption, so that only the end-points can read any sensitive data.
- Integrity of the communicated data through the use of message authentication codes, so that not eavesdropper could add, delete, modify or replay any command or response.
- Bi-directional authentication of both the HSM and the end-point, so that only authorized entities can establish an STC connection, and there can be no man-in-the-middle attack.

STC and NTL can co-exist on the same SafeNet Network HSM appliance, allowing you to configure some partitions to use STC, while other partitions use NTL. The client can also support both STC and NTLS links. However, all links from a specific client to a specific SafeNet Network HSM appliance can be either NTL or STC, but not both.

Secure Trusted Channel protects HSM and client communications using endpoint and message authentication, verification, and encryption. With the help of Secure Trusted Channel, HSM and client message integrity is ensured even if these messages are sent over public or other unsecured networks. Secure Trusted Channel links can be used to confidently deploy HSM services in cloud environments or in situations where message integrity is paramount.

Secure tunneling and messaging

STC connections are established in two distinct phases:

1. Secure tunnel creation: To ensure client integrity, STC performs bi-directional HSM/client authentication, and creates unique session keys for each STC connection, as described in Secure Tunnel Creation.
2. Secure message transport: To ensure message integrity, STC uses symmetric data encryption and message integrity verification, ensuring that any attempt to alter, insert,

or drop messages is detected by both end-points, resulting in immediate termination of the connection, as described in Secure Message Transport.

All messages protected outside the HSM

When STC is fully enabled on an HSM, all sensitive communications with the HSM are protected all the way into the HSM. That is, any messages exchanged between a client application and the HSM use STC encryption, authentication, and verification from the client interface to the HSM interface, regardless of whether those links traverse a network, or are internal to an HSM appliance (LunaSH to HSM) or SafeNet HSM client workstation (SafeNet Client to HSM). In addition, all STC links that use a network connection also use the same network protection as NTLS links, that is, they are wrapped using SSL.

To use STC, you must enable the following policies:

- HSM policy 39: Allow Secure Trusted Channel. This policy enables STC on the HSM, so that you can configure the HSM such that some partitions to use STC, while other partitions use NTLS. This policy can only be set by the HSM SO.
- Partition policy 37: Force Secure Trusted Channel. This policy forces the partition to use STC, and requires that HSM policy 39 is also set. For legacy partitions, this policy can be only be set by the HSM SO. For partitions with SO, this policy can only be set by the partition SO.

The procedure for creating an STC link between a client and a partition differs depending on whether the partition is a legacy partition or a partition with SO**.**

4.13.1 Creating An STC Link To A Legacy Partition

The procedure for creating an STC link to a legacy partition consists of the following major steps:

1. Enable the STC policy on the HSM and partition.
2. Export the partition identity public key to a file on the appliance.
3. Create the client token and identity.
4. Exchange the partition and client identity public keys.
5. Register the client identity public key to the partition.

6. Register the partition identity public key with the client.
7. Enable and verify the STC link.

Step 1: Enable the STC policy on the HSM and partition

1. Launch LunaSH and log in as the HSM SO.
2. Enter the following command to ensure that policy 39: Allow Secure Trusted Channel is enabled on the HSM:
   hsm showpolicies
   If it is not enabled, enter the following command to enable the policy:
   hsm changePolicy -policy 39 -value 1
3. Enable the STC admin channel to provide STC on all links (NTLS and STC) on the portion of the link from the appliance to the HSM
4. Enter the following command to ensure that policy 37: Force Secure Trusted Channel is enabled on the partition:
   partition showpolicies -partition <partition_name>
   If it is not enabled, enter the following command to enable the policy:
   partition changepolicy -partition <partition_name> -policy 37 -value 1

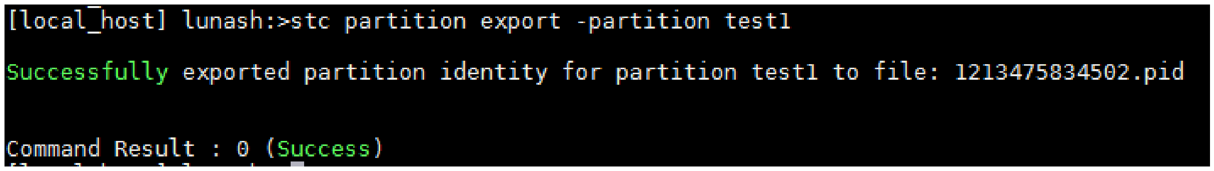
Step 2: Export the partition identity public key to a file on the appliance

This step is performed by the HSM SO. Exporting the partition identity public key creates the partition identity if it does not already exist. The public key is exported to a file named

<partition_serial_number>.pid on the appliance.

Enter the following command to export the partition's public key to a file:

stc partition export -partition <partition_name>



```
[local_host] lunash:>stc partition export -partition test1

Successfully exported partition identity for partition test1 to file: 1213475834502.pid


Command Result : 0 (Success)
```

Fig. 4.7 Partition Id export

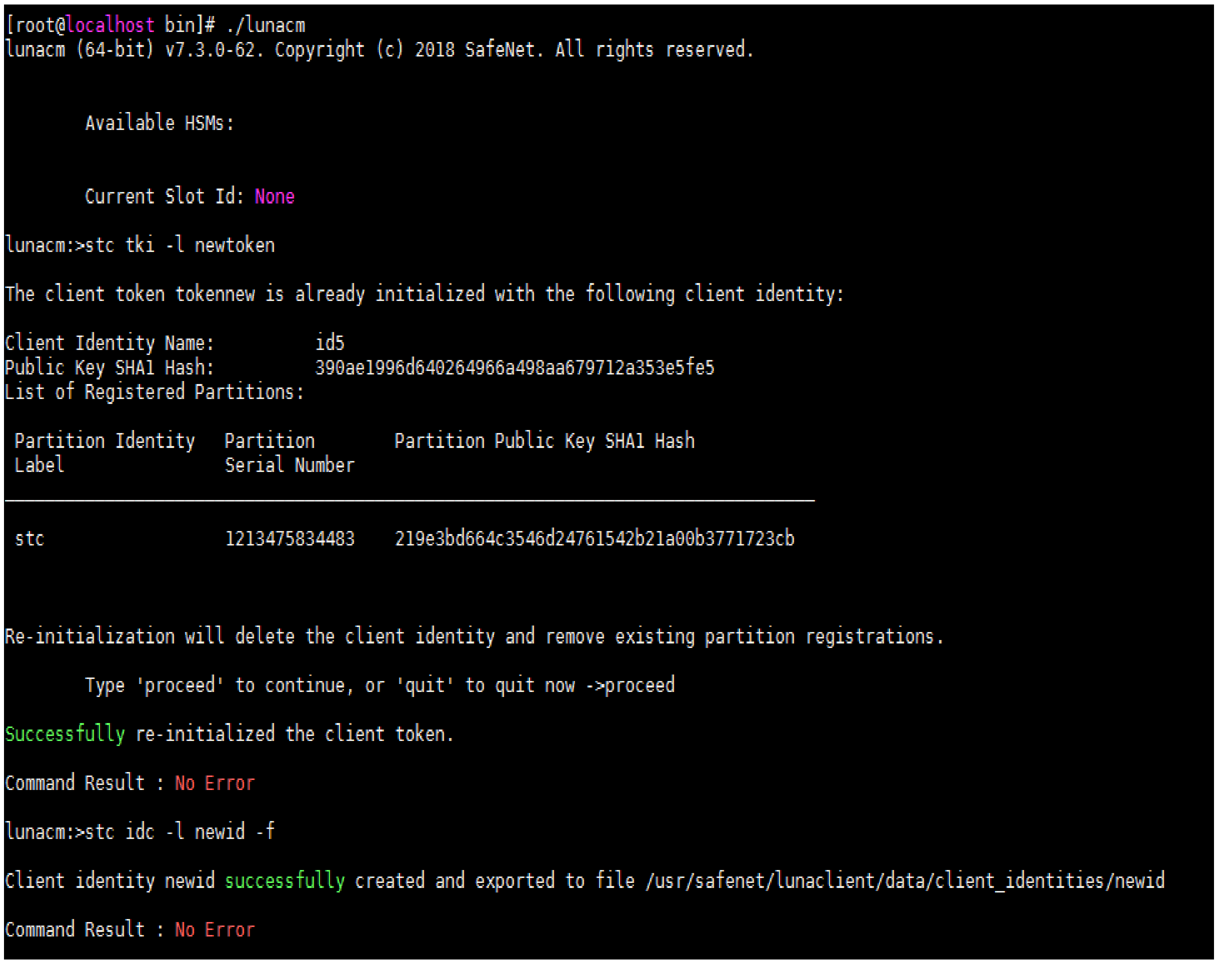
Step 3: Create the client token and identity

This step is performed by the root user on the SafeNet HSM client workstation, using Lunacm.

1. Open a SafeNet HSM client session:
    a. Open a command prompt or terminal window.
    b. Launch lunacm:

Windows	*C:\Program Files\SafeNet\LunaClient\bin\lunacm*
Linux/AIX	*/usr/safenet/lunaclient/data/bin/lunacm*
Solaris/HP-UX	*/opt/safenet/lunaclient/data/bin/lunacm*

2. Initialize the STC client software token, or insert the STC client hardware token you have prepared for this client:
   stc tokeninit -label <token_label>
3. Enter the following command to create a client identity on the token. The STC client identity public key is automatically exported to the *<luna_client_root_dir>/data/client_identities* directory:
   stc identitycreate -label <client_identity>
4. Exit LunaCM.



```
[root@localhost bin]# ./lunacm
lunacm (64-bit) v7.3.0-62. Copyright (c) 2018 SafeNet. All rights reserved.


        Available HSMs:


        Current Slot Id: None

lunacm:>stc tki -l newtoken

The client token tokennew is already initialized with the following client identity:

Client Identity Name:          id5
Public Key SHA1 Hash:          390ae1996d640264966a498aa679712a353e5fe5
List of Registered Partitions:

 Partition Identity   Partition        Partition Public Key SHA1 Hash
 Label                Serial Number

________________________________________________________________________________

 stc                  1213475834483    219e3bd664c3546d24761542b21a00b3771723cb



Re-initialization will delete the client identity and remove existing partition registrations.

        Type 'proceed' to continue, or 'quit' to quit now ->proceed

Successfully re-initialized the client token.

Command Result : No Error

lunacm:>stc idc -l newid -f

Client identity newid successfully created and exported to file /usr/safenet/lunaclient/data/client_identities/newid

Command Result : No Error
```

Fig. 4.8 Token and Id create

Step 4: Exchange the partition and client identity public keys

The STC identity public keys are exchanged as follows:

- the client identity public key is copied from the SafeNet HSM client data/client_identities directory to the SafeNet Network HSM appliance.
- the partition identity public key is copied from the appliance to the data/partition_identities directory on the SafeNet HSM client workstation.

Copying the public keys to or from the SafeNet Network HSM appliance is performed by the SafeNet Network HSM appliance administrator, using scp (UNIX/Linux) or pscp (Windows). Copying the public keys to or from the SafeNet HSM client workstation is performed by the root user on the SafeNet HSM client workstation.

1. Log in to the SafeNet HSM client workstation as the root user.
2. Go to the SafeNet HSM client data/client_identities directory:

Windows	*cd C:\Program Files\SafeNet\LunaClient\data\client_identities*
Linux/AIX	*cd /usr/safenet/lunaclient/data/client_identities*
Solaris/HP-UX	*cd /opt/safenet/lunaclient/data/client_identities*

3. Export the client identity public key to the HSM appliance, using pscp (Windows) or scp (Linux/UNIX). You require the SafeNet Network HSM appliance admin password to complete this step:

   Windows

   Syntax: *pscp [options] <source_filename> <user>@<host>:[<target_filename>]*

   Linux/UNIX

   Syntax: *scp [options] <source_filename> <user>@<host>:[<target_filename>]*
4. Go to the SafeNet HSM client data/partition_identities directory:

Windows	*cd/C:\ProgramFiles\SafeNet\LunaClient\data\ partition_identities*
Linux/AIX	*cd /usr/safenet/lunaclient/data/partition_identities*
Solaris/HP-UX	*cd /opt/safenet/lunaclient/data/partition_identities*

5. Use pscp (Windows) or scp (Linux/UNIX) to import the partition public key from the SafeNet Network HSM appliance to the data/partition_identities directory on the SafeNet HSM client workstation. You require the SafeNet Network HSM appliance admin password to complete this step:

   Windows

   Syntax: *pscp [options] <user>@<host>:<source_filename> <target_filename>*

   Linux/UNIX

   Syntax:*scp [options] <user>@<host>:<source_filename> <target_filename>*

Step 5: Register the client identity public key to the partition

Each client identity registered to a partition uses 2332 bytes of storage on the partition. Before registering a client identity to a partition, ensure that there is adequate free space.

This step is performed by the HSM SO. You can register multiple clients to a partition.

1. Launch LunaSH and log in as the HSM SO.
2. Enter the following command to register the client identity public key to the partition:
   stc client register -partition <partition_name> -label <client_label> -file <client_public_key>

Step 6: Register the partition identity public key to the client

This step is performed by the root user on the SafeNet HSM client workstation.

1. Log in to the SafeNet HSM client workstation as the root user.
2. Open a SafeNet HSM client session:
   a. Open a command prompt or terminal window.
   b. Launch lunacm:

Windows	C:\Program Files\SafeNet\LunaClient\bin\lunacm
Linux/AIX	/usr/safenet/lunaclient/data/bin/lunacm
Solaris/HP-UX	/opt/safenet/lunaclient/data/bin/lunacm

3.Enter the following command to register the partition identity public key to the client token:
stc partitionregister -file <partition_identity> [-label <partition_label>]

Step 7: Enable and verify the STC link

When you enable STC on the client, you must specify the SafeNet Network HSM appliance that hosts the partition you want to link to. This forces the client to use STC for all links to the specified SafeNet Network HSM appliance. Any existing NTLS links to the specified SafeNet Network HSM appliance will be terminated.
This step is performed by the root user on the SafeNet HSM client workstation.
1.Log in to the SafeNet HSM client workstation as the root user.

2.Open a SafeNet HSM client session:
a.Open a command prompt or terminal window.
b.Launch LunaCM:

Windows	C:\Program Files\SafeNet\LunaClient\bin\lunacm

Linux/AIX	/usr/safenet/lunaclient/data/bin/lunacm
Solaris/HP-UX	/opt/safenet/lunaclient/data/bin/lunacm

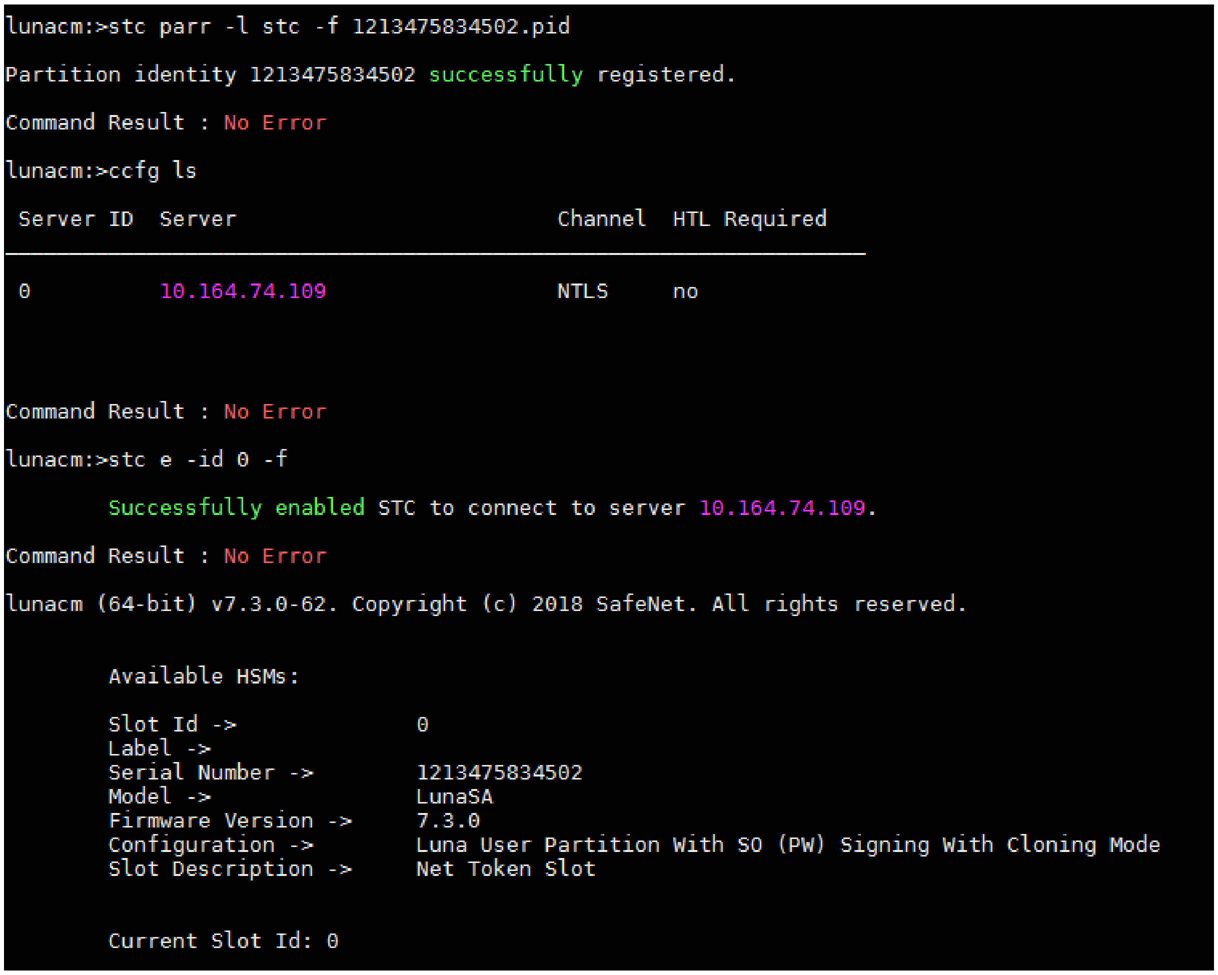
3.Enter the following command to determine the server ID of the SafeNet Network HSM appliance that hosts the partition:

lunacm:> clientconfig listservers

4.Enter the following command to enable the STC link:

stc enable -id <server_id>

At this point, LunaCM restarts. If successful, the partition is listed in the list of available HSMs. You can use the stc identityshow command to list the partitions registered to the client token.

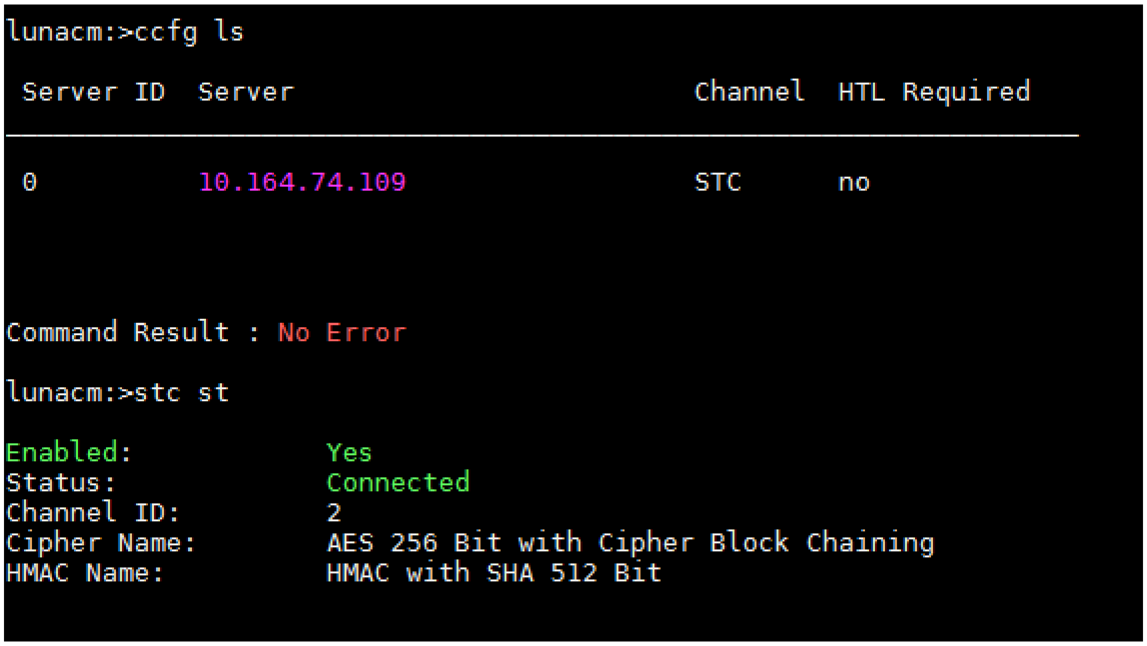


```
lunacm:>stc parr -l stc -f 1213475834502.pid

Partition identity 1213475834502 successfully registered.

Command Result : No Error

lunacm:>ccfg ls

 Server ID  Server                          Channel   HTL Required
__________________________________________________________________

 0          10.164.74.109                   NTLS      no



Command Result : No Error

lunacm:>stc e -id 0 -f

        Successfully enabled STC to connect to server 10.164.74.109.

Command Result : No Error

lunacm (64-bit) v7.3.0-62. Copyright (c) 2018 SafeNet. All rights reserved.


        Available HSMs:

        Slot Id ->              0
        Label ->
        Serial Number ->        1213475834502
        Model ->                LunaSA
        Firmware Version ->     7.3.0
        Configuration ->        Luna User Partition With SO (PW) Signing With Cloning Mode
        Slot Description ->     Net Token Slot


        Current Slot Id: 0
```

Fig. 4.9 STC Enable

5.Enter the following command to verify the link. This command displays the status of the STC link for the current slot:

lunacm:> stc status



```
lunacm:>ccfg ls

 Server ID  Server                        Channel  HTL Required
____________________________________________________________________

 0          10.164.74.109                 STC      no



Command Result : No Error

lunacm:>stc st

Enabled:            Yes
Status:             Connected
Channel ID:         2
Cipher Name:        AES 256 Bit with Cipher Block Chaining
HMAC Name:          HMAC with SHA 512 Bit
```

Fig. 4.10 STC Status

4.13.2 Creating An STC Link To A Partition With So

Creating an STC link to a partition with SO is performed entirely by the root user on the SafeNet HSM client workstation, using lunacm. The procedure consists of the following major steps:

1.Ensure that you have satisfied the prerequisite conditions.

2.Create the client token and identity.

3.Register the partition identity public key with the client.

4.Enable and verify the STC link.

5.Initialize the partition.

STC allows you to claim the partition as the holder of the partition public key, and creates a one-time temporary STC link to allow you to register the client to the partition. You must complete all of the steps in this procedure in a single lunacm session. If you do not, the partition is locked, and will not be accessible. The only workaround is for the HSM SO to delete the partition, create a new partition, and provide you with new partition public key so that you can try again.

Step 1: Ensure that you have satisfied the prerequisite conditions

Before attempting to create an STC link to a partition with SO, ensure that you have satisfied the following prerequisites:

1.You have the STC partition identity public key for the partition.

2.Confirm with the HSM SO that policy 39: Allow Secure Trusted Channel is enabled on the HSM.

Note: This procedure automatically registers the client identity to the partition. Each client identity registered to a partition uses 2332 bytes of storage on the partition. Before enabling the STC link, ensure that there is adequate free space on the partition.

Step 2: Create the client token and identity

1.Open a SafeNet HSM client session:

a. Open a command prompt or terminal window.

b. Launch lunacm:

Windows	C:\Program Files\SafeNet\LunaClient\bin\lunacm
Linux/AIX	/usr/safenet/lunaclient/data/bin/lunacm
Solaris/HP-UX	/opt/safenet/lunaclient/data/bin/lunacm

2. Initialize the STC client software token, or insert the STC client hardware token (SafeNet eToken 7300) you have prepared for this client:

If you are using an STC client software token, enter the following command to initialize the STC client token.

```
stc tokeninit -label <token_label>
```

3.Enter the following command to create a client identity on the token. The STC client identity public key is automatically exported to the <luna_client_root_dir>/data/client_identities directory:

```
stc identitycreate -label <client_identity>
```

Step 3: Register the partition identity public key to the client

1.Enter the following command to register the partition identity public key to the client token:

```
stc partitionregister -file <partition_identity> [-label <partition_label>]
```

2.If you were provided with the partition identity public key hash, enter the following command to verify that the hashes match:

```
stc identityshow
```

Step 4: Enable and verify the STC link

1.Enter the following command to determine the server ID of the SafeNet Network HSM appliance that hosts the partition:

```
lunacm:> clientconfig listservers
```

2.Enter the following command to enable the STC link:

```
stc enable -id <server_id>
```

3.Enter the following command to set the current slot to the slot containing the new partition:

```
slot set -slot <slot>
```

4.Enter the following command to verify the link:

```
lunacm:> stc status
```

Step 5: Initialize the partition

When you initialize the partition, the following actions are performed automatically:

•the client identity public key is registered to the partition.

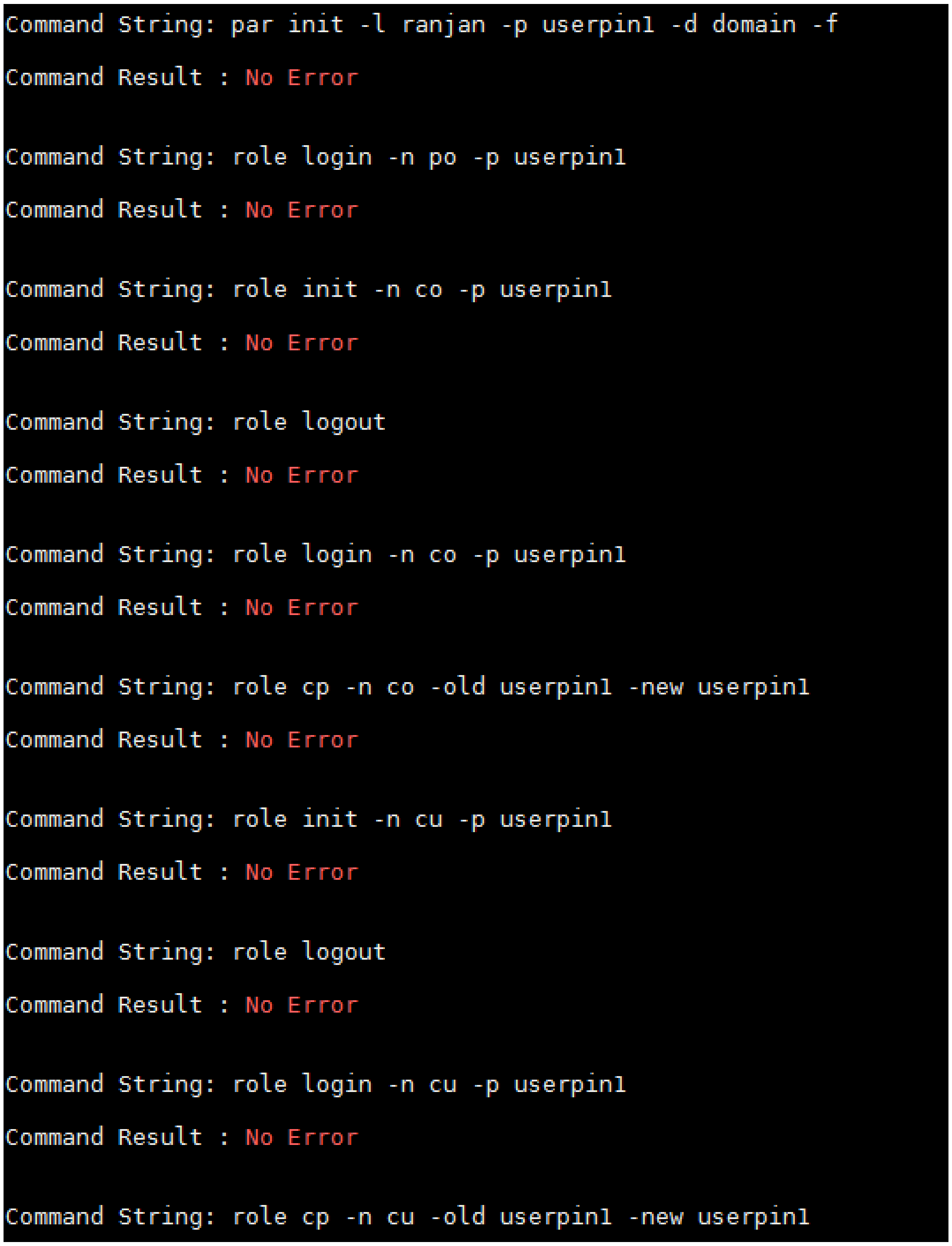
•partition policy 37: Force Secure Trusted Channel is enabled on the partition.

1.Set the current slot to the slot containing the uninitialized (unlabelled) partition.

2.Enter the following command to initialize the partition. On a password-authenticated HSM, you are prompted to specify the partition SO password and domain you want to use for the partition. On a PED-authenticated HSM, you are prompted to attend to the PED to imprint (or provide) the partition SO PED key and domain PED key:

```
partition initialize -label <partition_label>
Enter password for Partition SO: ********
Re-enter password for Partition SO: ********
Enter the domain name: *******
Re-enter the domain name: *******

lunacm:> slot list
```

You can now create the Crypto Officer and Crypto User roles on the partition.



```
Command String: par init -l ranjan -p userpin1 -d domain -f

Command Result : No Error

Command String: role login -n po -p userpin1

Command Result : No Error

Command String: role init -n co -p userpin1

Command Result : No Error

Command String: role logout

Command Result : No Error

Command String: role login -n co -p userpin1

Command Result : No Error

Command String: role cp -n co -old userpin1 -new userpin1

Command Result : No Error

Command String: role init -n cu -p userpin1

Command Result : No Error

Command String: role logout

Command Result : No Error

Command String: role login -n cu -p userpin1

Command Result : No Error

Command String: role cp -n cu -old userpin1 -new userpin1
```

Fig. 4.11 Partition and Roles Init

STC link is established.

CHAPTER – 5

MY LEARNING AND ROLE

5.1 Basics Of Cryptography

Cryptography uses mathematical algorithms and processes to convert intelligible plaintext into unintelligible cipher text, and vice versa.

Applications of cryptography include:

• Data encryption for confidentiality

• Digital signatures to provide non-repudiation (account- ability) and verify data integrity

• Certificates for authenticating people, applications and services, and for access control (authorization)

There are two main kinds of cryptography: shared secret and public key. In shared secret cryptography, sender and receiver use the same key for both encryption and decryption. Thus, many clients need to have the same key. Since encryption is presumably not available prior to key distribution, network- based key distribution is not a secure option. Other options, such as a secure courier, are expensive and slow. **Public key cryptography**, in contrast, uses pairs of keys: a public key that is widely available, and a different private key known only to the person, application or service that owns the keys. The public key can be transmitted unencrypted over insecure lines, since it is not a secret, while the private key must be kept secret. Thus, key distribution is greatly simplified using public key cryptography. The sender's private key may be used to produce a digital signature, an encrypted block of data which, when decrypted by the recipient, verifies the sender's identity (non- repudiation) as well as the integrity of the data.

5.2 PKI Components And Functions

There are three core functional components to a PKI(Public Key Infrastructure):

• The Certificate Authority (CA), an entity which issues certificates. One or more in-house servers, or a trusted third party such as VeriSign or GTE, can provide the CA function.

• The repository for keys, certificates and Certificate Revocation Lists (CRLs) is usually based on an Light- weight Directory Access Protocol (LDAP)-enabled directory service.

• A management function, typically implemented via a management console.

5.3 PKI Functions

The main functions of PKI are issuing certificates,storing them,and retrieving them.It ualso helps in revoking certificates, creating and publishing CRLs and lifecycle management of the key. Enhanced or emerging functions include time-stamping and policy-based certificate validation.

Issuing certificates

The CA signs the certificate, thereby authenticating the identity of the requestor, in the same way that a notary public vouches for the signature and identity of an individual. In addition, the CA “stamps” the certificate with an expiration date. The CA may return the certificate to the requesting system and/or post it in a repository.

Revoking certificates

A certificate may become invalid before the normal expiration of its validity period. For instance, an employee may quit or change names, or a private key may be compromised. Under such circumstances, the CA revokes the certificate by including the certificate’s serial number on the next scheduled CRL.

Storing and retrieving certificates and CRLs

Directory service is the most common way for storing and retrieving the certificates and CRLs, with access via LDAP. Other options which include X.500 compatible directories, HTTP, FTP, and e-mail providing trust. Each public key user must have at least one public key from a CA that the user trusts implicitly. Organizations can establish and maintain trust within a single security management domain through a thorough audit of the CA’s policies and procedures, repeated at regular intervals.

5.4 Manual Testing

I have done manual testing of SafeNet Luna Clients with SafeNet Luna HSM. I have performed two types of testing:

1. Sanity testing
2. Regression testing

Sanity testing is a software testing technique performed by the test team for some basic tests. This is usually done when there is some bug fixed and new build is released to check that its functionality is working as expected.Regression testing can also be done but it would take a lot of time to run full regression testingt.So,sanity testing usually saves our time and effort.

Regression testing could be a form of software package testing that verifies that software package that was antecedently developed and tested still performs properly when it had been modified or interfaced with differentsoftware package. Changes might embrace software package enhancements, patches, configuration changes, etc. throughout regression testing new software package bugs or regressions could also be uncovered. Sometimes a software package modification impact analysis is performed to see what areas can be suffering from the planned changes. These areas might embrace purposeful and non-functional areas of the system. the aim of regression testing is to confirm that changes like those mentioned on top of haven't introduced new faults. one amongst the mostreasons for regression testing is to see whether or not a modification in one a part of the software package affects different components of the software package. Common strategies of regression testing embrace rerunning antecedently completed tests and checking whether or not program behavior has modified and whether or notantecedently fastened faults have re-emerged. Regression testing may be performed to check a system expeditiouslyby consistently choosing the acceptable minimum set of tests required to adequately cowl a specific modification.

5.5 My Role

Role	**:**	**Intern**
Department	**:**	**HSM General Purpose Integration**

5.5.1 Assignments Carried Out

Automation of Network trust link on Linux Platform

Integration of SafeNet Luna HSM with Microsoft IIS Web Server

Integration of IBM HTTP SERVER with SafeNet Luna HSM

Integration of OCSP (Online Certificate Status Protocol) with SafeNet Luna HSM

Integration of SafeNet Luna HSM with JBOSS

Integration of SafeNet Luna HSM with Oracle TDE

5.5.2 Experience

It was a good opportunity to put in practice and develop organizational skills and learn concepts in digital security required in today's world. I did hands-on project with domain Cryptography operations and Integration of HSM with third party. So, overall It was a great opportunity for me for developing my inter-personal skills and making contacts which may prove of value in the future and to work with a hardworking team of HSM GP Integration .

CHAPTER – 6

CONCLUSION

The role assigned to me in the organization was carried out to my satisfaction as well as my team's expectations. I have added significant knowledge to myself after this industrial training. This has provided me an opportunity to do self-introspection of what value we have added to ourselves.

I have gathered the domain experience in the field of Digital Security, Hardware security modules, Cryptographic operations and concepts related. The training proved to be interesting with lots of things to be learned. It helped to acquire knowledge on punctuality, regularity and working environments in IT industries.

REFERENCES

1. SafeNet Network HSM documentation
2. Understanding Public key infrastructure (PKI)
3. ftp://ftp.rsa.com/pub/pdfs/understanding_pki.pdf
4. http://www.safenet-inc.com/about-safenet/
5. http://ryanstutorials.net/linuxtutorial/

APPENDIX

Assignment name : **Automation of Network trust link on Linux Platform**

Language used : **Python**

```
###############
# Name: ntls.py
# Description: Written in python using Python Modules:
#                   * OS
#                   * PEXPECT
#
#
# Imports:      ntls_details.py file
#                   * It contains all the details of Luna SA and Luna Client Machine
#
#
# This scripts automates the NTLS connection:
#
#

############
# This section imports the required libraries.

import os                          #This module provides a portable way of using operating
system dependent functionality.
import pexpect                     #This module create threads by making child
applications, controls them and respond to expected patterns in their output.
import ntls_details                #This imports the another .py file which contains all the
details of Luna SA and Luna Client Machine


#############
# FUNCTION
# Name:                fetch_details()
# Description: This function involves taking input from user about credentials of Luna SA and
validating the ip address user entered for Luna SA.


def fetch_details():
        '''This function involves fetching input from external py file about credentials of Luna
SA and validating the ip addresses of the same.
```

```
        Then this function will call client_processes function to execute commands at client
side'''

        # Declaring variables as global to be used in whole script

        global client_ip_address,luna_sa_ip,user_name,password,partition,hsm_password

        # client_ip_address is the IP address of the Luna Client machine
        # luna_sa_ip is the IP address of the Luna SA
        # user_name is the name of the account used for logging in the Luna SA
        # password is the confidential password for logging in Luna SA
        # hsm_passowrd is the confidential password of the HSM SO of Luna SA

        luna_sa_ip=ntls_details.luna_sa_ip
        user_name=ntls_details.user_name
        password=ntls_details.password
        partition=ntls_details.partition
        hsm_password=ntls_details.password
        client_ip_address=ntls_details.client_ip_address

        #Checking IP address validity for the ip address user entered for Luna SA
        ip_check(luna_sa_ip)
        ip_check(client_ip_address)


#########
# FUNCTION
# Name :        ip_check(ip_address)
# Description : This function splits the ip address to check its length and validate it.
# Input:        Takes ip adresss as parameter
# Output:       If the ip address is invalid,it terminates the execution of the script

def ip_check(ip_address):
        '''An Internet Protocol address (IP address) is a numerical label assigned to each device
for luna_sa_ip or network interface identification and location addressing.

        IPv4 addresses are 32 bit number represented in dot-decimal notation, which consists
of four decimal numbers, each ranging from 0 to 255, separated by dots'''

        parts_of_ip=ip_address.split(".")
#Splitting the ip address by decimal dots

        if len(parts_of_ip)<4 or len(parts_of_ip) > 4:
#Length of parts of ip splitted must be equal to 4 always
                print '\nInvalid IP address that you are trying to connect. Try again with correct
input. Error in:',ip_address
```

```
                os._exit(0)
        else:
                for i in range(0,3):
#To fetch the parts of ip from index 0 to 3
                        ip_part=int(parts_of_ip[i])
                        if ip_part < 0 or ip_part >= 255:
                                print '\nInvalid IP address that you are trying to connect. Try
again with correct input',ip_address
                                os._exit(0)
#Terminates the execution of script


#########
# FUNCTION
# Name :        client_processes()
# Description : This function executes the commands on client machine.

def client_processes():
        '''This functions executes all the commands on client machine.

        It also executes command of vtl utility such as registering the server,creating vertificate
at client.
        It also uses scp command to transfer certificates between Luna SA and client machine.'''

        try:
                os.chdir("/usr/safenet/lunaclient/bin")
#Changing the working directory
        except OSError:
                print 'Check that Luna Client is installed on the system.' #If path not
found,OSError exception is thrown.
                os._exit(0)
#terminates the execution

        #Checking if server.pem already exists in the directory.
        # If it does, it will delete the older version and replace it with new one.
        status=os.popen("ls").read()
        if 'server.pem' in status:
                print "\nPrevious version of server.pem found.\nDeleting..."
                os.system("rm server.pem -f")
 # remove any older version of server.pem

        #Copying server.pem to the Luna SA machine

        #var_command is variable used to store the command to be next executed on server
        var_command ="scp {0}@{1}:server.pem .".format(user_name,luna_sa_ip)
        child=pexpect.spawn(var_command)
```

```
          # flag is the variable used to check the output status of the command
          # if output of pexpect.spawn is ".*password.*" then flag will get 0 as output
          # if EOF is encountered instead of "password" then the value of flag will be 1
          flag=child.expect(['.*password.*',pexpect.EOF])
          if flag==0:
                    child.sendline(password)
                    child.expect(pexpect.EOF)
                    print "Copying server.pem successful."
          elif flag==1:
                    print 'Error occured while transfering server certificate to luna_sa_ip machine.'
                    os._exit(0) #terminates the execution


          #Check if server already registered.
          status=os.popen("./vtl listservers").read()
          if luna_sa_ip in status:
                    print "Server already registered.Deleting the old entry..."
                    var_command="./vtl deleteserver -n {0} -f".format(luna_sa_ip)
#VTL utility command
                    os.system(var_command)


          #Adding server to client
          var_command="./vtl addserver -n {0} -c server.pem".format(luna_sa_ip)
#VTL utility command
          status=os.system(var_command)
          if status!=0:
                    print "Error occured while registering server with the client."
                    os._exit(0)
#terminates the execution

          var_command="./vtl createcert -n {0}".format(client_ip_address)
#VTL utility command
          status=os.system(var_command)
          if status!=0:
                    print "Error occured while creating client certificate."
                    os._exit(0)
#terminates the execution


          #Transfering certficates to server
          extension='.pem'
          file_name=client_ip_address+extension                                    #Certificate
generated in form of .pem file
```

```
	var_command	="scp	/usr/safenet/lunaclient/cert/client/%s	%s@%s:"
%(file_name,user_name,luna_sa_ip)
#SCP copies the file between the two machines
	child=pexpect.spawn(var_command)
	flag=child.expect(['.*password.*',pexpect.EOF])
	if flag==0:
		child.sendline(password)
		child.expect(pexpect.EOF)
		print "Client certificate transfered to server."
	elif flag==1:
		print "EOF exception caught. Try again."
		os._exit(0)
#terminates the execution
	else:
		print "Timeout occured.Try again."
		os._exit(0)
#terminates the execution
	child.close()



####################
# FUNCTION
# Name :	luna_ssh_processes()
# Description : This function makes SSH with Luna SA and executes commands there.
# Process:	SSH (Secure SHell) is a network protocol which provides a replacement for
insecure remote login and command execution facilities

def luna_ssh_processes():
	global child
	#var_command is variable used to store the command to be next executed on server
	# SSH connection is created to the remote Luna SA machine to execute commands
	var_command="ssh {0}@{1}".format(user_name,luna_sa_ip)
	#SSH command : ssh user_name@ip_address
	child=pexpect.spawn(var_command)
	# flag is the variable used to check the output status of the command
	flag=child.expect(['.*password.*',pexpect.TIMEOUT])
	if flag==0:
		child.sendline(password)
		flag=child.expect('.*lunash.*')
	if flag==0:
		print "SSH session created."
		child.sendline('hsm login')
		flag=child.expect('.*password.*')
	if flag==0:
```

```
                child.sendline(hsm_password)
                flag=child.expect(['.*successful.*',pexpect.TIMEOUT])
        if flag==0:
                print "Login successful as HSM Administrator."



#########
# FUNCTION
# Name :        client_register_precheck()
# Description : This function checks if client is already registered with Luna SA.

def client_register_precheck():
        '''It checks if client is already registered with Luna SA.

        If the entry exists, it deletes the old entry.'''

        #Checking if client is already registered with Luna SA
        var_command='client list\n'

        #Creating a file to record the output of the shell in file
        child.logfile=open("/home/admin/Logs/client.txt","w")
        child.sendline(var_command)
        flag=child.expect(".*lunash.*")
        if flag==0:
                if client_ip_address in open('/home/admin/Logs/client.txt').read():
                        print "Client is already registered.Deleting the old entry..."
                        var_command="client delete  -c %s -f" %(client_ip_address)
                        child.sendline(var_command)
                        flag=child.expect(".*success.*")
                        if flag==0:
                                client_register()
                        else:
                                print "Error occured while deleting the old entry. Try again."
                                os._exit(0)
                else:
                        client_register()

#########
# FUNCTION
# Name :        client_register()
# Description : This function registers the client on Luna SA.

def client_register():

        #Registering client on server
```

```
		if child.expect(".*lunash.*")==0:
			var_command="client register -c %s -ip %s -f"
%(client_ip_address,client_ip_address)
#Client register on lunash
			child.sendline(var_command)
			flag=child.expect(['.*lunash.*',pexpect.EOF,pexpect.TIMEOUT])
			if flag==0:
				var_command="client assignpartition -c {0} -p
{1}".format(client_ip_address,partition)  #Assigning partition to Client registered
				child.sendline(var_command)
				flag=child.expect(['.*success.*',pexpect.TIMEOUT])
				if flag==0:
					print 'Partition assigned to client\n\n'
					print '-------NTLS Created.---------'
				else:
					print 'Partition not assigned.Error occured.'
					os._exit(0)
			#terminates the execution
			else:
				print "client registration failed"
				os._exit(0)
			#terminates the execution


##############
# FUNCTION
# MAIN()

def main():
	'''This script automates the NTLS coonection between Client and Luna SA.'''

	#FUNCTION CALLS

	#Calling fetch_details function to fetch the details of the NTLS connection to be
established.
	fetch_details()

	#Calling client_processes to execute commands on Luna Client machine
	client_processes()

	#Calling luna_ssh_processes function to execute commands at Luna SA
	luna_ssh_processes()

	#Checking if the client is already registered on Luna SA or not
	client_register_precheck()
```

```
if __name__=="__main__":
        main()
```