The GNU Privacy Guard Team

GnuPG

2.3.7

Integration Guide

CryptoServer HSM

Utimaco SecurityServer Software 4.45.5

utimaco®

Imprint

Copyright 2026	Utimaco IS GmbH
	Krefelder Straße 220
	52070 Aachen
	Germany
Phone	AMERICAS +1-844-UTIMACO (+1 844-884-6226)
	EMEA +49 800-627-3081
	APAC +81 800-919-1301
Internet	https://support.hsm.utimaco.com/
e-mail	support@utimaco.com
Document Version	1.0.0
Date	2026-03-04
Status	PUBLISHED
Document No.	IG-2026-0030

All rights reserved	No part of this documentation may be reproduced in any form (printing, photocopy or according to any other process) without the written approval of Utimaco IS GmbH or be processed, reproduced or distributed using electronic systems. Utimaco IS GmbH reserves the right to modify or amend the documentation at any time without prior notice. Utimaco IS GmbH assumes no liability for typographical errors and damages incurred due to them. All trademarks and registered trademarks are the property of their respective owners.

Table of Contents

1 About This Guide 5
1.1 Target Audience for This Guide 5
1.2 Document Conventions 5
1.3 Abbreviations 6
2 Overview 8
2.1 GnuPG (GPG) 8
2.2 Utimaco CryptoServer HSM 8
3 Integration Requirements and Prerequisites 9
3.1 Tested Versions 9
3.2 Software Requirements 9
3.3 Hardware Requirements 10
3.4 Prerequisites 10
4 Integrating GnuPG on Linux 11
4.1 Installing Dependent Packages for GnuPG 11
4.1.1 Installing Libgpg-error 11
4.1.2 Installing Libgcrypt 14
4.1.3 Installing Libassuan 17
4.1.4 Installing Libksba 19
4.1.5 Installing NPTH 22
4.1.6 Installing Pinentry 23
4.1.7 Installing GnuPG 25
4.2 Installing GnuPG-PKCS11-SCD 29
4.3 PKCS#11 Configuration for CryptoServer 33
4.3.1 Create SO User and Initialize a Slot 34
4.4 Configuring GnuPG to Use Utimaco HSM 34
4.4.1 Generating Key and Certificate for GnuPG 35
4.4.2 Adding certificate to Gnupg 39
4.4.3 Signing, Encryption, Decryption and Verification with GnuPG 41
4.4.4 RPM Signing and Verification with GnuPG 44
4.4.4.1 RPM Signing 44
4.4.4.2 Signed RPM Verification 45

5 Troubleshooting 48
6 Further Information 49
7 References 50

1 About This Guide

This guide provides an integration explaining how to integrate an Utimaco CryptoServer Hardware Security Module (HSM) with GnuPG. Utimaco HSM securely stores the keys and certificate used by GnuPG for encryption, decryption, signing and verification.

1.1 Target Audience for This Guide

This guide is intended for administrators of GnuPG and of Utimaco HSMs.

1.2 Document Conventions

The following conventions are used in this guide:

Convention	Use	Example
Bold	Items of the Graphical User Interface (GUI), e.g., menu options	Select **Details** and click on **Properties** button
`Monospaced`	Code that is given for explanation or as an example, file paths	`certreq.exe -new request.inf IISCertRequest.csr`
Italic	References and important terms	Operating system listed in *Tested Versions*

Table 1: Document conventions

We use special icons to highlight the most important notes and information.

Here you will find important safety information that should be followed.

Here you will find additional notes or supplementary information.

This message marks the result expected after the successful execution of an instruction.

1.3 Abbreviations

The following abbreviations are used in this guide:

Abbreviation	Meaning
BSD	Berkeley Source Distribution
CD	Compact Disc
CSADM	CryptoServer Command-line Administration Tool
EPEL	Extra Packages for Enterprise Linux
GnuPG (GPG)	GNU Privacy Guard
GUI	Graphical User Interface
HSM	Hardware Security Module
ID	Identity
IP	Internet Protocol
Abbreviation	*Meaning*
LAN	Local Area Network

Abbreviation	Meaning
NPTH	New GNU Portable Threads
PCIe	PCI Express Interface
PIN	Personal Identification number
PKCS#11	Public-Key Cryptography Standard #11
RPM	Red Hat Package Manager
RSA	Rivest-Shamir-Adleman
SCD	Smart Card Daemon
SO	Security Officer
SSH	Secure Shell or Secure Socket Shell
URL	Uniform Resource Locator

Table 2: List of abbreviations

2 Overview

2.1 GnuPG (GPG)

GnuPG is a complete and free implementation of the OpenPGP standard as defined by RFC4880 (also known as PGP). GnuPG allows you to encrypt and sign your data and communications; it features a versatile key management system, along with access modules for all kinds of public key directories. GnuPG, also known as GPG, is a command line tool with features for easy integration with other applications. GnuPG also provides support for S/MIME and Secure Shell (ssh).

Gnupg-pkcs1-scd is a project to implement a BSD-licensed smart-card daemon to enable the use of PKCS#11 tokens with GnuPG.

2.2 Utimaco CryptoServer HSM

CryptoServer is a hardware security module developed by Utimaco IS GmbH. CryptoServer is a physically protected specialized computer unit designed to perform sensitive cryptographic tasks and to securely manage as well as store cryptographic keys and data. It can be used as a universal, independent security component for heterogeneous computer systems.

3 Integration Requirements and Prerequisites

Ensure the system environment you will be using meets the following hardware and software requirements.

This guide assumes that the user has already installed and configured required Software.

3.1 Tested Versions

The integrations that have been successfully tested with the Utimaco HSM with GnuPG.

Operating System	GnuPG Version	GnuPG PKCS11 SCD Version	Utimaco Security Server Version	Utimaco HSM
RHEL8	2.3.7	0.10.0	SecurityServer 4.45.5	CryptoServer CSe-Series/Se-Series

Table 3: List of tested versions

3.2 Software Requirements

Software	Software Requirements
GnuPG	GnuPG 2.3.7
GnuPG PKCS11 SCD	GnuPG PKCS11 SCD 0.10.0
HSM Interface	SecurityServer PKCS#11 Provider

Table 4: List of software requirements

3.3 Hardware Requirements

Hardware	Hardware Requirements
Utimaco LAN HSM	CryptoServer CSe-Series/Se-Series LAN with firmware SecurityServer 4.45.5.0 or higher
Utimaco PCI-e HSM	CryptoServer CSe-Series/Se-Series PCI-e with firmware SecurityServer 4.45.5.0 or higher

Table 5: List of hardware requirements

Setup an account on the Utimaco support portal and request download access at the following URL.

https://support.hsm.utimaco.com

3.4 Prerequisites

Before you begin, please ensure that you have installed/setup:

- CryptoServer is setup and configured. Refer the CryptoServer documentations to setup the HSM
- CryptoServer Default Admin should be replaced with a new admin user
- Operating system listed in Tested Versions
- SecurityServer listed in Tested Version
- Public and private key pair must be created and stored onto each HSM. Refer the CryptoServer documentations to setup the keys
- PKCS#11 library is setup and configured as per the environment. Refer the CryptoServer documentations to setup and configure the PKCS#11 library for CryptoServer
- Familiarize yourself with the GnuPG. Refer GnuPG Documentation portal for more information

4 Integrating GnuPG on Linux

4.1 Installing Dependent Packages for GnuPG

GnuPG requires the following dependent packages:

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

4.1.1 Installing Libgpg-error

1. Download the libgpg-error installation file

›_ Console

```
# wget https://www.gnupg.org/ftp/gcrypt/libgpg-error/libgpg-error1.45.tar.bz2
```

```
[root@rk ~]# wget https://www.gnupg.org/ftp/gcrypt/libgpg-error/libgpg-error-1.45.tar.bz2
--2022-09-04 10:13:43--  https://www.gnupg.org/ftp/gcrypt/libgpg-error/libgpg-error-1.45.tar.bz2
Resolving www.gnupg.org (www.gnupg.org)... 217.69.76.60, 2001:aa8:fff1:2100::60
Connecting to www.gnupg.org (www.gnupg.org)|217.69.76.60|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1015954 (992K) [text/plain]
Saving to: 'libgpg-error-1.45.tar.bz2'

libgpg-error-1.45.tar.bz2           100%[=============================================================================>] 992.14K  1.22MB/s    in 0.8s

2022-09-04 10:13:45 (1.22 MB/s) - 'libgpg-error-1.45.tar.bz2' saved [1015954/1015954]
```

Figure 1 : Downloading libgpg-error

2. Extract the file

›_ Console

```
# tar -xjf libgpg-error-1.45.tar.bz2
```

3. Go to the directory where the file is extracted

›_ Console

```
# cd libgpg-error-1.45
```

4. Run the following command to compile and install

›_ Console

```
# ./configure
# make
# make install
```

```
[root@rk libgpg-error-1.45]# make install
Making install in m4
make[1]: Entering directory '/root/libgpg-error-1.45/m4'
make[2]: Entering directory '/root/libgpg-error-1.45/m4'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/libgpg-error-1.45/m4'
make[1]: Leaving directory '/root/libgpg-error-1.45/m4'
Making install in src
make[1]: Entering directory '/root/libgpg-error-1.45/src'
make  install-am
make[2]: Entering directory '/root/libgpg-error-1.45/src'
make[3]: Entering directory '/root/libgpg-error-1.45/src'
 /usr/bin/mkdir -p '/usr/local/lib'
 /bin/sh ../libtool   --mode=install /usr/bin/install -c   libgpg-error.la '/usr/local/lib'
libtool: install: /usr/bin/install -c .libs/libgpg-error.so.0.33.0 /usr/local/lib/libgpg-error.so.0.33.0
libtool: install: (cd /usr/local/lib && { ln -s -f libgpg-error.so.0.33.0 libgpg-error.so.0 || { rm -f libgpg-error.so.0 && ln -s libgpg-error.so.0.33.0
 libgpg-error.so.0; }; })
libtool: install: (cd /usr/local/lib && { ln -s -f libgpg-error.so.0.33.0 libgpg-error.so || { rm -f libgpg-error.so && ln -s libgpg-error.so.0.33.0 lib
gpg-error.so; }; })
libtool: install: /usr/bin/install -c .libs/libgpg-error.lai /usr/local/lib/libgpg-error.la
libtool: finish: PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin:/sbin" ldconfig -n /usr/local/lib
----------------------------------------------------------------------
Libraries have been installed in:
   /usr/local/lib

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,-rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to `/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------
 /usr/bin/mkdir -p '/usr/local/bin'
  /bin/sh ../libtool   --mode=install /usr/bin/install -c gpg-error '/usr/local/bin'
libtool: install: /usr/bin/install -c .libs/gpg-error /usr/local/bin/gpg-error
 /usr/bin/mkdir -p '/usr/local/bin'
 /usr/bin/install -c gpgrt-config gpg-error-config '/usr/local/bin'
 /usr/bin/mkdir -p '/usr/local/share/aclocal'
 /usr/bin/install -c -m 644 gpg-error.m4 gpgrt.m4 '/usr/local/share/aclocal'
 /usr/bin/mkdir -p '/usr/local/include'
 /usr/bin/install -c -m 644 gpg-error.h gpgrt.h '/usr/local/include'
 /usr/bin/mkdir -p '/usr/local/lib/pkgconfig'
 /usr/bin/install -c -m 644 gpg-error.pc '/usr/local/lib/pkgconfig'
make[3]: Leaving directory '/root/libgpg-error-1.45/src'
make[2]: Leaving directory '/root/libgpg-error-1.45/src'
make[1]: Leaving directory '/root/libgpg-error-1.45/src'
Making install in doc
make[1]: Entering directory '/root/libgpg-error-1.45/doc'
make[2]: Entering directory '/root/libgpg-error-1.45/doc'
 /usr/bin/mkdir -p '/usr/local/bin'
  /bin/sh ../libtool   --mode=install /usr/bin/install -c yat2m '/usr/local/bin'
libtool: install: /usr/bin/install -c yat2m /usr/local/bin/yat2m
sed '/^##/ d' ./errorref.txt >errorref.txt.x
echo "# Installed by libgpg-error 1.45" >>errorref.txt.x
/bin/sh /root/libgpg-error-1.45/build-aux/install-sh -d /usr/local/share/libgpg-error
/usr/bin/install -c -m 644 errorref.txt.x /usr/local/share/libgpg-error/errorref.txt
 /usr/bin/mkdir -p '/usr/local/share/info'
 /usr/bin/install -c -m 644 ./gpgrt.info '/usr/local/share/info'
 install-info --info-dir='/usr/local/share/info' '/usr/local/share/info/gpgrt.info'
 /usr/bin/mkdir -p '/usr/local/share/man/man1'
 /usr/bin/install -c -m 644 gpgrt-config.1 '/usr/local/share/man/man1'
make[2]: Leaving directory '/root/libgpg-error-1.45/doc'
make[1]: Leaving directory '/root/libgpg-error-1.45/doc'
Making install in tests
```

Figure 2 : Installing libgpg-error

```
make[1]: Entering directory '/root/libgpg-error-1.45/tests'
make[2]: Entering directory '/root/libgpg-error-1.45/tests'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/libgpg-error-1.45/tests'
make[1]: Leaving directory '/root/libgpg-error-1.45/tests'
Making install in po
make[1]: Entering directory '/root/libgpg-error-1.45/po'
installing cs.gmo as /usr/local/share/locale/cs/LC_MESSAGES/libgpg-error.mo
installing da.gmo as /usr/local/share/locale/da/LC_MESSAGES/libgpg-error.mo
installing de.gmo as /usr/local/share/locale/de/LC_MESSAGES/libgpg-error.mo
installing eo.gmo as /usr/local/share/locale/eo/LC_MESSAGES/libgpg-error.mo
installing es.gmo as /usr/local/share/locale/es/LC_MESSAGES/libgpg-error.mo
installing fr.gmo as /usr/local/share/locale/fr/LC_MESSAGES/libgpg-error.mo
installing hu.gmo as /usr/local/share/locale/hu/LC_MESSAGES/libgpg-error.mo
installing it.gmo as /usr/local/share/locale/it/LC_MESSAGES/libgpg-error.mo
installing ja.gmo as /usr/local/share/locale/ja/LC_MESSAGES/libgpg-error.mo
installing nl.gmo as /usr/local/share/locale/nl/LC_MESSAGES/libgpg-error.mo
installing pl.gmo as /usr/local/share/locale/pl/LC_MESSAGES/libgpg-error.mo
installing pt.gmo as /usr/local/share/locale/pt/LC_MESSAGES/libgpg-error.mo
installing ro.gmo as /usr/local/share/locale/ro/LC_MESSAGES/libgpg-error.mo
installing ru.gmo as /usr/local/share/locale/ru/LC_MESSAGES/libgpg-error.mo
installing sr.gmo as /usr/local/share/locale/sr/LC_MESSAGES/libgpg-error.mo
installing sv.gmo as /usr/local/share/locale/sv/LC_MESSAGES/libgpg-error.mo
installing tr.gmo as /usr/local/share/locale/tr/LC_MESSAGES/libgpg-error.mo
installing uk.gmo as /usr/local/share/locale/uk/LC_MESSAGES/libgpg-error.mo
installing vi.gmo as /usr/local/share/locale/vi/LC_MESSAGES/libgpg-error.mo
installing zh_CN.gmo as /usr/local/share/locale/zh_CN/LC_MESSAGES/libgpg-error.mo
installing zh_TW.gmo as /usr/local/share/locale/zh_TW/LC_MESSAGES/libgpg-error.mo
if test "libgpg-error" = "gettext-tools"; then \
  /usr/bin/mkdir -p /usr/local/share/gettext/po; \
  for file in Makefile.in.in remove-potcdate.sin quot.sed boldquot.sed en@quot.header en@boldquot.header insert-header.sin Rules-quot   Makevars.templat
e; do \
    /usr/bin/install -c -m 644 ./$file \
                     /usr/local/share/gettext/po/$file; \
  done; \
  for file in Makevars; do \
    rm -f /usr/local/share/gettext/po/$file; \
  done; \
else \
  : ; \
fi
make[1]: Leaving directory '/root/libgpg-error-1.45/po'
Making install in lang
make[1]: Entering directory '/root/libgpg-error-1.45/lang'
Making install in cl
make[2]: Entering directory '/root/libgpg-error-1.45/lang/cl'
make[3]: Entering directory '/root/libgpg-error-1.45/lang/cl'
make[3]: Nothing to be done for 'install-exec-am'.
 /usr/bin/mkdir -p '/usr/local/share/common-lisp/source/gpg-error'
 /usr/bin/install -c -m 644 gpg-error.asd gpg-error-package.lisp gpg-error.lisp '/usr/local/share/common-lisp/source/gpg-error'
 /usr/bin/mkdir -p '/usr/local/share/common-lisp/source/gpg-error'
 /usr/bin/install -c -m 644 gpg-error-codes.lisp '/usr/local/share/common-lisp/source/gpg-error'
make[3]: Leaving directory '/root/libgpg-error-1.45/lang/cl'
make[2]: Leaving directory '/root/libgpg-error-1.45/lang/cl'
make[2]: Entering directory '/root/libgpg-error-1.45/lang'
make[3]: Entering directory '/root/libgpg-error-1.45/lang'
make[3]: Nothing to be done for 'install-exec-am'.
make[3]: Nothing to be done for 'install-data-am'.
make[3]: Leaving directory '/root/libgpg-error-1.45/lang'
make[2]: Leaving directory '/root/libgpg-error-1.45/lang'
make[1]: Leaving directory '/root/libgpg-error-1.45/lang'
make[1]: Entering directory '/root/libgpg-error-1.45'
make[2]: Entering directory '/root/libgpg-error-1.45'
make[2]: Nothing to be done for 'install-exec-am'.
make  install-data-hook
make[3]: Entering directory '/root/libgpg-error-1.45'
make[3]: Nothing to be done for 'install-data-hook'.
make[3]: Leaving directory '/root/libgpg-error-1.45'
make[2]: Leaving directory '/root/libgpg-error-1.45'
make[1]: Leaving directory '/root/libgpg-error-1.45'
[root@rk libgpg-error-1.45]#
```

4.1.2 Installing Libgcrypt

1. Download the libgcrypt installation file

›_ Console

```
# wget https://www.gnupg.org/ftp/gcrypt/libgcrypt/libgcrypt-1.10.1.tar.bz2
```

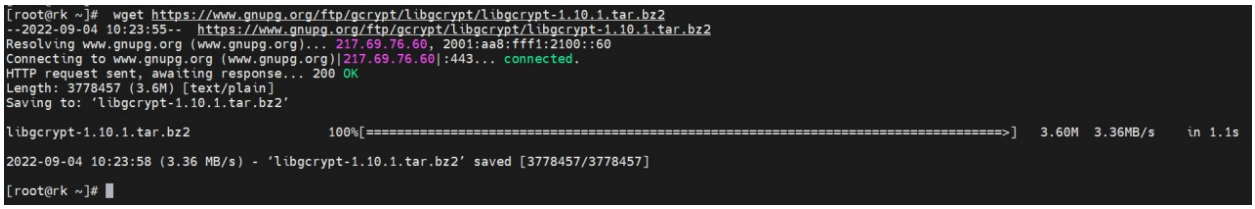

Figure 3 : Downloading libgcrypt

2. Extract the file

›_ Console

```
# tar -xjf libgcrypt-1.10.1.tar.bz2
```

3. Go to the directory where the file is extracted

›_ Console

```
# cd libgcrypt-1.10.1
```

4. Run the following command to compile and install

›_ Console

```
# ./configure
# make
# make install
```

```
[root@rk libgcrypt-1.10.1]# make install
Making install in compat
make[1]: Entering directory '/root/libgcrypt-1.10.1/compat'
make[2]: Entering directory '/root/libgcrypt-1.10.1/compat'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/libgcrypt-1.10.1/compat'
make[1]: Leaving directory '/root/libgcrypt-1.10.1/compat'
Making install in mpi
make[1]: Entering directory '/root/libgcrypt-1.10.1/mpi'
make[2]: Entering directory '/root/libgcrypt-1.10.1/mpi'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/libgcrypt-1.10.1/mpi'
make[1]: Leaving directory '/root/libgcrypt-1.10.1/mpi'
Making install in cipher
make[1]: Entering directory '/root/libgcrypt-1.10.1/cipher'
make[2]: Entering directory '/root/libgcrypt-1.10.1/cipher'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/libgcrypt-1.10.1/cipher'
make[1]: Leaving directory '/root/libgcrypt-1.10.1/cipher'
Making install in random
make[1]: Entering directory '/root/libgcrypt-1.10.1/random'
make[2]: Entering directory '/root/libgcrypt-1.10.1/random'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/libgcrypt-1.10.1/random'
make[1]: Leaving directory '/root/libgcrypt-1.10.1/random'
Making install in src
make[1]: Entering directory '/root/libgcrypt-1.10.1/src'
make[2]: Entering directory '/root/libgcrypt-1.10.1/src'
 /usr/bin/mkdir -p '/usr/local/lib'
 /bin/sh ../libtool   --mode=install /usr/bin/install -c   libgcrypt.la '/usr/local/lib'
libtool: install: /usr/bin/install -c .libs/libgcrypt.so.20.4.1 /usr/local/lib/libgcrypt.so.20.4.1
libtool: install: (cd /usr/local/lib && { ln -s -f libgcrypt.so.20.4.1 libgcrypt.so.20 || { rm -f libgcrypt.so.20 && ln -s libgcrypt.so.20.4.1 libgcrypt.so.20; }; })
libtool: install: (cd /usr/local/lib && { ln -s -f libgcrypt.so.20.4.1 libgcrypt.so || { rm -f libgcrypt.so && ln -s libgcrypt.so.20.4.1 libgcrypt.so; }; })
libtool: install: /usr/bin/install -c .libs/libgcrypt.lai /usr/local/lib/libgcrypt.la
libtool: finish: PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin:/sbin" ldconfig -n /usr/local/lib
----------------------------------------------------------------------
Libraries have been installed in:
   /usr/local/lib

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,-rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to `/etc/ld.so.conf'
```

Figure 4 : Installing libgcrypt

```
See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------
 /usr/bin/mkdir -p '/usr/local/bin'
  /bin/sh ../libtool   --mode=install /usr/bin/install -c dumpsexp hmac256 mpicalc '/usr/local/bin'
libtool: install: /usr/bin/install -c dumpsexp /usr/local/bin/dumpsexp
libtool: install: /usr/bin/install -c hmac256 /usr/local/bin/hmac256
libtool: install: /usr/bin/install -c .libs/mpicalc /usr/local/bin/mpicalc
 /usr/bin/mkdir -p '/usr/local/bin'
 /usr/bin/install -c libgcrypt-config '/usr/local/bin'
 /usr/bin/mkdir -p '/usr/local/share/aclocal'
 /usr/bin/install -c -m 644 libgcrypt.m4 '/usr/local/share/aclocal'
 /usr/bin/mkdir -p '/usr/local/include'
 /usr/bin/install -c -m 644 gcrypt.h '/usr/local/include'
 /usr/bin/mkdir -p '/usr/local/lib/pkgconfig'
 /usr/bin/install -c -m 644 libgcrypt.pc '/usr/local/lib/pkgconfig'
make[2]: Leaving directory '/root/libgcrypt-1.10.1/src'
make[1]: Leaving directory '/root/libgcrypt-1.10.1/src'
Making install in doc
make[1]: Entering directory '/root/libgcrypt-1.10.1/doc'
make  install-am
make[2]: Entering directory '/root/libgcrypt-1.10.1/doc'
make[3]: Entering directory '/root/libgcrypt-1.10.1/doc'
make[3]: Nothing to be done for 'install-exec-am'.
 /usr/bin/mkdir -p '/usr/local/share/info'
 /usr/bin/install -c -m 644 ./gcrypt.info ./gcrypt.info-1 ./gcrypt.info-2 '/usr/local/share/info'
 install-info --info-dir='/usr/local/share/info' '/usr/local/share/info/gcrypt.info'
 /usr/bin/mkdir -p '/usr/local/share/man/man1'
 /usr/bin/install -c -m 644 hmac256.1 '/usr/local/share/man/man1'
make[3]: Leaving directory '/root/libgcrypt-1.10.1/doc'
make[2]: Leaving directory '/root/libgcrypt-1.10.1/doc'
make[1]: Leaving directory '/root/libgcrypt-1.10.1/doc'
Making install in tests
make[1]: Entering directory '/root/libgcrypt-1.10.1/tests'
make[2]: Entering directory '/root/libgcrypt-1.10.1/tests'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/libgcrypt-1.10.1/tests'
make[1]: Leaving directory '/root/libgcrypt-1.10.1/tests'
make[1]: Entering directory '/root/libgcrypt-1.10.1'
make[2]: Entering directory '/root/libgcrypt-1.10.1'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/libgcrypt-1.10.1'
make[1]: Leaving directory '/root/libgcrypt-1.10.1'
[root@rk libgcrypt-1.10.1]#
```

4.1.3 Installing Libassuan

1. Download the libassuan installation file

›_ Console

```
# wget https://www.gnupg.org/ftp/gcrypt/libassuan/libassuan-2.5.5.tar.bz2
```

```
[root@rk ~]#  wget https://www.gnupg.org/ftp/gcrypt/libassuan/libassuan-2.5.5.tar.bz2
--2022-09-04 10:37:50--  https://www.gnupg.org/ftp/gcrypt/libassuan/libassuan-2.5.5.tar.bz2
Resolving www.gnupg.org (www.gnupg.org)... 217.69.76.60, 2001:aa8:fff1:2100::60
Connecting to www.gnupg.org (www.gnupg.org)|217.69.76.60|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 572263 (559K) [text/plain]
Saving to: 'libassuan-2.5.5.tar.bz2'

libassuan-2.5.5.tar.bz2                 100%[=======================================================================================>] 558.85K   844KB/s    in 0.7s

2022-09-04 10:37:51 (844 KB/s) - 'libassuan-2.5.5.tar.bz2' saved [572263/572263]

[root@rk ~]#
```

Figure 5 : Downloading libgassuan

2. Extract the file

›_ Console

```
# tar -xjf libassuan-2.5.5.tar.bz2
```

3. Go to the directory where the file is extracted

›_ Console

```
# cd libassuan-2.5.5
```

4. Run the following command to compile and install

›_ Console

```
# ./configure
# make
# make install
```

```
[root@rk libassuan-2.5.5]# make install
Making install in m4
make[1]: Entering directory '/root/libassuan-2.5.5/m4'
make[2]: Entering directory '/root/libassuan-2.5.5/m4'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/libassuan-2.5.5/m4'
make[1]: Leaving directory '/root/libassuan-2.5.5/m4'
Making install in src
make[1]: Entering directory '/root/libassuan-2.5.5/src'
make  install-am
make[2]: Entering directory '/root/libassuan-2.5.5/src'
make[3]: Entering directory '/root/libassuan-2.5.5/src'
 /usr/bin/mkdir -p '/usr/local/lib'
 /bin/sh ../libtool   --mode=install /usr/bin/install -c   libassuan.la '/usr/local/lib'
libtool: install: /usr/bin/install -c .libs/libassuan.so.0.8.5 /usr/local/lib/libassuan.so.0.8.5
libtool: install: (cd /usr/local/lib && { ln -s -f libassuan.so.0.8.5 libassuan.so.0 || { rm -f libassuan.so.0 && ln -s libassuan.so.0.8.5 libassuan.so.0; }; })
libtool: install: (cd /usr/local/lib && { ln -s -f libassuan.so.0.8.5 libassuan.so || { rm -f libassuan.so && ln -s libassuan.so.0.8.5 libassuan.so; }; })
libtool: install: /usr/bin/install -c .libs/libassuan.lai /usr/local/lib/libassuan.la
libtool: finish: PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin:/sbin" ldconfig -n /usr/local/lib
----------------------------------------------------------------------
Libraries have been installed in:
   /usr/local/lib

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,-rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to `/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------
 /usr/bin/mkdir -p '/usr/local/bin'
 /usr/bin/install -c libassuan-config '/usr/local/bin'
make  install-exec-hook
make[4]: Entering directory '/root/libassuan-2.5.5/src'
make[4]: Nothing to be done for 'install-exec-hook'.
make[4]: Leaving directory '/root/libassuan-2.5.5/src'
 /usr/bin/mkdir -p '/usr/local/share/aclocal'
```

Figure 6 : Installing libassuan


```
[root@rk libassuan-2.5.5]# make install
Making install in m4
make[1]: Entering directory '/root/libassuan-2.5.5/m4'
make[2]: Entering directory '/root/libassuan-2.5.5/m4'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/libassuan-2.5.5/m4'
make[1]: Leaving directory '/root/libassuan-2.5.5/m4'
Making install in src
make[1]: Entering directory '/root/libassuan-2.5.5/src'
make  install-am
make[2]: Entering directory '/root/libassuan-2.5.5/src'
make[3]: Entering directory '/root/libassuan-2.5.5/src'
 /usr/bin/mkdir -p '/usr/local/lib'
 /bin/sh ../libtool   --mode=install /usr/bin/install -c   libassuan.la '/usr/local/lib'
libtool: install: /usr/bin/install -c .libs/libassuan.so.0.8.5 /usr/local/lib/libassuan.so.0.8.5
libtool: install: (cd /usr/local/lib && { ln -s -f libassuan.so.0.8.5 libassuan.so.0 || { rm -f libassuan.so.0 && ln -s libassuan.so.0.8.5 libassuan.so.0; }; })
libtool: install: (cd /usr/local/lib && { ln -s -f libassuan.so.0.8.5 libassuan.so || { rm -f libassuan.so && ln -s libassuan.so.0.8.5 libassuan.so; }; })
libtool: install: /usr/bin/install -c .libs/libassuan.lai /usr/local/lib/libassuan.la
libtool: finish: PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin:/sbin" ldconfig -n /usr/local/lib
----------------------------------------------------------------------
Libraries have been installed in:
   /usr/local/lib

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,-rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to `/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------
 /usr/bin/mkdir -p '/usr/local/bin'
 /usr/bin/install -c libassuan-config '/usr/local/bin'
make  install-exec-hook
make[4]: Entering directory '/root/libassuan-2.5.5/src'
make[4]: Nothing to be done for 'install-exec-hook'.
make[4]: Leaving directory '/root/libassuan-2.5.5/src'
 /usr/bin/mkdir -p '/usr/local/share/aclocal'
```


4.1.4 Installing Libksba

1. Download the libksba installation file

›_ Console

```
# wget https://www.gnupg.org/ftp/gcrypt/libksba/libksba-1.6.0.tar.bz2
```

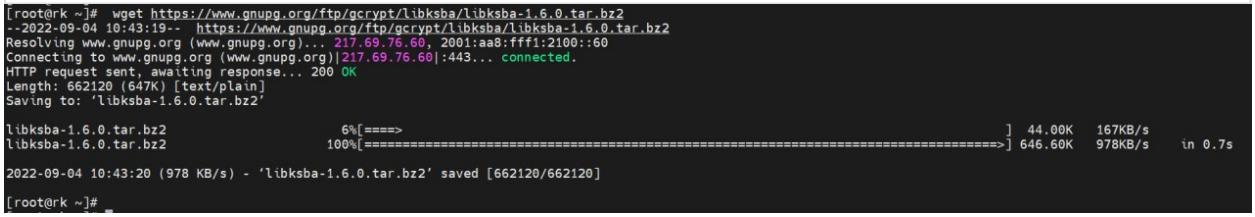

Figure 7 : Downloading libksba

2. Extract the file

›_ Console

```
# tar -xjf libksba-1.6.0.tar.bz2
```

3. Go to the directory where the file is extracted

›_ Console

```
# cd libksba-1.6.0
```

4. Run the following command to compile and install

›_ Console

```
# ./configure
# make
# make install
```

```
[root@rk libksba-1.6.0]# make install
Making install in m4
make[1]: Entering directory '/root/libksba-1.6.0/m4'
make[2]: Entering directory '/root/libksba-1.6.0/m4'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/libksba-1.6.0/m4'
make[1]: Leaving directory '/root/libksba-1.6.0/m4'
Making install in gl
make[1]: Entering directory '/root/libksba-1.6.0/gl'
make  install-am
make[2]: Entering directory '/root/libksba-1.6.0/gl'
make[3]: Entering directory '/root/libksba-1.6.0/gl'
make[3]: Nothing to be done for 'install-exec-am'.
make[3]: Nothing to be done for 'install-data-am'.
make[3]: Leaving directory '/root/libksba-1.6.0/gl'
make[2]: Leaving directory '/root/libksba-1.6.0/gl'
make[1]: Leaving directory '/root/libksba-1.6.0/gl'
Making install in src
make[1]: Entering directory '/root/libksba-1.6.0/src'
make  install-am
make[2]: Entering directory '/root/libksba-1.6.0/src'
make[3]: Entering directory '/root/libksba-1.6.0/src'
 /usr/bin/mkdir -p '/usr/local/bin'
 /usr/bin/install -c ksba-config '/usr/local/bin'
 /usr/bin/mkdir -p '/usr/local/lib'
 /bin/sh ../libtool   --mode=install /usr/bin/install -c   libksba.la '/usr/local/lib'
libtool: install: /usr/bin/install -c .libs/libksba.so.8.14.0 /usr/local/lib/libksba.so.8.14.0
libtool: install: (cd /usr/local/lib && { ln -s -f libksba.so.8.14.0 libksba.so.8 || { rm -f libksba.so.8 && ln -s libksba.so.8.14.0 libksba.so.8; }; })
libtool: install: (cd /usr/local/lib && { ln -s -f libksba.so.8.14.0 libksba.so || { rm -f libksba.so && ln -s libksba.so.8.14.0 libksba.so; }; })
libtool: install: /usr/bin/install -c .libs/libksba.lai /usr/local/lib/libksba.la
libtool: finish: PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin:/sbin" ldconfig -n /usr/local/lib
----------------------------------------------------------------------
Libraries have been installed in:
   /usr/local/lib

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,-rpath -Wl,LIBDIR' linker flag
```

Figure 8 : Installing libksba

```
   - have your system administrator add LIBDIR to `/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------
 /usr/bin/mkdir -p '/usr/local/share/aclocal'
 /usr/bin/install -c -m 644 ksba.m4 '/usr/local/share/aclocal'
 /usr/bin/mkdir -p '/usr/local/include'
 /usr/bin/install -c -m 644 ksba.h '/usr/local/include'
 /usr/bin/mkdir -p '/usr/local/lib/pkgconfig'
 /usr/bin/install -c -m 644 ksba.pc '/usr/local/lib/pkgconfig'
make[3]: Leaving directory '/root/libksba-1.6.0/src'
make[2]: Leaving directory '/root/libksba-1.6.0/src'
make[1]: Leaving directory '/root/libksba-1.6.0/src'
Making install in tests
make[1]: Entering directory '/root/libksba-1.6.0/tests'
make  install-am
make[2]: Entering directory '/root/libksba-1.6.0/tests'
make[3]: Entering directory '/root/libksba-1.6.0/tests'
make[3]: Nothing to be done for 'install-exec-am'.
make[3]: Nothing to be done for 'install-data-am'.
make[3]: Leaving directory '/root/libksba-1.6.0/tests'
make[2]: Leaving directory '/root/libksba-1.6.0/tests'
make[1]: Leaving directory '/root/libksba-1.6.0/tests'
Making install in doc
make[1]: Entering directory '/root/libksba-1.6.0/doc'
make[2]: Entering directory '/root/libksba-1.6.0/doc'
make[2]: Nothing to be done for 'install-exec-am'.
 /usr/bin/mkdir -p '/usr/local/share/info'
 /usr/bin/install -c -m 644 ./ksba.info '/usr/local/share/info'
 install-info --info-dir='/usr/local/share/info' '/usr/local/share/info/ksba.info'
make[2]: Leaving directory '/root/libksba-1.6.0/doc'
make[1]: Leaving directory '/root/libksba-1.6.0/doc'
make[1]: Entering directory '/root/libksba-1.6.0'
make[2]: Entering directory '/root/libksba-1.6.0'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/libksba-1.6.0'
make[1]: Leaving directory '/root/libksba-1.6.0'
[root@rk libksba-1.6.0]#
```

4.1.5 Installing NPTH

1. Download the npth installation file

›_ Console

```
# wget https://www.gnupg.org/ftp/gcrypt/npth/npth-1.6.tar.bz2
```

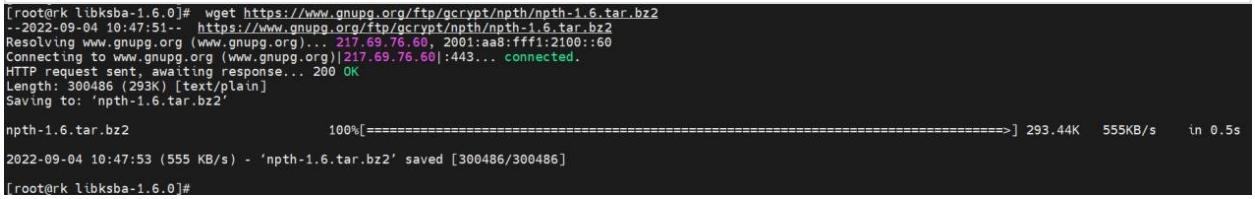

Figure 9 : Downloading npth

2. Extract the file

›_ Console

```
# tar -xjf npth-1.6.tar.bz2
```

3. Go to the directory where the file is extracted

›_ Console

```
# cd npth-1.6
```

4. Run the following command to compile and install

›_ Console

```
# ./configure
# make
# make install
```

```
[root@rk npth-1.6]# make install
Making install in src
make[1]: Entering directory '/root/libksba-1.6.0/npth-1.6/src'
make[2]: Entering directory '/root/libksba-1.6.0/npth-1.6/src'
 /usr/bin/mkdir -p '/usr/local/lib'
 /bin/sh ../libtool   --mode=install /usr/bin/install -c   libnpth.la '/usr/local/lib'
libtool: install: /usr/bin/install -c .libs/libnpth.so.0.1.2 /usr/local/lib/libnpth.so.0.1.2
libtool: install: (cd /usr/local/lib && { ln -s -f libnpth.so.0.1.2 libnpth.so.0 || { rm -f libnpth.so.0 && ln -s libnpth.so.0.1.2 libnpth.so.0; }; })
libtool: install: (cd /usr/local/lib && { ln -s -f libnpth.so.0.1.2 libnpth.so || { rm -f libnpth.so && ln -s libnpth.so.0.1.2 libnpth.so; }; })
libtool: install: /usr/bin/install -c .libs/libnpth.lai /usr/local/lib/libnpth.la
libtool: finish: PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin:/sbin" ldconfig -n /usr/local/lib
----------------------------------------------------------------------
Libraries have been installed in:
   /usr/local/lib

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,-rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to `/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------
 /usr/bin/mkdir -p '/usr/local/include'
 /usr/bin/install -c -m 644 npth.h '/usr/local/include'
make[2]: Leaving directory '/root/libksba-1.6.0/npth-1.6/src'
make[1]: Leaving directory '/root/libksba-1.6.0/npth-1.6/src'
Making install in tests
make[1]: Entering directory '/root/libksba-1.6.0/npth-1.6/tests'
make[2]: Entering directory '/root/libksba-1.6.0/npth-1.6/tests'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/libksba-1.6.0/npth-1.6/tests'
make[1]: Leaving directory '/root/libksba-1.6.0/npth-1.6/tests'
make[1]: Entering directory '/root/libksba-1.6.0/npth-1.6'
make[2]: Entering directory '/root/libksba-1.6.0/npth-1.6'
 /usr/bin/mkdir -p '/usr/local/bin'
 /usr/bin/install -c npth-config '/usr/local/bin'
 /usr/bin/mkdir -p '/usr/local/share/aclocal'
 /usr/bin/install -c -m 644 npth.m4 '/usr/local/share/aclocal'
make[2]: Leaving directory '/root/libksba-1.6.0/npth-1.6'
make[1]: Leaving directory '/root/libksba-1.6.0/npth-1.6'
[root@rk npth-1.6]#
```

Figure 10 : Installing npth

4.1.6 Installing Pinentry

1. Download the pinentry installation file

›_ Console

```
# wget https://www.gnupg.org/ftp/gcrypt/pinentry/pinentry-1.2.0.tar.bz2
```

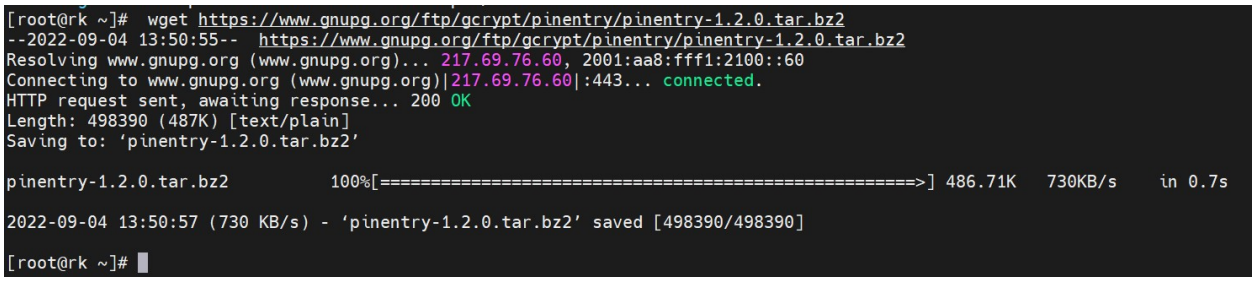

Figure 11 : Downloading pinentry

2. Extract the file

›_ Console

```
# tar -xjf pinentry-1.2.0.tar.bz2
```

3. Go to the directory where the file is extracted

›_ Console

```
# cd pinentry-1.2.0
```

4. Run the following command to compile and install

›_ Console

```
# ./configure --enable-pinentry-tty
# make
# make install
```

```
[root@rk pinentry-1.2.0]# make install
Making install in m4
make[1]: Entering directory '/root/pinentry-1.2.0/m4'
make[2]: Entering directory '/root/pinentry-1.2.0/m4'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/pinentry-1.2.0/m4'
make[1]: Leaving directory '/root/pinentry-1.2.0/m4'
Making install in secmem
make[1]: Entering directory '/root/pinentry-1.2.0/secmem'
make[2]: Entering directory '/root/pinentry-1.2.0/secmem'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/pinentry-1.2.0/secmem'
make[1]: Leaving directory '/root/pinentry-1.2.0/secmem'
Making install in pinentry
make[1]: Entering directory '/root/pinentry-1.2.0/pinentry'
make[2]: Entering directory '/root/pinentry-1.2.0/pinentry'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/pinentry-1.2.0/pinentry'
make[1]: Leaving directory '/root/pinentry-1.2.0/pinentry'
Making install in tty
make[1]: Entering directory '/root/pinentry-1.2.0/tty'
make[2]: Entering directory '/root/pinentry-1.2.0/tty'
 /usr/bin/mkdir -p '/usr/local/bin'
  /usr/bin/install -c pinentry-tty '/usr/local/bin'
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/pinentry-1.2.0/tty'
make[1]: Leaving directory '/root/pinentry-1.2.0/tty'
Making install in doc
make[1]: Entering directory '/root/pinentry-1.2.0/doc'
make[2]: Entering directory '/root/pinentry-1.2.0/doc'
make[2]: Nothing to be done for 'install-exec-am'.
 /usr/bin/mkdir -p '/usr/local/share/info'
 /usr/bin/install -c -m 644 ./pinentry.info '/usr/local/share/info'
 install-info --info-dir='/usr/local/share/info' '/usr/local/share/info/pinentry.info'
make[2]: Leaving directory '/root/pinentry-1.2.0/doc'
make[1]: Leaving directory '/root/pinentry-1.2.0/doc'
make[1]: Entering directory '/root/pinentry-1.2.0'
make[2]: Entering directory '/root/pinentry-1.2.0'
(cd /usr/local/bin; \
rm -f pinentry; \
ln -s pinentry-tty pinentry)
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/pinentry-1.2.0'
make[1]: Leaving directory '/root/pinentry-1.2.0'
[root@rk pinentry-1.2.0]#
```

Figure 12 : Installing pinentry

4.1.7 Installing GnuPG

1. Download the gnupg installation file

›_ Console

```
# wget https://www.gnupg.org/ftp/gcrypt/gnupg/gnupg-2.3.7.tar.bz2
```

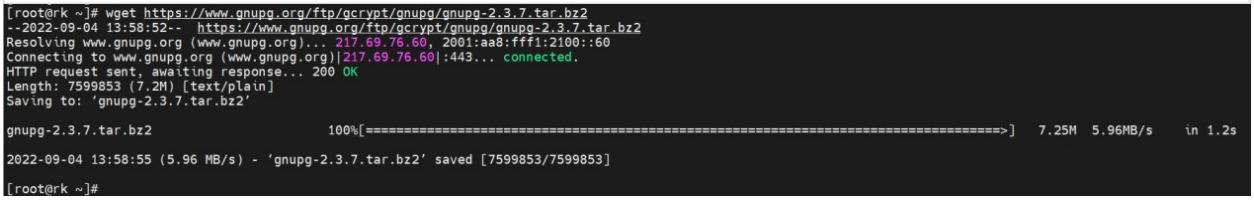

Figure 13 : Downloading gnupg

2. Extract the file

›_ Console

```
# tar -xjf gnupg-2.3.7.tar.bz2
```

3. Go to the directory where the file is extracted

›_ Console

```
# cd gnupg-2.3.7
```

4. Run the following command to compile and install

›_ Console

```
# ./configure
# make
# make install
```

```
[root@rk gnupg-2.3.7]# make install
Making install in m4
make[1]: Entering directory '/root/gnupg-2.3.7/m4'
make[2]: Entering directory '/root/gnupg-2.3.7/m4'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/gnupg-2.3.7/m4'
make[1]: Leaving directory '/root/gnupg-2.3.7/m4'
Making install in common
make[1]: Entering directory '/root/gnupg-2.3.7/common'
make  install-am
make[2]: Entering directory '/root/gnupg-2.3.7/common'
make[3]: Entering directory '/root/gnupg-2.3.7/common'
make[3]: Nothing to be done for 'install-exec-am'.
make[3]: Nothing to be done for 'install-data-am'.
make[3]: Leaving directory '/root/gnupg-2.3.7/common'
make[2]: Leaving directory '/root/gnupg-2.3.7/common'
make[1]: Leaving directory '/root/gnupg-2.3.7/common'
Making install in regexp
make[1]: Entering directory '/root/gnupg-2.3.7/regexp'
make  install-am
make[2]: Entering directory '/root/gnupg-2.3.7/regexp'
make[3]: Entering directory '/root/gnupg-2.3.7/regexp'
make[3]: Nothing to be done for 'install-exec-am'.
make[3]: Nothing to be done for 'install-data-am'.
make[3]: Leaving directory '/root/gnupg-2.3.7/regexp'
make[2]: Leaving directory '/root/gnupg-2.3.7/regexp'
make[1]: Leaving directory '/root/gnupg-2.3.7/regexp'
Making install in kbx
make[1]: Entering directory '/root/gnupg-2.3.7/kbx'
make[2]: Entering directory '/root/gnupg-2.3.7/kbx'
 /usr/bin/mkdir -p '/usr/local/bin'
  /usr/bin/install -c kbxutil '/usr/local/bin'
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/root/gnupg-2.3.7/kbx'
make[1]: Leaving directory '/root/gnupg-2.3.7/kbx'
Making install in g10
make[1]: Entering directory '/root/gnupg-2.3.7/g10'
make[2]: Entering directory '/root/gnupg-2.3.7/g10'
make  install-exec-hook
make[3]: Entering directory '/root/gnupg-2.3.7/g10'
running install-exec-hook
 /usr/bin/mkdir -p '/usr/local/bin'
 /usr/bin/install -c                    gpg '/usr/local/bin/gpg'
 /usr/bin/install -c                    gpgv '/usr/local/bin/gpgv'
make[3]: Leaving directory '/root/gnupg-2.3.7/g10'
/bin/sh ../build-aux/mkinstalldirs /usr/local/share/gnupg
mkdir -p -- /usr/local/share/gnupg
/usr/bin/install -c -m 644 ./distsigkey.gpg \
                    /usr/local/share/gnupg/distsigkey.gpg
make[2]: Leaving directory '/root/gnupg-2.3.7/g10'
make[1]: Leaving directory '/root/gnupg-2.3.7/g10'
Making install in sm
make[1]: Entering directory '/root/gnupg-2.3.7/sm'
```

Figure 14 : Installing gpg

```
make[3]: Entering directory '/root/gnupg-2.3.7/tests/gpgme'
make[3]: Nothing to be done for 'install-exec-am'.
make[3]: Nothing to be done for 'install-data-am'.
make[3]: Leaving directory '/root/gnupg-2.3.7/tests/gpgme'
make[2]: Leaving directory '/root/gnupg-2.3.7/tests/gpgme'
Making install in pkits
make[2]: Entering directory '/root/gnupg-2.3.7/tests/pkits'
make[3]: Entering directory '/root/gnupg-2.3.7/tests/pkits'
make[3]: Nothing to be done for 'install-exec-am'.
make[3]: Nothing to be done for 'install-data-am'.
make[3]: Leaving directory '/root/gnupg-2.3.7/tests/pkits'
make[2]: Leaving directory '/root/gnupg-2.3.7/tests/pkits'
Making install in .
make[2]: Entering directory '/root/gnupg-2.3.7/tests'
make[3]: Entering directory '/root/gnupg-2.3.7/tests'
make[3]: Nothing to be done for 'install-exec-am'.
make[3]: Nothing to be done for 'install-data-am'.
make[3]: Leaving directory '/root/gnupg-2.3.7/tests'
make[2]: Leaving directory '/root/gnupg-2.3.7/tests'
make[1]: Leaving directory '/root/gnupg-2.3.7/tests'
make[1]: Entering directory '/root/gnupg-2.3.7'
(set -e; cd bin; \
 for i in gpg gpgv; \
     do ln -sf ../g10/$i .; done; \
 for i in gpgsm; \
     do ln -sf ../sm/$i .; done; \
 for i in gpg-agent; \
     do ln -sf ../agent/$i .; done; \
 for i in dirmngr; \
     do ln -sf ../dirmngr/$i .; done; \
 for i in gpgconf gpg-connect-agent gpgtar gpg-card; \
     do ln -sf ../tools/$i .; done; \
 cd ../libexec ; \
 for i in keyboxd; \
     do ln -sf ../kbx/$i .; done; \
 for i in scdaemon; \
     do ln -sf ../scd/$i .; done; \
 for i in gpg-preset-passphrase; \
     do ln -sf ../agent/$i .; done; \
 echo "created links to binaries" )
created links to binaries
make[2]: Entering directory '/root/gnupg-2.3.7'
make[2]: Nothing to be done for 'install-exec-am'.
 /usr/bin/mkdir -p '/usr/local/share/doc/gnupg'
 /usr/bin/install -c -m 644 README '/usr/local/share/doc/gnupg'
make  install-data-hook
make[3]: Entering directory '/root/gnupg-2.3.7'
make[3]: Nothing to be done for 'install-data-hook'.
make[3]: Leaving directory '/root/gnupg-2.3.7'
make[2]: Leaving directory '/root/gnupg-2.3.7'
make[1]: Leaving directory '/root/gnupg-2.3.7'
[root@rk gnupg-2.3.7]#
```

5. Export PATH and LD_LIBRARY_PATH variable

›_ Console

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

6. Verify that the GnuPG has been installed successfully.

›_ Console

```
# gpg --version
```

```
[root@rk gnupg-2.3.7]# gpg --version
gpg (GnuPG) 2.3.7
libgcrypt 1.10.1
Copyright (C) 2021 Free Software Foundation, Inc.
License GNU GPL-3.0-or-later <https://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: /root/.gnupg
Supported algorithms:
Pubkey: RSA, ELG, DSA, ECDH, ECDSA, EDDSA
Cipher: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH,
        CAMELLIA128, CAMELLIA192, CAMELLIA256
AEAD: EAX, OCB
Hash: SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Compression: Uncompressed
[root@rk gnupg-2.3.7]#
```

Figure 15 : Verifying gpg version

4.2 Installing GnuPG-PKCS11-SCD

1. (Optional) Install EPEL repository if it doesn't exit

›_ Console

```
# dnf install https://dl.fedoraproject.org/pub/epel/epel-release-
latest8.noarch.rpm
```

2. Install pkcs11-helper-devel package

›_ Console

```
# dnf install pkcs11-helper-devel
```

3. Download the gnupg-pkcs11-scd installation file

›_ Console

```
# wget https://github.com/alonbl/gnupg-pkcs11-scd/releases/download/gnupgpkcs11-
scd-0.10.0/gnupg-pkcs11-scd-0.10.0.tar.bz2
```

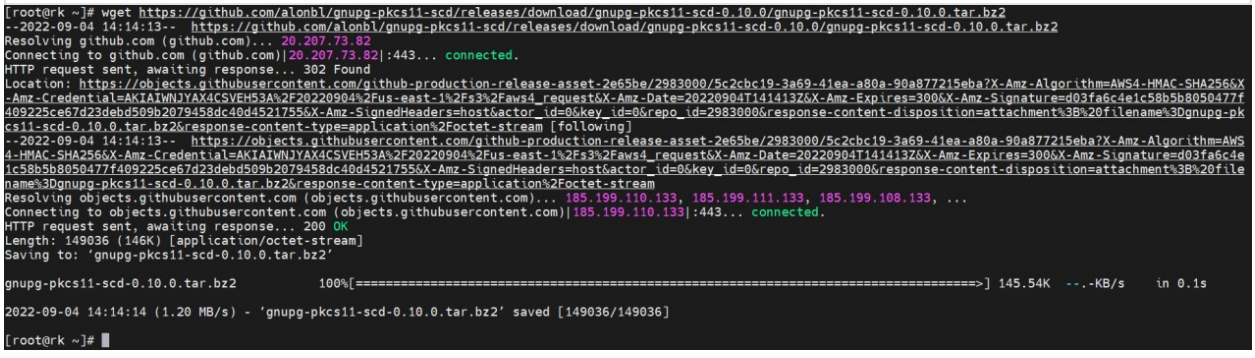

Figure 16 : Downloading gnupg-pkcs11-scd

4. Extract the file

›_ Console

```
# tar -xjf /gnupg-pkcs11-scd-0.10.0.tar.bz2
```

5. Go to the directory where the file is extracted

›_ Console

```
# cd gnupg-pkcs11-scd-0.10.0
```

6. Run the following command to compile and install

›_ Console

```
# ./configure --with-libgpg-error-prefix=/usr/local --with-libassuanprefix=/usr/
local --with-libgcrypt-prefix=/usr/local
# make
# make install
```

```
[root@rk gnupg-pkcs11-scd-0.10.0]# make install
Making install in gnupg-pkcs11-scd
make[1]: Entering directory '/root/gnupg-pkcs11-scd-0.10.0/gnupg-pkcs11-scd'
make[2]: Entering directory '/root/gnupg-pkcs11-scd-0.10.0/gnupg-pkcs11-scd'
 /usr/bin/mkdir -p '/usr/local/bin'
  /usr/bin/install -c gnupg-pkcs11-scd '/usr/local/bin'
 /usr/bin/mkdir -p '/usr/local/share/doc/gnupg-pkcs11-scd'
 /usr/bin/install -c -m 644 gnupg-pkcs11-scd.conf.example '/usr/local/share/doc/gnupg-pkcs11-scd'
 /usr/bin/mkdir -p '/usr/local/share/man/man1'
 /usr/bin/install -c -m 644 gnupg-pkcs11-scd.1 '/usr/local/share/man/man1'
make[2]: Leaving directory '/root/gnupg-pkcs11-scd-0.10.0/gnupg-pkcs11-scd'
make[1]: Leaving directory '/root/gnupg-pkcs11-scd-0.10.0/gnupg-pkcs11-scd'
Making install in gnupg-pkcs11-scd-proxy
make[1]: Entering directory '/root/gnupg-pkcs11-scd-0.10.0/gnupg-pkcs11-scd-proxy'
make[2]: Entering directory '/root/gnupg-pkcs11-scd-0.10.0/gnupg-pkcs11-scd-proxy'
make[2]: Leaving directory '/root/gnupg-pkcs11-scd-0.10.0/gnupg-pkcs11-scd-proxy'
make[1]: Leaving directory '/root/gnupg-pkcs11-scd-0.10.0/gnupg-pkcs11-scd-proxy'
Making install in distro
make[1]: Entering directory '/root/gnupg-pkcs11-scd-0.10.0/distro'
Making install in debian
make[2]: Entering directory '/root/gnupg-pkcs11-scd-0.10.0/distro/debian'
make  install-am
make[3]: Entering directory '/root/gnupg-pkcs11-scd-0.10.0/distro/debian'
make[4]: Entering directory '/root/gnupg-pkcs11-scd-0.10.0/distro/debian'
make[4]: Nothing to be done for 'install-exec-am'.
make[4]: Nothing to be done for 'install-data-am'.
make[4]: Leaving directory '/root/gnupg-pkcs11-scd-0.10.0/distro/debian'
make[3]: Leaving directory '/root/gnupg-pkcs11-scd-0.10.0/distro/debian'
make[2]: Leaving directory '/root/gnupg-pkcs11-scd-0.10.0/distro/debian'
Making install in rpm
make[2]: Entering directory '/root/gnupg-pkcs11-scd-0.10.0/distro/rpm'
make[3]: Entering directory '/root/gnupg-pkcs11-scd-0.10.0/distro/rpm'
make[3]: Nothing to be done for 'install-exec-am'.
make[3]: Nothing to be done for 'install-data-am'.
make[3]: Leaving directory '/root/gnupg-pkcs11-scd-0.10.0/distro/rpm'
make[2]: Leaving directory '/root/gnupg-pkcs11-scd-0.10.0/distro/rpm'
make[2]: Entering directory '/root/gnupg-pkcs11-scd-0.10.0/distro'
make[3]: Entering directory '/root/gnupg-pkcs11-scd-0.10.0/distro'
make[3]: Nothing to be done for 'install-exec-am'.
make[3]: Nothing to be done for 'install-data-am'.
make[3]: Leaving directory '/root/gnupg-pkcs11-scd-0.10.0/distro'
make[2]: Leaving directory '/root/gnupg-pkcs11-scd-0.10.0/distro'
make[1]: Leaving directory '/root/gnupg-pkcs11-scd-0.10.0/distro'
make[1]: Entering directory '/root/gnupg-pkcs11-scd-0.10.0'
make[2]: Entering directory '/root/gnupg-pkcs11-scd-0.10.0'
make[2]: Nothing to be done for 'install-exec-am'.
 /usr/bin/mkdir -p '/usr/local/share/doc/gnupg-pkcs11-scd'
 /usr/bin/install -c -m 644 COPYING README '/usr/local/share/doc/gnupg-pkcs11-scd'
make[2]: Leaving directory '/root/gnupg-pkcs11-scd-0.10.0'
make[1]: Leaving directory '/root/gnupg-pkcs11-scd-0.10.0'
[root@rk gnupg-pkcs11-scd-0.10.0]#
```

Figure 17 : Installing gnupg-pkcs11-scd

After rebooting the server, export the path and library path every time.

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

```
export LD_LIBRARY_PATH=/usr/local/lib
```

4.3 PKCS#11 Configuration for CryptoServer

1. Create the directory `/etc/utimaco.` Locate the Utimaco PKCS#11 configuration file in your SecurityServer directory, `Linux/x86-64/Crypto_APIS/PKCS11_R3/sample`. Copy the Utimaco PKCS#11 configuration file `cs_pkcs11_R3.cfg` into /etc/utimaco directory

›_ Console

```
# mkdir /etc/utimaco
# cd <install directory>/Software/Linux/x86-64/Crypto_APIs/PKCS11_R3/sample # cp
cs_pkcs11_R3.cfg /etc/utimaco
# cd /etc/utimaco
```

2. Edit the `cs_pkcs11_R3.cfg` file and make the appropriate changes to the file

cs_pkcs11_R3.cfg

```
[Global]

# For unix:
Logpath = /tmp

# Loglevel (0 = NONE; 1 = ERROR; 2 = WARNING; 3 = INFO; 4 = TRACE)
Logging = 1

# Set the Device to connect with
[CryptoServer] # Device specifier  Device = <HSM_IP>
```

For more information regarding the commands and command parameters please check the Utimaco CryptoServer documentation. The device may be a CryptoServer (PCIe or LAN) device. The device line will follow one of these patterns, based on the HSM form-factor:

`Device = 288@<HSM IP address>` *Hardware (LAN) HSM*

OR

```
Device = /dev/cs2.0 Hardware (PCIe) HSM
```

To make your testing easier, it would be good to enable the PKCS#11 log file. That can be enabled by editing the Logging ***Loglevel****. Set the* LogPath *and Logging* Loglevel *to* 1*. For testing you may want to increase it to* 4*.*

The added **LogPath** points to a writable directory, not to a file.
If you encounter problems, check the log file named `cs_pkcs11_R3.log` in the **LogPath** defined directory. When you are done testing, you should change Logging to 1 or 2. This will limit the logging to only critical and important messages.

4.3.1 Create SO User and Initialize a Slot

You should initialize a slot with a custom label using p11tool2.

First using p11tool2 create, the SO or Security Officer and then using p11tool2 command initialize the Slot that you want to use, and the slot user as shown below.

›_ Console

```
# ./p11tool2 slot=<slot_no> Label=<token_label> Login=ADMIN,ADMIN.key
InitToken=<SO_PIN>
# ./p11tool2 slot=0 LoginSO=<SO_PIN> InitPin=<CryptoUser_PIN>
```

This CryptoUser PIN is the slot PIN that will be used in this guide for every crypto operations.

4.4 Configuring GnuPG to Use Utimaco HSM

1. Run the following command to automatically create directory structure for gnupg

›_ Console

```
# gpg --list-keys
```

```
[root@rk ~]# gpg --list-keys
gpg: directory '/root/.gnupg' created
gpg: keybox '/root/.gnupg/pubring.kbx' created
gpg: /root/.gnupg/trustdb.gpg: trustdb created
[root@rk ~]#
```

Figure 18 : Listing gpg keys

2. Copy the sample file from resource/gnupg-pkcs11-scd-0.10.0/gnupg-pkcs11-scd/gnupg-pkcs11-scd.conf.example to ~/.gnupg/gnupg-pkcs11-scd.conf

›_ Console

```
# cp gnupg-pkcs11-scd-0.10.0/gnupg-pkcs11-scd/gnupg-pkcs11-scd.conf.example /
root/.gnupg/gnupg-pkcs11-scd.conf
```

3. Open the file /root/.gnupg/gnupg-pkcs11-scd.conf and make the following changes

›_ Console

```
pin-cache 0

providers p1

provider-p1-library /opt/utimaco/lib/libcs_pkcs11_R3.so
```

4. Create a file /root/.gnupg/gpg-agent.conf and add the following content in it

›_ Console

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

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

4.4.1 Generating Key and Certificate for GnuPG

1. Generate the key pair using below p11tool2 command

›_ Console

```
# p11tool2 Slot=21 LoginUser=123456

PubKeyAttr=CKA_LABEL="GPGPublicKey",CKA_ID=0x45

PrvKeyAttr=CKA_LABEL="GPGPrivateKey",CKA_ID=0x45 GenerateKeyPair=RSA
```

Only RSA key is supported with GnuPG PKCS11 SCD.

2. Verify that the keys are generated

›_ Console

```
# p11tool2 Slot=21 LoginUser=123456 ListObjects
```

```
[root@rk ~]# p11tool2 slot=21 LoginUser=123456 ListObjects


CKO_PUBLIC_KEY:

+ 1.1
  CKA_KEY_TYPE                    = CKK_RSA
  CKA_LABEL                       = GPGPublicKey
  CKA_ID                          = 0x45 (E)


CKO_PRIVATE_KEY:

+ 2.1
  CKA_KEY_TYPE                    = CKK_RSA
  CKA_SENSITIVE                   = CK_TRUE
  CKA_EXTRACTABLE                 = CK_FALSE
  CKA_LABEL                       = GPGPrivateKey
  CKA_ID                          = 0x45 (E)
[root@rk ~]#
```

Figure 19 : Listing keys on HSM slot

3. Install opensc and openssl-pcs11

›_ Console

```
# dnf install opensc openssl-pkcs11
```

4. Open openssl shell and load the dynamic engine

›_ Console

```
# openssl

OpenSSL> engine  dynamic -pre SO_PATH:/usr/lib64/engines-1.1/pkcs11.so pre
ID:pkcs11 -pre LIST_ADD:1 -pre LOAD -pre MODULE_PATH:/opt/utimaco/lib/
libcs_pkcs11_R3.so
```

```
[root@rk ~]# openssl
OpenSSL>  engine  dynamic -pre SO_PATH:/usr/lib64/engines-1.1/pkcs11.so -pre ID:pkcs11 -pre LIST_ADD:1 -pre LOAD -pre MODULE_PATH:/opt/uti
maco/lib/libcs_pkcs11_R3.so
(dynamic) Dynamic engine loading support
[Success]: SO_PATH:/usr/lib64/engines-1.1/pkcs11.so
[Success]: ID:pkcs11
[Success]: LIST_ADD:1
[Success]: LOAD
[Success]: MODULE_PATH:/opt/utimaco/lib/libcs_pkcs11_R3.so
Loaded: (pkcs11) pkcs11 engine
```

Figure 20 : Loading dynamic engine in openssl

5. Run the following command to generate a self-signed certificate. Provide slot PIN when prompted.

›_ Console

```
OpenSSL> req -x509 -engine pkcs11 -keyform engine -new -key

"pkcs11:token=gpgslot;object=GPGPrivateKey" -sha256 -out gpgcert.pem -subj "/
CN=test.utimaco.com"
```

```
OpenSSL> req -x509 -engine pkcs11 -keyform engine -new -key "pkcs11:token=gpgslot;object=GPGPrivateKey" -sha256 -out gpgcert.pem -subj "/C
N=test.utimaco.com"
engine "pkcs11" set.
Enter PKCS#11 token PIN for gpgslot:
OpenSSL> exit
[root@rk ~]#
```

Figure 21 : Generating self-signed certificate

Here gpgslot is the token label and GPGPrivateKey is the key on the Utimaco HSM.

After this a certificate gpgcert.pem is generated.

Type exit to exit from openssl prompt

It is recommended to use CA signed certificate for production environment.

6. Convert the certificate from pem to der

›_ Console

```
# openssl x509 -outform der -in gpgcert.pem -out gpgcert.der
```

7. Import the certificate to Utimaco HSM

›_ Console

```
# pkcs11-tool --module /opt/utimaco/lib/libcs_pkcs11_R3.so -l --writeobject
gpgcert.der --type cert --token-label gpgslot --id 45 --label "GPGCert"
```

```
[root@rk ~]# pkcs11-tool --module /opt/utimaco/lib/libcs_pkcs11_R3.so -l --write-object gpgcert.der --type cert --token-label gpgslot --id 45 --label "GPGCert"
Logging in to "gpgslot".
Please enter User PIN:
Created certificate:
Certificate Object; type = X.509 cert
  label:      GPGCert
  subject:    DN: CN=test.utimaco.com
  ID:         45
[root@rk ~]#
```

Figure 22 : Importing certificate to Utimaco HSM

Here gpgslot is the token label, 45 is the CKA_ID of the private key on the HSM and gpgcert.der is the certificate name.

8. Verify that the certificate has been imported to Utimaco HSM

›_ Console

```
# p11tool2 slot=21 LoginUser=123456 ListObjects
```

```
[root@rk ~]# p11tool2 slot=21 LoginUser=123456 ListObjects


CKO_CERTIFICATE:

+ 1.1
  CKA_CERTIFICATE_TYPE          = CKC_X_509
  CKA_LABEL                     = GPGCert
  CKA_ID                        = 0x45 (E)
  CKA_SUBJECT                   =
                0x301B3119 30170603 5504030C 10746573 |0 1 0   U    tes|
                  742E7574 696D6163 6F2E636F 6D       |t.utimaco.com   |



CKO_PUBLIC_KEY:

+ 2.1
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_LABEL                     = GPGPublicKey
  CKA_ID                        = 0x45 (E)


CKO_PRIVATE_KEY:

+ 3.1
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_SENSITIVE                 = CK_TRUE
  CKA_EXTRACTABLE               = CK_FALSE
  CKA_LABEL                     = GPGPrivateKey
  CKA_ID                        = 0x45 (E)
[root@rk ~]#
```

Figure 23 : Listing keys and certificate on HSM slot

4.4.2 Adding certificate to Gnupg

1. Create master key based on existing key

›_ Console

```
# gpg --expert --full-generate-key
```

2. Select option (14) Existing key from card

```
[root@rk ~]# gpg --expert --full-generate-key
gpg (GnuPG) 2.3.7; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Please select what kind of key you want:
   (1) RSA and RSA
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
   (7) DSA (set your own capabilities)
   (8) RSA (set your own capabilities)
   (9) ECC (sign and encrypt) *default*
  (10) ECC (sign only)
  (11) ECC (set your own capabilities)
  (13) Existing key
  (14) Existing key from card
Your selection? 14
```

Figure 24 : GPG command to select existing key from HSM

This will list the serial number of the HSM slot and existing keys which has corresponding Certificate.

```
Serial number of the card: D2760001240111503131 7B9EABF61111
Available keys:
   (1) 2E29D3545DE6ED5B3604679EF2CE02781B856A44 pkcs11:model=CryptoServer;token=gpgslot;manufacturer=Utimaco%20IS%20GmbH;serial=CS711108_0021;id=E rsa2048
```

Figure 25 : List existing keys on HSM through gpg

3. Enter the number for the keys you want use

```
Your selection? 1
```

Figure 26 : Selecting key number

4. Enter Q then provide key expiry, real name, and email address. Provide slot PIN when prompted.

```
Possible actions for this RSA key:
Current allowed actions:

   (Q) Finished

Your selection? Q
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0) 0
Key does not expire at all
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

Real name: test@utimaco.com
Email address: test@utimaco.com
Comment:
You selected this USER-ID:
    "test@utimaco.com <test@utimaco.com>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
Please enter the PIN (pkcs11:model=CryptoServer;token=gpgslot;manufacturer=Utimaco IS GmbH;serial=CS711108_0021) to unlock the card
PIN:
Please enter the PIN (pkcs11:model=CryptoServer;token=gpgslot;manufacturer=Utimaco IS GmbH;serial=CS711108_0021) to unlock the card
PIN:
gpg: revocation certificate stored as '/root/.gnupg/openpgp-revocs.d/B88C00E416F5E1A19AC3C5FDC9E4FA546398322E.rev'
public and secret key created and signed.

pub   rsa2048 2022-09-04 [SCEA]
      B88C00E416F5E1A19AC3C5FDC9E4FA546398322E
uid                      test@utimaco.com <test@utimaco.com>

[root@rk ~]#
```

Figure 27 : Finishing gpg Key Generate Command

5. List the keys

›_ Console

```
# gpg --list-keys
```

```
[root@rk ~]# gpg --list-keys
/root/.gnupg/pubring.kbx
------------------------
pub   rsa2048 2022-09-04 [SCEA]
      B88C00E416F5E1A19AC3C5FDC9E4FA546398322E
uid           [ultimate] test@utimaco.com <test@utimaco.com>

[root@rk ~]#
```

Figure 28 : gpg list keys

4.4.3 Signing, Encryption, Decryption and Verification with GnuPG

1. Create a sample message file

›_ Console

```
# echo "Welcome to Utimaco" > message.txt
```

2. Sign the file using the key name.

›_ Console

```
# gpg --output message.txt.signed --sign --default-key test@utimaco.com
message.txt
```

Provide slot PIN when prompted. This will generate a signed file message.txt.signed

```
[root@rk ~]# gpg --output message.txt.signed --sign --default-key test@utimaco.com  message.txt
gpg: using "test@utimaco.com" as default secret key for signing
Please enter the PIN (pkcs11:model=CryptoServer;token=gpgslot;manufacturer=Utimaco IS GmbH;serial=CS711108_0021) to unlock the card
PIN:
[root@rk ~]#
```

Figure 29 : Signing the file

3. Verify the file message.txt.signed

›_ Console

```
# gpg --verify message.txt.signed
```

```
[root@rk ~]# gpg --verify message.txt.signed
gpg: Signature made Sun 04 Sep 2022 06:45:27 PM UTC
gpg:                using RSA key B88C00E416F5E1A19AC3C5FDC9E4FA546398322E
gpg:                issuer "test@utimaco.com"
gpg: Good signature from "test@utimaco.com <test@utimaco.com>" [ultimate]
[root@rk ~]#
[root@rk ~]#
```

Figure 30 : Verifying the signed file

4. Encrypt the file. Provide recipient user ID when prompted

›_ Console

```
# gpg --output message.txt.enc --encrypt  message.txt
```

```
[root@rk ~]# gpg --output message.txt.enc --encrypt  message.txt
You did not specify a user ID. (you may use "-r")

Current recipients:

Enter the user ID.  End with an empty line: test@utimaco.com

Current recipients:
rsa2048/C9E4FA546398322E 2022-09-04 "test@utimaco.com <test@utimaco.com>"

Enter the user ID.  End with an empty line:
[root@rk ~]#
```

Figure 31 : Encrypting the file

This will generate an encrypted file message.txt.enc

5. Decrypt the encrypted file

›_ Console

```
# gpg --output message.txt.dec --decrypt  message.txt.enc
```

Provide slot PIN when prompted. This will generate a decrypted file message.txt.dec

```
[root@rk ~]# gpg --output message.txt.dec --decrypt  message.txt.enc
gpg: encrypted with rsa2048 key, ID C9E4FA546398322E, created 2022-09-04
      "test@utimaco.com <test@utimaco.com>"
Please enter the PIN (pkcs11:model=CryptoServer;token=gpgslot;manufacturer=Utimaco IS GmbH;serial=CS711108_0021) to unlock the card
PIN:
[root@rk ~]#
```

Figure 32 : Decrypting the file

6. Verify the content of original file

›_ Console

```
# cat message.txt.dec
```

Figure 33 : Original content of the file

4.4.4 RPM Signing and Verification with GnuPG

4.4.4.1 RPM Signing

1. Create file `/root/.rpmmacros` in user's home directory and add the following content in it

›_ Console

```
%_signature gpg
%_gpg_path /root/.gnupg
%_gpg_name test@utimaco.com
%_gpg /usr/local/bin/gpg
%_gpg_sign_cmd %{_gpg} gpg --force-v3-sigs --batch --verbose --no-armor -no-
secmem-warning -u "%{_gpg_name}" -sbo %{_signature_filename} --digestalgo
filename}
```

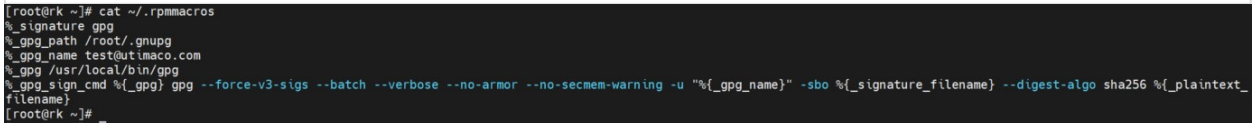

Figure 34 : Content of rpmmacros file

Here:

- `/root/.gnupg` is the base directory for gnupg
- test@utimaco.com is the key name
- `/usr/local/bin/gpg` is the path for gpg
- `%{_gpg} gpg --force-v3-sigs --batch --verbose --no-armor --no-secmem-warning -u "%{_gpg_name}" -sbo %{_signature_filename} --digest-algo filename}` is the gpg command that will be used for signing rpm

2. Sign the file using below command

›_ Console

```
# rpm --addsign <rpm_file>
```

Provide the slot PIN when prompted.

3. If you want to sign it again then run the below command. Provide the slot PIN when prompted.

›_ Console

```
# rpm --resign <rpm_file>
```

```
[root@rk ~]# rpm --resign shim-15-8.el7.src.rpm
shim-15-8.el7.src.rpm:
Please enter the PIN (pkcs11:model=CryptoServer;token=gpgslot;manufacturer=Utimaco IS GmbH;serial=CS711108_0021) to unlock the card
PIN:
Please enter the PIN (pkcs11:model=CryptoServer;token=gpgslot;manufacturer=Utimaco IS GmbH;serial=CS711108_0021) to unlock the card
PIN:
[root@rk ~]#
```

Figure 35 : Resigning the rpm file

4.4.4.2 Signed RPM Verification

1. Export the public key to a file

›_ Console

```
# gpg --export --armor test@utimaco.com > gpgpub.key
```

```
[root@rk ~]# gpg --export --armor test@utimaco.com > gpgpub.key
[root@rk ~]#
[root@rk ~]#
[root@rk ~]# cat gpgpub.key
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQENBGMfXCkBCACVGh+swd3jPotiDfFqRblaNiCrqm6yQKGkurTjrSjhkO7SquQP
lxwqnO7NqgXuCfank8NxtL9z1dLqx7w7YBI3pB1hZHSBTSubyo34ekrfX6IMpZek
beOvK4eRbW2rXz0+Kl4oLKR2iaanyZM+kAZdHD0WL8POhzeokl5M8IlXF8WwyD5V
fePviAEhxu+h0ADrSIswZ2uw2JsGfqRkYFEVvxSnP3xtuwc6qVJY+/IWPz2f12uE
3Lcl1eggVSokNjwytsuFE9EV/oy4hPgiHhL+7miahLAYsLbnsdxAHWdkECJxF7to
ML+zgvRTzC8emGZGbgsW6vUCTfUl7s4xQ3FvABEBAAG0I3Rlc3RAdXRpbWFjby5j
b20gPHRlc3RAdXRpbWFjby5jb20+iQFPBBMBCAA5FiEE8UhJQyFSmtlyUkp7uEOU
fabkfGUFAmMfXCkCGy8FCwkIBwICIgIGFQoJCAsCAhYAAh4HAheAAAoJELhDlH2m
5HxlGbwH/0Z7TUdLiDUWEZrBOBCNOf+imizuAn8IB+TAL8kUcDrfOIUc1X6Ee1T3
zoyRwmPM3nATwKofrOMbUR+tE/OEKF//3cfLLGkDuFqK52IOKNUafn1G4X20bCZs
eCNBm4lkHI0rodSdX18jZ1JpqSLTprJTuNZRHLtpt6rE4ksbtvrKeB07a1nFvIQ1
b5AmJAq1Lk7p81jKEQ7dDYYHUoOjF+M9htK84mfqDQQmjCnWAJ6UZBZrLsbVuvdC
fIsnwWKhMDJdi7MoqhcCYlhezahRQtKQlPC7JyMycYvwLuSvHtfrXwmRb6W+k958
9EwbX/2KN+vgFpPlnrp5TT3jA68v3Fg=
=wLyJ
-----END PGP PUBLIC KEY BLOCK-----
[root@rk ~]#
```

Figure 36 : Exporting the public key to a file

2. Import the public key

›_ Console

```
# rpm --import gpgpub.key
```

```
[root@rk ~]# rpm --import gpgpub.key
[root@rk ~]#
```

Figure 37 : Importing the public key to rpm db

3. Verify the signature of the signed rpm

›_ Console

```
# rpm --checksig <signed_rpm_file>
```

```
[root@rk ~]# rpm --checksig shim-15-8.el7.src.rpm
shim-15-8.el7.src.rpm: digests signatures OK
[root@rk ~]#
```

Figure 38 : Verifying the signed rpm file

4. Verify the signing information

›_ Console

```
# rpm -qpi shim-15-8.el7.src.rpm
```

Signature field contains the signing information

```
[root@rk ~]# rpm -qpi shim-15-8.el7.src.rpm
Name        : shim
Version     : 15
Release     : 8.el7
Architecture: x86_64
Install Date: (not installed)
Group       : Unspecified
Size        : 1291770
License     : BSD
Signature   : RSA/SHA256, Tue 20 Sep 2022 11:03:11 AM UTC, Key ID b843947da6e47c65
Source RPM  : (none)
Build Date  : Thu 30 Jul 2020 08:36:31 PM UTC
Build Host  : kbuilder.bsys.centos.org
Relocations : (not relocatable)
Packager    : CentOS BuildSystem <http://bugs.centos.org>
Vendor      : CentOS
URL         : http://www.codon.org.uk/~mjg59/shim/
Summary     : First-stage UEFI bootloader
Description :
Initial UEFI bootloader that handles chaining to a trusted full bootloader
under secure boot environments.
[root@rk ~]#
```

Figure 39 : Verifying the signed rpm file signature information

This completes the Integration of GnuPG with Utimaco HSM.

5 Troubleshooting

Error	Diagnosis
LoginUser= failed: 05.12.2021 23:45:45 src/p11adm_R2.c[429] p11_login: C_Login [type=1] returned Error 0x00000102 (CKR_USER_PIN_NOT_INITIALIZED)	PKCS#11 Slot is not initialized.
The CryptoServer PKCS#11 Library R3 is not initialized. Error CKR_CRYPTOKI_NOT_INITIALIZED occurred	PKCS#11 Slot is not initialized.

Table 6: List of Error and its Diagnosis

6 Further Information

This document forms a part of the information and support which is provided by the Utimaco IS GmbH. Additional documentation can be found on the product CD in the Documentation directory.

All CryptoServer product documentation is also available at the Utimaco IS GmbH website:

http://hsm.utimaco.com

7 References

Reference	Title/Company	Document No.
[CSADMIN]	CryptoServer – csadm Manual/Utimaco IS GmbH	2009-0003
[CSTrSh]	CryptoServer Troubleshooting/Utimaco IS GmbH	M011-0008-en
[CSADMIN2]	CryptoServer_csadm_Manual_Systemadministrators.pdf	2009-0003
[CSP11Tool2]	CryptoServer_p11tool2_Manual.pdf	2012-0004
[CSPKCSM]	CryptoServer - PKCS#11 P11CAT Manual	M013-0001-en
[CSLAN5]	CryptoServerLAN_Manual_Systemadministrators.pdf	2018-0004