
Overview

This document describes how to configure Communications Server for Linux on Sytem z, V6.4 (CS Linux) or Communications Server for Data Center Deployment, V7 for Linux on System z to use the z90Crypto adapter that runs on z9, z10, and EC12 System z hardware. This is a new function added as APAR LI74667 as part of the CS Linux 6.4.0.1 PTF.

CS Linux needs additional configuration parameters to point the SSL code (GSKit) to the hardware driver, secondary key database file, Token, certificate, Token password, and z90Crypto options (number generation options, symmetric certification, etc.).

This document makes reference to the IBM Redpaper: "Using Cryptographic Adapters for Web Servers with Linux on IBM System z9 and zSeries" at:

<http://www.redbooks.ibm.com/abstracts/redp4131.html?Open>

This is a good reference to PKCS#11 implementation for WebSphere and Apache Web Servers. Many of the same configuration options apply to the CS Linux TN3270 Server.

Table of Contents:

Table of Contents

Configuration:.....	1
Certificate settings:.....	3
Operations:.....	7

Configuration:

The configuration parameters for z90Crypto support are supplied as environment variables for the CS Linux TN3270 Server and are not configured in the node configuration file (/etc/opt/ibm/sna/sna_node.cfg). The implementation described in this document allows for the settings to be filed and managed in its own configuration file.

The z90Crypto card must be configured in “accelerator” mode and not “co-processor” mode for these configuration parameters to work. The accelerator mode provides fast connection setup paths for the crypto support. Co-processor mode is meant for short transactional data that requires redundant checking. Using the z90Crypto in accelerator mode allows for more transactions to be established per minute. Once sessions establish using the z90Crypto card, the encryption/decryption is done using the CPACF firmware on z9 or z10. Session initialization is the most process intensive time when using SSL sessions. The “accelerator” provides the means to do that initialization in hardware.

Communications Server for Linux on System z Crypto Adapter Support

2

Any changes to the z90Crypto configuration will not take effect until the CS Linux software is stopped and restarted with the following environment variables set in the environment of CS Linux.

These parameters should be defined in a single place for ease of management. For example, create a file in the location where CS Linux environment settings are stored:

`/etc/opt/ibm/sna/environment`

The permissions of this file must be at least "read + execute" for root to execute this at startup.

`-r-xr-x--- 1 root root 418 /etc/opt/ibm/sna/environment`

File contents:

```
-----  
# CS Linux Hardware Crypto configuration  
  
# TN_HW_CRYPT_SUPPORTED=0  
# enables hardware adapter support (0=disabled,1=enabled)  
export TN_HW_CRYPT_SUPPORTED=1  
  
# TN_HW_CRYPT_PATH=hw_driver_path_string  
# full path to hardware driver for card  
export TN_HW_CRYPT_PATH=/usr/lib/pkcs11/PKCS11_API.so  
  
# TN_HW_CRYPT_KEYRING_PATH=key_database_path_string  
# full path to "secondary" keyring database  
# may be optional, depending on the specific crypto adapter  
export TN_HW_CRYPT_KEYRING_PATH=/etc/opt/ibm/sna/security/ibmcs.kdb  
  
# TN_HW_CRYPT_TOKEN_LABEL=token_name  
# set using the pkcsconf utility  
export TN_HW_CRYPT_TOKEN_LABEL=CEX2A0 # on RHEL 6: CEX3A1  
  
# TN_HW_CRYPT_KEY_LABEL=certificate_key_name  
# the label identifying which certificate to use from the keyring  
# database  
export TN_HW_CRYPT_KEY_LABEL=ibmcs-keyA  
  
# TN_HW_CRYPT_TOKEN_PWD=encrypted_for_hw_token_access  
# the encrypted password generated by the snatncrypt tool and used  
# to access the hardware token  
export TN_HW_CRYPT_TOKEN_PWD=1cc773934d462ea3a0efc1518dcf480b  
  
# TN_HW_CRYPT_SYMM=0  
# sets Symmetric processing option (0=off, 1=on)  
export TN_HW_CRYPT_SYMM=0
```

```
# TN_HW_CRYPT_DIGEST=0
# sets Digest option (0=off, 1=on)
# Digest keeps a cache of key exchanges to reduce duplication.
# Do not use digest for z90Crypto in accelerator mode.
export TN_HW_CRYPT_DIGEST=0

# TN_HW_CRYPT_RANDOMDATAGEN=0
# sets Random Data Generation option (0=off, 1=on)
# The z90Crypto card does not generate random numbers when
# used in “accelerator” mode.
# Do not use random data generation for z90Crypto in accelerator
mode.
export TN_HW_CRYPT_RANDOMDATAGEN=0

# TN_HW_CRYPT_PUBLICKEYGEN=0
# sets Public Key Generation option (0=off, 1=on)
export TN_HW_CRYPT_PUBLICKEYGEN=1

# end of CS Linux crypto configuration file
```

Certificate settings:

The certificates for the secure TN3270 Server sessions are held on the PKCS#11 device. The z90Crypto adapter only holds Personal certificates. It does not hold Signer certificates. When you open the z90Crypto library file to store certificates, GSKit will ask for a “secondary” key database to store Signer certificates. Therefore, both the hardware driver and the secondary key database must be specified when using the z90Crypto card with the CS Linux TN3270 Server.

For the following steps, refer to the IBM Redpaper: *Using Cryptographic Adapters for Web Servers with Linux on IBM System z9 and zSeries* (link provided in the overview above), pages 12-13:

In the following steps, if running on RHEL 6 replace `-c 0` with `-c 1`.

```
1) Load the drivers:
modprobe z90crypt
pkcs11_startup
service pkcsslotd start
```

Communications Server for Linux on System z Crypto Adapter Support

4

2) Create a Token definition for the PKCS#11 database that the card will use to track certificates for this Linux image. The Token label is arbitrary. In this example, we choose CEX2A0 for the label since we are using the Cryptographic Express2 Accelerator card. Use the -I option:

```
/usr/bin/pkcsconf -c 0 -I
```

You will be prompted to enter additional data:

Enter the SO PIN: ****so_pin**** (default SO PIN is 87654321)

Enter a unique token label: CEX2A0 (on RHEL 6: CEX3A1)

Now, take this Token label and set the environment variable in the environment settings file:

```
export TN_HW_CRYPT_TOKEN_LABEL=CEX2A0 # on RHEL 6: CEX3A1
```

3) Create an SO PIN (used for administration of the PKCS#11 device), using the -P option :

```
/usr/bin/pkcsconf -c 0 -P
```

You will be prompted to enter additional data:

Enter the SO PIN: ****old_so_pind****

Enter the new SO PIN: ****new_so_pin****

Re-enter the new SO PIN: ****new_so_pin****

4) Create a User PIN for the Token. Use the -u option:

```
/usr/bin/pkcsconf -c 0 -u
```

You will be prompted to enter additional data:

Enter the SO PIN: ****so_pin****

Enter the new user PIN: ****user_pin****

Re-enter the new user PIN: ****user_pin****

5) Change the User PIN for the Token. This step is required because the software requires that the PIN be reset after it is first created. To access the Token, applications like the CS Linux TN3270 Server must provide this PIN. Use the -p option:

```
/usr/bin/pkcsconf -c 0 -p
```

You will be prompted to enter additional data:

Enter user PIN: ****old_user_pin****

Enter the new user PIN: ****new_user_pin****

Re-enter the new user PIN: ****new_user_pin****

Communications Server for Linux on System z Crypto Adapter Support

5

6) Encrypt the Token user PIN so that it can be passed into the TN3270 Server to decrypt and pass into the PKCS#11 device:

```
/opt/ibm/sna/bin/snatncrypt **user_pin**  
943378FA34F44490
```

Now, take this encrypted key (the above numbers are an example of the output of the snatncrypt command) and set the environment variable in the environment file:

```
export TN_HW_CRYPT_TOKEN_PWD=943378FA34F44490
```

7) If running on RHEL 6, complete this step; otherwise, skip to the next step.

Create a new file: /etc/opt/ibm/sna/security/pkcs11.cfg with the content:

```
library = /usr/lib/pkcs11/PKCS11_API.so  
name = CEX3A1  
#slot = 1  
description = description  
attributes(*,CKO_PRIVATE_KEY,*) = {  
    CKA_PRIVATE=true  
    CKA_TOKEN=true  
}
```

Edit /opt/ibm/sna/ibm-java-s390-60/jre/lib/security/java.security to add the following line immediately below security.provider.10:

```
security.provider.11=com.ibm.crypto.pkcs11impl.provider.IBM  
PKCS11Impl /etc/opt/ibm/sna/security/pkcs11.cfg
```

8) Create a Token certificate in the PKCS#11 device and create the secondary key database. Run the SNA Key Manager program to create or access the Key database files:

```
snakeyman &
```

Select the "Open" option and set the type to "CMS Cryptographic Token". Enter the path to the PKCS#11 device driver. For example:

Key database type: CMS Cryptographic Token

File Name: PKCS11_API.so

Location: /usr/lib/pkcs11/

Select OK when done.



Communications Server for Linux on System z Crypto Adapter Support

6

On RHEL 6, the Key database type may be either of:

PKCS11Direct – The file name and location need to be filled in as shown.

PKCS11Config – The file name and location will be blank and grayed out.

Next, enter the Token Label (from step 1) and Token Password (from step 4). Select either "Create new" or "Open existing" for the secondary key database file. Do not use the default TN3270 Server software key database: /etc/opt/ibm/sna/ibmcs.kdb. This should only be used for software encryption (the default). For this example, change the location to /etc/opt/ibm/sna/security/ but keep the same file name.

The screenshot shows a dialog box titled "Open Cryptographic Token". It contains the following fields and options:

- Cryptographic Token Label:** A text box containing "CEX2A0".
- Cryptographic Token Password:** A text box containing "*****".
- Instructions:** A text block stating: "Some cryptographic tokens have limited capacity, and are unable to hold the signer certificates required to receive or import a personal certificate. If the selected cryptographic token has such a restriction, you may choose to open a secondary key database file to provide the extra capacity to hold signer certificates."
- Options:**
 - ☒ Open existing secondary key database file
 - ☐ Create new secondary key database file
- Key database type:** A dropdown menu showing "CMS".
- File Name:** A text box containing "ibmcs.kdb" with a "Browse..." button to its right.
- Location:** A text box containing "/etc/opt/ibm/sna/security/".
- Buttons:** "OK" and "Cancel" buttons at the bottom.

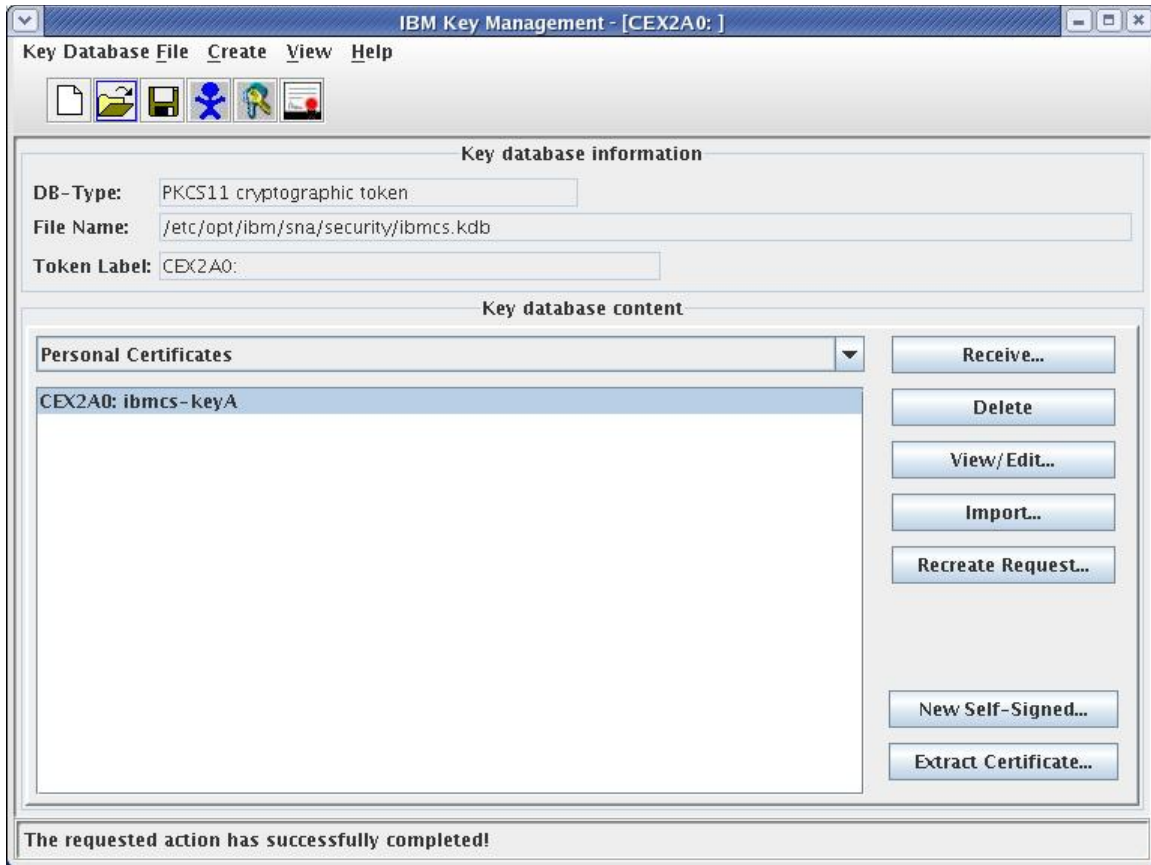
Set this environment variable in the environment file:

export TN_HW_CRYPT_KEYRING_PATH=/etc/opt/ibm/sna/security/ibmcs.kdb

Communications Server for Linux on System z Crypto Adapter Support

7

7) Create a self-signed certificate, or request, purchase, and receive a CA generated certificate for this host under the Personal Certificates section. The procedure to do this is not covered in this document.



The certificate key name you assign to the certificate will be used in the environment to specify which key the TN3270 Server is to use for the hardware encryption support:

```
export TN_HW_CRYPT_KEY_LABEL=ibmcs-keyA
```

If you created a self-signed certificate, you must extract the certificate to give to the clients for authentication. Use the Base64-encoded_ascii format (.ARM file) to use with clients like PCOMM or RUMBA.

If you use a certificate from a Certificate Authority (CA), then the clients must recognize (have the signer certificate for) that CA.

Operations:

Communications Server for Linux on System z Crypto Adapter Support

8

Prior to CS Linux on System z v6.4.0.2, the implementation of the z90Crypto support requires that the environment variables be set for the TN3270 Server hardware cryptographic settings each time the CS Linux starts.

In order to activate these settings, these environment variables must be set when the CS Linux daemon starts. To do this, enter the following lines, in **bold**, into the /etc/init.d/snastart file (as shown, immediately following the line "start)" and preceding the comment "start daemons"):

```
case $1 in
    start)

#
# test that tn3270 crypto settings file exists
#
if [[ -x /etc/opt/ibm/sna/environment ]]; then
#
# Include the file so export statement falls within scope
#
. /etc/opt/ibm/sna/environment
fi

# start daemons
/opt/ibm/sna/bin/sna start

# Uncomment the following lines to start the node
# /opt/ibm/sna/bin/snaadmin query_node 2>&1 > /dev/null
# if [ $? = 0 ]
# then
# /opt/ibm/sna/bin/snaadmin init_node
# fi
touch /var/lock/subsys/snastart
;;
```

To assure that this happens, always use the Linux "service" command to activate CS Linux:

```
service snastart start
```

This will call the service in the /etc/init.d directory (just like at boot time) and tell the snastart script to perform the "start" processing. If you instead use the "sna start" command, you must set the environment variables within the bash shell that you are issuing the command from; this can be done by sourcing the configuration file:

```
. /etc/opt/ibm/sna/environment
```


Communications Server for Linux on System z Crypto Adapter Support

9

By using the "service snastart start" command, you assure that the settings are established and are consistent for each activation of the CS Linux daemon.

Starting with Communications Server for Linux on System z V6.4.0.2, the environment setting file is automatically read if it exists. It should be a file with execute permissions, located at: /etc/opt/ibm/sna/environment. There is no requirement to start the server using the /etc/init.d/snastart script. Start the server using "sna start".

You may stop the SNA software with either "sna stop" or "service snastart stop". This will not affect the environment definitions defined for the TN3270 Server.