Axway, Inc.

Luna_client

A program for testing connections and logins from a Linux system to a Luna Hardware Security Module (HSM)

Bill Shine and Bruce Tachoir
10-24-2018

Introduction

The program luna_client contained in the "luna_client.tar" file is a program for testing the connection between a Validation Authority (VA) Responder server running Linux and a Hardware Security Module (HSM). When the correct command line parameters are given to the luna_client program, luna_client actually connects to the HSM and does a login to it. This provides the analyst with a verification that the HSM is running, the HSM password is correct and the HSM is configured to communicate with the VA server.

The program luna_client is contained in the tar file build_luna_client.tar. Also included in this file is the source code, include files and library necessary to build your own copy of luna client. Additionally, you will need to have several standard Linux commands, such as "make" and the GNU GCC compiler suite. Libraries specific to Luna HSMs are included in the tar file.

Luna_client: build and install

What follows is a listing of all the files contained in build_luna_client. Notice that the program itself, namely luna_client, is available in the tar file. This means if you do not have all the required prerequisites for building linu_client or just don't want to go through all the steps to build it, you can install this prebuilt version by un-taring the tar file, move into the directory build_luna_client and enter 'make install'. The executable will then be copied to the /usr/sbin directory.

```
> tar tf build_luna_client.tar
```

```
build_luna_client/
build_luna_client/include/
build_luna_client/include/RSA/
build_luna_client/include/RSA/pkcs11.h
build_luna_client/include/RSA/pkcs11t.h
build_luna_client/include/RSA/pkcs11f.h
build_luna_client/include/cryptoki_v2.h
build_luna_client/include/sfnt_extensions.h
build_luna_client/include/cryptoki.h
build_luna_client/lib/
build_luna_client/lib/libcknfast.lib
build_luna_client/luna_client.c
build_luna_client/Makefile
build_luna_client/luna_client
```

Next, we look at the makefile used to build and install the luna_client command. This makefile is of course the one contained in the tar file.

```
> cat build_luna_client/Makefile
```

```
CC=gcc

###CFLAGS=-DUNIX -DOS_UNIX -I/usr/safenet/lunaclient
```

```
CFLAGS=-DUNIX -DOS_UNIX -I./include
###LINKFLAGS= -L/usr/safenet/lunaclient/lib -lCryptoki2_64 -ldl
LINKFLAGS= -L./lib -lCryptoki2_64 -ldl

SRCS=luna_client.c
OBJS=luna_client.o

### Note that in the preprocessor defs for the program we define
### "OS_LINUX".  This pulls in the linux specific code in cryptoki_v2.h
### and excludes the  windows code...

luna_client : clean $(OBJS)
        $(CC) -o luna_client $(OBJS) $(LINKFLAGS) $(LIBS)

luna_client.o : luna_client.c
        $(CC) $(CFLAGS) -c luna_client.c

clean :
        rm -f luna_client $(OBJS)

install :
       cp luna_client /usr/sbin/
```

```
 > cd build_luna_client
 > touch luna_client.c
 > make
rm -f luna_client luna_client.o
gcc -DUNIX -DOS_UNIX -I./include -c luna_client.c
gcc -o luna_client luna_client.o -L./lib -lCryptoki2_64 -ldl
 >

 > which luna_client
/usr/sbin/luna_client
```

Running the Luna_client program:

Luna_client with no command line arguments produces the following usage message:

```
 > luna_client

         Usage:

        luna_client SECRET_KEY label_buffer_size(256) id_buffer_size(20)
```

Next, we show the correct way to run luna_client where the Luna password and VA server password (should be the same) is substituted for SECRET_KEY.

```
 > luna_client SECRET_KEY 140 20
```

```
        Args: userPin= SECRET_KEY
        label_bufSize=140(normal value: 256)
        id_bufSize=20 (normal value: 20)

 Slot count: 1

 start session: rv is 0 session handle is 1
 Tried logging in using SECRET_KEY rv = 0
 C_findObjectsInit return value is 0 (0==success)
 C_FindObjects returns 1 objects, return value 0
 Calling getAttributeValue
 GetAttributeValue returned 0

 Found a key:
        Key label: 0002 RSAPrv 10/04/18 15:44:13
        Key ID: 6e:43:cf:89:b3:6a:3c:b5:64:c6:5b:20:70:3a:f4:1d:ea:85:05:01:
 End session, rv = 0
```

Next, we see that if either label_buffer_size or id_buffer_sizeis too small, the program will terminate with a 336 return error code.

```
 > luna_client SECRET_KEY 14 20

        Args: userPin= SECRET_KEY
        label_bufSize=14(normal value: 256)
        id_bufSize=20 (normal value: 20)

 Slot count: 1

 start session: rv is 0 session handle is 1
 Tried logging in using SECRET_KEY rv = 0
 C_findObjectsInit return value is 0 (0==success)
 C_FindObjects returns 1 objects, return value 0
 Calling getAttributeValue
 GetAttributeValue returned 336

Error at get attribute value: 336


 > luna_client SECRET_KEY 140 19

        Args: userPin= SECRET_KEY
        label_bufSize=140(normal value: 256)
        id_bufSize=19 (normal value: 20)

 Slot count: 1

 start session: rv is 0 session handle is 1
 Tried logging in using SECRET_KEY rv = 0
 C_findObjectsInit return value is 0 (0==success)
```

```
C_FindObjects returns 1 objects, return value 0
Calling getAttributeValue
GetAttributeValue returned 336
```

Error at get attribute value: 336