Scripts

These UNIX shell scripts automate the input of XML requests to, and process the resulting output from, the Northbound Interface (NBI) Application Programmers Interface (API) of the Cisco Prime Provisioning network management application.

This appendix contains information about the following scripts:

- srDump
- taskdump
- taskDump
- upinterface
- VrfPing

README File

The README file contains an example of a working script file and describes the required environment variables and parameters, and the location for optional files.

Scripts Main directory

This section describes the scripts in the main directory. See the "Script Subdirectories" section on page C-20 for more information about these optional files required by the UNIX shell scripts in the main scripts directory.

Note These scripts work with either the Sybase and Oracle database.

env

The **env** file contains all of the environment or UNIX shell variable definitions required by all of the UNIX shell scripts in the main scripts directory. All existing UNIX shell scripts in this directory reference the **env** file. Any new scripts created must also include a reference to this file.

changeMaxRoutes

This script changes the maximum allowed VPN routes for the service request links that belong to the specified VPN and Customer. It also downloads the **maxRoutes** value to the PE devices that belong to the service request links.

Command Syntax

changeMaxRoutes [-v *vpnName***] [-c** *customerName***] -m** *maxroutes*

Table C-1 changeMaxRoutes Command Options

Option	Description
-v *vpnName*	VPN name. Optional parameter.
-c *customerName*	Customer name. Optional parameter.
-m *maxroutes*	The maximum number of VPN routes allowed in the device configuration. Required parameter.

STDOUT

```
State Success
OutputString
ilpe3.cisco.com|V1:test_vpn|change
ilpe3.cisco.com|V1:test_vpn|change
ilpe2.cisco.com|V2:test_vpn-s|change
ilpe2.cisco.com|V3:newvpn|nochange: Reason- since could not set maxroutes in repository
```

Where State is either *Success* or *Failure.* The OutputString is:

```
<pename>|<vrf name>|<change or nochange: Reason- >
```

LOG

The log information is stored in <Prime Provisioning log Location>/http.0.* in XML format.

The information stored depends on the log level. Log levels range from **SEVERE** to **FINEST**, and are set using **Administration > Control Center > Hosts > Configuration > Logging > Default > Loglevel**.

changepasswd

This script causes the Prime Provisioning application to change the password on a specified device.

Command Syntax

changepasswd -f *inputfilename* **[-log** *logfilename***] | changepasswd -help**

STDOUT

0 for success, 1 for failure

Log Name

The default log name is $PRIMEF_HOME/tmp/changepasswd.log.*$$*, where *$$* is the process ID. An alternate log file name can be specified in the input parameters.

Log Output Example

```
-------------------------------------
opening the repository
input: 3550_6-1|NbiRegion|test|test1|test2
rpmname: 3550_6-1
regionname: NbiRegion
newusername: test
newpassword: test1
newenpassword: test2
After constructTibrvMsg Password ID:: 43835
RPM 3550_6-1 Success
********
input:
```

changepw

This script causes the Prime Provisioning application to change the password on a specified device (single instance).

Command Syntax

changepw *host password enablepassword authpassword encrpassword*

Example:

changepw ilpe2 xyz abc qrs 123 asf

Fcollect

This script collects device configurations on a supplied list of devices.

Command Syntax

collect *device1* **[***device_list***]**

Example:

collect ensw2950-1 ensw2950-2

collectConfig

Use this script to collect the device configuration for a specified device (**rpmName**). The device configuration is stored in the directory $PRIMEF_HOME/tmp in a file named after the device. You can list multiple device names (multiple **rpmName** parameters).

Command Syntax

collectConfig -r *rpmName*

STDOUT

0 for success, 1 for failure.

File Name

$PRIMEF_HOME/tmp/device, where *device* is the name of the device (for example, 3550_6-1).

File Output Example

```
3550_6-1#term len 0
3550_6-1#show run
Building configuration...

Current configuration : 10676 bytes
!
version 12.1
no service pad
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
hostname 3550_6-1
!
enable secret 5 $1$xHqv$.pVjEARI1vXrJ7tK1S0qa1
!
errdisable recovery cause l2ptguard
errdisable recovery interval 5000
```

```
ip subnet-zero
!
...
```

Log Name

$PRIMEF_HOME/tmp/collectConfig.log

Log Output Example

```
Mon Aug  2 14:13:36 PDT 2004:  collectConfig started
-------------------------------------------
collectConfig request created for device: 3550_6-1
-------------------------------------------

saving config for device 3550_6-1 in the directory /opt/vpnsc/iscadmin/tmp
```

deletece

Deletes all CE devices in the repository that have no service requests associated with them.

Command Syntax

deletece [-p *pvc_id*]

Optionally, you can specify a single CE device using the pvc_id value with the -p script option flag.

STDOUT

```
Deleting unused CEs
c1234
The number of CEs deleted: 1
The number of Sites deleted: 0
To view the log file, please see /tmp/deletecefile
```

Log Name

$PRIMEF_HOME/tmp/deletecefile

Log Output Example

```
Deleting unused CEs
Start TIME = Mon Aug  2 15:24:36 PDT 2004
c1234
The number of CEs deleted: 1
The number of Sites deleted: 0
End TIME = Mon Aug  2 15:25:07 PDT 2004
```

deletesr

This script performs the following actions:

1) Decommissions the specified list of service requests (SRs).

2) Runs a report on all specified SRs returning the jobId and status.

3) Runs a purge request for a closed SR.

Command Syntax

deletesr *RJobId* **[***SRJobId***, ...]**

Example:

deletesr 5,6,7

where 5, 6, and 7 are the SRJobIds.

deleteSR

This script performs the following actions:

- Decommissions the list of service requests from the Prime Provisioning database. The service request is specified using the **SRJobId**.
- Produces an audit report for all specified service requests, returning the associated **JobId** and state for each one.
- Purges service requests in the *Closed* state.

The audit report can be disabled using the **-noaudit** option flag.

Command Syntax

deleteSR *SRJobId* **[***SRJobId***, ...]**

STDOUT

```
113     CLOSED - purging
Purge complete
```

deleteUnusedCpes

This script performs the following actions:

1. Finds all CPEs that are not part of an MPLS SR.
2. Deletes these CPEs.
3. Deletes their corresponding target devices.
4. Deletes any sites that no longer have any CPEs.

Note This script only checks for CPEs that are part of an MPLS SR. If any CPE is part of any other type of SR (L2VPN, for example), the script will fail.

Command Syntax

deleteUnusedCpes

deployallsr

This script performs the following actions:

- Finds all the MPLS service requests that are in the *Requested* state.
- Deploys the above listed MPLS service requests to the Prime Provisioning-managed network.
- The SRs are deployed in batches of 100.

Command Syntax

deployallsr [-outdir *dir_name*] **[-log** *log_file_name*]

Log Name

$PRIMEF_HOME/tmp/deployallsr.log.1897_08_03_04_09_06_29

Where 1897 is the process ID, and the remaining numbers are the date and time. An alternate log file name can be specified in the input parameters.

Log Output Example

```
SRs to be deployed...
146
Task deployment state: Completed
DateTime,SRJobID,State
2004-08-03 10:07:03,146,CLOSED
```

deployAllSR

This script deploys all MPLS SRs that are in the *Requested* state sequentially. It performs an audit by default unless the arguement **-noaudit** is passed.

It performs the following actions:

1. Finds all MPLS SRs that are in the *Requested* state.
2. Deploys each of these SRs.

Command Syntax

deployAllSR [-noaudit]

deploysr

This script performs the following actions:

- Deploys all service requests listed in the input parameters, regardless of the state.
- Produces an audit report for all specified service requests, returning the associated **JobId** and state for each one.

Command Syntax

deploysr *SR_ID* **[-noaudit] [-force]**

Table C-2 deploysr Command Options

Option	Description
SR ID	Service request ID.
-noaudit	An audit is performed by default unless the argument **-noaudit** is passed.
-force	The service request is deployed by default unless the argument **-force** is passed.

Example:

deploysr 5

where 5 is the SR ID.

STDOUT

None, unless there is an error. The following is an example of an error output:

```
The SR with ID: 1 does not exist in the Database!
```

downinterface

Use this script to turn off or shut down a given network interface (**interfaceName**) on a given device (**rpmName**). This script logs into the listed RPM device and inserts the **shutdown** IOS command on the specified interface.

Command Syntax

downinterface -rpm *rpmName* **[-user** *userName***] -pw** *userPassword* **-enableuser** *enableUserName* **-enablepw** *enablePassword* **-interface** *interfaceName* **[-log** *logFilename***]**

Table C-3 downinterface Command Options

Command Option	Description
-rpm	Hostname (or IP address) of the RPM (PE device). Required parameter.
-user	Login username. This parameter is only required if both the username and password are required for login.
-pw	Login password. Required parameter.
-enableuser	Enable username. This parameter is only required if both the username and password are required to enter enable mode.
-enablepw	Enable password. Required parameter.
-interface	The complete interface name (for example, Switch1.1). Required parameter.
-log	Log filename. Optional parameter. If not specified, the file downinterface.log is created in the $ECSP_HOME/tmp directory.

STDOUT

Non-zero exit code if there is an error.

getfile

This script will get the latest config for a device from the repository.

Command Syntax

getfile {{-device *device_name***} [-dirname** *outputFileDirectory***] {-outfile** *outputFileName***}} | {-r** *deviceName***}}**

Table C-4 getfile Command Options

Command Option	Description
-device *deviceName*	Mandatory. Name of device from which the config is to be collected.
-dirname *outpuFileDirectory*	Name of the directory, where the output file is located. [default '/tmp']
-outfile *outputFileName*	Mandatory. Specify output file name.
-r *deviceName*	Use this option if only the device name is to be specified. This will use the default values **/tmp** for output file directory and **tmp** for output file name.

Example:

getfile -device ilpe2 -dirname /tmp/xyz -outfile outputfile

getpe

This script provides a report for all PE device names and associated IP addresses contained in the Prime Provisioning database. The display is sent to the computer screen by default, or you can specify an output file, using the **-f filename** script option flag.

Command Syntax

getpe {-f *filename* **| -help**}

STDOUT

Creating getpe.txt in current directory

File Name

Default is getpe.txt (found in the directory where the script was executed). An alternate file name can be specified in the input parameters.

File Output Example

```
atlnga95r11-0038|null
```

```
stlsmo95r10-0063|null
stlsmo95r11-0064|null
washdc95r10-0068|null
washdc95r11-0069|null
atlnga95r12-0051|null
nycmny95r12-0057|null
okldca95r10-0059|null
okldca95r11-0060|null
cmbrma95r11-0084|null
dllstx95r13-0062|null
lsanca95r12-0054|null
okldca95r12-0061|null
clmboh95r10-0078|135.184.109.52
clmboh95r11-0079|null
atlnga95r10-0037|null
lsanca95r10-0035|10.20.21.136
lsanca95r11-0036|null
dllstx95r10-0033|null
dllstx95r11-0034|null
chcgil95r10-0039|135.184.14.155
```

getPEs

This script will print all of the names of the PEs along with their management IP addresses.

Command Syntax

getPEs

modifyce

This script modifies the CE device names in the Prime Provisioning database. The **inputfilename** parameter is used to specify the CE device names to be changed.

For example, the following input file:

1234 5678

4321 8765

makes these modifications:

- The site named C1234 is changed to C5678
- The device named c1234 is changed to c5678
- The site named C4321 is changed to C8765
- The device named c4321 is changed to c8765

Command Syntax

modifyce -input *filename* **[-log** *logFileName***]**

To send the output to a log file, use the **-log** script option and specify a **logFileName**.

STDOUT

0 for success, 1 for failure.

Log Name

Default log name is $PRIMEF_HOME/tmp/modifyce.log.$$

Where *$$* is the UNIX process id assigned to this script when it is run. An alternate log file name can be specified in the input parameters.

Log Output Example

```
**********************************************************************
*
*


Tue Aug  3 09:19:19 PDT 2004
********Detailed log messages for each of CE and it's Site name modification****
***
Success: Site with the name C1234 changed to C4321 and it's CE name changed from
 c1234 to c4321
******All the given CE names and it's Site name changed successfully!*******

**********************************************************************
```

purgeces

This script purges all closed SRs/CEs belonging to a VPN.

It performs the following actions:

1. Finds all closed SRs that are associated with the specified VPN.
2. Purges these SRs.
3. Deletes any CPEs and sites that are no longer used.

Command Syntax

purgeces [<VPN_NAME> | all]

Table C-5 purgeces Command Options

Option	Description
VPN_NAME	Purges closed SRs/CEs belonging to VPN_NAME.
all	Purges all closed SRs/CEs.

Example:

purgeces vpn1

This purges all CE's belonging to vpn1.

purgeConfigs

This script performs the following actions:

1. Runs a report to determine, which devices are candidates to have their configs removed.
2. Creates one or more collect config tasks. These task will perform collect config tasks on devices, which have exceeded the recommended number of stored configs.

Command Syntax

purgeConfigs [-t configThreshold]

Example: purgeConfigs -t 2

purgesrs

This script performs the following actions:

1. Finds all service requests in the Prime Provisioning database that are in the *Closed* state.
2. Purges or removes each of these service requests from the Prime Provisioning database.

If you specify a file and filename that contains a list of service request job IDs (**SRJobId**), only the service requests listed in the file are purged, and only if they are in the *Closed* state.

To purge service requests regardless of the state use the **-force** script option flag.

Command Syntax

purgesrs [-file <filename>] [-log <logFileName>] [-force]

If no arguments are given, all service requests in the *Closed* state are purged.

Table C-6 purgesrs Command Options

Option	Description
-file <filename>	The file containing the list of service requests to be purged.
-log <logFileName>	The log output file name.
-force	All service requests in <filename> are force purged

Log Name

Specified on the command line.

Log Output Example

```
SR with Id 140403 was purged
```

removesr

Use this script to change a specified service request to the *Decommissioned* state. The service request remains in the Prime Provisioning database but is not deployed. Use the job ID (**SRJobId**) to specify the service request to decommission.

Command Syntax

removesr SRJobId

STDOUT

New SR created 140403

showces

This script performs the following actions:

1. Shows all SRs/CEs belonging to a specified VPN.
2. Shows all SRs/CEs.

Command Syntax

showces [-h | -n *vpnName* **| -a]**

(use only one option with this script)

Table C-7 showces Command Options

Option	Description
-h	Prints help message.
-n *vpnName*	Prints SRs/CEs belonging to <vpn_name>
-a	Prints all SRs/CEs

Example:

showces -n vpn1

where vpn1 is the name of the vpn. This command displays all the ces'es related to vpn1.

showsr

This script performs the following actions:

- Finds all the MPLS service requests in the Prime Provisioning database which are not in the *Deployed*, *Functional*, or *Closed* state.
- Finds the VPNs associated with each MPLS service request.
- Finds the PE and CE devices associated with each MPLS service request.
- Displays this information in a table format.

When no arguments are specified, the output lists all service requests that are not in the *Deployed*, *Functional*, or *Closed* state.

Command Syntax

showsr [-a] [*last_N_sr***] [***sr_state***]**

showsr [-p *pvc_id***]**

showsr [-v *vpn_name***]**

Table C-8 showsr Command Options

Option	Description
-a	Prints all service requests regardless of the state.
last_N_sr	Truncates the number of service requests reported, regardless of the state.
sr_state	Reports only service requests in a specified state. Valid [sr_state] values are: REQUESTED, PENDING, FAILED_DEPLOY, INVALID, DEPLOYED, BROKEN, FUNCTIONAL, LOST, CLOSED, FAILED_AUDIT and WAIT_DEPLOY.
last_N_sr [*sr_state*]	Prints the last (N) of service requests in a specified state. If last_N_sr = 0, all service requests in state [sr_state] are printed.
-p *pvc_id*	Reports only service requests with a specific device ID.
-v *vpn_name*	Reports only service requests with a specific VPN name.

STDOUT

Job_ID SR_STATE PE_ROUTER CE_ROUTER VPN_ID CREATION_DATE_TIME

149 DEPLOYED dllstx95r10-0033 c1333698 V34 2004-1-27 15:56:18

srdump

This script performs one of these actions:

- Returns information about all service requests in the Prime Provisioning database, which contain the network device specified by the *pvc_id* parameter.
- Returns information about the service request designated by the *sr_id*. The **-sr** script option is required when requesting *sr_id*.

Command Syntax

srdump *pvc_id* **[-disable] [-configlet]**

srdump -sr *sr_id* **[-configlet]**

Table C-9 srdump Command Options

Option	Description
-sr	Indicates that the required argument refers to a service request ID. If **-sr** is not specified, a PVC device name must be defined.
sr_id | *pvc_id*	The required identification number of the service request for this report. • service request ID, *sr_id* • PVC device name, *pvc_id*
-disable	Disables full reports. Only a brief report is displayed for each service request. Use this option to reduce the amount of data reported. This option is only available with the *pvc_id* argument.
-configlet	Prints the configlet for each service request.

STDOUT

```
CREATION_TIME             2004-1-27 16:12:30
MODIFICATION_TIME         2004-1-27 16:12:41
SR_ID                     147
OpType                    ADD
SR_STATE                  DEPLOYED
CE_NAME                   c1331520.customer
PE_NAME                   nycmny95r11-0044.noc.att.com
BGP_AS                    65000
CE_ADDR                   128.222.253.118/30
CE_ENCAP                  FRAME_RELAY
CE_INTERFACE              Serial0.100
CE_DLCI/CE_VCD            777
CE_VCI                    -1
CE_VPI                    -1
NEIGHBOR_AS_OVERRIDE      false
PE_ADDR                   128.222.253.117/30
PE_ENCAP                  ATM
PE_INTERFACE              Switch1.216
PE_IF_SHUTDOWN            false
PE_VCD                    1
PE_VCI                    216
PE_VPI                    0
Vrf_Rd_Overwrite_Enabled  false
CERC                      any_to_any
IsHub                     true
HUB_RT                    13979:34
RD                        13979:34
VRF_NAME                  34
PE_CE_PROTOCOL            BGP

 Last State Change Comment: -1


------------------------------
 no  rate-limit input access-group rate-limit 6 56000 16000 32000 conform-action transmit
exceed-action set-prec-transmit 1
exit
int Switch1.216
 no  rate-limit input access-group rate-limit 7 208000 40000 80000 conform-action transmit
exceed-action set-prec-transmit 4
exit
```

```
int Switch1.216
 no   service-policy output COS_POLICY4:1
 pvc 0/216
 no   service-policy output COS_POLICY4:1
exit
int Switch1.216 point-to-point
 ip accounting precedence input
 rate-limit input access-group rate-limit 8 8000 8000 8000 conform-action
set-prec-continue 0 exceed-action set-prec-continue 0
 rate-limit input access-group rate-limit 7 8000 8000 8000 conform-action transmit
exceed-action set-prec-transmit 4
 rate-limit input access-group rate-limit 6 8000 8000 8000 conform-action transmit
exceed-action set-prec-transmit 1
 pvc 0/216
 service-policy output COS_POLICY3:1
 tx-ring-limit 3
exit
ip vrf 34
 maximum routes 4500 75
router bgp 13979
address-family ipv4 vrf 34
 default-information originate
 maximum-paths eibgp 6
 neighbor 128.222.253.118 route-map set-CE-local-pref in
exit
-------------------------------
```

srDump

This script performs the following actions:

Command Syntax

srDump [-d] [-c] [-h] [-s] *id*

Table C-10 srDump Command Options

Option	Description
-d	Disables full reports (only a brief report will be displayed for each service request).
-c	Prints the configlet for each service request.
-h	Displays the basic help.
-s	Indicates that the required argument, <id>, refers to a service request ID. If -s is not specified, it refers to a PVC device name.
id	The required identification number of the service request (with the -s option) or PVC device name for this report.

PVC ARGUMENT (CPE id)

Example: **srDump [-d] [-c]** *PVC_Id*

1) Prints a list of SRs associated with a CPE

- If more than one SR is found, the following brief output is printed:

  LocatorId

  State

  Organization

- If only one SR is found, the following detailed information is printed:

  CreationTime

  ModificationTime

  LocatorId

  State

  Organization

  MvrfCe

  pe

  CE_Intf_Address

  CE_DLCI

  CE_Intf_Encap

  CE_Facing_MVRFCE_Intf_Address

  CE_VCD

  CE_VCI

  CE_VPI

  Vrf_Rd_Overwrite_Enabled

  PE_Intf_Address

  PE_DLCI

  PE_Intf_Encap

  PE_Intf_Shutdown

  PE_Intf_Address

  PE_Facing_MVRFCE_Intf_Address

  PE_VCD

  PE_VCI

  PE_VPI

  Overidden_Rd

  Overidden_Vrf_Name

  MVRFCE_CE_Routes_To_Site_IP_Address

2) Prints only a brief summary if the **-d** flag is used.

3) Prints the configlet also if the **-c** flag is used.

SR ARGUMENT

Example: **srDump [-d] [-c] [-s]** *SR_Id*

1) The **-s** flag is required to make the script treat the *id* variable as an SR ID.

2) The **-c** flag also prints the configlet if the **-configlet** flag is used.

3) Prints the detailed information of the SR (see above).

(The **-d** flag will produce a less detailed output.)

taskdump

This script provides information about service request tasks. Indicate the detail of the report by specifying either a:

- service request ID (*sr_id*)
- task name (*task_name*)

Command Syntax

taskdump -h | *sr_id* | *task_name* **[-verbose]**

Table C-11 taskdump Command Options

Option	Description
-h	Prints the help message.
sr_id	Obtain information about tasks associated with service request.
task_name	Obtain information about a specified task.
-verbose	Obtain detailed task information.

STDOUT

```
Date: 2004-08-03T09:10:41 Level: INFO Message: Open repository succeeded
 ======== Creating ProvDrvSR succeeded for Job#140418SR#140423
     Date: 2004-08-03T09:11:07 Level: INFO Message: MPLS_VPN_Link[ 140413 ] Status [[
c2571924 ] Successful Deployment<br>[ dllstx95r10-0033 ] Successful Deployment<br>]
     Date: 2004-08-03T09:11:08 Level: INFO Message: Open repository succeeded
 ======== Creating ProvDrvSR succeeded for Job#140418SR#140423
bash-2.05b$ taskdump 140418
     Date: 2004-08-03T09:10:41 Level: INFO Message: Open repository succeeded
 ======== Creating ProvDrvSR succeeded for Job#140418SR#140423
     Date: 2004-08-03T09:11:07 Level: INFO Message: MPLS_VPN_Link[ 140413 ] Status [[
c2571924 ] Successful Deployment<br>[ dllstx95r10-0033 ] Successful Deployment<br>]
     Date: 2004-08-03T09:11:08 Level: INFO Message: Open repository succeeded
 ======== Creating ProvDrvSR succeeded for Job#140418SR#140423
```

taskDump

This script provides information about service request tasks.

Command Syntax

taskDump <[-A] [-I *id*] **[-S] [-R]> [-r] [-a] [-f** *file_name*] **[-h]**

(One or more of the first four options is required for this script.)

Table C-12 taskDump Command Options

Option	Description
-A	Dump all persistent tasks.
-I	Dump specific persistent task name, requires *id*
-S	Dump the related scheduled tasks.
-R	Dump all active runtime tasks.
-r	Dump related runtime tasks.
-a	Dump persistent task actions.
-f	Create a configuration file, requires *file_name*
-h	Display the brief help utility.

upinterface

Use this script to turn on (or turn up) a given network interface (**interfaceName**) on a given device (**rpmName**). This script logs into the specified RPM device and inserts the **no shutdown** IOS command on the specified interface.

Command Syntax

upinterface -rpm *rpmName* **[-user** *userName*] **-pw** *userPassword* **-enableuser** *enableUserName* **-enablepw** *enablePassword* **-interface** *interfaceName* **[-log** *logFileName*]

Table C-13 upinterface Command Options

Command Option	Description
-rpm	Hostname (or IP address) of the RPM (PE device). Required parameter.
-user	Login username. This parameter is only required if both the username and password are required for login.
-pw	Login password. Required parameter.
-enableuser	Enable username. This parameter is only required if both the username and password are required to enter enable mode.
-enablepw	Enable password. Required parameter.
-interface	The complete interface name (for example, Switch1.1). Required parameter.
-log	Log filename. Optional parameter. If not specified, the file upinterface.log is created in the $ECSP_HOME/tmp directory.

STDOUT

Non-zero exit code if there is an error.

VrfPing

VrfPing checks the connectivity between the PE and CE by executing the **traceroute vrf** and **ping atm** commands. If the **traceroute vrf** command succeeds, **VrfPing** returns with an exit status of 0. The **ping atm** command is executed only if the VCI value is specified with the **-vci** option and the **traceroute** command fails.

The exit states of VrfPing are:

- 0 - **traceroute** command successful.
- 1 - **traceroute** command failed. **ping atm** command successful (if vci was specified).
- 2 - **traceroute** command failed. **ping atm** command failed.

Command Syntax

VrfPing -pe *pe_name* **-ce** *ce_name* **-vrf** *vrf_name* **[-vci** *vci_value***] [-user** *user_name***] -pw** *user_passwd* **[-enuser** *enable_username***] -enpw** *enable_passwd* **[-log** *log_file_name***]**

Table C-14 VrfPing Options

Option	Description
-pe	Hostname (or IP address) of the PE device (RPM). Required parameter.
-ce	VPN interface address of the CE device. Required parameter.
-vrf	VRF name. Required parameter.
-vci	VCI value of the ATM subinterface.
-user	Login username. Required only if both username and password are required for login.
-pw	Login password. Required parameter.
-enuser	Enable username for PE. Required only if both username and password are required for login.
-enpw	Enable password for the PE device.
-log	Log file name. This parameter is optional. If not specified, the file vrfping.log is created in the $ECSP_HOME/tmp directory.

STDOUT

Non-zero exit code if there is an error.

Script Subdirectories

These subdirectories are located in the scripts main directory.

util

This directory contains UNIX shell scripts that are used by the UNIX shell scripts in the main scripts directory. They perform utility functions which might be used by any of the UNIX shell scripts in the main directory. Users that create or modify scripts in the main directory have reference to these utility scripts, but they cannot be used directly or modified.

xml

This directory contains input request XML template files. The main directory UNIX shell scripts read, copy, and modify the copied XML template file to generate inputs for the Prime Provisioning NBI. The files in this directory are not modified throughout the process.

filters

This directory contains variables, used by the UNIX shell scripts in the main directory, to filter the responses generated by the Prime Provisioning NBI before the response data is formatted for output to the user. As you create or modify UNIX shell scripts in the main directory, you might need to modify or add new filter files to this directory.

queries

This directory contains input request XML template files, similar to those in the xml subdirectory, but these files are in a different and more detailed format. The main directory UNIX shell scripts use the files in this directory in much the same way those in the *xml* directory are used. The resulting output from the NBI API are more detailed, and the scripts using the files of this directory can generate more detailed and formatted output to present to the user.