GnuPG

INTEGRATION GUIDE

SAFENET LUNA HSM

SAFENET DATA PROTECTION ON DEMAND

Document Information

Document Part Number	007-013996-001
Release Date	13 March 2020

Revision History

Revision	Date	Reason
B	13 March 2020	Update

Trademarks, Copyrights, and Third-Party Software

Disclaimer

All information herein is either public information or is the property of and owned solely by Gemalto NV. and/or its subsidiaries who shall have and keep the sole right to file patent applications or any other kind of intellectual property protection in connection with such information.

Nothing herein shall be construed as implying or granting to you any rights, by license, grant or otherwise, under any intellectual and/or industrial property rights of or concerning any of Gemalto's information.

This document can be used for informational, non-commercial, internal and personal use only provided that:

- The copyright notice below, the confidentiality and proprietary legend and this full warning notice appear in all copies.
- This document shall not be posted on any network computer or broadcast in any media and no modification of any part of this document shall be made.

Use for any other purpose is expressly prohibited and may result in severe civil and criminal liabilities.

The information contained in this document is provided "AS IS" without any warranty of any kind. Unless otherwise expressly agreed in writing, Gemalto makes no warranty as to the value or accuracy of information contained herein.

The document could include technical inaccuracies or typographical errors. Changes are periodically added to the information herein. Furthermore, Gemalto reserves the right to make any change or improvement in the specifications data, information, and the like described herein, at any time.

Gemalto hereby disclaims all warranties and conditions with regard to the information contained herein, including all implied warranties of merchantability, fitness for a particular purpose, title and non-infringement. In no event shall Gemalto be liable, whether in contract, tort or otherwise, for any indirect, special or consequential damages or any damages whatsoever including but not limited to damages resulting from loss of use, data, profits, revenues, or customers, arising out of or in connection with the use or performance of information contained in this document.

Gemalto does not and shall not warrant that this product will be resistant to all possible attacks and shall not incur, and disclaims, any liability in this respect. Even if each product is compliant with current security standards in force on the date of their design, security mechanisms' resistance necessarily evolves according to the state of the art in security and notably under the emergence of new attacks. Under no circumstances, shall Gemalto be held liable for any third party actions and in particular in case of any successful attack against systems or equipment incorporating Gemalto products. Gemalto disclaims any liability with respect to security for direct, indirect, incidental or consequential damages that result from any use of its products. It is further stressed that independent testing and verification by the person using the product is particularly encouraged, especially in any application in which defective, incorrect or insecure functioning could result in damage to persons or property, denial of service or loss of privacy.

CONTENTS

PREFACE

This guide describes the steps involved in integrating SafeNet Luna HSM and SafeNet Data Protection on Demand service (DPoD) with GnuPG, also known as GPG or Gnu Privacy Guard. It contains the following chapters:

> Getting Started describes the third party applications, supported platforms, prerequisites, and the setup required for GnuPG.

> Integrating GnuPG with SafeNet HSMs explains the steps involved in integrating SafeNet Luna HSM with GnuPG.

Audience

This document is intended to guide security administrators through the steps for integrating GnuPG with SafeNet Luna HSMs and HSM on Demand service.

All products manufactured and distributed by Gemalto, Inc. are designed to be installed, operated, and maintained by personnel who have the knowledge, training, and qualifications required to safely perform the tasks assigned to them. The information, processes, and procedures contained in this document are intended for use by trained and qualified personnel only.

Document Conventions

This section provides information on the conventions used in this document.

Notes

Notes are used to alert you to important or helpful information.

NOTE: Take note. Notes contain important or helpful information.

Cautions

Cautions are used to alert you to important information that may help prevent unexpected results or data loss.

CAUTION! Exercise caution. Caution alerts contain important information that may help prevent unexpected results or data loss.

Warnings

Warnings are used to alert you to the potential for catastrophic data loss or personal injury.

****WARNING** Be extremely careful and obey all safety and security measures. In this situation you might do something that could result in catastrophic data loss or personal injury**

Command Syntax and Typeface Conventions

Convention	Description
Bold	The bold attribute is used to indicate the following: > Command-line commands and options (Type dir /p.) > Button names (Click Save As.) > Check box and radio button names (Select the Print Duplex check box.) > Window titles (On the Protect Document window, click Yes.) > Field names (User Name: Enter the name of the user.) > Menu names (On the File menu, click Save.) (Click Menu > Go To > Folders.) > User input (In the Date box, type April 1.)
Italic	The italic attribute is used for emphasis or to indicate a related document. (See the *Installation Guide* for more information.)
Double quote marks	Double quote marks enclose references to other sections within the document.
<variable>	In command descriptions, angle brackets represent variables. You must substitute a value for command line arguments that are enclosed in angle brackets.
[optional] [<optional>]	Square brackets enclose optional keywords or <variables> in a command line description. Optionally enter the keyword or <variable> that is enclosed in square brackets, if it is necessary or desirable to complete the task.
[a | b | c] [<a> | <b> | <c>]	Square brackets enclose optional alternate keywords or variables in a command line description. Choose one command line argument enclosed within the braces, if desired. Choices are separated by vertical (OR) bars.

{ a | b | c }
{ <a> | <b> | <c> }
	Braces enclose required alternate keywords or <variables> in a command line description. You must choose one command line argument enclosed within the braces. Choices are separated by vertical (OR) bars.

Support Contacts

If you encounter a problem while installing, registering, or operating this product, refer to the documentation. If you cannot resolve the issue, contact your supplier or Gemalto Customer Support.

Gemalto Customer Support operates 24 hours a day, 7 days a week. Your level of access to this service is governed by the support plan arrangements made between Gemalto and your organization. Please consult this support plan for further information about your entitlements, including the hours when telephone support is available to you.

Customer Support Portal

The Customer Support Portal, at https://supportportal.thalesgroup.com is a where you can find solutions for most common problems. The Customer Support Portal is a comprehensive, fully searchable database of support resources, including software and firmware downloads, release notes listing known problems and workarounds, a knowledge base, FAQs, product documentation, technical notes, and more. You can also use the portal to create and manage support cases.

> **NOTE:** You require an account to access the Customer Support Portal. To create a new account, go to the portal and click on the **REGISTER** link.

Telephone Support

If you have an urgent problem, or cannot access the Customer Support Portal, you can contact Gemalto Customer Support by telephone at +1 410-931-7520. Additional local telephone support numbers are listed on the support portal.

Email Support

You can also contact technical support by email at technical.support@gemalto.com.

CHAPTER 1: Getting Started

This chapter covers the following topics:

About SafeNet Luna HSMs and SafeNet Data Protection on Demand

You can use SafeNet Luna HSMs and SafeNet Data Protection on Demand service (DPoD) to store GPG signing, encryption, and authentication keys. GnuPG is a command line tool that allows you to encrypt and sign your data and communications. The benefits of securing GPG keys with SafeNet HSM include:

- Secure generation, storage, and protection of keys on FIPS 140-2 level 3 validated hardware.
- Full life cycle management of keys.
- Access to the HSM audit trail*.
- Adoption of cloud services with confidence.

*HSMoD services do not have access to the secure audit trail

SafeNet Luna HSM: SafeNet Luna HSM appliances are purposefully designed to provide a balance of security, high performance, and usability that makes them an ideal choice for enterprise, financial, and government organizations. SafeNet Luna HSMs physically and logically secure cryptographic keys and accelerate cryptographic processing. The SafeNet Luna HSM on premise offerings include the SafeNet Luna Network HSM, SafeNet PCIe HSM, and SafeNet Luna USB HSMs.

SafeNet Data Protection on Demand (DPoD): SafeNet DPoD is a cloud-based platform that provides on-demand HSM and Key Management services through a simple graphical user interface. With DPoD, security is simple, cost effective, and easy to manage because there is no hardware to buy, deploy, and maintain. As an Application Owner, you click and deploy services, generate usage reports, and maintain only the services that you need.

Third Party Applications

This integration uses the following third party application:

- GnuPG (GPG)

Supported Platforms

This integration is supported/verified with SafeNet Luna HSMs and SafeNet Data Protection on Demand on the following operating systems:

Operating Systems	GnuPG	SafeNet HSM
RedHat Enterprise Linux 7.7 (64 bit)	GPG v2.0.22	Appliance Software v7.3.0 Firmware v7.3.0 SafeNet Luna HSM Client 7.3.0
RedHat Enterprise Linux 7.0 (64 bit)	GPG v2.0.22	Appliance Software v6.2.2 Firmware 6.24.6 SafeNet Luna HSM Client 6.2.2
RedHat Enterprise Linux 6.5 (64 bit)	GPG v2.0.14	Appliance Software v6.3.0 Firmware 6.10.9 SafeNet Luna HSM Client 6.3.0

Operating Systems	GnuPG	SafeNet DPoD
RedHat Enterprise Linux 7.7 (64 bit)	GPG v2.0.22	SafeNet Data Protection on Demand HSMoD Service Client v10.1

Prerequisites

Before beginning the integration, ensure you complete the following processes:

> Configuring the SafeNet Luna HSM

> Provisioning the HSM on Demand Service

> Installing GPG-Dependent Packages

> Installing Pinentry Package

> Installing GPG Packages

> Installing gnupg-pkcs11-scd smart-card daemon and pkcs11-helper

Configuring the SafeNet Luna HSM

If you are using a SafeNet Luna HSM:

1. Verify the HSM is set up, initialized, provisioned, and ready for deployment. Refer to the *SafeNet Luna HSM Product Documentation* for more information.
2. Create a partition on the HSM that will be used by GnuPG (GPG) later on.

3. If you are using a SafeNet Luna Network HSM, register a client for the system and assign the client to the partition to create an NTLS connection. Initialize the Crypto Officer and Crypto User roles for the registered partition.

4. Ensure that each partition is successfully registered and configured. The command to see the registered partitions is:

```
# /usr/safenet/lunaclient/bin/lunacm

lunacm (64-bit) v7.3.0-165. Copyright (c) 2018 SafeNet. All rights reserved.

Available HSMs:

Slot Id ->                0

Label ->                  gpgpartition

Serial Number ->          1280780175949

Model ->                  LunaSA 7.3.0

Firmware Version ->       7.3.0

Configuration ->          Luna User Partition With SO (PW) Key Export With
Cloning Mode

Slot Description ->       Net Token Slot

Current Slot Id: 0
```

> **NOTE:** Follow the *SafeNet Luna Network HSM Product Documentation* for steps to create the NTLS connection, initialize the partitions, and initialize the Security Officer, Crypto Officer, and Crypto User roles.

Provisioning the HSM on Demand service

This service enables your client machine to access an HSM application partition for storing cryptographic objects used by your applications. Application partitions can be assigned to a single client, or multiple clients can be assigned to and share a single application partition.

You need to provision your application partition by initializing the following roles:

- **Security Officer (SO)** - Responsible for setting the partition policies and for creating the Crypto Officer.
- **Crypto Officer (CO)** - Responsible for creating, modifying, and deleting crypto objects within the partition. The CO can use the crypto objects and create an optional, limited-capability role called Crypto User that can use the crypto objects but cannot modify them.
- **Crypto User (CU)** – An optional role that can use crypto objects while performing cryptographic operations.

> **NOTE:** The HSMoD service client package is a zip file that contains system information required to connect your client machine to an existing HSM on Demand service.Refer to the *SafeNet Data Protection on Demand Application Owner Quick Start Guide* for more information about provisioning the HSM on Demand service and create a service client.

Constraints on HSMoD Services

Take the following limitations into consideration when integrating your application with an HSMoD Service:

- **HSM on Demand Service in FIPS mode:** HSMoD services operate in a FIPS and non-FIPS mode. If your organization requires non-FIPS algorithms for your operations, ensure you enable the **Allow non-FIPS approved algorithms** check box when configuring your HSM on Demand service. The FIPS mode is enabled by default. Refer to the “Mechanism List” in the *SDK Reference Guide* for more information about the available FIPS and non-FIPS algorithms.
- **Verifying HSM on Demand <slot> value:** LunaCM commands work on the current slot. If there is only one slot, then it is always the current slot. If you are completing an integration using HSMoD services, you need to verify the slot where you have sent the commands. If there is more than one slot, then use the **slot set** command to direct a command to a specified slot. You can use the slot list to map the slot numbers with HSMoD services.

Installing GPG-Dependent Packages

Before you begin the integration process, you need to install the following GPG-dependent packages from https://www.gnupg.org/download/index.html:

- npth
- libgpg-error
- libgcrypt
- libksba
- libassuan

Installing Pinentry Package

To authenticate partition access for GPG, you can use the **salogin** utility, which gets automatically installed along with the SafeNet Luna Client software. However, if you do not want to use the **salogin** utility, you can install the **Pinentry** package available at https://www.gnupg.org/download/index.html.

Installing GPG Package

After building and installing the above packages, you need to install the **GPG** package available at https://www.gnupg.org/download/index.html

Installing PKCS11-Helper and GnuPG-PKCS11 SCD

After installing GPG, you need to install **pkcs11-helper** and **gnupg-pkcs11-scd** and libraries.

> **NOTE:** While building the gnupg-pkcs11-scd daemon, the development packages associated with these libraries are subsequently used at runtime. To keep the new library versions separate from the versions that are already installed, run the **export LD_LIBRARY_PATH=/usr/local/lib** command**.**

- pkcs11-helper (https://github.com/OpenSC/pkcs11-helper/releases)
- gnupg-pkcs11-scd (https://github.com/alonbl/gnupg-pkcs11-scd/releases/)

CHAPTER 2: Integrating GnuPG with SafeNet HSMs

This chapter covers the following topics:

> Accessing SafeNet HSM

> Configuring the gnupg-pcs11-scd.conf file

> Generating Keys and Certificates

> Configuring GPG to use the PKCS#11 Smart Card Daemon

> Testing GPG

Accessing SafeNet HSM

You can use either of the following methods to access SafeNet HSM on GPG:

> Using salogin utility

> Using Pinentry

Using salogin utility

The persistent session allows the GPG to access the HSM object without prompting the password every time.

> **NOTE:** Persistent Session is not supported in DPoD so salogin will not work. For DPoD go to Using Pinentry section.

To open the persistent session using **salogin** utility, perform the following steps:

1. Add the following text in the **/etc/Chrystoki.conf** file:

```
Misc = {
   AppIdMajor=1;
   AppIdMinor=1;
}
```

2. Run the following command to open the authenticated persistent session to access the HSM object:

```
# ./salogin -o -s 0 -i 1:1 -p <partition_password>
```

Where `-s` represent the slot_id and `-i` represent the AppId set in the **Chrystoki.conf** file.

Using Pinentry

To use Pinentry, you need to add the following text in the **/root/.gnupg/gpg-agent.conf** file.

```
pinentry-program /usr/local/bin/pinentry
```

> **NOTE:** If **gpg-agent.conf file** doesn’t exist, you need to create it at **/root/.gnupg/** directory.

Configuring the gnupg-pcs11-scd.conf file

> **NOTE:** Skip this step if you are using the **salogin** utility.

To configure the gnupg-pkcs11-scd.conf file:

1. Add/modify the following lines to **/root/.gnupg/gnupg-pkcs11-scd.conf** file

   ```
   provider-p1-allow-protected-auth

   provider-p1-cert-private

   provider-p1-private-mask 0
   ```

> **NOTE:** If **gnupg-pkcs11-scd.conf** file doesn’t exist, you need create this file and copy all the contents from **/usr/local/etc/gnupg-pkcs11-scd.conf.example** or **/usr/local/share/doc/gnupg-pkcs11-scd/gnupg-pkcs11-scd.conf**.

Generating Keys and Certificates

After creating the NTLS connection with HSM, follow the below steps to generate the RSA key pair on HSM. GPG uses several asymmetric key pairs as part of the keychain configuration. These are signing, encryption, and authentication key pairs. It is possible to use the same key pair for all three functions. To generate keys and certificates:

1. Generate the RSA key pair on SafeNet Luna HSM using the **CMU** utility provided with Luna Client in **/usr/safenet/lunaclient/bin** directory. Provide the partition password when prompted.

   ```
   # ./cmu generatekeypair -modulusBits=2048 -publicExponent=65537 -
   labelPublic=GPG-Sign-Pub –labelPrivate=GPG-Sign-Priv -id=11111101 -sign=T -
   verify=T -encrypt=T -decrypt=T

   Please enter password for token in slot 0 : ********

   Select RSA Mechanism Type -

   [1] PKCS [2] FIPS 186-3 Only Primes [3] FIPS 186-3 Auxiliary Primes : 1
   ```

   Select RSA Mechanism Type as [1] PKCS

> **NOTE:** CMU command option might be slightly differ in other versions of Luna Client, kindly refer to the Luna SA documentation for exact options.

2. List the contents generated on HSM partition and note down the handle of public/private key. Provide the partition password when prompted.

   ```
   # ./cmu list

   Please enter password for token in slot 0 : ********

   handle=34       label=GPG-Sign-Pub

   handle=35       label=GPG-Sign-Priv
   ```

3. Generate the self-signed certificate from the generated public/private key. Provide the partition password and certificate attributes when prompted.

```
# ./cmu selfsigncertificate -publichandle=34 -privatehandle=35 -
startDate=20200225 -endDate=20251025 -serialNumber=0133337A -
keyusage=digitalsignature,keyencipherment -label=GPG-Sign

Please enter password for token in slot 0 : ********

Enter Subject 2-letter Country Code (C) : IN

Enter Subject State or Province Name (S) : UPST

Enter Subject Locality Name (L) : NOIDA

Enter Subject Organization Name (O) : GEMALTO

Enter Subject Organization Unit Name (OU) : IDPS

Enter Subject Common Name (CN) : GPG-Signing

Enter EMAIL Address (E) :
```

> **NOTE:** Self-signed certificate is used for test purpose, in production environment facing internet, create the certificate request and signed it by the Trusted Certificate Authority.

4. If you want to use different Encryption and Authentication Keys/Certificates, then repeat the above steps

> **NOTE:** Ensure that the **id** and **label** for every key/certificate is different.

Configuring GPG to use the PKCS#11 Smart Card Daemon

Perform the following steps to configure the gpg-agent that uses the smart card daemon to access the keys on HSM:

1. Add the following line to**/root/.gnupg/gpg-agent.conf** file.,

```
scdaemon-program /usr/local/bin/gnupg-pkcs11-scd
```

> **NOTE:** if **/root/.gnupg/gpg-agent.conf** file is not present, then create the file and add the above lines.

2. Add/modify the following lines to **/root/.gnupg/gnupg-pkcs11-scd.conf** file available at the :

```
providers p1

provider-p1-library /usr/safenet/lunaclient/lib/libCryptoki2_64.so
```

> **NOTE:** If **gnupg-pkcs11-scd.conf** file doesn't exist, you need create this file and copy all the contents from **/usr/local/etc/gnupg-pkcs11-scd.conf.example** or **/usr/local/share/doc/gnupg-pkcs11-scd/gnupg-pkcs11-scd.conf**.

3. Set the following environment variables to use the installed GPG. .

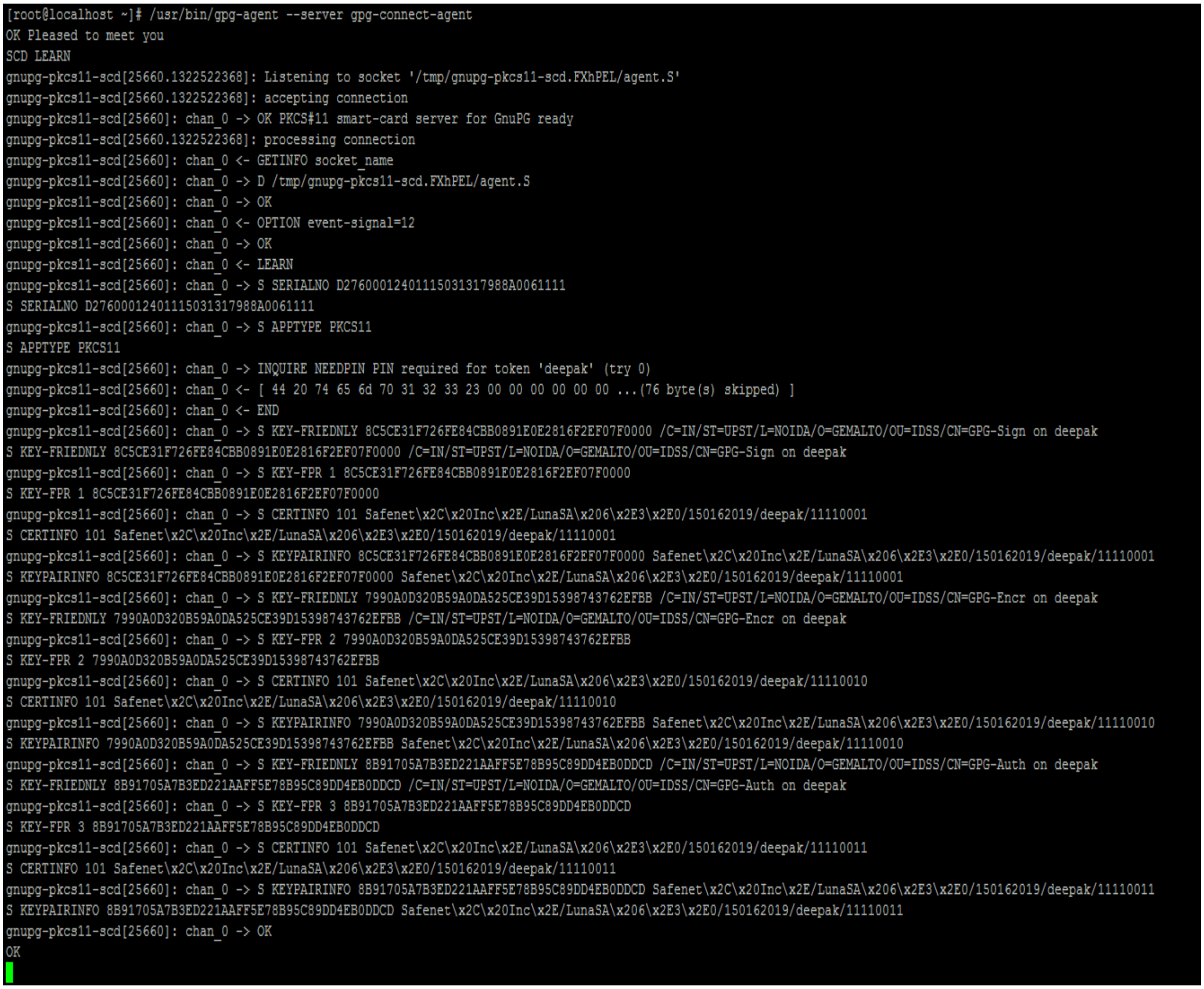
```
# export LD_LIBRARY_PATH=/usr/local/lib:$LD_LIBRARY_PATH

# export PATH=/usr/local/bin:$PATH
```

4. Execute the following command to connect the agent to HSM and get the keys from HSM:

```
# gpg-agent --server gpg-connect-agent
```

5. At the prompt, enter **SCD LEARN**. The pinentry program pop ups and prompts for the partition password. The output of the command will be similar to the following:



```
[root@localhost ~]# /usr/bin/gpg-agent --server gpg-connect-agent
OK Pleased to meet you
SCD LEARN
gnupg-pkcs11-scd[25660.1322522368]: Listening to socket '/tmp/gnupg-pkcs11-scd.FXhPEL/agent.S'
gnupg-pkcs11-scd[25660.1322522368]: accepting connection
gnupg-pkcs11-scd[25660]: chan_0 -> OK PKCS#11 smart-card server for GnuPG ready
gnupg-pkcs11-scd[25660.1322522368]: processing connection
gnupg-pkcs11-scd[25660]: chan_0 <- GETINFO socket_name
gnupg-pkcs11-scd[25660]: chan_0 -> D /tmp/gnupg-pkcs11-scd.FXhPEL/agent.S
gnupg-pkcs11-scd[25660]: chan_0 -> OK
gnupg-pkcs11-scd[25660]: chan_0 <- OPTION event-signal=12
gnupg-pkcs11-scd[25660]: chan_0 -> OK
gnupg-pkcs11-scd[25660]: chan_0 <- LEARN
gnupg-pkcs11-scd[25660]: chan_0 -> S SERIALNO D27600012401115031317988A0061111
S SERIALNO D27600012401115031317988A0061111
gnupg-pkcs11-scd[25660]: chan_0 -> S APPTYPE PKCS11
S APPTYPE PKCS11
gnupg-pkcs11-scd[25660]: chan_0 -> INQUIRE NEEDPIN PIN required for token 'deepak' (try 0)
gnupg-pkcs11-scd[25660]: chan_0 <- [ 44 20 74 65 6d 70 31 32 33 23 00 00 00 00 00 00 ...(76 byte(s) skipped) ]
gnupg-pkcs11-scd[25660]: chan_0 <- END
gnupg-pkcs11-scd[25660]: chan_0 -> S KEY-FRIEDNLY 8C5CE31F726FE84CBB0891E0E2816F2EF07F0000 /C=IN/ST=UPST/L=NOIDA/O=GEMALTO/OU=IDSS/CN=GPG-Sign on deepak
S KEY-FRIEDNLY 8C5CE31F726FE84CBB0891E0E2816F2EF07F0000 /C=IN/ST=UPST/L=NOIDA/O=GEMALTO/OU=IDSS/CN=GPG-Sign on deepak
gnupg-pkcs11-scd[25660]: chan_0 -> S KEY-FPR 1 8C5CE31F726FE84CBB0891E0E2816F2EF07F0000
S KEY-FPR 1 8C5CE31F726FE84CBB0891E0E2816F2EF07F0000
gnupg-pkcs11-scd[25660]: chan_0 -> S CERTINFO 101 Safenet\x2C\x20Inc\x2E/LunaSA\x206\x2E3\x2E0/150162019/deepak/11110001
S CERTINFO 101 Safenet\x2C\x20Inc\x2E/LunaSA\x206\x2E3\x2E0/150162019/deepak/11110001
gnupg-pkcs11-scd[25660]: chan_0 -> S KEYPAIRINFO 8C5CE31F726FE84CBB0891E0E2816F2EF07F0000 Safenet\x2C\x20Inc\x2E/LunaSA\x206\x2E3\x2E0/150162019/deepak/11110001
S KEYPAIRINFO 8C5CE31F726FE84CBB0891E0E2816F2EF07F0000 Safenet\x2C\x20Inc\x2E/LunaSA\x206\x2E3\x2E0/150162019/deepak/11110001
gnupg-pkcs11-scd[25660]: chan_0 -> S KEY-FRIEDNLY 7990A0D320B59A0DA525CE39D15398743762EFBB /C=IN/ST=UPST/L=NOIDA/O=GEMALTO/OU=IDSS/CN=GPG-Encr on deepak
S KEY-FRIEDNLY 7990A0D320B59A0DA525CE39D15398743762EFBB /C=IN/ST=UPST/L=NOIDA/O=GEMALTO/OU=IDSS/CN=GPG-Encr on deepak
gnupg-pkcs11-scd[25660]: chan_0 -> S KEY-FPR 2 7990A0D320B59A0DA525CE39D15398743762EFBB
S KEY-FPR 2 7990A0D320B59A0DA525CE39D15398743762EFBB
gnupg-pkcs11-scd[25660]: chan_0 -> S CERTINFO 101 Safenet\x2C\x20Inc\x2E/LunaSA\x206\x2E3\x2E0/150162019/deepak/11110010
S CERTINFO 101 Safenet\x2C\x20Inc\x2E/LunaSA\x206\x2E3\x2E0/150162019/deepak/11110010
gnupg-pkcs11-scd[25660]: chan_0 -> S KEYPAIRINFO 7990A0D320B59A0DA525CE39D15398743762EFBB Safenet\x2C\x20Inc\x2E/LunaSA\x206\x2E3\x2E0/150162019/deepak/11110010
S KEYPAIRINFO 7990A0D320B59A0DA525CE39D15398743762EFBB Safenet\x2C\x20Inc\x2E/LunaSA\x206\x2E3\x2E0/150162019/deepak/11110010
gnupg-pkcs11-scd[25660]: chan_0 -> S KEY-FRIEDNLY 8B91705A7B3ED221AAFF5E78B95C89DD4EB0DDCD /C=IN/ST=UPST/L=NOIDA/O=GEMALTO/OU=IDSS/CN=GPG-Auth on deepak
S KEY-FRIEDNLY 8B91705A7B3ED221AAFF5E78B95C89DD4EB0DDCD /C=IN/ST=UPST/L=NOIDA/O=GEMALTO/OU=IDSS/CN=GPG-Auth on deepak
gnupg-pkcs11-scd[25660]: chan_0 -> S KEY-FPR 3 8B91705A7B3ED221AAFF5E78B95C89DD4EB0DDCD
S KEY-FPR 3 8B91705A7B3ED221AAFF5E78B95C89DD4EB0DDCD
gnupg-pkcs11-scd[25660]: chan_0 -> S CERTINFO 101 Safenet\x2C\x20Inc\x2E/LunaSA\x206\x2E3\x2E0/150162019/deepak/11110011
S CERTINFO 101 Safenet\x2C\x20Inc\x2E/LunaSA\x206\x2E3\x2E0/150162019/deepak/11110011
gnupg-pkcs11-scd[25660]: chan_0 -> S KEYPAIRINFO 8B91705A7B3ED221AAFF5E78B95C89DD4EB0DDCD Safenet\x2C\x20Inc\x2E/LunaSA\x206\x2E3\x2E0/150162019/deepak/11110011
S KEYPAIRINFO 8B91705A7B3ED221AAFF5E78B95C89DD4EB0DDCD Safenet\x2C\x20Inc\x2E/LunaSA\x206\x2E3\x2E0/150162019/deepak/11110011
gnupg-pkcs11-scd[25660]: chan_0 -> OK
OK
```

NOTE: If you open the persistent session via **salogin,** the password prompt will not appear.

6. Look for the line **S KEY-FRIENDLY**, identify the signing/encryption/authentication certificate by the appropriate Common name (CN), and copy the 20 byte SHA-1 hash in the **gnupg-pkcs11-scd.conf** file as follows:

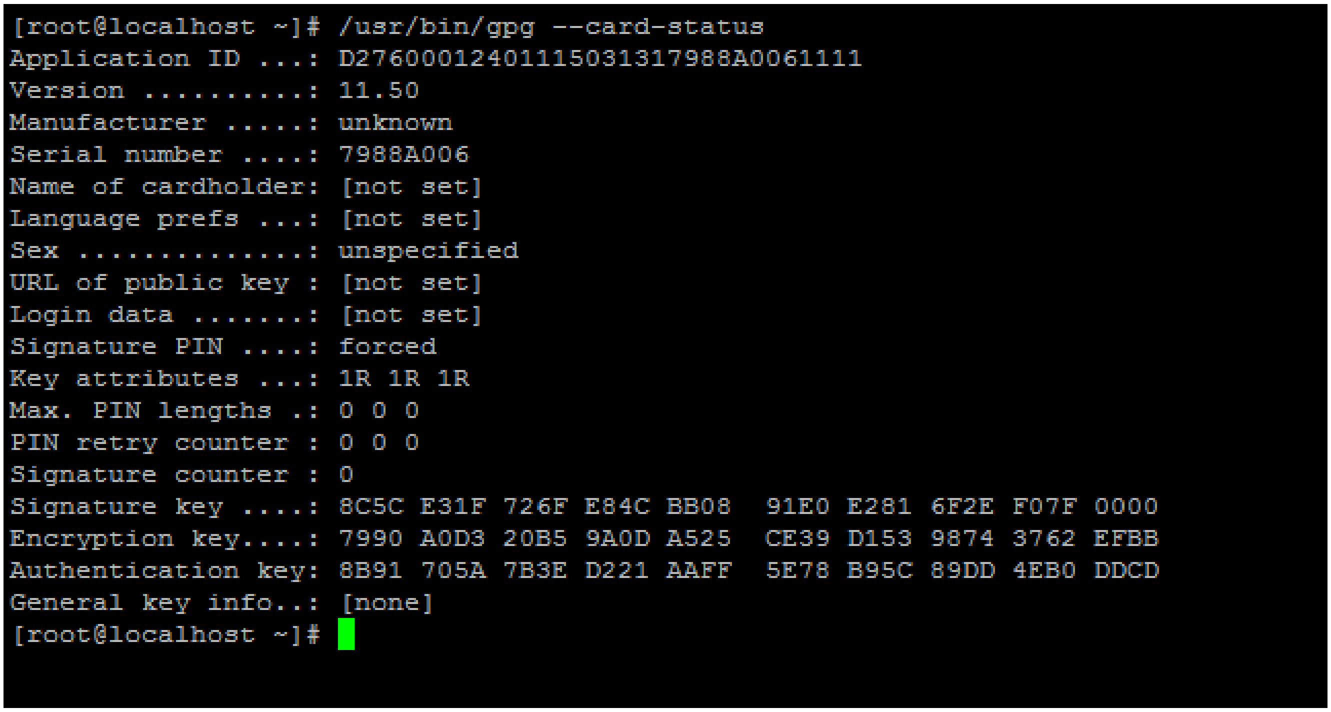
```
openpgp-sign 8C5CE31F726FE84CBB0891E0E2816F2EF07F0000

openpgp-encr 7990A0D320B59A0DA525CE39D15398743762EFBB

openpgp-auth 8B91705A7B3ED221AAFF5E78B95C89DD4EB0DDCD
```

7. Use the following command to enable GPG to discover all useful information of the card (or HSM partition in this case):

```
# gpg --card-status
```



```
[root@localhost ~]# /usr/bin/gpg --card-status
Application ID ...: D27600012401115031317988A0061111
Version ..........: 11.50
Manufacturer .....: unknown
Serial number ....: 7988A006
Name of cardholder: [not set]
Language prefs ...: [not set]
Sex ..............: unspecified
URL of public key : [not set]
Login data .......: [not set]
Signature PIN ....: forced
Key attributes ...: 1R 1R 1R
Max. PIN lengths .: 0 0 0
PIN retry counter : 0 0 0
Signature counter : 0
Signature key ....: 8C5C E31F 726F E84C BB08  91E0 E281 6F2E F07F 0000
Encryption key....: 7990 A0D3 20B5 9A0D A525  CE39 D153 9874 3762 EFBB
Authentication key: 8B91 705A 7B3E D221 AAFF  5E78 B95C 89DD 4EB0 DDCD
General key info..: [none]
[root@localhost ~]#
```

8. Execute the following commands to generate the GPG virtual keys. Note that the keys are not actually generated on the local host and only a reference to the HSM keys is returned and registered by GPG.

```
# gpg --card-edit
```

```
# Command> admin
```

```
# Command> generate
```

You need to provide the following inputs:

a. Respond "y" to Replace existing keys?

b. Do not backup keys if prompted.

c. Set the expiry parameter

d. Provide the key name when prompted for Real name.

   Note this name as it will be used to reference the GPG and RPM signing key going forward.

Testing GPG

You can test GPG in the following use-cases:

> File Signing and Verification

> RPM Signing and Verification

File Signing and Verification

To sign and verify a file, perform the following steps:

1. Run the following command:

```
# gpg --sign --default-key <Your key name> somefile
```

2. Provide the partition password. When signing is completed, then the file **somefile.gpg** will be created containing the original file contents and the signature.

   > **NOTE:** If you open persistent session via **salogin**, you will not be asked to provide the password.

3. The original file can be verified and recovered by running the following command:

   ```
   # gpg somefile.gpg
   ```

   Contents of the original file and recovered file will be the same.

RPM Signing and Verification

RPM Signing uses GPG under the hood for key management and crypto functions. Some GPG options can be invoked via the RPM command line, but some experimentation may be required in order to get the desired signature options and attributes. Follow the below sections sequentially to sign and verify the rpms.

1. **Environment variables**

   At a minimum the following environment setting may be required in order for RPM (via GPG) to call the pinentry program:

   ```
   GPG_TTY=$(tty)

   export GPG_TTY
   ```

2. **Signing an RPM**

   A script is required to demonstrate some of the common options that contribute to the signing operation and signature format. Create the script from the source listing, modify the script to reference the appropriate key name and then execute as follows:

   ```
   ./<scriptname> <your_rpm>
   ```

   An example of the script is provided below, copy and paste the below snippet in the file called **rpmsigntest**:

   ```
   #!/usr/bin/expect --

   spawn rpm --define "_gpg_name  GPG-Sign" --resign {*}$argv

   expect {

   "Enter pass phrase:" { send "\r" ; exp_continue }

   eof

   }
   ```

   Where GPG-Sign is your signing key name.

   > **NOTE:** Ensure that your system has expect installed, if not, install the expect using the below command:
   >
   > ```
   > # yum install expect expectk
   > ```

   Now run the script to sign the RPM, for example:

   ```
   # ./rpmsigntest example.rpm

   spawn rpm --define _gpg_name  GPG-Sign --resign example.rpm
   ```

```
Enter pass phrase:
Pass phrase is good.
example.rpm:
```

The pinentry program pop ups and prompts for the partition password but if the persistent session is opened via SALOGIN then there will be no prompt for password. Each RPM signing involves three separate signing operations and you receive prompt for the partition password each time.

3. **Verify a signed RPM**

   In order to verify the signature on the RPM file, the public key associated with the signing key needs to be imported into the GPG keychain.

   a. Export the key using the command:

   ```
   gpg --export --armor <your_keyname> > <your_keyfile>
   ```

   For example:

   ```
   # /usr/bin/gpg --export --armor GPG-Sign>gpg.key
   ```

   b. Import the key using the command:

   ```
   rpm –import <your_keyfile>
   ```

   For example:

   ```
   # rpm --import gpg.key
   ```

   c. Verify the signed RPM

   ```
   rpm --checksig <your_rpm>
   ```

   For example:

   ```
   # rpm --checksig example.rpm
   example.rpm: rsa sha1 (md5) pgp md5 OK
   ```

Unattended RPM Signing

You may need to perform the RPM signing as part of the product build and RPM creation processes. Thus it would be impractical for someone to enter the partition password three times for each RPM signature. To perform unattended rpm signing:

1. Start the gpg-agent daemon using the following command:

   ```
   gpg-agent --daemon [additional options – refer to gpg-agent man pages]
   ```

   > **NOTE:** If the gpg-agent is started in daemon mode at the command line however, it will remain in the background and the partition password (after the initial invocation) will be cached for subsequent use. On the first rpm signing, the pinentry prompts for the password once.

2. Create a simple expect script

   ```
   #!/usr/bin/expect --
   spawn rpm --define "_gpg_name <your keyname here>" --resign {*}$argv
   expect {
   ```

```
"Enter pass phrase:" { send "\r" ; exp_continue }

eof

}
```

> **NOTE: Expect** is an extension to the Tcl scripting language and is a program used to automate interactions with other applications that expose a text terminal interface. The easiest way to install it on a RedHat/CentOS system is with yum as follows:
>
> **# yum install expect expectk**

3. Sign the rpm

```
# ./rpm-sign.exp <rpm_name>
```