securosys

SWISS SECURITY TECHNOLOGIES
FOR COMMUNICATIONS SYSTEMS

Primus PKCS#11 OpenSSL/Apache Integration

Contained OpenSSL1.0.2u and Apache 2.4.46

For Primus HSM or Clouds HSM

Valid for PKCS#11 Provider v1.7.x/v1.8.x

Primus HSM PKCS#11 Integration Guide for
OpenSSL1.0.2u and Apache v2.4.46
contained in the provider delivery

Securosys SA, Max-Höggerstrasse 2
CH-8048 Zürich, Switzerland
Tel. +41 44 552 31 00 • www.securosys.com
info@securosys.com

Document Information and Revision Control

Version	Date	Author	Description, Changes
1	---	PM	Initial document (extracted from PKCS#11 User Guide)
2	08.06.2021	PM	Updated for PKCS#11 Provider v1.7 using OpenSSL v1.0.2u and Apache v2.4.46
3	08.09.2023	PM	Minor corrections and updates.

File: PrimusAPI_P11-OpenSslApacheProv_AN-E03.docx

Table of Contents

1 Introduction

This application note describes the steps how to configure Primus HSM PKCS#11 Provider v1.7.x/v1.8.x on Unix to use the included OpenSSL v1.0.2u and Apache v2.4.46, delivered with the provider package.

Please note:

- OpenSSL v1.0.2u is the latest public available update from Dec 2019 (EoL). Further updates are only available with OpenSSL Premium Level Support.
- Securosys will not maintain further the provided OpenSSL and Apache distributions.
- Primus PKCS#11 Provider v2.x does no longer include OpenSSL and Apache distributions.

We recommend using the latest OpenSSL (with EC support) and latest Apache of your Linux distribution, based on the p11-kit (PKCS#11 module handler). Therefore, consult the application note PrimusHSM_P11-Kit-Tool_AN-Enn.pdf, downloadable from the Securosys Support Portal.

1.1 Notations and Symbols

Notes are used to notify you about useful or important information. The following element is used

NOTE: contains helpful or important information

Cautions are used to notify you about important information that may help prevent unexpected results or data loss. The following element is used

CAUTION: contains important information that may help prevent unexpected results or data loss

Warnings are used to notify you of potential data loss. The following element is used

WARNING: be careful and obey all instructions. You might do something that could result in data loss

Command lines and their output are boxed using `Consolas` font. The command is printed **`bold`**, values to adapt by the user are marked in **`bold blue`**, and comments are marked in *`italic brown`*., e.g.

```
ppin -u -e HSM_USERNAME

********************
Primus Permanent PIN
********************
Provide setup password for 'HSM_USERNAME': <enter User Setup Password, no echo>
Provide PKCS11 password for 'HSM_USERNAME': <enter PKCS#11 PIN, no echo>
…
```

1.2 References and More Information

- Primus HSM, PKCS#11 Provider, User Guide PrimusAPI_PKCS11-UserGuide_UG-Enn.pdf
- Application Note: Primus PKCS#11 Integration into p11-kit to use latest OpenSSL and Apache/Nginx, PrimusHSM_P11-Kit-Tool_AN-Enn.pdf
- OpenSSL web site https://www.openssl.org/
- Apache web-server web site https://www.apache.org/ and https://httpd.apache.org/

Application Notes are downloadable from the Securosys Support Portal.

1.3 Glossary

Acronym	Definition
HSM	Hardware Security Module
PKCS#11	Public-Key Cryptography Standards – API to cryptographic tokens
ECC	Elliptic-Curve Cryptography
AN	Application Note
UG	User Guide

1.4 Requirements

Supported platforms: see PKCS#11 Provider User Guide.

1.5 Software and Hardware Used for this Setup

- Primus HSM with firmware 2.9.2
- Virtualization Software (VMware Desktop 15.5.6)
- CentOS Linux release 7.9.2009
- Primus HSM PKCS#11 Provider, PrimusAPI_PKCS11-1.7.32-rhel7-x86_64.tar.gz
- opensc-0.19.0

2 Installation Procedure

This procedure provides a straightforward integration process, which has been tested. Please take notice that there may be other ways to achieve it. This guide assumes that you are familiar with the Primus HSM, Linux operating system installation and configuration procedures, OpenSSL configuration, and Apache configuration and does not cover every step of the hardware and software setup process. The application note does not cover firewall nor SELinux configuration topics.

The following installation and configuration description are based on CentOS 7 Linux and may slightly differ for other Linux distributions.

2.1 Setup the HSM

Setting-up the Primus HSM hardware or your Clouds HSM partition is not part of this application note. Please refer to the corresponding User Guide PrimusAPI_PKCS11-UserGuide_UG-Enn.pdf.

2.2 Installing the Primus HSM PKCS#11 Provider

Basic installation of Primus PKCS#11 provider v1.7 is according PKCS#11 Provider User Guide (PrimusAPI_PKCS11-UserGuide_UG-Enn.pdf), downloadable from the Securosys Support Portal.

Adapt the existing PATH variable setup to include provider, OpenSSL, and Apache bin folders:

- /usr/local/primus/bin
- /usr/local/primus/openssl/1.0.2u/bin
- /usr/local/primus/apache/2.4.46/bin

```
export PRIMUS_HOME=/usr/local/primus
export PATH=$PRIMUS_HOME/bin:$PRIMUS_HOME/openssl/1.0.2u/bin:$PRIMUS_HOME/apache/2.4.46/bin:$PATH
export LD_LIBRARY_PATH=$PRIMUS_HOME/openssl/1.0.2u/lib:$LD_LIBRARY_PATH
```

Ensure that OpenSSL 1.0.2u from the Primus provider package is now used with the following command:

```
openssl version
OpenSSL 1.0.2u  20 Dec 2019
```

Ensure that Apache/httpd 2.4.46 from the Primus provider package is now used with the following command:

```
httpd -v
Server version: Apache/2.4.46 (Unix)
```

Depending on the used OS you might get the following error message (e.g. Ubuntu 20, Debian 10):

```
httpd: error while loading shared libraries: libpcre.so.1: cannot open shared object
file: No such file or directory
```

In such error case search for the libpcre library location and add a symbolic link, e.g.:

```
sudo find / -name "libpcre.so*"
/usr/lib/x86_64-linux-gnu/libpcre.so.3
/usr/lib/x86_64-linux-gnu/libpcre.so.3.13.3
```

and add symbolic link for libpcre.so.1 to the available library, e.g. for Ubuntu 20 or Debian 10:

```
sudo ln -s /usr/lib/x86_64-linux-gnu/libpcre.so.3 /usr/lib/x86_64-linux-gnu/libpcre.so.1
```

2.3 Additional "primus.cfg" Configuration File Parameters

Insert [1]the following openssl section in `primus.cfg` configuration file before the last line with the closing bracket (/*end primus */) and adapt the **blue** colored parameters (slot, PKCS#11 PIN) according your HSM setup.

```
  /*--- OPENSSL CONFIGURATION SECTION ------------------------------------*/
  /*    for the included OpenSSL engine 1.0.2u only */
  openssl:
  {
    slotId       = 0; /* slot id to be used for OpenSSL */
    lib          = "/usr/local/primus/lib/libprimusP11.so";
    user_pin     = "PKCS#11_UserPIN"
    deferredInit = false;
    enableDigest = false;
    enableRand   = true;
    enableRSA    = true;
    enableSSLv3  = false;
    enableTrcLvl = true;
  }; /* end openssl */
}; /* end primus */       ← Closing bracket
```

2.3.1 Section openssl

Optional OpenSSL configuration (**only used for the OpenSSL version 1.0.2u** contained in the provider delivery). The OpenSSL configuration references the HSM configuration file.
The `openssl:{ … };` block contains the fields:

Attributes	Content description
slotId	Indicate which slot to use in the `hsmx:{ … slotx: { id = x;} … };` section
user_pin	User PIN login (`CK_USER`), your PKCS#11 password.
lib	The fully qualified path to the PKCS#11 library or symlink.
deferredInit	True or false. Set to 'true' when used in conjunction with Apache/SSL
enableDigest	True or false. When '`true`', uses the Primus HSM digest functions. Set this value to '`false`' when using `s_server/s_client` openssl command line options and Apache/SSL
enableRand	True or false. When set to '`true`' uses the Primus HSM RNG. Set to '`false`' with Apache/SSL
enableRSA	True or false. When set to '`true`', enables RSA with the Primus HSM
enableSSLv3	True or false. When set to '`true`', enables SSLv3/TLSv1_1 support for Apache/mod_ssl
enableTrcLvl	True or false. When set to '`true`' sets the syslog log mask defined by trace level when initializing the Primus OpenSSL Engine.

[1] The section may already exist in case of migrating from an older configuration file.

3 OpenSSL

3.1 Configuration

Adapt the section "openssl" of the configuration file primus.cfg according chapter 2.3.1:

```
    slotId       = 0; /* slot id to be used for OpenSSL */
    user_pin     = "Your PKCS#11 Password"
```

Add a symbolic link (if not existing):

```
sudo ln -sf /usr/local/primus/lib/libeprimus.so /usr/local/primus/openssl/1.0.2u/lib/engines/libprimus.so
```

And edit the openssl.cnf file in

```
/usr/local/primus/openssl/1.0.2u/ssl/openssl.cnf
```

as follow (contained in provided template):

```
[engine_section]
primus = primus_section

[primus_section]
engine_id = primus
dynamic_path = primus
```

and add further changes according your requirements.

3.2 OpenSSL Engine Test

```
openssl engine primus -t
(primus) Primus engine support
     [ available ]
```

3.3 OpenSSL Rand

```
openssl rand -hex -engine primus 24
engine "primus" set.
fb73a1d84c6f3b8921385f63dee84c9820116c7da2d2bc33
```

3.4 OpenSSL Digest

```
cp /usr/local/primus/bin/sign.txt .

openssl dgst -md5 -engine primus sign.txt
engine "primus" set.
MD5(sign.txt)= c068a3d70a9b57325384a8748b2d9cb3
```

```
openssl dgst -sha1 -engine primus sign.txt
engine "primus" set.
SHA1(sign.txt)= 29f5389986e44dc8b96df9e849df1afdefa96899
```

```
openssl dgst -sha224 -engine primus sign.txt
engine "primus" set.
SHA224(sign.txt)= 93e024d1db52c81e56df5280aa09d5338bf75a45a16554acb1027113
```

```
openssl dgst -sha256 -engine primus sign.txt
engine "primus" set.
SHA256(sign.txt)= 9a649857db05c9660788a34e3528c873390cb21d95387183de551b291167093a
```

```
openssl dgst -sha384 -engine primus sign.txt
engine "primus" set.
SHA384(sign.txt)=059f084019897172d9f0dd1ee941f420f2d6487fdcf2bc0f0bc07b735f8fbe70b743b44
2a1168d98b82e0f86f8563eaf
```

```
openssl dgst -sha512 -engine primus sign.txt
engine "primus" set.
SHA512(sign.txt)=0b2b1ff8ba9dcfbf16cfcf44ff7fb40843132ca8295c8471e807de114a5bb0742654413
e0bdfc2e3b2012ce2223a245351f5e2caa7b8ddd6140cb21e8712a00d
```

3.5 OpenSSL RSA

3.5.1 genrsa

```
openssl genrsa -engine primus -out primus.rsa.private.key 2048
engine "primus" set.
Generating RSA private key, 2048 bit long modulus
e is 65537 (0x10001)
```

Extract the public key into a separate file:

```
openssl rsa -in primus.rsa.private.key -pubout -out primus.rsa.public.key
writing RSA key
```

To initialize the seed with your own random data from `./myseed.rnd` for key generation:

```
openssl genrsa -engine primus -rand ./myseed.rnd 2048
engine "primus" set.
1139 semi-random bytes loaded
Generating RSA private key, 2048 bit long modulus
e is 65537 (0x10001)
```

3.5.2 rsautl

rsautl is obsolete since 2010 and replaced by pkeyutl.

3.5.2.1 RSA Sign

```
openssl rsautl -in sign.txt -inkey primus.rsa.private.key -keyform ENGINE -pkcs -sign
-engine primus -hexdump
engine "primus" set.
engine "primus" set.
0000 - 14 80 d7 c8 4a 6c 51 63-b1 68 3c 9b b4 a1 97 33   ....JlQc.h<....3
0010 - 96 0c fb f5 77 2b c1 f1-e0 34 2b 2c 09 7f d7 a4   ....w+...4+,....
0020 - 9a c1 12 2b b6 52 f3 7f-7c 9a 5f fd e7 c0 46 ee   ...+.R..|._...F.
0030 - 8b 23 93 81 48 94 ca cd-d1 81 9f de 47 f0 e3 7e   .#..H.......G..~
0040 - fd f3 fd 1f 22 78 ac 89-6e 95 a0 d8 d1 7c 8c 6e   ....."x..n....|.n
0050 - 0f 21 fd 9a 15 00 36 c8-a8 cf 39 90 d6 4c e7 72   .!....6...9..L.r
0060 - 1d a8 64 f2 f6 05 be 46-19 a9 7a 03 82 b4 5e c4   ..d....F..z...^.
0070 - bd 2c 38 b5 cd c8 33 f7-a3 38 e8 13 0e fd ea 2f   .,8...3..8.....
0080 - c9 62 d5 f3 da a1 bb 80-78 cf c8 a2 78 1e 72 38   .b......x...x.r8
0090 - d4 85 61 c3 e7 5f d7 ec-6f 75 af d2 6c 83 9d 2e   ..a.._..ou..l...
00a0 - 7c bd 52 b7 f5 26 af 2b-ba c8 fd d0 db a0 e0 e1   |.R..&.+........
00b0 - 16 45 97 3b 56 e5 0f 57-78 53 73 08 d5 2e 5c 6d   .E.;V..WxSs...\m
00c0 - c2 20 4f 8e 7e 19 8e 27-e4 a5 94 cb 31 13 13 a6   . O.~..'....1...
00d0 - 33 29 e8 e5 68 6e e5 8c-28 60 69 f4 6c fb d3 65   3)..hn..(`i.l..e
00e0 - 58 30 ac a4 e0 02 d4 a2-87 18 4f e5 c9 89 77 83   X0........O...w.
00f0 - dd 80 8b 20 14 c8 4a 42-f6 fb cf 28 3b 1e 59 7f   ... ..JB...(;.Y.
```

3.5.2.2 RSA Sign & Verify

```
openssl dgst -sha256 -sign primus.rsa.private.key -out sign.txt.sha256 -keyform ENGINE
-engine primus sign.txt

openssl dgst -sha256 -verify primus.rsa.public.key -signature sign.txt.sha256 -keyform
ENGINE -engine primus sign.txt
engine "primus" set.
Verified OK
```

3.5.2.3 RSA Encrypt & Decrypt PKCS1

```
openssl rsautl -encrypt -engine primus -keyform ENGINE -pkcs -inkey
primus.rsa.public.key -pubin -in sign.txt -out sign.enc

openssl rsautl -decrypt -engine primus -keyform ENGINE -pkcs -inkey
primus.rsa.private.key -in sign.enc -out sign.out.txt
```

3.5.2.4 RSA Encrypt & Decrypt RAW

```
openssl rsautl -encrypt -engine primus -keyform ENGINE -raw -inkey primus.rsa.public.key
-pubin -in sign.txt -out sign.enc

openssl rsautl -decrypt -engine primus -keyform ENGINE -raw -inkey
primus.rsa.private.key -in sign.enc -out sign.out.txt
```

Note, as `rsautl` uses the RSA algorithm directly, it can only be used with small input files (<key size)!

3.5.2.5 RSA Encrypt & Decrypt OAEP

Unsupported as there is no mechanism to pass in the hash argument

```
openssl rsautl -encrypt -engine primus -keyform ENGINE -oaep -inkey
primus.rsa.public.key -pubin -in sign.txt -out sign.enc

openssl rsautl -decrypt -engine primus -keyform ENGINE -oaep -inkey
primus.rsa.private.key -in sign.enc -out sign.out.txt
```

3.5.2.6 RSA Encrypt & Decrypt SSLv2

Unsupported sslv2 Padding

```
openssl rsautl -encrypt -engine primus -keyform ENGINE -ssl -inkey primus.rsa.public.key
-pubin -in sign.txt -out sign.enc

openssl rsautl -decrypt -engine primus -keyform ENGINE -ssl -inkey
primus.rsa.private.key -in sign.enc -out sign.out.txt
```

3.5.3 genpkey

```
openssl genpkey -algorithm RSA -pkeyopt rsa_keygen_bits:2048 -pkeyopt
rsa_keygen_pubexp:65537 -engine primus
engine "primus" set.
…
```

3.5.4 pkeyutl

3.5.4.1 sign/verify raw input (PKCS1)

```
openssl pkeyutl -sign -engine primus -keyform ENGINE -inkey primus.rsa.private.key -in
sign.txt -out sign.txt.bin

openssl pkeyutl -verify -engine primus -keyform ENGINE -pubin -inkey
primus.rsa.public.key -in sign.txt -sigfile sign.txt.bin
```

3.5.4.2 sign/verify PKCS1[2]

```
openssl pkeyutl -sign -pkeyopt digest:sha256 -engine primus -keyform ENGINE -inkey
primus.rsa.private.key -in sign.txt -out sign.txt.bin

openssl pkeyutl -verify -pkeyopt digest:sha256 -engine primus -keyform ENGINE -pubin
-inkey primus.rsa.public.key -in sign.txt -sigfile sign.txt.bin

openssl pkeyutl -sign -pkeyopt rsa_padding_mode:pkcs1 -pkeyopt digest:sha256 -engine
primus -keyform ENGINE -inkey primus.rsa.private.key -in sign.txt -out sign.txt.bin

openssl pkeyutl -verify -pkeyopt rsa_padding_mode:pkcs1 -pkeyopt digest:sha256 -engine
primus -keyform ENGINE -pubin -inkey primus.rsa.public.key -in sign.txt -sigfile
sign.txt.bin
```

3.5.4.3 sign/verify PSS

```
openssl pkeyutl -sign -pkeyopt rsa_padding_mode:pss -pkeyopt digest:sha256 -pkeyopt
rsa_mgf1_md:sha256 -pkeyopt rsa_pss_md:sha256 -pkeyopt rsa_pss_saltlen:-1 -engine primus
-keyform ENGINE -inkey primus.rsa.private.key -in sign.txt -out sign.txt.bin

openssl pkeyutl -verify -pkeyopt rsa_padding_mode:pss -pkeyopt digest:sha256 -pkeyopt
rsa_mgf1_md:sha256 -pkeyopt rsa_pss_md:sha256 -pkeyopt rsa_pss_saltlen:-1 -engine primus
-keyform ENGINE -pubin -inkey primus.rsa.public.key -in sign.txt -sigfile sign.txt.bin
```

3.5.4.5 sign/verify RAW

```
openssl pkeyutl -sign -pkeyopt rsa_padding_mode:raw -engine primus -keyform ENGINE
-inkey primus.rsa.private.key -in sign.txt -out sign.txt.bin

openssl pkeyutl -verify -pkeyopt rsa_padding_mode:raw -engine primus -keyform ENGINE
-pubin -inkey primus.rsa.public.key -in sign.txt -sigfile sign.txt.bin
```

3.5.4.6 encrypt/decrypt PKCS1

```
openssl pkeyutl -encrypt -engine primus -keyform ENGINE -pubin -inkey
primus.rsa.public.key -in sign.txt -out sign.txt.bin

openssl pkeyutl -decrypt -engine primus -keyform ENGINE -inkey primus.rsa.private.key
-in sign.txt.bin -out sign.txt.clear

openssl pkeyutl -encrypt -pkeyopt rsa_padding_mode:pkcs1 -engine primus -keyform ENGINE
-pubin -inkey primus.rsa.public.key -in sign.txt -out sign.txt.bin

openssl pkeyutl -decrypt -pkeyopt rsa_padding_mode:pkcs1 -engine primus -keyform ENGINE
-inkey primus.rsa.private.key -in sign.txt.bin -out sign.txt.clear
```

3.5.4.7 encrypt/decrypt RAW

```
openssl pkeyutl -encrypt -pkeyopt rsa_padding_mode:raw -engine primus -keyform ENGINE
-pubin -inkey primus.rsa.public.key -in sign.txt -out sign.txt.bin

openssl pkeyutl -decrypt -pkeyopt rsa_padding_mode:raw -engine primus -keyform ENGINE
-inkey primus.rsa.private.key -in sign.txt.bin -out sign.txt.clear
```

[2] The input file sign.txt must have exactly the length of 256 bits.

3.5.4.8 encrypt/decrypt OAEP

```
openssl pkeyutl -encrypt -pkeyopt rsa_padding_mode:oaep -pkeyopt rsa_mgf1_md:sha256
-pkeyopt rsa_oaep_md:sha256 -pkeyopt hexseed:ffeeaabbcceeff -engine primus -keyform
ENGINE -pubin -inkey primus.rsa.public.key -in sign.txt -out sign.txt.bin

openssl pkeyutl -decrypt -pkeyopt rsa_padding_mode:oaep -pkeyopt rsa_mgf1_md:sha256
-pkeyopt rsa_oaep_md:sha256 -pkeyopt hexseed:ffeeaabbcceeff -engine primus -keyform
ENGINE -inkey primus.rsa.private.key -in sign.txt.bin -out sign.txt.clear
```

3.5.4.9 Self-signed Certificate, CSR and Friends

Create a key pair and self-signed certificate:

```
openssl req -engine primus -new -newkey rsa:2048 -days 365 -nodes -x509 -outform DER
-out cert.der
```

Create a key pair and CSR:

```
openssl req -engine primus -nodes -newkey rsa:2048 -out csr.der -outform DER -subj
"/C=CH/O=HSM/OU=Primus/CN=primus"
```

Create a CSR using an existing key (primus.rsa.private.key):

```
openssl req -engine primus -new -key primus.rsa.private.key -keyform ENGINE -out csr.der
-outform DER -subj "/C=CH/ST=ZH/L=Zuerich/O=Securosys/OU=IT/CN=primus"
```

3.5.5 pkey

This command line option is not supported. The key material is stored on the Primus HSM.

3.5.6 s_server

Configure the `primus.cfg` openssl section with '`enableDigest=false`"

sslv2, sslv3 protocols are not supported

Create a key (if not previously done)

```
openssl req -engine primus -new -newkey rsa:2048 -days 365 -nodes -x509 -keyout
server.key -out server.crt
```

Start the openssl server using the key and certificate:

```
openssl s_server -accept 44444 -cert server.crt -key server.key -keyform ENGINE     -
engine primus -debug -www -tls1_2
```

Connect with the openssl client:

```
openssl s_client -connect localhost:44444 -tls1_2 -debug
…
Peer signing digest: SHA512
Server Temp Key: ECDH, P-256, 256 bits
---
SSL handshake has read 1420 bytes and written 433 bytes
---
New, TLSv1/SSLv3, Cipher is ECDHE-RSA-AES256-GCM-SHA384
Server public key is 2048 bit
Secure Renegotiation IS supported
```

```
Compression: NONE
Expansion: NONE
No ALPN negotiated
SSL-Session:
    Protocol  : TLSv1.2
    Cipher    : ECDHE-RSA-AES256-GCM-SHA384
    Session-ID: 9DDDAD17F6C3B498BBDFEAF782112EBA9B5651077C5A09238A7668A8FCF5BCCD
    Session-ID-ctx:
    Master-Key:
C29F20BCB76AEF4E8FFDF160958D44384CA7ECE2F31D07ED5C0EC461E3EB29659C310EA8E355ABE993CA4C1E
3E410C71
    Key-Arg   : None
    PSK identity: None
    PSK identity hint: None
    SRP username: None
    TLS session ticket lifetime hint: 7200 (seconds)
    TLS session ticket:
    0000 - 00 8a 4c ac eb 9b 03 6f-3f e5 c9 91 33 90 a6 1b   ..L....o?...3...
    0010 - ca 0b 45 a9 f9 82 b5 e1-5b d7 ef d7 3f 41 cd 15   ..E.....[...?A..
    0020 - 7b 85 4e 0f 59 f4 b8 e8-f4 71 a5 a1 79 72 47 0f   {.N.Y....q..yrG.
    0030 - da 83 c7 c9 76 aa 84 e9-90 86 4c f0 83 ea fa b3   ....v.....L.....
    0040 - 15 6c 34 93 52 2a ce 54-c5 de 64 50 f4 4f 02 25   .l4.R*.T..dP.O.%
    0050 - 98 e2 b2 8c a6 2c 43 66-26 23 30 b7 4e 63 5c ca   .....,Cf&#0.Nc\.
    0060 - 81 9a 71 02 18 a9 53 cd-af dc 43 40 64 22 90 9f   ..q...S...C@d"..
    0070 - 4c 74 67 0e cd 4d 23 71-fa 11 7a 57 84 01 04 d7   Ltg..M#q..zW....
    0080 - 81 b1 7e 48 0e 37 d2 f3-f2 a3 ab 45 c5 19 4d 8c   ..~H.7.....E..M.
    0090 - 2e 76 ee 0d 13 46 dc 67-06 bf 1b 04 b7 ff ac b5   .v...F.g........

    Start Time: 1516176680
    Timeout   : 300 (sec)
    Verify return code: 18 (self signed certificate)
```

3.5.7 Key Files

The private and public key files are base64 encoded OpenSSL "keys" (references). The generated private key files can only be used with the OpenSSL `-keyform ENGINE` option (reference to key on HSM).

3.6 OpenSSL Test Scripts

3.6.1 Test OpenSSL

Run the OpenSSL tests using the command line

```
test_openssl.sh
+ export OPENSSL=/usr/local/primus/openssl/1.0.2u/bin/openssl
+ OPENSSL=/usr/local/primus/openssl/1.0.2u/bin/openssl
+ /usr/local/primus/openssl/1.0.2u/bin/openssl engine primus -vvvv -c
(primus) Primus engine support
[… snipped …]
```

3.6.2 Test OpenSSL HW versus SW

Key Invalidation must be disabled to run the following test successfully!

To test OpenSSL using software and hardware keys, following preparation is necessary

- Import the sample RSA public key '`/usr/local/primus/bin/sft.rsa.public.key.pem`' onto the HSM partition.

- Import the sample RSA private key '`/usr/local/primus/bin/sft.rsa.private.key.pem`' onto the HSM partition.
- Copy '`/usr/local/primus/bin/sign-padded-256.bin`' to your working directory
- Copy '`/usr/local/primus/bin/sft.rsa.private.key.pem`' and '`/usr/local/primus/bin/sft.rsa.public.key.pem`' to your working directory and copy the files

Below is an example of the complete procedure.

Convert PEM key files to DER format:

```
openssl rsa -pubin -pubout -inform PEM -outform DER -in
/usr/local/primus/bin/sft.rsa.public.key.pem -out sft.rsa.public.key.der

openssl rsa -inform PEM -outform DER -in /usr/local/primus/bin/sft.rsa.private.key.pem
-out sft.rsa.private.key.der
```

Import keys in DER format onto HSM partition using pkcs11-tool:

```
pkcs11-tool --module /usr/local/primus/lib/libprimusP11.so --slot 0 -l -p <User PIN>
--label sender_pub -w sft.rsa.public.key.der -y pubkey --usage-sign --usage-decrypt

pkcs11-tool --module /usr/local/primus/lib/libprimusP11.so --slot 0 -l -p <User PIN>
--label sender_pub -w sft.rsa.private.key.der -y privkey --usage-sign --usage-decrypt
```

Copy and rename PEM files to your working directory:

```
cp /usr/local/primus/bin/sft.rsa.p* .
cp sft.rsa.public.key.pem primus.rsa.public.key.pem
cp sft.rsa.private.key.pem primus.rsa.private.key.pem
```

Copy sign-padded-256.bin to working directory:

```
cp /usr/local/primus/bin/sign-padded-256.bin .
```

Execute the test script:

```
test_openssl_engine_sw_vs_hw.sh
TESTS BEGIN
Checking RSA none
========
testing pkeyutl encryption with rsa NONE
hw encryption
engine "primus" set.
hw decryption from hw encryption
engine "primus" set.
OK
sw decryption from hw encryption
------OK
sw encryption
 [… snipped …]
```

4 Apache SSL

The provided Apache 2.4.46 requires the provided OpenSSL 1.0.2u.

4.1 Configuration

A pre-configured Apache based on OpenSSL 1.0.2u is available in

```
/usr/local/primus/apache/2.4.46
```

The Apache server configuration files are located in

```
/usr/local/primus/apache/2.4.46/conf
/usr/local/primus/apache/2.4.46/conf/extra
```

4.2 Enabling SSL with Primus HSM

The 'envvars' file in

```
/usr/local/primus/apache/2.4.46/bin/envvars
```

references the LD_LIBRARY_PATH variable to pick up the OpenSSL crypto and SSL libraries in

```
/usr/local/primus/openssl/1.0.2u/lib
```

OpenSSL is pre-configured to use the Primus HSM. Please refer to the OpenSSL section 2.3.1 for configuration details. Configure the openssl section within `/etc/primus/primus.cfg` and adapt the **blue** colored parameters:

```
      /* This is the openssl configuration section */
      openssl:  {
             slotId  = 0;
             user_pin   = " PKCS#11_UserPIN ";
             lib   = "/usr/local/primus/lib/libprimusP11.so";
             deferredInit = true;
             enableDigest = false;
             enableRand = false;
             enableRSA = true;
             enableSSLv3 = false;
             enableTrcLvl = true;
      }; /* end openssl */
```

- enableSSLv3=false -> to handle TLSv1_2
- enableSSLv3=true-> to handle SSLv3 or TLSv1_1

4.3 Add Apache Account to Primus Group

The user account used for running apache (usually `daemon`) must be included in the user group of the PKCS#11 provider users (`primus`) or adapt according your installation:

```
usermod -a -G primus daemon
```

4.4 Apache httpd.conf

Activate the Primus HSM in the SSL configuration section (contained in provided template):

```
# Secure (SSL/TLS) connections
Include conf/extra/httpd-ssl.conf
<IfModule ssl_module>
      SSLRandomSeed startup builtin
```

```
      SSLRandomSeed connect builtin
      SSLCryptoDevice primus
</IfModule>
```

4.4.1 Server Key and Certificate

Generate a sample self-signed server certificate and key

```
openssl req -engine primus -new -newkey rsa:2048 -days 365 -nodes -x509 -keyout
server.key -out server.crt
```

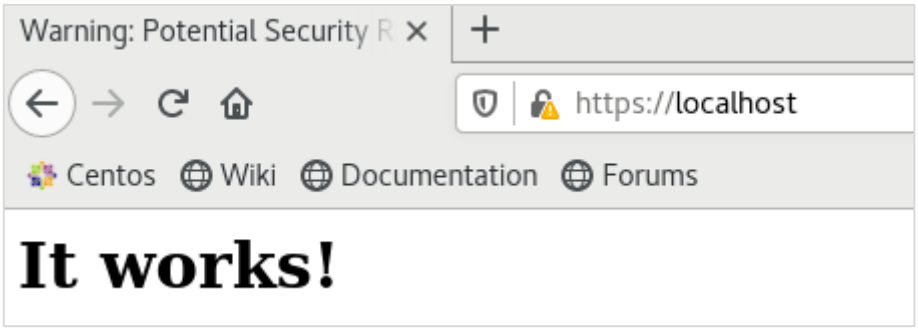
Copy server.key and server.crt to /usr/local/primus/apache/2.4.46/conf

```
sudo cp server.key /usr/local/primus/apache/2.4.46/conf
sudo cp server.crt /usr/local/primus/apache/2.4.46/conf
```

Start the Apache server

```
sudo /usr/local/primus/apache/2.4.46/bin/apachectl -k start
```

Connecting to https://localhost and accepting the self-signed certificate should show the following page:



Or connect with openssl client:

```
openssl s_client -connect localhost:443 -tls1_2
```

4.4.2 Importing PFX/PKCS#12

If you are in possession of an existing SSL software key, you can import the PFX/PKCS#12 file onto a Primus HSM partition. For this, you use the '`importPFX`'' command line. This utility extracts the certificates and certificate chain (if present) from the PFX/PKCS#12 file and writes them out to stdout or to files (one file for the certificate matching the key and one file containing the certificate chain). The software private key (support for RSA only) gets imported onto the Primus HSM partition and a modified software version of the key is written to a PEM file. The generated private key PEM file corresponds to the PEM version when generating the private key directly on the HSM.

4.4.2.1 Environment

Setup the correct `LD_LIBRARY_PATH` as shown in chapter 2.2:

```
export LD_LIBRARY_PATH=/usr/local/primus/lib:/usr/local/primus/openssl/1.0.2u/lib
```

4.4.2.2 Import PFX Command

```
importPFX -h
**********
Import PFX
**********
parameters:
-k: optional, PEM private key output file. Prints to stdout if not provided
-p: optional, PEM certificate (matching key) output file. Prints to stdout if not provided
-m: optional, PEM certificate chain (other certificates) output file. Prints to stdout if not
provided
commands:
ImportPFX -h: this message.
ImportPFX -x <pfx/pkcs12 file> -e <user name>: import private key from PFX to HSM partition.
```

Example, having sample.pfx as PKCS#12 container:

```
importPFX -x ./sample.pfx -e myUser -k ~/private.key -p ~/certificate.pem -m
~/certificate.chain.pem
```

Produces three files from the `sample.pfx`:

- `private.key` (references the HSM key)
- `certificate.pem` containing the certificate matching your private key
- `certificate.chain.pem` containg the optional certificate chain included in the PFX/PKCS#12 file

The `private.key` file is the file you use in your SSL configuration to access the hardware based private key stored on the Primus HSM partition.

Note: importPFX works currently only on PKCS#11 slot 0.