



SCMSH v1.04

(A CLI for Managing Cluster)

User Guide

Version 1.04

SCMSH v1.04 User Guide

Release: 1.04

Document release date: 6/8/2012

Copyright © 2012 Super Micro Computer, Inc.

All Rights Reserved.

Legal Notices

This software and documentation is the property of Super Micro Computer, Inc., and supplied only under a license. Any use or reproduction of this software is not allowed, except as expressly permitted by the terms of said license.

Information in this document is subject to change without notice.

Trademark Notice

All trademarks and copyrights referred to are the property of their respective owners.

Contacting Supermicro

Headquarters

Address: Super Micro Computer, Inc.
980 Rock Ave.
San Jose, CA 95131 U.S.A.

Tel: +1 (408) 503-8000

Fax: +1 (408) 503-8008

Email: marketing@supermicro.com (General Information)
support@supermicro.com (Technical Support)

Web Site: www.supermicro.com

Europe

Address: Super Micro Computer B.V.
Het Sterrenbeeld 28, 5215 ML
's-Hertogenbosch, The Netherlands

Tel: +31 (0) 73-6400390

Fax: +31 (0) 73-6416525

Email: sales@supermicro.nl (General Information)
support@supermicro.nl (Technical Support)
rma@supermicro.nl (Customer Support)

Asia-Pacific

Address: Super Micro Computer, Inc.
4F, No. 232-1, Liancheng Rd.
Chung-Ho 235, Taipei County
Taiwan, R.O.C.

Tel: +886-(2) 8226-3990

Fax: +886-(2) 8226-3991

Web Site: www.supermicro.com.tw

Technical Support:

Email: support@supermicro.com.tw

Tel: +886-(2)-8226-5990

Contents

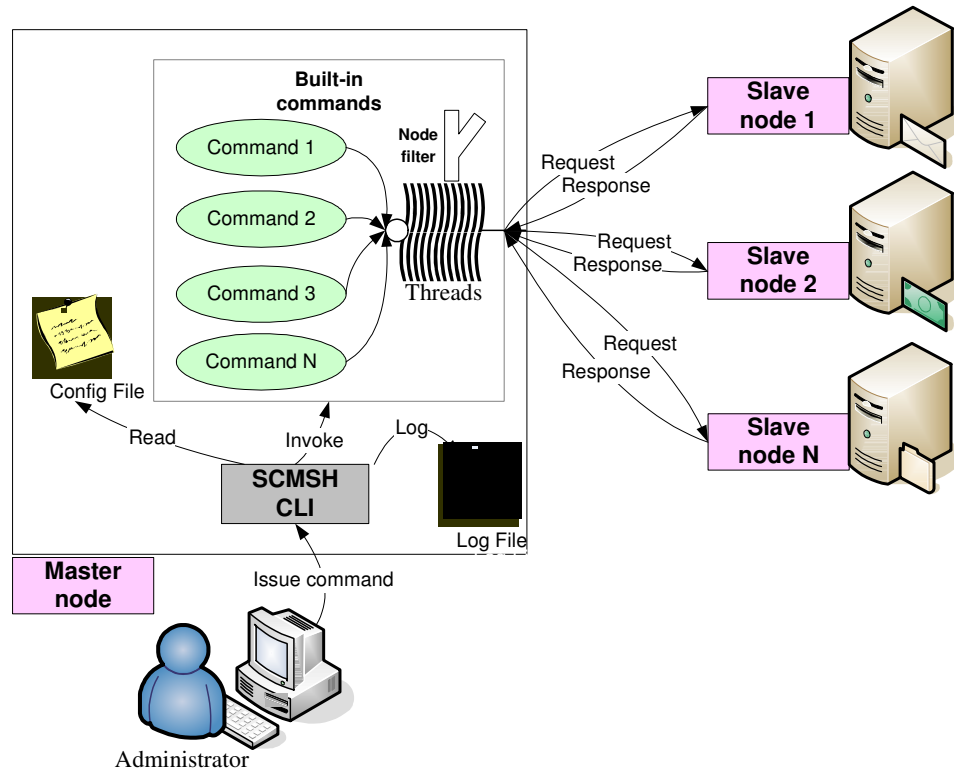
1. SCMSH Overview	3
1.1 SCMSH System Architecture	3
1.2 SCMSH Features.....	3
1.3 System Requirements.....	4
1.3.1 SCMSH.....	4
1.3.2 SCMNODE	4
1.4 Third Party Software	4
2. Installation	5
2.1 Installing SCMSH.....	5
2.2 Installing SCMNODE	5
3. Un-Installing.....	6
3.1 Un-Installing SCMSH	6
3.2 Un-Installing SCMNODE.....	6
4. Configuration	7
4.1 SCMSH Configuration.....	7
4.1.1 Ssl_master.cfg	7
4.1.2 cmd.cfg	7
4.1.3 node.cfg.....	8
4.1.4 cmdseq.cfg	9
4.2 SCMNODE Configuration	9
4.2.1 ssl_slave.cfg	9
5. Using product key for SCMSH	9
5. Using product key for SCMSH	10
5.1 Activating product key	10
5.2 Activating off-line product key	10
5.3 Deactivating product key.....	10
5.4 Extend trial	11
6 Using SCMSH in Interactive Mode	12
6.1 Defining Managed Slave Nodes/Groups	12
6.2 Running SCMSH	12
6.3 Running SCMSH shell options.....	12
6.4 Some Basic Commands and Usages.....	13
6.5 Built-In Command Sets	14
6.5.1 [x9biosfw] Command Set	14
6.5.2 [biosfw] Command Set.....	16
6.5.3 [ipmifw] Command Set.....	17

6.5.4 [system/mem] Command Set.....	17
6.5.5 [system/power] Command Set	17
6.5.6 [sdii] Command Set.....	17
6.6 Log of Executed Commands.....	18
6.6.1 scmsh.log.....	18
6.6.2 cmdexedump Folder	18
7. Using SCMSH in Batch Mode	19
7.1 Batch Mode	19
7.2 Batch Mode with Pipe	19
8. Using SCMNODE	20
8.1 Nodedaemon	20
9. Troubleshooting	21
9.1 Errors While Loading Shared Libraries	21
Appendix	22
Using the OpenSSL Utility to Create Certificate and Keys.....	22

1. SCMSH Overview

SCMSH is a simple command line interface designed for making management of lots of servers easier. Through SCMSH of the master node, you can simultaneously execute predefined commands on managed servers (slave nodes), with SCMNODE installed, and get results of the execution.

1.1 SCMSH System Architecture



1.2 SCMSH Features

- Supports batch mode and interactive mode.
- Supports group management.
- Records executed commands and results.
- Displays the execution time on each managed system.
- Asks for confirmation when critical commands are about to execute, such as reboot or shutdown.
- Supports SSL encrypted channel.
- Supports command sets in a tree structure.
- Supports user defined commands with shell script.

1.3 System Requirements

1.3.1 SCMSH

- **Hardware**
 - 20 MB free disk space
 - 256 MB available RAM
 - Ethernet network interface cards
- **Operating System**
 - Red Hat Enterprise Linux Server 6.x x86_64

1.3.2 SCMNODE

- **Hardware**
 - Supermicro servers
 - 25 MB free disk space
 - 256 MB available RAM
 - Ethernet network interface cards
- **Operating System**
 - Red Hat Enterprise Linux Server 6.x x86_64

1.4 Third Party Software

The following open source libraries are used by SCMSH:

Library	License
C++ STL	GPL
Boost	Boost Software License
log4cxx	Apache License
Simpleopt	MIT
OpenSSL	OpenSSL/SSLeay License

2. Installation

2.1 Installing SCMSH

1. Please pre-install `apr` and `apr-util` packages.
2. Extract *install_scmsh.tar.gz* and change to *install_scmsh* directory on the master node.

```
# tar zxvf install_scmsh.tar.gz
# cd install_scmsh
# ./install.sh    (The default path of SCMSH is "/opt/scmsh/")
```

2.2 Installing SCMNODE

1. Please pre-install `apr` and `apr-util` packages.
2. Extract *install_scmnode.tar.gz* and change to *install_scmnode* directory on slave nodes.

```
# tar zxvf install_scmnode.tar.gz
# cd install_scmnode
# ./install.sh    (The default path of SCMNODE is "/opt/scmnode/")
```

Note: The *test.crt.pem* of SCMSH, and *test.crt.pem* or *test.pri.pem* of SCMNODE are SSL certificates and keys used for tests only. For the sake of security, please create your own SSL certificate and keys. Please refer to [Using the OpenSSL Utility to Create Certificate and Keys](#) for creating SSL certificates and keys.

3. Un-Installing

3.1 Un-Installing SCMSH

Run *uninstall.sh* in *install_scmsh* directory to remove SCMSH from system.

```
# install_scmsh/uninstall.sh
```

3.2 Un-Installing SCMNODE

Run *uninstall.sh* in *install_scmnode* directory to remove SCMNODE from system.

```
# install_scmnode/uninstall.sh
```


4. Configuration

4.1 SCMSH Configuration

4.1.1 Ssl_master.cfg

CERTIFICATE_FILE=/opt/scmsh/test_cert.pem

The ***CERTIFICATE_FILE*** must point to the SSL certificate file you created for tests. Please refer to [Using the OpenSSL Utility to Create Certificate and Keys](#) for creating SSL certificates and keys.

4.1.2 cmd.cfg

This file is used to keep definitions of commands with shell scripts.

Format:

```
[cmdset]
name=mandatory
description=mandatory
confirmtype=optional [NONE|INHERITANCE|ONCE|EACH]
confirmmsg=optional
[cmd]
name=mandatory
description=mandatory
type=mandatory
cmd=mandatory
destination=mandatory
confirmtype=optional [NONE|INHERITANCE|ONCE|EACH]
confirmmsg=optional
```


For example, the following defines a **system/power** command set and a **restart** command.

```
[cmdset]
name=system/power
description=Commands for power control.
timeout=3500

[cmd]
name=restart
description=Restart the system
type=script
cmd=reboot
cmd=echo reboot
destination=slave
confirmtype=EACH
confirmmsg=Do you really want to reboot the system?
timeout=4500
```

4.1.3 node.cfg

Format of a node:

node_name=mac,ip[or hostname]:port

For example:

```
node_0000=AA:BB:CC:DD:00:00,localhost:4000
node_0001=AA:BB:CC:DD:00:01,localhost:4001
node_0002=AA:BB:CC:DD:00:02,localhost:4002
```

Format of a group:

group_name= any_node_or_group [,any_node_or_group] +

For example

```
group_00=node_0000,node_0001,node_0002
group_all=group_00,group_01,group_02
```

Note: Nodes need to be defined before groups in *node.cfg*.

4.1.4 cmdseq.cfg

This file is created after running SCMSH. It keeps only a sequence number for the next executed command. Do not delete or revise the content of this file, since if this file is deleted the sequence number for the next command is reset to 0.

4.2 SCMNODE Configuration

4.2.1 ssl_slave.cfg

CERTIFICATE_FILE must point to the SSL certificate file for test or your created file.

PRIVATE_KEY_FILE must point to the SSL private key for test or your created file.

Please refer to [Using the OpenSSL Utility to Create Certificate and Keys](#) for creating SSL certificates and keys.

CERTIFICATE_FILE=/opt/scmnode/test_crt.pem
PRIVATE_KEY_FILE=/opt/scmnode/test_pri.pem

5. Using product key for SCMSH

5.1 Activating product key

It needs to activate a product key for SCMSH before you start using it.

```
[root@X9DRT-HIBFF scmsh]# ./run_scmsh
This product is not activated! [1]
Please key in product key to activate product:
AAAA-BBBB-CCCC-DDDD-EEEE-FFFF-GGGG
Activate product successfully.
```

5.2 Activating off-line product key

If your system does not have connection to internet, you could activate an off-line product key.

1. Get an off-line activation request file by using [-olreq | --offlinerequestfile] option.

```
[root@X9DRT-HIBFF scmsh]# ./run_scmsh -olreq request.xml
Off-line activate request file is created successfully.
```

2. Email the off-line activation request file to your vendor and get an off-line activation response file from your vendor.
3. Active the product key with an off-line activation response file by using [-olact | --offlineactivate] option.

```
[root@X9DRT-HIBFF scmsh]# ./run_scmsh -olact ActivationResponse.xml
Off-line activate from file is successfully.
```

5.3 Deactivating product key

Please contact your vendor for deactivating a product key.

5.4 Extend trial

If you need to continue using the trial version after it's expired, you must get a trial extension key from your vendor.

```
[root@X9DRT-HIBFF scmsh] # ./run_scmsh  
Trial is expired!  
If you want to extend your trial, please contact your vendor!  
Please key in trail extension key from your vendor:  
HHHH-IIII-JJJJ-KKKK-LLLL-MMMM-NNNN
```


6 Using SCMSH in Interactive Mode

The **scmsh** file has been the only executable binary file, for SCMSH in the default path “/opt/scmsh/”. Please run **run_scmsh** for a shell in interactive mode.

6.1 Defining Managed Slave Nodes/Groups

Please refer to the [node.cfg](#) file format to define your managed slave nodes and groups. The original content of *node.cfg* is used for tests on a local system.

6.2 Running SCMSH

In a console on the master node, change the path to **/opt/scmsh/** and run **./run_scmsh**, you will see a SCMSH prompt:

```
[root@X9DRT-HIBFF scmsh]# ./run_scmsh
Log property file: /opt/scmsh/log4cxx.properties
Supermicro(c) SCMSH (A CLI interface for managing cluster.)
Version 0.3, Build 20120120
[X9DRT-HIBFF]$
```

Execute the “**help**” or “**??**” command for information of running of all built-in commands.

6.3 Running SCMSH shell options

- **-h:** Display the help information.
- **-dekey:** Deactivate a product key.
- **-olreq:** Create a file to request off-line activation.
Example: `./scmsh -olreq offlineactivaterequest.xml`
- **-olact:** Activate a off-line product key.
Example: `./scmsh -olact offlineactivaterequest.xml`
- **-oldereq:** Create a file to request off-line deactivation.
Example: `./scmsh -oldereq offlinedeactivaterequest.xml`
- **-chkact:** Check activation and update product key.
- **-chknodecfg:** Check if IP and MAC addresses match in specified node.cfg file.
Example: `./scmsh -chknodecfg node.cfg`

- **-correctmac:** Correct MAC address to match IP address in specified node.cfg file.
Example: ./scmsh -correctmac node.cfg
- **-searchnode:** Search active slave nodes by IP, IP range and port.
Example: ./scmsh -searchnode 192.168.10.1,192.168.10,101 4422
Example: ./scmsh -searchnode 192.168.10.1-192.168.10.100 4422
- **-addnode:** Add active slave nodes into specified node.cfg file by IP, IP range and port.
Example: ./scmsh -addnode 192.168.10.1,192.168.10,101 4422 node.cfg
Example: ./scmsh -addnode 192.168.10.1-192.168.10.100 4422 node.cfg

6.4 Some Basic Commands and Usages

1. **? command**

This command displays information about usage in the current level.

2. **?? command**

This command displays all information about usages.

3. **clear command**

This command clears the console.

4. **get file command**

This command tries to get a specified file from a slave node's 4422 directory, where 4422 is the TCP port used by nodedaemon.

5. **help, - -help command**

This command displays all information about usages.

6. **history command**

This command shows the history of executed commands.

7. To repeat a specific command in the history, follow these steps:

1. Press the **Up** or **Down** key to forward or return to display a command in the history. Then press the **Enter** key to execute it.
2. Execute the "**!nnn**" command to repeat a specific command in the history.
nnn is the number displayed with the command in the history.

8. **list command**

This lists all possible commands and the command set in the current level.

9. **list group command**

This command lists all managed groups of slave nodes.

10. **list node command**

This lists all managed slave nodes.

11. **quit command**

This command quits the SCMSH.

12. **save file** command

This command tries to save a specified file into a slave node's 4422 directory, where 4422 is the TCP port used by nodedaemon.

13. **show license info** command

This command shows product key and related license information.

14. Change command level

The multiple or tree-like structure command set can be defined in SCMSH.

Use the following commands to go to different command levels:

- **cd command_set_name**
Gets into a command set named **command_set_name**.
- **cd ..**
Gets out of a command set.
- **cd /**
Gets out of all command set.

6.5 Built-In Command Sets

For details of the commands in the following command set, please run **scmsh** and execute the **help** command in the specific command set.

6.5.1 [x9biosfw] Command Set

This includes commands for BIOS firmware of **X9** boards: backup bios config, flash bios fw, get bios built date from system, get bios id from system, restore bios config, and save bios fw.

Note: The “backup bios config” and “restore bios config” commands of x9biosfw command set are removed if the product key does not enable this feature for trial version.

“backup bios config” command

When you run “backup bios config” command in script mode on a slave node, a BIOS configuration file will be saved. You can manually change it with any text editor and remove unrelated BIOS setup options. And then you can restore the changed BIOS configuration to other slave nodes.

Here is an example of changing the **boot order** in BIOS configuration file in script

mode.

The original **boot order** boots from Hard Drive first:

```

Setup Question    = Boot Option #
Token    =07 // Do NOT change this line
Offset    =00
Width     =02
ListOrder =[0002] Hard Drive
           [0001] USB
           [0003] Network Card
           [0005] UEFI: Built-in EFI Shell

```

By using the following change, it can boot from Hard Drive first:

```

Setup Question    = Boot Option #
Token    =07 // Do NOT change this line
Offset    =00
Width     =02
ListOrder =[0003] Network Card
           [0002] Hard Drive
           [0001] USB
           [0005] UEFI: Built-in EFI Shell

```

When you run “backup bios config” command in raw mode on a slave node, three BIOS configuration files will be saved. The other two files are .hii and .list files.

(Note: The formats of script mode and raw mode BIOS configuration files are different.)

Here is an example of changing the **boot order** in BIOS configuration file in raw mode.

The original **boot order** boots from USB first:

```

GUID
45cf35f6-0d6e-4d04-856a-0370a5b16f53
Attributes
00000007
Variable Name
DefaultBootOrder
Variable Data
01 00 02 00 03 00 05 00

```


By using the following change, it can boot from Hard Drive first:

```
GUID
45cf35f6-0d6e-4d04-856a-0370a5b16f53
Attributes
00000007
Variable Name
DefaultBootOrder
Variable Data
03 00 01 00 02 00 05 00
```

“restore bios config” command

You can run “restore bios config” command to restore changed BIOS configuration file to other specified slave nodes.

6.5.2 [biosfw] Command Set

This includes commands for BIOS firmware of **X8/H8** boards: check bios info, check bios layout, check bios memory chip, check tool version, clear cmos checksum, dump cmos memory, flash bios, flash cmos memory, save cmos memory, save current bios, and validate bios id.

- **check bios info:** Check BIOS information.
Example: check bios info -n node_0000.
- **check bios layout:** Display the layout of the BIOS ROM file.
Option: -f, --file: The BIOS ROM file to check.
Example: check bios layout -f file -n node_0000.
- **check bios memory chip:** Check and show the memory chip used for BIOS.
Example: check bios memory chip -n node_0000
- **clear cmos checksum:** Clear the CMOS checksum.
Example: clear cmos checksum -n node_0000
- **dump cmos memory:** Dump the content of the current CMOS memory.
Example: dump cmos memory -n node_0000
- **flash bios fw:** Flash the file contents into the BIOS ROM chip.
Option: -f, --file: The BIOS ROM file to flash.
-x, --skipid: Disable Supermicro BIOS ID checking during flash.
-r, --reboot: Reboot system after flash BIOS.
Example: flash bios fw -f file -n node_0000
- **flash cmos memory:** Read the content of CMOS memory from a file and flash

it

Option:-f, --file: The file for flashing the CMOS.

Example: flash cmos memory -f file -n node_0000

- **save cmos memory:** Save the content of the current CMOS memory to file.
Option:-f, --file: This file is used to save the content of the current CMOS memory.
Example: save cmos memory -f file -n node_0000
- **save bios fw:** Save the current BIOS ROM chip contents to file.
Option:-f, --file: The file to save the content of the current BIOS.
Example: save bios fw -f file -n node_0000
- **validate bios id:** Validate the BIOS ID in the BIOS ROM file to see if it can flash on Supermicro mother board.
Option:-f, --file: The BIOS ROM file is used to validate.
Example: validate bios id -f -n node_0000

6.5.3 [ipmifw] Command Set

This includes commands for IPMI firmware (get ipmi fw revision, flash ipmi fw).

- **flash ipmi fw:** Flash ipmi firmware.
Option:-f, --file: this is ipmi FW image file that will be flashed.
Example: flash ipmi fw -f fwFile -n node_0000.
- **get ipmi fw revision:** Get IPMI FW revision on system
Example: get ipmi fw revision -n node_0000

6.5.4 [system/mem] Command Set

This includes commands for checking memory information.

- **mem:** Use **top** to check the memory usage.
- **swap:** Use **top** to check the swap usage.

6.5.5 [system/power] Command Set

This includes commands for power control.

- **power off:** Power off the system.
- **restart:** Restart the system.

6.5.6 [sdii] Command Set

This includes shell script commands for health monitoring from **SDII for Linux**.

- **failed items:** Check monitoring items in bad status.

- **failed items num**: Check the number of monitoring items in bad status.
- **fan**: Check fan monitoring items.
- **ok items**: Check monitoring items in good status.
- **ok items num**: Check the number of monitoring items in good status.
- **switch**: Check switch monitoring items.
- **temperature**: Check temperature monitoring items.
- **version**: Check the version of the SDII for Linux.
- **voltage**: Check voltage monitoring items.

Note: To ensure the commands in the [sdii] command set to work properly, **SDII for Linux** must be installed in the slave nodes. If **SDII for Linux** is not installed, the response would be empty.

6.6 Log of Executed Commands

6.6.1 scmsh.log

SCMSH records every executed command in this file.

For example:

```
2012-01-20 18:17:45,999 INFO : (0)list
2012-01-20 18:17:48,296 INFO : (1)cd biosfw
2012-01-20 18:17:49,459 INFO : (2)clear
```

The number in “()” stands for the sequence number of the executed command.

6.6.2 cmdexedump Folder

This folder, created automatically when using SCMSH, keeps execution results for every executed command. Each result is saved in a separate log file, *cmd_xxx.log*, where xxx maps to each executed command's sequence number.

7. Using SCMSH in Batch Mode

This feature allows SCMSH to be integrated easily with other shell scripts or programs.

7.1 Batch Mode

In a console on the master node, running a command in the following format executes a specific command of a command set.

```
# scmsh [command_set_name]+/command [command_option]
```

For example, the following command executes the **mem** command of the **system/mem** command set to check the memory status of slave node **node_0000**.

```
# scmsh system/mem/mem -n node_0000  
[node_0000] (547ms)  
Mem:   3107072k total,   645460k used,  2461612k free,   33944k buffers  
#
```

7.2 Batch Mode with Pipe

The following command does the same thing as Batch Mode, but this time it uses a pipe.

```
# echo "system/mem/mem -n node_0000" | scmsh  
[node_0000] (541ms)  
Mem:   3107072k total,   646948k used,  2460124k free,   34156k buffers  
#
```


8. Using SCMNODE

So far there is only one executable binary file, ***nodedaemon***, for SCMNODE in the default path “/opt/scmnode/”.

8.1 Nodedaemon

The ***nodedaemon*** is installed and runs on slave nodes to accept requests of command execution from ***scmsh*** in the master node, and responds with the execution result back. TCP port 4422 is used by ***nodedaemon***.

After install and before reboot the system, you need to start the ***nodedaemon*** manually.

```
[root@localhost scmnode]# /etc/init.d/scmnode start
Starting Supermicro SCM node daemon...
[root@localhost scmnode]# Listening on port 4422....
SSL configuration file: /opt/scmnode/ssl_slave.cfg
```

Note: The ***nodedaemon*** is configured as a service and will be automatically running after reboot.

9. Troubleshooting

9.1 Errors While Loading Shared Libraries

Q: When I run **scmsh** or **nodedaemon**, it says “error while loading shared libraries: libxxx.so: cannot open shared object file: No such file or directory”. What should I do?

A: Please make sure the path to **scmsh**, i.e., */opt/scmsh*, or a path to **nodedaemon**, i.e., */opt/scmnode*, is in the **LD_LIBRARY_PATH** system environment variable. After installation of SCMSH or SCMNODE, these paths should be set automatically.

Q: When I run **scmsh** or **nodedaemon**, it says “error while loading shared libraries: ./libxxx.so: cannot restore segment prot after reloc: Permission denied”. What should I do?

A: This usually results from SELINUX (Security Extension Linux). In some Linux distributions with 2.6 kernels, SELINUX is active by default and affects some system behaviors, including the loading of shared libraries.

You can do one of the following things to correct this:

- Disable SELINUX.
- Add “SELINUX=disabled” into the */etc/sysconfig/selinux* file and restart the system.

Use the *chcon* command to change the SELINUX security context for the specific share library.

Example: run the following command:

```
# chcon -t texrel_shlib_t path/to/your/shared/library/libxxx.so
```


Appendix

Using the OpenSSL Utility to Create Certificate and Keys

Please follow the steps below to create certificate and keys:

1. Create a private key by

```
# openssl genrsa -des3 -out your_pri.pem 2048
```

or

```
# openssl genrsa -out your_pri.pem 2048
```

If "-des3" option is used, it will ask for a password to protect the created private key. The number 2048 is the size of the key.

2. Create a certificate request by

```
# openssl req -new -key your_pri.pem -out your_cert.csr
```

A certificate request can then be sent to a certificate authority to get it signed into a certificate. If you need a self-signed certificate, or if you have your own certificate authority, you may sign it yourself.

3. Create a self-signed test certificate by

```
# openssl x509 -req -days 1095 -in your_cert.csr -signkey your_pri.pem -out
```

or

```
# openssl req -new -x509 -key your_pri.pem -out your_cert.pem -days 1095
```

4. Create a public key from the private key by

```
# openssl rsa -in your_pri.pem -pubout > your_pub.pem
```