NORTEL NETWORKS

Passport 7400, 15000, 20000

Commands

241-5701-050

Passport 7400, 15000, 20000

Commands

Publication: 241-5701-050
Document status: Standard
Document version: 5.2S2
Document date: December 2003

Printed in Canada

Publication history

December 2003

5.2S2 Standard
General availability. Contains information on Passport 7400, Passport 15000, and Passport 20000 for the PCR 5.2 GA release

List of tables

About this document

241-5701-050 Passport 7400, 15000, 20000 Commands explains how to use commands to operate and maintain a Passport node or network through a text interface device such as a VT100 terminal. This document contains an alphabetical reference of most Passport commands. Each command description includes detailed examples.

The following topics are discussed in this section:

Who should read this document and why

This guide is for persons who perform operational, administration, and maintenance tasks.

What you need to know

This guide assumes that you understand the architecture and operation of Passport and DPN-100 products. Knowledge of DPN-100 is required if your network incorporates a call server resource module (CSRM). You also require a basic knowledge of UNIX.

You can acquire product knowledge by reading 241-5701-030 *Passport 7400, 15000, 20000 Overview*. Before undertaking operations and maintenance activities, you should have an understanding of the following product areas:

- Passport concepts
  - — Passport hardware and software
  - — Passport installation, commissioning, and provisioning
  - — Passport-to-Passport interworking
  - — Passport-to-DPN-100 interworking
- DPN-100 concepts
  - — routing, trunking, and addressing
  - — network installation, provisioning, operation, and maintenance
  - — service protocols
- UNIX
  - — a working knowledge of UNIX workstations
  - — a working knowledge of the UNIX operating system, its facilities and commands (such skills are those typically acquired through an introductory UNIX course, and through experience performing tasks such as file creation, maintenance, and manipulation)
- standard network operations and maintenance activities
- Preside Multiservice Data Manager workstation concepts

What's new in this document

The following features were added to this document:

- "CAS usability enhancements" (page 35)
- "Single shelf HSM" (page 35)

Other changes made to this document include the following:

- The sections "Reset <processor> command" (page 244) and "Switchover Lp command" (page 320) were updated with information about performing a CP switchover.

CAS usability enhancements

The following sections were updated for this feature:

- “Display command” (page 121)
- “Find command” (page 137)
- “List command” (page 159)

Single shelf HSM

The following section was updated for this feature:

- “Activate Prov command” (page 48)

  Note: This feature is not generally available. Contact Nortel Networks for more information on this feature.

Text conventions

This document uses the following text conventions:

- `nonproportional spaced plain type`

  Nonproportional spaced plain type represents system generated text or text that appears on your screen.

- `nonproportional spaced bold type`

  Nonproportional spaced bold type represents words that you should type or that you should select on the screen.

- *italics*

  Statements that appear in italics in a procedure explain the results of a particular step and appear immediately following the step.

  Words that appear in italics in text are for naming.

- [`optional_parameter`]

  Words in square brackets represent optional parameters. The command can be entered with or without the words in the square brackets.

- <general_term>

  Words in angle brackets represent variables which are to be replaced with specific values.

- UPPERCASE,lowercase

  Passport commands are not case-sensitive and do not have to match commands and parameters exactly as shown in this document, with the exception of string options values (for example, file and directory names) and string attribute values.

  There are two exceptions. Passport distinguishes between upper- and lowercase characters in attribute values and option values that are strings. Filenames, for example, are string option values and are case sensitive.

- |

  This symbol separates items from which you may select one; for example, ON|OFF indicates that you may specify ON or OFF. If you do not make a choice, a default ON is assumed.

- ...

  Three dots in a command indicate that the parameter may be repeated more than once in succession.

The term absolute pathname refers to the full specification of a path starting from the root directory. Absolute pathnames always begin with the slash (/) symbol. A relative pathname takes the current directory as its starting point, and starts with any alphanumeric character (other than /).

Entering commands

To successfully enter a command, you must consider the following:

- “Abbreviations” (page 37)
- “Keyboard shortcuts” (page 38)

Abbreviations

Many verbs, options, components, groups, and attributes have explicit abbreviations. For example, the explicit abbreviation for the verb display is *d* and the explicit abbreviation for the component *FileSystem* is *fs*. To display information about the file system, you can enter the whole name:

```
display FileSystem
```

Or its abbreviation:

```
d Fs
```

You can also abbreviate a command by entering only the first few characters of any of its parts. Passport will attempt to match the characters you typed to a valid command. For example, you can abbreviate the *FileSystem* component in the display FileSystem command using its first four characters:

```
display File
```

Since there are no other displayable components that begin with the characters *File*, Passport can match this abbreviation to a single valid command.

In some cases, the abbreviation matches more than one valid command, giving you an ambiguous match. Passport will prompt you to specify more characters to clear up the ambiguity. For example, you can use the list and the listFile verb with the *FileSystem* component. If you enter the following command, you will get an ambiguous match between the two verbs:

```
li File
```

If you intended to use the listFile verb, you would have to enter the first five characters to clear up the ambiguity, or use its explicit abbreviation of *ls*.

Some verbs have a minimum number of matching characters to prevent you from accidently entering a command that can have a serious impact on the system. The activate verb, for example, requires that you enter at least the first three characters. The following command gives an unambiguous match, but fails because it only uses the first two characters of the activate verb:

```
ac prov
```

You can find the explicit abbreviation for parts of a command and the minimum matching characters required for a verb using the help command (see “Help command” (page 146)). The explicit abbreviation is in parenthesis following the full name. The minimum matching characters are indicated in uppercase in the full verb name. For example, the activate verb is represented as ACTivate.

Keyboard shortcuts

The text interface provides a set of keyboard shortcuts you can use in either provisioning or operational mode. The shortcuts allow you to edit the command line and control the display of command responses. The table “Keyboard shortcuts” (page 38) summarizes these shortcuts.

Table 1
Keyboard shortcuts

Key	Description
Up Arrow and Down Arrow	Recalls recently used commands for edit or reuse. Passport stores the last 10 commands in a queue. Each time you press the Up Arrow key, you step back through the command queue. The Down Arrow key steps forward through the command queue.
Left Arrow and Right Arrow	Moves the cursor over the command without affecting the characters in the command.
Backspace or Delete	Deletes the character preceding the cursor.
Control-D	Deletes the character under the cursor.
Control-A	Moves the cursor to the beginning of the command line.
Control-E	Moves the cursor to the end of the command line.
Control-H	Moves the cursor back one character at a time.
Control-J or Control-M	Inserts a paragraph return.
Control-K	Deletes text from the character under the cursor to the end of the command line.
Control-N	Moves the cursor down one line at a time.

(Sheet 1 of 2)

Table 1 (continued)
Keyboard shortcuts

Key	Description
Control-P	Moves the cursor up one line at a time.
Return or Enter	Sends the command to Passport for processing.
Control-S	Suspends the response from a command. This shortcut is useful where a command response is longer than 24 lines. If you suspend output for a long period of time, the text interface discards some subsequent output. If the session is registered to display a data stream (alarm, SCN, log, or debug data), the session discards all the data generated while response is suspended. If you suspend a telnet interface for a long time and a large amount of subsequent output is queued, the underlying TCP connection can terminate, bringing down the telnet session.
Control-Q	Resumes suspended output.
Control-C	Cancels a response. One Control-C cancels only the current response, and does not affect other queued responses.

(Sheet 2 of 2)

Command wildcarding

Command wildcarding means using a wildcard character (*) or wildcard pattern to match more than one component or attribute. The commands display, list, and find support wildcarding.

Passport has two kinds of wildcarding:

- “Type wildcarding” (page 40)
- “Instance wildcarding” (page 40)

You cannot combine the two types of wildcarding in a single command.

Type wildcarding

Type wildcarding allows you to view information about all subcomponents of a component. You can match all subcomponents by substituting an asterisk for the component type. For example, the following command lists all *Bus* or *FabricCard*, *Card*, and *Test* subcomponents of the *Shelf* component:

```
list Shelf *
```

Type wildcarding only works if the asterisk is the last parameter of the command. In other words, you can only substitute an asterisk for the last component type.

Instance wildcarding

With instance wildcarding, you can view information about instances of a particular component type.

You can match all instances by substituting a wildcard (*) for the instance value. For example, the following command lists the attributes in the OsiState group from all instances of the *Lp* component:

```
display Lp/* OsiState
```

You can match a subset of instances by substituting a wildcard pattern for the instance value. For example, the following command will list all full provisioning views that also contain “Test” within the name portion of the view:

```
list Prov View/*Test*.full.*
```

You can substitute a wildcard (*) for more than one instance to view all instances from multiple component types in the component hierarchy. For example, the following command lists all instances of the *V35* component from all instances of the *Lp* component:

```
list Lp/* V35/*
```

For multi-indexed components you can substitute a wildcard (*) or a wildcard pattern for one or more of the indexes. You must specify the first index in a multi-index component and any unspecified indexes will default to *. For example, the command:

```
display Vr/* IP Fwd/*,255.*
```

will default to:

```
display Vr/* IP Fwd/*,255.*,*,*
```

You can match an expression using = (equal) or != (not equal) as valid operators when specifying attribute values. For example, the following command lists all LPs that are active:

```
list Lp/(usageState = active)
```

and the following command lists all LPs that are not active:

```
list Lp/(usageState != active)
```

You can match an expression by joining up to three attributes using the following operators: AND and OR. For example, the following command lists all the *fruni* components that have a locked open system interconnection (OSI) administrative state or a disabled OSI operational state.

```
list fruni/(adminState = locked OR operationalState =
disabled)
```

Note: The operators = and ! have a higher a precedence over the operators AND and OR.

You can also apply both instance filtering and value-based filtering to the same component instance. For example

```
display Laps/2* Sts/*(operationalState=disabled)
```

Command categories

Most Passport commands fall into one of the following categories as shown in “Command categories” (page 42).

Each of the following sections briefly explains the category of the command and lists, in alphabetical order, the commands that fall into the category.

Table 2
Command categories

Category and purpose	Command
Session Session commands allow you to manage your Passport session or the sessions of other users.	“Cancel command” (page 65) “Clear Nmis <interface> Session command” (page 87) “Logout command” (page 182) “Me command” (page 186) “Quit command” (page 224) “Telnet Vr command” (page 330)
Common You can use the common verbs with most components to form the common commands. Some components prevent the use of a particular common verb. To determine if a common verb works on a component, use the help command (see “Help command” (page 146)).	“Add command” (page 56) “Delete command” (page 118) “Display command” (page 121) “Find command” (page 137) “Help command” (page 146) “List command” (page 159) “Lock command” (page 177) “Set command” (page 272) “Unlock command” (page 360)

(Sheet 1 of 4)

Table 2 (continued)
Command categories

Category and purpose	Command
Provisioning The provisioning commands allow you to modify the configuration of your Passport node. To execute most of these commands you must be in provisioning mode (see “Start Prov command” (page 287)).	“Activate Prov command” (page 48) “Apply Prov command” (page 60) “Check Prov command” (page 71) “Clear Prov command” (page 91) “Commit Prov command” (page 96) “Confirm Prov command” (page 100) “Continue Prov command” (page 103) “Copy Prov command” (page 111) “End Prov command” (page 135) “Load Prov command” (page 173) “Save Prov command” (page 264) “Start Prov command” (page 287) “Stop Prov command” (page 300) “Tidy Prov command” (page 335)

(Sheet 2 of 4)

Table 2 (continued)
Command categories

Category and purpose	Command
File system The Passport file system stores the files necessary to the operation of the node. You can manage the file system and the files stored on it using these commands.	“Chdir Fs command” (page 68) “Copy Fs command” (page 106) “Format Fs Disk” (page 142) “ListFile Fs command” (page 168) “Lock command” (page 177) “MakeDirectory Fs command” (page 184) “Move Fs command” (page 188) “Protect Fs command” (page 215) “Remove Fs command” (page 232) “Synchronize Fs command” (page 327) “Touch Fs command” (page 344) “Unlock command” (page 360) “Unprotect Fs command” (page 363) “WorkingDir Fs command” (page 366)
Software Use the Software commands to download new software applications to your node and to delete old software applications.	“Remove Sw Av command” (page 236) “Start Sw Dld command” (page 291) “Stop Sw Dld command” (page 303) “Tidy Sw command” (page 341)
Logical processor Use the Logical processor commands to control the logical processors on your node.	“Lock command” (page 177) “ReloadCp Lp command” (page 229) “Reset <processor> command” (page 244) “Restart <processor> command” (page 253) “Switchover Lp command” (page 320) “Unlock command” (page 360)
(Sheet 3 of 4)	

Table 2 (continued)
Command categories

Category and purpose	Command
Line automatic protection switching on Passport 7400 Use these commands to control the automatic protection switching (APS) links on your node.	"Clear Aps command on Passport 7400" (page 83) "ProtectionLockout Aps command on Passport 7400" (page 218) "Switch Aps command on Passport 7400" (page 306)
Line automatic protection switching on Passport 15000 and 20000 Use these commands to control the line automatic protection switching (LAPS) links on your node.	"Clear Laps command on Passport 15000 and 20000" (page 85) "ProtectionLockout Laps command on Passport 15000 and 20000" (page 221) "Switch Laps command on Passport 15000 and 20000" (page 313)
Testing The Testing commands allow you to perform diagnostic tests on various parts of your node.	"Run Shelf Test command" (page 261) "Start <hardware> Test command" (page 282) "Stop <hardware> Test command" (page 296)
Routing The Routing commands allow you to clear connections, determine network paths, and to optimize the routing system.	"Ping Rtg Dpn <node> command" (page 207) "Ping <connection> Vc command" (page 201)
(Sheet 4 of 4)	

How to get more help

For information on training, problem reporting, and technical support, see the "Nortel Networks support services" section in the *product overview document.*

Chapter 1
Command reference

This chapter provides detailed descriptions of all Passport commands. In the descriptions, some commands have a variable in angle brackets (<>) as part of their name. These commands work with more than one component.

To create a valid command, replace the variable (and its angle brackets) with a specific component name. For example, the clear Nmis <interface> Session command works with three different components, including Nmis Telnet Session. The following is a valid clear Nmis <interface> Session command:

```
clear Nmis Telnet Session/1
```

The detailed descriptions include the following sections:

- Properties. This section lists the operator modes in which you can use the command, and the minimum impact and scope you need to execute the command. It can list other properties of the command as well.
- Syntax. This section lists the syntax for the most common uses of the command.
- Parameters. This section details all the parameters of the command, including its options.
- Examples. This section gives practical examples of how to use the command.
- Related Information. This section lists other information in this document related to the command.

Activate Prov command

Use the activate Prov (act Prov) command to bring into service either the edit view or a saved view.

> ***Note:*** Complete a semantic check of the edit view (using the check Prov command) before issuing the activate Prov command. If you do not complete a semantic check for the view, Passport automatically starts the check before the activate process begins.

When you activate a provisioning view, the provisioning system checks that all the software necessary to activate the components in the view is running on the module. If any necessary software is missing, the command fails and the system displays a list of missing software.

CAUTION
If you do not confirm the activate Prov command (see "Confirm Prov command" (page 100)) within 20 minutes of issuing it, a system restart occurs and the system reactivates the last committed view as the current view (see "Commit Prov command" (page 96)). A service disruption occurs while the restart is in progress.

Activating a view can delete certain components in the current view. Some components temporarily go out of service if the changes are critical to the provisioned data.

An activation can also result in a CP restart or reload, causing all services on the Passport node to fail. When this happens, wait until the restart or reload is complete and then login again and continue with the confirm Prov command. If an activation results in a CP reset or reload, and the edit view is not saved, the system puts it into a temporary saved view.

For more information on the activate Prov command, see

- "Properties for activate Prov" (page 49)
- "Syntax for activate Prov" (page 49)

Properties for activate Prov

The following table describes the properties of the activate Prov command.

Table 3
Properties of the activate Prov command

Mode	Impact	Scope	Component
Operational or Provisioning	Service	Application	ProvisioningSystem (Prov)

Syntax for activate Prov

The following table describes the syntax for the activate Prov command.

Table 4
Syntax of the activate Prov command

If you want to...	Use the following syntax:
Activate the edit view, making it the current view	activate Prov
Activate a saved view, making it the current view	activate -file(<view_name>) Prov
Force the activation of an unchecked view	activate -file(<view_name>) -force Prov
Enable the automatic pause of a hitless software upgrade	activate -Pause Prov ***Note:*** This feature is not generally available. Contact Nortel Networks for more information on this feature.
(Sheet 1 of 2)	

Table 4 (continued)
Syntax of the activate Prov command

If you want to...	Use the following syntax:
Disable the automatic pause of a hitless software upgrade	activate -noPause Prov
Force the activation when a restore is possible	activate -force Prov
(Sheet 2 of 2)	

Parameters for activate Prov

There are two types of parameters for the activate Prov command:

- "Activating a saved view parameter for activate Prov" (page 50)
- "Force override for activate Prov" (page 50)
- "Disabling the automatic pause parameter in case of faults occurring during a hitless software upgrade" (page 52)

Activating a saved view parameter for activate Prov

You can activate a saved view instead of the edit view using the file option:

```
-file(<view_name>)
```

Enter the name of the saved view using the three-part view filename. For information on the filename syntax, refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview*.

If you issue the activate Prov command in operational mode, you must use the file option. You cannot use the activate Prov command with the file option if another operator session is currently in provisioning mode.

Force override for activate Prov

Semantic check

You can override the check for the proper software using the force option:

```
-force
```

The force option also overrides critical warnings while continuing to activate the view.

When you activate an unchecked provisioning view, the provisioning system runs a semantic check. If there are semantic problems with the view, the system issues errors and warnings. Some warnings are critical. For example, you can receive warnings to restart a specific service, to activate the specified provisioning data change, or to restart the complete system. If a semantic check error or critical warning occurs, then the system aborts the activate Prov command. For critical warnings, you can override this behavior using the force option. You cannot use the force option to override semantic check errors.

Restore possible

If the journaled current view can be restored, then the activate Prov command may cause the journaled current view to be purged, and a restore to no longer be possible. In this case, the -force option is required for the activation to proceed. Note that a restore is only possible if the attribute *restorePossible* is set to yes.

Enabling saving of journal log files for activate Prov

By default, the saving of journal log files is enabled. By disabling journal log files, the saving of journal log files will be turned off for any subsequent activations. By enabling journal log files, the saving of journal log files will be turned on for any subsequent activations.

Note: You need to commit the configuration after enabling or disabling journal log file saving.

Enabling the automatic pause parameter for activate Prov when starting a hitless software upgrade

At the start of a hitless software upgrade, you can enable the automatic pause that occurs before migration switchover during a software migration activation:

`-Pause`

When a hitless software upgrade is activated with the Pause option, the software migration activation pauses before migration switchover occurs.

If faults occur before the pause, the faults will be included in the reason for the pause. If faults occur during the pause, a list of the faults is provided and the continue prov command will fail. If this occurs, you can:

- abort the migration
- fix the condition and continue by issuing the continue prov command
- proceed with the migration by issuing the continue -force prov command

Note 1: The -pause and -noPause parameters are mutually exclusive. If you issue the -pause parameter, you cannot issue the -noPause parameter.

Note 2: If the migration pauses because of a fault, or if a fault occurs during a pause, you must issue the continue -force prov command if you wish to continue despite the fault condition.

Note 3: This feature is not generally available. Contact Nortel Networks for more information on this feature

Disabling the automatic pause parameter in case of faults occurring during a hitless software upgrade

At the start of a hitless software upgrade, you can disable the automatic pause that occurs before migration switchover during a software migration activation:

```
-noPause
```

When a hitless software upgrade is activated with the noPause option, the software migration activation does not pause before migration switchover occurs, regardless of service affecting alarms.

Note: The -pause and -noPause parameters are mutually exclusive. If you issue the -pause parameter, you cannot issue the -noPause parameter.

Examples for activate Prov

The following examples show how to use the activate Prov command:

- “Activating the edit view in provisioning mode example” (page 53)
- “Activating a saved view in operational mode example” (page 53)
- “Activating a saved view and overriding the system check for proper software example” (page 54)

Activating the edit view in provisioning mode example

You want to activate the edit view in provisioning mode. You enter the following command:

```
activate Prov
```

An alarm warns all operators on the node that an activation has occurred. The following message prompts you to confirm the activation by issuing the confirm Prov command:

```
   Prov; <timestamp>
SET warning operator operationalCondition    70000007
   ADMIN: unlocked     OPER: enabled      USAGE: busy
   AVAIL:               PROC: initializing  CNTRL:
   ALARM:              STBY: notSet       UNKNW: false
   Id: 0D        Rel: Lp/0
   Com: Activation complete. Enter 'confirm prov' to
   confirm activation or rollback will occur in 20
   minutes.
   Int: 0/1/2/16663; casAlarm.cc; 686; p5.0d.19
  You have 20 minutes in which to confirm activation.
```

Activating a saved view in operational mode example

You want to activate the saved view DZL_test.full.023 in operational mode. The proper software for the saved view is currently running on the node. You enter the following command:

```
activate -file(DZL_test.full.023) Prov
```

An alarm warns all operators on the node that an activation has occurred. The message prompts you to confirm the activation by issuing the confirm Prov command.

```
Prov
    Invoking Load prov command.
    Prov;
SET warning operator
<timestamp> operationalCondition      70000007
.
.   <text deleted>
.
```

```
Int: 15/1/3/39955; casCPmainEntry.cc;470; p4.0d.26
Loaded file DZL_test.full.023.
You have 20 minutes in which to confirm activation.
```

Activating a saved view and overriding the system check for proper software example

You want to activate a saved view DZL_test.full.024, but the node is not currently running the proper software for the saved view. You attempt to activate the view using the following command:

```
activate -file(DZL_test.full.024) Prov
```

You receive a number of error messages ending with the following response:

```
Too many errors loading file DZL_test.full.024.
command failed
```

The command does not activate the saved view. However, you still want to activate the view regardless of the software. You override the system check for proper software using the following command:

```
activate -file(DZL_test.full.024) -force Prov
```

You receive a number of error messages, ending with the following response:

```
Loaded or applied file DZL_test.full.024.
```

Related information for activate Prov

See the following for information related to the activate Prov command:

- “Activate Prov command” (page 48)
- “Apply Prov command” (page 60)
- “Check Prov command” (page 71)
- “Clear Prov command” (page 91)
- “Commit Prov command” (page 96)
- “Confirm Prov command” (page 100)
- “Continue Prov command” (page 103)
- “Copy Prov command” (page 111)

- “End Prov command” (page 135)
- “Load Prov command” (page 173)
- “Restore Prov command” (page 258)
- “Save Prov command” (page 264)
- “Start Prov command” (page 287)
- “Stop Prov command” (page 300)
- “Tidy Prov command” (page 335)

Add command

The add (a) command lets you create a component (with its default values) in the edit view. You can only use the add command in provisioning mode.

The system adds all mandatory subcomponents, but shows the first six only. If there are more than six subcomponents, a message indicates the number of unseen subcomponents.

To add a component for a new feature or application, the software application version list or feature list must specify that feature or application. For details on specific components, see 241-5701-060 *Passport 7400, 15000, 20000 Components*.

For more information on the add command, see

- "Properties for add" (page 56)
- "Syntax for add" (page 57)
- "Parameters for add" (page 57)
- "Examples for add" (page 58)
- "Related information for add" (page 59)

Properties for add

The following table describes the properties of the add command.

> ***Note:*** Some components require a higher impact. See 241-5701-060 *Passport 7400, 15000, 20000 Components* for information on specific components or use the help command.

Table 5
Properties of the add command

Mode	Impact	Scope	Components
Provisioning	Configuration	Depends on component	All provisionable

Syntax for add

The following table describes the syntax for the add command.

Table 6
Syntax of the add command

If you want to...	Use the following syntax:
Add a new component	add <component_name>
Add a new component and any missing superior components	add -superiors <superior_component_name> <component_name>

Parameters for add

There are two parameter types for the add command:

- “Add a component parameter for add” (page 57)
- “Add superior components parameter for add” (page 57)

Add a component parameter for add

You specify which component you want to add to the edit view using the component of the command:

```
<component_name>
```

Add superior components parameter for add

To create a component, all of its superior component(s) must already exist. If they do not, rather than adding the superior components separately, you can add them automatically using the superiors option:

```
-superiors
```

You must specify the name of all superior components in the command.

Examples for add

The following examples shows how to use the add command:

- “Adding a component example” (page 58)
- “Adding a component and all its superior components example” (page 58)

Adding a component example

You want to create a new *virtualRouter* (*Vr*) component. You enter the following command:

```
add vr/0
```

You receive the following response:

```
Vr/0
   The following components have been created:
     Vr/0 Mm
     Vr/0 QosP/0
     Vr/0
ok
```

Adding a component and all its superior components example

You want to create a new *Ip* component under a new *Vr* component (*Vr/1*), and have the Passport software automatically add the superior *Vr* component. You enter the following command:

```
add -superiors vr/1 Ip
```

You receive the following response:

```
Vr/1 Ip
   The following components have been created:
     Vr/1 Mm
     Vr/1 QosP/0
     Vr/1
     Vr/1 Ip Arp
     Vr/1 Ip Icmp
     Vr/1 Ip RelayBC
        and another 3 component(s).
```

Related information for add

See the following for information related to the add command:

- “Clear Prov command” (page 91)
- “Copy Prov command” (page 111)
- “Delete command” (page 118)
- “List command” (page 159)
- “Start Prov command” (page 287)
- Components and commands. Refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview* for details.

Apply Prov command

Use the apply Prov (app Prov) command to apply the changes in a saved view to the edit view. Use only saved views with a delta format with this command. You can determine the format of all saved views on your node using the following command:

```
display Prov View/* formats
```

Use the apply Prov command only if you have saved the edit view or it is identical to the current view. For information on views, see 241-5701-045 *Passport 7400, 15000, 20000 Management System User Interface Guide*.

When you apply a provisioning view, the provisioning system checks to ensure that the software necessary to activate the components in the view is running on the switch. If any software is missing, the command fails and the system displays a list of the missing software. If the information in the saved view differs from the edit view, then the edit view prevails and the system displays warning messages. Note any conflicts and resolve them as necessary.

For more information on the apply Prov command, see

- “Syntax for apply Prov” (page 61)
- “Syntax for apply Prov” (page 61)
- “Parameters for apply Prov” (page 61)
- “Examples for apply Prov” (page 62)
- “Related information for apply Prov” (page 64)

Properties for apply Prov

The following table describes the properties of the apply Prov command.

Table 7
Properties of the apply Prov command

Mode	Impact	Scope	Component
Provisioning	Configuration	Application	ProvisioningSystem (Prov)

Syntax for apply Prov

The following table describes the syntax for the apply Prov command.

Note: The apply Prov command must always include the file option.

Table 8
Syntax of the apply Prov command

If you want to...	Use the following syntax:
Apply the changes in a saved view to the edit view	apply -file(<view_name>) Prov
Apply the changes in a saved view to the edit view, and override any system checks for necessary software	apply -force -file(<view_name>) Prov

Parameters for apply Prov

There are two types of parameters for the apply Prov command:

- “Applying a saved view parameter for apply Prov” (page 62)
- “System check override parameter for apply Prov” (page 62)

Applying a saved view parameter for apply Prov

You can specify that the provisioning system apply the changes in a saved view to the edit view, using the file option:

```
-file(<view_name>)
```

Enter the name of the saved view. Refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview* for information on the filename syntax of saved views.

System check override parameter for apply Prov

You can specify that the provisioning system override the system check using the force option:

```
-force
```

This option is useful if you know you have all the necessary software, and you want to skip the system check.

The force option overrides the check for proper software, and applies the saved view regardless of the software running on the node. The force option also ensures that the provisioning system continues to process the entire view rather than stopping after five semantic errors.

Examples for apply Prov

The following examples show how to use the apply Prov command:

- “Applying changes in the saved view to the edit view example” (page 62)
- “Applying changes in a saved view to the edit view and skipping the system check example” (page 63)

Applying changes in the saved view to the edit view example

You want to apply changes in the saved view AXL_test.full.003 to the edit view. You verify that the saved view has a delta format using the following command:

```
display Prov View/AXL_test.full.003 formats
```

You receive the following response:

```
Prov View/AXL_test.full.003
   formats = delta
```

Since the saved view has a delta format, you enter the following command:

```
apply -file(AXL_test.full.003) Prov
```

The system applies the changes from the saved view to the edit view, and resolves or reports conflicts. You receive the following response:

```
Prov
   Loaded or applied file AXL_test.full.003.
ok
```

Applying changes in a saved view to the edit view and skipping the system check example

You want to apply changes in a saved view to the edit view, but want to skip the system check for the proper software. In this example, the node is missing some of the proper software. If you try to apply the view without the force option, you receive a number of error messages ending with the following response:

```
Too many errors loading file AXL_test.full.004.
command failed
```

In this situation, the system detects the errors, and aborts the command.

To force the node to ignore the error messages and apply the view, you enter the following command:

```
apply -force -file(AXL_test.full.004) Prov
```

You receive a number of error messages, ending with the following response:

```
Loaded or applied file AXL_test.full.004.
```

Related information for apply Prov

See the following for information related to the apply Prov command:

- “Activate Prov command” (page 48)
- “Check Prov command” (page 71)
- “Load Prov command” (page 173)
- “ReloadCp Lp command” (page 229)
- “Save Prov command” (page 264)
- Saved view filename syntax. Refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview* for more details.

Cancel command

The cancel command allows you to cancel a wildcarded command and most commands that issue multiple responses or very long responses.

The cancel command works on wildcarded display commands and on all list commands. It only cancels the last command for which you have received part of the response.

You can type out cancel while the command is running, or you can use the keyboard shortcut of Control-C for Local and Telnet.

You can use the cancel command on the Preside Multiservice Data Manager Command Console.

For more information on the cancel command, see

- "Properties for cancel" (page 65)
- "Syntax for cancel" (page 66)
- "Canceling a command example" (page 66)
- "Related information for cancel" (page 67)

Properties for cancel

The following table describes the properties of the cancel command.

Table 9
Properties of the cancel command

Mode	Minimum Impact	Component
Operational or Provisioning	Passive	Does not affect components

Syntax for cancel

The following table describes the syntax for the cancel command.

Table 10
Syntax of the cancel command

If you want to...	Use the following syntax:
Cancel the last wildcarded command you entered	cancel

Canceling a command example

You want to view all the attributes for logical processor 1. When you enter the command, you accidently type an asterisk for the instance number:

```
display -noTabular Lp/*
```

The long and mostly unwanted command response begins to display:

```
Lp/0
   adminState = unlocked
   operationalState = enabled
   usageState       = active
   availabilityStatus =
```

Without waiting for a the command prompt, you enter the cancel command:

```
cancel
```

Alternatively, you press Control-C. The command stops and the following message displays:

```
  Command cancelled.
command failed
```

Related information for cancel

See the following for information related to the cancel command:

- “Display command” (page 121)
- “List command” (page 159)
- “Command wildcarding” (page 39)

Chdir Fs command

The chdir Fs (cd Fs) command allows you to change your current working directory on the Passport file system.

If you do not specify a directory, most file system commands assume you want to use your current working directory. You can determine the name of your current working directory using the workingDir Fs command.

For more information on the chdir Fs command, see

Properties for chdir Fs

The following table describes the properties of the chdir Fs command.

Table 11
Properties of the chdir Fs command

Mode	Impact	Scope	Component
Operational or Provisioning	Passive	Device	FileSystem (Fs)

Syntax for chdir Fs

The following table describes the syntax for the chdir Fs command.

Table 12
Syntax of the chdir Fs command

If you want to...	Use the following syntax:
Change to a new working directory	chdir -path(<directorypath>) Fs

New working directory parameter for chdir Fs

You can specify the new working directory using the path option:

```
-path(<directorypath>)
```

Enter the absolute or relative path for the new working directory. Enclose the path in double quotation marks. For information on path syntax, refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview*.

You must enter this option.

Changing the current working directory example

Your current working directory is the root directory (/). You want to change it to the provisioning directory. You enter the following command:

```
chdir -path(“provisioning”) Fs
```

You check your current working directory using the workingDir Fs command:

```
workingDir Fs
```

You receive the following response:

```
Fs
   /provisioning
```

You now want to change your current directory to the /spooled/closed directory using its absolute path. You enter the following command:

```
chdir -path("/spooled/closed") Fs
```

Now you want to change to the opened subdirectory of the spooled directory using a relative pathname. You enter the following command:

```
chdir -path("../opened") Fs
```

Related information for chdir Fs

See the following for information related to the chdir Fs command:

- "ListFile Fs command" (page 168)
- "WorkingDir Fs command" (page 366)

Check Prov command

Use the check Prov command to run a semantic check for errors on the provisioning data in the edit view. Use this command before the activate Prov command.

> ***Note:*** If you have changed a *Software*, *LogicalProcessor*, *LogicalProcessorType*, or *Card* component, you cannot run the check Prov command while a start Sw Dld, tidy Sw, or a remove Sw command is in progress.

During a full semantic check, the system issues a warning for each component that will go out of service because of critical provisioning data changes. When you activate the edit view, these components go out of service and then return to service after the activate is complete. See “Activate Prov command” (page 48) for more information on view activation.

For more information on the check Prov command, see

Properties for check Prov

The following table describes the properties of the check Prov command.

Table 13
Properties of the check Prov command

Mode	Impact	Scope	Components
Provisioning	Configuration	Application	ProvisioningSystem (Prov)

Syntax for check Prov

The following table describes the syntax for the check Prov command.

Table 14
Syntax of the check Prov command

If you want to...	Use the following syntax:
Perform a semantic check on the entire edit view	check Prov
Perform a semantic check on only those components in the edit view that are new or changed from the current view	check -changed Prov
Perform a semantic check on a single component and its subcomponents	check -component(<component_name>) Prov
Perform a semantic check on a single component only (and not its subcomponents)	check -component(<component_name>) -nonrecursive Prov
Perform a semantic check but stop it after the system encounters the first error	check -stopOnError Prov

Parameters for check Prov

There are four types of parameters for the check Prov command:

- “Check changed components parameter for check Prov” (page 73)
- “Check component parameter for check Prov” (page 73)

Check changed components parameter for check Prov

You can specify that the provisioning system check only components in the edit view that are new or changed from the current view using the changed option:

-changed

You cannot use this option in conjunction with the component option.

This option is useful if you wish to save the time involved in performing a full semantic check. However, to activate the edit view, the system must successfully complete a full semantic check.

Check component parameter for check Prov

You can specify that the provisioning system check a specific component and its subcomponents, using the component option:

-component(<component_name>)

This option requests a semantic check of the indicated component and its subcomponents. This option starts a fully recursive check of the component, including all its subcomponents and their attributes. This option does not provide out of service warnings.

You cannot use this option in conjunction with the changed option.

This option is useful when changes in a provisioning session affect only a single component. This option is much faster than performing a full semantic check on all components. However, to activate the edit view, the system must successfully complete a full semantic check.

Nonrecursive parameter for check Prov

You can specify that the provisioning system perform a nonrecursive check on a component using the nonrecursive option:

-nonrecursive

This option limits a semantic check to a single component. The system does not check subcomponents and their attributes.

You must use this option in conjunction with the component option.

This option is useful when changes in a provisioning session affect only the top layer of a single component. This option is much faster than performing a full semantic check. However to activate the edit view, the system must successfully complete a full semantic check.

Stop check at first error parameter for check Prov

You can specify that the provisioning system stop the check when the first error is found, using the stopOnError option:

```
-stopOnError
```

This option stops the semantic check at the first error.

If you do not use this option, the provisioning system attempts to find as many errors as possible, up to a maximum of five. If the system detects five errors, it aborts the semantic check.

Examples for check Prov

The following examples show how to use the check Prov command:

- “Checking the provisioning data of a view example” (page 75)
- “Checking the provisioning data for new or changed components example” (page 75)
- “Checking the provisioning data of a component and its subcomponents example” (page 76)
- “Checking a component only example” (page 77)
- “Stop checking the provisioning data of a view after encountering an error example” (page 77)

Checking the provisioning data of a view example

You want to start a semantic check with no options. You enter the following command:

```
check Prov
```

You receive the following response:

```
Prov
ok
```

OR

A warning appears to indicate which components will be taken out of service as a result of the change. In this example, after a critical change to the *FrameRelayUni/2* component, the following message appears:

```
Prov
   Warning: The following components will be taken out
   of service:
FrUni/2
```

Checking the provisioning data for new or changed components example

For the this example, assume that you added two ATM interfaces (*AtmIf/5* and *AtmIf/21*) without assigning them to a logical processor.

You complete a check of any new or changed components using the following command:

```
check -changed Prov
```

You receive the following response:

```
AtmIf/5
   Error:
      Related Component:  AtmIf/5
      Related Attribute:  interfaceName
      Reason:  Unable to retrieve the LP Info from the
               Interface Name of the atmIf.
AtmIf/5 ConnMap
   Error:
      Reason:  Unable to retrieve the LP Info from the
```

```
                Interface Name of the atmIf.
AtmIf/21
   Error:
      Related Component:  AtmIf/21
      Related Attribute:  interfaceName
      Reason:  Unable to retrieve the LP Info from the
               Interface Name of the atmIf.
AtmIf/21 ConnMap
   Error:
      Reason:  Unable to retrieve the LP Info from the
               Interface Name of the atmIf.
Prov
   Semantic check failed. This view cannot be
   activated.
command failed
```

The check Prov command is unsuccessful in this example because the changed option checks all the attributes and subcomponents of both new components.

Checking the provisioning data of a component and its subcomponents example

For this example, assume that you added two ATM interfaces (AtmIf/5 and AtmIf/21) without assigning them to a logical processor.

To complete a semantic check of the *AtmIf/21* component and all its subcomponents, you enter the following command:

```
check -component(AtmIf/21) Prov
```

You receive following response:

```
AtmIf/21
   Error:
      Related Component:  AtmIf/21
      Related Attribute:  interfaceName
      Reason:  Unable to retrieve the LP Info from the
               Interface Name of the atmIf.
AtmIf/21 ConnMap
   Error:
      Reason:  Unable to retrieve the LP Info from the
               Interface Name of the atmIf.
```

```
Prov
   Semantic check failed. This view cannot be
   activated.
command failed
```

The check Prov command is unsuccessful in this example because the component option used on its own recursively checks all the attributes and subcomponents of the *AtmIf/21* component.

Checking a component only example

You made provisioning changes affecting the *Lp/2* component. However, your changes did not affect any of its many provisioned subcomponents. You have more changes to make before activating the edit view. To save time, you only check the *Lp/2* component (and none of its subcomponents) using the following command:

```
check -component(Lp/2) -nonrecursive Prov
```

You receive the following response:

```
Prov
ok
```

Stop checking the provisioning data of a view after encountering an error example

To complete a semantic check of all changed components, and stop the check after the first error is encountered, you enter the following command:

```
check -stopOnError Prov
```

You receive the following response:

```
AtmIf/5
   Error:
      Related Component:  AtmIf/5
      Related Attribute:  interfaceName
      Reason:  Unable to retrieve the LP Info from the
               Interface Name of the atmIf.
Prov
   Too many semantic errors found, checking aborted.
   This view cannot be activated.
command failed
```

Related information for check Prov

See the following for information related to the check Prov command:

- “Activate Prov command” (page 48)
- “Commit Prov command” (page 96)
- “Confirm Prov command” (page 100)
- “Start Prov command” (page 287)
- “Stop Prov command” (page 300)

Clear -notificationId(<id>) -comment(<text>) aaserv command

Use the clear -notificationId(<id>) -comment(<text>) aaserv command to remove an alarm associated with a specific notification ID from the active alarm list (AAList).

For more information on the clear -notificationId(<id>) -comment(<text>) aaserv command, see

- "Properties for clear -notificationId(<id>) -comment(<text>) aaserv" (page 79)
- "Syntax for clear -notificationId(<id>) -comment(<text>) aaserv" (page 80)
- "Examples of the clear -notificationId(<id>) -comment(<text>) aaserv command" (page 81)
- "Related information for clear -notificationId(<id>) -comment(<text>) aaserv" (page 82)

Properties for clear -notificationId(<id>) -comment(<text>) aaserv

The following table describes the properties of the clear -notificationId(<id>) -comment(<text>) aaserv command:

Table 15
Properties of the clear -notificationId(<id>) -comment(<text>) aaserv command

Mode	Impact	Scope	Component
Operational	Service	Device	ActiveAlarmServices

Syntax for clear -notificationId(<id>) -comment(<text>) aaserv

The following table describes the syntax for the clear -notificationId(<id>) -comment(<text>) aaserv command.

Table 16
Syntax of the clear -notificationId(<id>) -comment(<text>) aaserv command

If you want to...	Use the following syntax:
Remove an active alarm with a specific notification ID from the active alarm list (AAList)	clear -notificationId(id) aaserv
Add an ASCII comment to the specific active alarm that is being removed from the AAList	clear -notificationId (id) -comment (text) aaserv

Examples of the clear -notificationId(<id>) -comment(<text>) aaserv command

You are user “debug”, and have two active alarms with notification IDs 082A and 082B that currently appear on the AAList:

```
Col/log Sp; 2002-06-18 15:42:56.09
SET major operator operationalCondition 00001000
ADMIN: locked OPER: enabled USAGE: idle
AVAIL: PROC: CNTRL:
ALARM: STBY: notSet UNKNW: false
Id: 082A Rel: Lp/0
Com: The component is locked
Int: 0/0/2/7470; osiState.cc; 669; VER4.1.10

Col/alarm Sp; 2002-06-18 15:42:57.33
SET major operator operationalCondition 00001000
ADMIN: locked OPER: enabled USAGE: idle
AVAIL: PROC: CNTRL:
ALARM: STBY: notSet UNKNW: false
Id: 082B Rel: Lp/0
Com: The component is locked
Int: 0/0/2/7468; osiState.cc; 669; VER4.1.10
```

To clear the 082A alarm from the AAList, enter the following command:

```
clear -notificationId(082A) aaserv
```

A response appears, stating that this alarm has been manually cleared by the user “debug”.

```
Col/alarm Sp; 2002-06-19 12:14:22.67
CLR cleared operator operationalCondition 00001000
ADMIN: unlocked OPER: enabled USAGE: active
AVAIL: PROC: CNTRL:
ALARM: STBY: notSet UNKNW: true
Id: 082A Rel: Lp/0
Com: Action by debug. This alarm has been manually
cleared.
The component is locked
Int: 0/0/2/7468; osiState.cc; 669; VER4.1.10
AAServ
ok 2002-06-19 12:14:22.68
```

The alarm is no longer listed in the AAList, and the action of clearing the alarm is logged in the standard operator log stream.

> ***Note:*** If a notification ID is not specified or does not exist, an error response is generated.

You can also add a custom comment in addition to the response that appears stating that the alarm is removed. The size of this comment is limited to 100 characters. To clear the same alarm, but this time with a custom comment attached, enter the following command:

```
clear -notificationId(082A) -comment(This is a test.)
aaserv
```

A response appears, stating that this alarm has been manually cleared by debug. In addition, the comment appears, stating the custom value that was entered (in this case, that this is a test).

```
Col/log Sp; 2002-06-19 12:15:58.46
CLR cleared operator operationalCondition 00001000
ADMIN: unlocked OPER: enabled USAGE: active
AVAIL: PROC: CNTRL:
ALARM: STBY: notSet UNKNW: true
Id: 082A Rel: Lp/0
Com: Action by debug. This alarm has been manually
cleared. This is a test.
The component is locked
Int: 0/0/2/7470; osiState.cc; 669; VER4.1.10
AAServ
ok 2002-06-19 12:15:58.47
```

As in the first example, the alarm is no longer listed in the AAList, and the manual clear (with the custom comment) is logged in the standard operator log stream.

Related information for clear -notificationId(<id>) -comment(<text>) aaserv

See the following for information related to the clear -notificationId(<id>) -comment(<text>) aaserv command:

- “Replay aaserv command” (page 239)

Clear Aps command on Passport 7400

Use the clear Aps command to remove manual overrides to the automatic selection of the active line of a link protected by automatic protection switching (APS). This command clears the effects of the protectionLockout Aps and the switch Aps commands.

An APS-protected link has two lines: working and protection. The working line normally carries the data. The protection line acts as a backup to the working line. The line that is currently carrying the data is called the active line. When no manual overrides are in effect, the APS system automatically determines the active line.

For more information on the clear Aps command, see

- "Properties for clear Aps" (page 83)
- "Syntax for clear Aps" (page 84)
- "Component instance parameter for clear Aps" (page 84)
- "Clearing APS overrides example" (page 84)
- "Related information for clear Aps" (page 84)

Properties for clear Aps

The following table describes the properties of the clear Aps command:

Table 17
Properties of the clear Aps command

Mode	Impact	Scope	Component
Operational or Provisioning	Service	Device	AutomaticProtectionSwitching (Aps)

Syntax for clear Aps

The following table describes the syntax for the clear Aps command.

Table 18
Syntax of the clear Aps command

If you want to...	Use the following syntax:
Remove all manual overrides on an APS-protected line	clear Aps/<n>

Component instance parameter for clear Aps

An Aps component instance represents a link that is protected by APS. The Aps component manages both the working and protection lines. To remove manual overrides on a particular link, you specify the instance number of the Aps component:

```
Aps/<n>
```

The instance value can be any integer between 0 and 15999.

Clearing APS overrides example

You have manually overridden the automatic selection of the active line on an APS-protected link (represented by Aps/4) using either the protectionLockout Aps command or the switch Aps command. You now want to remove the overrides. You enter the following command:

```
clear Aps/<n>
```

The APS system now automatically selects the active line for the link.

Related information for clear Aps

See the following for information related to the clear Aps command:

- “ProtectionLockout Aps command on Passport 7400” (page 218)
- “Switch Aps command on Passport 7400” (page 306)

Clear Laps command on Passport 15000 and 20000

Use the clear Laps command to remove manual overrides to the automatic selection of the active line of a link protected by line automatic protection switching (Line APS). This command clears the effects of the protectionLockout Laps and the switch Laps commands.

A Line APS-protected link has two lines: working and protection. The working line normally carries the data. The protection line acts as a backup to the working line. The line that is currently carrying the data is called the active line. When no manual overrides are in effect, the line APS system automatically determines the active line.

For more information on the clear Laps command, see

- "Properties for clear Laps" (page 85)
- "Syntax for clear Laps" (page 86)
- "Component instance parameter for clear Laps" (page 86)
- "Clearing LAPS overrides example" (page 86)
- "Related information for clear Laps" (page 86)

Properties for clear Laps

The following table describes the properties of the clear Laps command:

Table 19
Properties of the clear Laps command

Mode	Impact	Scope	Component
Operational or Provisioning	Service	Device	Line AutomaticProtectionSwitching (Laps)

Syntax for clear Laps

The following table describes the syntax for the clear Aps command.

Table 20
Syntax of the clear Laps command

If you want to...	Use the following syntax:
Remove all manual overrides on an LAPS-protected line	clear Laps/<n>

Component instance parameter for clear Laps

An Laps component instance represents a link that is protected by line APS. The Laps component manages both the working and protection lines. To remove manual overrides on a particular link, you specify the instance number of the Laps component:

```
Laps/<n>
```

The instance value can be any integer between 0 and 15999.

Clearing LAPS overrides example

You have manually overridden the automatic selection of the active line on a LAPS-protected link (represented by Laps/4) using either the protectionLockout Laps command or the switch Laps command. You now want to remove the overrides. You enter the following command:

```
clear Laps/<n>
```

The LAPS system now automatically selects the active line for the link.

Related information for clear Laps

See the following for information related to the clear Laps command:

- “ProtectionLockout Laps command on Passport 15000 and 20000” (page 221)
- “Switch Laps command on Passport 15000 and 20000” (page 313)

Clear Nmis <interface> Session command

The clear Nmis <interface> Session command lets you terminate an individual user session and close its TCP connection. You can terminate user sessions on the telnet, FTP, and FMIP network management interfaces. For information on these interfaces, see 241-5701-045 *Passport 7400, 15000, 20000 Management System User Interface Guide*.

If you clear a telnet session which has an outgoing client connection, the client connection also terminates.

For more information on the clear Nmis <interface> Session command, see

Properties for clear Nmis <interface> Session

The following table describes the properties of the clear Nmis <interface> Session command.

Table 21
Properties of the clear Nmis <interface> Session command

Mode	Impact	Scope	Components
Operational or Provisioning	System Administration	Application	NetworkManagementInterfaceSystem Telnet Session (Nmis Telnet Session) NetworkManagementInterfaceSystem Fmip Session (Nmis Fmip Session) NetworkManagementInterfaceSystem Ftp Session (Nmis Ftp Session)

Syntax for clear Nmis <interface> Session

The following table describes the syntax for the clear Nmis <interface> Session command.

Table 22
Syntax of the clear Nmis <interface> Session command

If you want to...	Use the following syntax:
Terminate a Telnet user session	clear Nmis Telnet Session/<n>
Terminate an FMIP user session	clear Nmis Fmip Session/<n>
Terminate an FTP user session	clear Nmis Ftp Session/<n>

Session parameters for clear Nmis <interface> Session

You specify which user session you want to terminate using the Session subcomponents of the *NetworkManagementInterfaceSystem* (*Nmis*) component.

An instance of a *Session* component represents each user session. If you display the *Session* subcomponents of a network management interface, you get information on all the current sessions. For example, the following command displays the current FTP sessions:

```
display Nmis Ftp Session/*
```

Table 23
Session parameters for the clear Nmis <interface> Session command

Component	Description
Nmis	Always enter this component.
Telnet Session/<n>	If you want to clear a telnet user session, enter the number of the telnet user session. If the telnet session has an outgoing client connection, the client connection also terminates.
(Sheet 1 of 2)	

Table 23 (continued)
Session parameters for the clear Nmis <interface> Session command

Component	Description
Fmip Session/<n>	If you want to clear an FMIP user session, enter the number of the FMIP user session.
Ftp Session/<n>	If you want to clear an FTP session, enter the number of the FTP user session.
(Sheet 2 of 2)	

Clearing a session example

You are the system administrator and you want to terminate a telnet user session. To determine the instance number of the session, you display all current telnet user sessions using the following command:

```
display Nmis Telnet Session/*
```

You receive the following response:

```
Nmis Telnet Session/*
+=======+--------+----------+---------------+-----+
|Session| userid |dataStr   |    address    |port |
|       |        | eams     |               |     |
|       |        |a d l s r |               |     |
|       |        |l e o c t |               |     |
|       |        |a b g n s |               |     |
+=======+--------+----------+---------------+-----+
|      1|maryb   |* . . .  .|132.0.1.180    |36135|
|      2|stevd   |* . . . . |132.0.1.181    |36136|
```

From this display, you determine that you want to terminate session 2. You enter the following command:

```
clear Nmis Telnet Session/2
```

Related information for clear Nmis <interface> Session

For more information related to the clear Nmis <interface> Session command see the following:

- “Display command” (page 121)
- “Logout command” (page 182)
- “Me command” (page 186)

Clear Prov command

Use the clear Prov command to remove nonpermanent components and their subcomponents from the edit view. This command does not affect permanent components (components that you cannot delete).

CAUTION
Possible loss of provisioning data
The clear Prov command deletes all nonpermanent components from the edit view. If you have not saved your provisioning data in a saved view before using the clear Prov command, you will lose your provisioning data.

For more information on the clear Prov command, see

- "Properties for clear Prov" (page 91)
- "Syntax for clear Prov" (page 92)
- "Deleting the components of a removed feature parameter for clear Prov" (page 92)
- "Examples for clear Prov" (page 93)
- "Related information for clear Prov" (page 95)

Properties for clear Prov

The following table describes the properties of the clear Prov command:

Table 24
Properties of the clear Prov command

Mode	Impact	Scope	Components
Provisioning	System Administration	Application	ProvisioningSystem (Prov)

Syntax for clear Prov

The following table describes the syntax for the clear Prov command.

Table 25
Syntax of the clear Prov command

If you want to...	Use the following syntax:
Remove all nonpermanent components and their subcomponents from the edit view	clear Prov
Remove all nonpermanent components and their subcomponents associated with a removed feature from the edit view	clear -removedFeature Prov

Deleting the components of a removed feature parameter for clear Prov

You can specify that the provisioning system remove only the components associated with a removed feature using the removedFeature option:

```
-removedFeature
```

Use this option when you have removed one or more features from one or more LPTs. It specifies that the system remove from the edit view all components, both permanent and nonpermanent, which have had their associated software unloaded.

Examples for clear Prov

The following examples show how to use the clear Prov command:

- “Clearing all nonpermanent components from the edit view example” (page 93)
- “Clearing components associated with a removed feature from the edit view example” (page 93)

Clearing all nonpermanent components from the edit view example

You want to clear all nonpermanent components from the edit view. First, to display all the top-level components, you enter the following command:

```
list
```

You receive the following response:

```
FrUni/1
Sw
Shelf
```

To delete the nonpermanent components, you enter the following command:

```
clear Prov
```

You issue the list command again.

```
list
```

You receive the following response:

```
Sw
Shelf
```

Clearing components associated with a removed feature from the edit view example

This example shows how to clear all components for a removed feature. Assume that you have removed the frameRelayUni (FrUni) feature software from the feature list of all LPs.

To display all the top-level components, you enter the following command:

```
list
```

You receive the following response:

```
Col/*
Sw
Shelf
Lp/*
Rtg
Mod
Ac
Npi/*
FrUni/*
```

To clear the components for a removed feature, you enter the following command:

```
clear -removedFeature Prov
```

To display all the top-level components, you enter the following command:

```
list
```

You receive the following response:

```
Col/*
Sw
Shelf
Lp/*
Rtg
Mod
Ac
Npi/*
```

Related information for clear Prov

See the following for information related to the clear Prov command:

- “Copy Prov command” (page 111)
- “Delete command” (page 118)
- “List command” (page 159)
- “Start Prov command” (page 287)

Commit Prov command

Use the commit Prov command to set the current view of the provisioning data as the committed view. This activity can entail saving the current view in portable or commit formats.

The committed view is the configuration to which the node defaults following a CP reset, restart, or switchover. The following conditions must be met before you can commit the current view:

- You must store the current view in a saved view (See “Save Prov command” (page 264)) prior to committing it. If you have not previously saved the current view with a unique filename, you can both save and commit it by using the file option (See “Specifying a file name parameter for commit Prov” (page 98)) to specify a filename for the view.
- In a dual-CP node, disks must be synchronized.

In a dual-CP node, the commit Prov command can cause a reload of the standby CP.

Use the commit Prov command after successfully issuing the activate Prov and confirm Prov commands. If you issue the commit Prov command before the confirm Prov command, the system automatically confirms the view before committing the view. If you issue the commit Prov command from operational mode, it fails if another operator session is in provisioning mode.

For more information on the commit Prov command, see

Properties for commit Prov

The following table describes the properties of the commit Prov command.

Table 26
Properties of the commit Prov command

Mode	Impact	Scope	Components
Operational or Provisioning	Service	Application	ProvisioningSystem (Prov)

Syntax for commit Prov

The following table describes the syntax for the commit Prov command.

Table 27
Syntax of the commit Prov command

If you want to...	Use the following syntax:
Make the current view the committed view	commit Prov
Specify a file name for the committed view	commit -file(<view_name>) Prov

Parameters for commit Prov

There are two types of parameters for the commit Prov command: .

- “Force override parameter when restore is possible for commit Prov” (page 97)
- “Specifying a file name parameter for commit Prov” (page 98)

Force override parameter when restore is possible for commit Prov

If the journaled current view can be restored, then the commit Prov command will cause the journaled current view to be purged, and a restore will no longer be possible. In this case, the -force option is required for the commit Prov command to proceed. Note that a restore is only possible if the attribute *restorePossible* is set to yes.

Specifying a file name parameter for commit Prov

You can specify a name for the committed view using the file option:

```
-file(<view_name>)
```

Enter a name for the saved version of the current view. For information on filename syntax, refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview*.

Examples for commit Prov

The following examples show how to use the commit Prov command:

- “Committing the current view example” (page 98)
- “Specifying a filename for the committed view example” (page 98)

Committing the current view example

You want to make the current view of the provisioning data the committed view. For this example, the current view is PPS.full.001. You enter the following:

```
commit Prov
```

You receive the following response:

```
Prov
   Saving the current view into PPS.full.001 (with
   commit,portable formats) ...
Prov
   The committed file is PPS.full.001.
ok
```

Specifying a filename for the committed view example

You want to specify a filename (for this example, commit1) to save the committed view. You enter the following:

```
commit -file(commit1) Prov
```

You receive the following response:

```
Prov
   Saving the current view into commit1.full.001 (with
   commit,portable formats)
```

```
   ...
Prov
   The committed file is commit1.full.001.
ok
```

Related information for commit Prov

See the following for information related to the commit Prov command:

- “Activate Prov command” (page 48)
- “Confirm Prov command” (page 100)
- “Save Prov command” (page 264)

Confirm Prov command

Issue the confirm Prov command after the activate Prov command to activate the edit view. After the confirmation is complete, the current view is the same as the edit view.

You must confirm an the activation of a new view within 20 minutes. After 20 minutes without confirmation, the node restarts using the last committed view. Passport generates a warning alarm every five minutes until you enter the confirm Prov command.

CAUTION
If you do not confirm the activate Prov command within 20 minutes of issuing it (see “Activate Prov command” (page 48)) a system restart occurs and the system reactivates the last committed view (see “Commit Prov command” (page 96)). A service disruption occurs while the restart is in progress.

You can use the confirm Prov command in either provisioning or operational mode. If you are in operational mode when you issue the confirm Prov command, and another operator is in provisioning mode, the command fails.

For more information on the confirm Prov command, see

- “Properties for confirm Prov” (page 101)
- “Syntax for confirm Prov” (page 101)
- “Confirming the activation of a new view into the current view example” (page 101)
- “Related information for confirm Prov” (page 102)

Properties for confirm Prov

The following table describes the properties of the confirm Prov command.

Table 28
Properties of the confirm Prov command

Mode	Impact	Scope	Components
Operational or Provisioning	Service	Application	ProvisioningSystem (Prov)

Syntax for confirm Prov

The following table describes the syntax for the confirm Prov command.

Table 29
Syntax of the confirm Prov command

If you want to...	Use the following syntax:
Confirm a new provisioning view into the current view, after issuing the activate Prov command	confirm Prov

Confirming the activation of a new view into the current view example

You want to confirm activation of a new provisioning view into the current view. You enter the following command:

```
confirm Prov
```

You receive the following response:

```
    Prov; 1998-11-20 14:10:47.49
CLR cleared operator
operationalCondition         70000007
   ADMIN: un

ed      OPER: enabled       USAGE: busy
   AVAIL:               PROC: reporting     CNTRL:
   ALARM:             STBY: notSet       UNKNW: false
   Id: 0F000021  Rel: Lp/0
   Com: Activation confirmed. Rollback will not occur.
   Int: 15/1/3/63783;
casResolverVerbProvConfirm.cc;171; p4.0d.27
Prov
```

If you enter the confirm Prov command before the activate Prov command, you receive the following response:

```
Prov
  You must activate the edit view first, then confirm.
command failed
```

This message indicates that the confirm Prov command failed because you did not activate the new provisioning view.

Related information for confirm Prov

See the following for information related to the confirm Prov command:

- “Activate Prov command” (page 48)
- “Commit Prov command” (page 96)

Continue Prov command

The continue Prov command enables a hitless software upgrade that has been paused to continue. When issued, the continue Prov command causes the migration shelf to reset and the migration switchover to continue.

For more information on the continue Prov command, see

Properties for continue Prov

The following table describes the properties of the continue Prov command.

Table 30
Properties of the continue Prov command

Mode	Impact	Scope	Components
Provisioning	Configuration	Application	ProvisioningSystem (Prov)

Syntax for continue Prov

The following table describes the syntax for the continue command.

Table 31
Syntax of the continue command

If you want to...	Use the following syntax:
Restart a paused software migration activation	continue Prov
Force the migration activation to continue by resetting the shelf	continue -force Prov

Force parameter for continue Prov

You can specify that the hitless software migration activation continues by using the force option. The force option causes the migration shelf to reset and the migration switchover to continue.

Examples for continue Prov

The following examples illustrate how to use the continue Prov command:

- “Continuing a hitless software upgrade” (page 104)
- “Forcing a hitless software upgrade to continue” (page 105)

Continuing a hitless software upgrade

If an application does not behave properly during a migration switchover, then the hitless software upgrade automatically pauses before migration switchover occurs. This pause allows you to review all visible migration alarms before continuing or stopping the hitless software upgrade. To continue the hitless software upgrade, issue the following command:

```
continue Prov
```

Forcing a hitless software upgrade to continue

If you have configured a spare CP but it is unavailable, the migration activation fails. To force the migration activation to continue, reset the shelf by issuing the following command:

```
continue -force Prov
```

Note: If the migration pauses because of a fault, or if a fault occurs during a pause, you must issue the continue -force prov command if you wish to continue despite the fault condition.

Related information for continue Prov

See the following for information related to the continue Prov command:

- "Activate Prov command" (page 48)
- "Apply Prov command" (page 60)
- "Check Prov command" (page 71)
- "Clear Prov command" (page 91)
- "Commit Prov command" (page 96)
- "Confirm Prov command" (page 100)
- "Continue Prov command" (page 103)
- "Copy Prov command" (page 111)
- "End Prov command" (page 135)
- "Load Prov command" (page 173)
- "Save Prov command" (page 264)
- "Start Prov command" (page 287)
- "Stop Prov command" (page 300)
- "Tidy Prov command" (page 335)

Copy Fs command

The copy Fs (cp Fs) command lets you copy files or directories on the Passport file system. You can copy an individual source file or a source directory and all its subdirectories to a destination on the file system.

The copy Fs command does not overwrite existing files. If you specify an existing file as a destination, the command fails. If you want to overwrite a file, you must first remove it using the remove Fs command.

When you copy a protected file, the new file is not protected.

When you use this command, avoid copying files into the root directory (/). Reserve the root directory for directories only.

For more information on the copy Fs command, see

- "Properties for copy Fs" (page 106)
- "Syntax for copy Fs" (page 107)
- "Parameters for copy Fs" (page 107)
- "Examples for copy Fs" (page 108)
- "Related information for copy Fs" (page 110)

Properties for copy Fs

The following table describes the properties of the copy Fs command.

Table 32
Properties of the copy Fs command

Mode	Impact	Scope	Components
Operational or Provisioning	Configuration	Device	FileSystem (Fs)

Syntax for copy Fs

The following table describes the syntax for the copy command.

Table 33
Syntax of the copy command

If you want to...	Use the following syntax:
Copy a file	copy -source(<filepath>) -destination(<filepath>) Fs
Copy a file into a directory	copy -source(<filepath>) -destination(<directorypath>) Fs
Copy a directory and all its contents (including all subdirectories) into a directory	copy -source(<directorypath>) -destination(<directorypath>) -recursive Fs

Parameters for copy Fs

There are two types of parameters for the copy Fs command:

- “Source and destination parameters for copy Fs” (page 107)
- “Copy subdirectories parameter for copy Fs” (page 108)

Source and destination parameters for copy Fs

You can specify which file or directory you want to copy using the source option. You can specify where you want to put the copy using the destination option.

For information on path syntax, refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview*.

Table 34
Source and destination options for the copy Fs command

Option	Description
-source (<filepath>) -source (<directorypath>)	Enter the absolute or relative path to the source file or directory. If you specify the path of a directory, the destination must be also be a directory and you must use the recursive option. You must enter this option.
-destination (<filepath>) -destination (<directorypath>)	Enter the absolute or relative path to the destination file or directory. If you are copying a file to directory, the directory must already exist on the file system. You must enter this option.

Copy subdirectories parameter for copy Fs

You can specify that you want to copy all the subdirectories of a directory using the recursive option:

```
-recursive
```

When you use the recursive option, Passport creates the complete subdirectory hierarchy of the source directory in the destination directory and copies all the files into the destination directory hierarchy.

If the destination directory exists, Passport creates a directory in it with the same name as the source and copies all the contents of the source directory into that new directory. If the destination directory does not exist, Passport creates it and copies all the contents of the source directory into it.

Examples for copy Fs

The following examples illustrate how to use the copy Fs command:

- “Copying a file example” (page 109)
- “Copying a directory and its subdirectories example” (page 109)

Copying a file example

Your current working directory is /spooled/closed/account. You want to copy the file account.old in the current working directory into the /tmp directory. You enter the following command:

```
copy -source("account.old") -destination("/tmp") Fs
```

You now want to copy that same file into the /spooled/closed directory, which is one directory above your current working directory. You enter the following command:

```
copy -source("account.old") -destination("..") Fs
```

Copying a directory and its subdirectories example

Your current working directory is /spooled. You want to copy all the contents of the closed directory, including all subdirectories into the /tmp directory. You enter the following command:

```
copy -source("closed") -destination("/tmp") -recursive
Fs
```

The/ tmp directory now has a closed subdirectory with the same contents as the /spooled/closed directory.

You then decide that you want to copy the same closed directory into a new, directory called /closedtmp. You enter the following command:

```
copy -source("closed") -destination("/closedtmp")
-recursive Fs
```

There is now a new /closedtmp directory with the same contents of the / spooled/closed directory.

Related information for copy Fs

See the following for information related to the copy Fs command:

- “ListFile Fs command” (page 168)
- “MakeDirectory Fs command” (page 184)
- “Move Fs command” (page 188)
- “Remove Fs command” (page 232)
- “WorkingDir Fs command” (page 366)

Copy Prov command

Use the copy Prov command to copy the entire current view into the edit view, to copy individual components from the current view into the edit view, or to copy components within the edit view alone.

The following are characteristics of the copy Prov command:

- If you use the command alone with no options, to copy a full view, the system copies all components and related data such that the edit view is identical to the current view.
- If you use the command to copy a component or partial view (edit view to edit view or current view to edit view), the system can change data so that it does not conflict with existing data (for example, DNAs, passwords, and all link attributes). In situations where the system does change data, a warning appears to specify that the system has modified values.

For more information on the copy Prov command, see

- "Properties for copy Prov" (page 111)
- "Syntax for copy Prov" (page 112)
- "Parameters for copy Prov" (page 113)
- "Examples for copy Prov" (page 114)
- "Related information for copy Prov" (page 117)

Properties for copy Prov

The following table describes the properties of the copy Prov command.

Table 35
Properties of the copy Prov command

Mode	Impact	Scope	Components
Provisioning	Configuration	Application	ProvisioningSystem (Prov)

Syntax for copy Prov

Copy Prov allows you to copy entire views or partial views from both the current view and the edit view into the edit view. The view from which you copy is the source view.

The default source view depends on the options you specify:

- If you do not use any options, the default source view is the current view, and the system copies the complete current view into the edit view. This is useful if you have made changes to the edit view but for whatever reason wish to return to the pre-change configuration.
- If you use the source option only, the default source view is the current view.
- If you use both the source and the destination options, the default source view is the edit view. However, you can use the current option to change the default source view to the current view.

The following table describes the syntax for the copy Prov command.

Table 36
Syntax of the copy Prov command

If you want to...	Use the following syntax:
Copy the current view into the edit view	copy Prov
Copy a component and its subcomponents into the edit view	copy -source(<component_name>) Prov
Copy a component from the edit view and save it under another name, also within the edit view	copy -source(<component_name>) -destination(<component_name>) Prov
(Sheet 1 of 2)	

Table 36 (continued)
Syntax of the copy Prov command

If you want to...	Use the following syntax:
Copy a component from the current view and save it under another name in the edit view	copy -source(<component_name>) -current -destination(<component_name>) Prov
Copy a component without its subcomponents from the current view to the edit view	copy -source(<component_name>) -nonrecursive Prov
(Sheet 2 of 2)	

Parameters for copy Prov

There are four types of parameters for the copy Prov command:

- "Copying a component parameter for copy Prov" (page 113)
- "Copying and renaming a component parameter for copy Prov" (page 114)
- "Copying from the current view parameter for copy Prov" (page 114)
- "Nonrecursive parameter for copy Prov" (page 114)

Copying a component parameter for copy Prov

You can specify that the provisioning system copy a component into the edit view using the source option:

```
-source(<component_name>)
```

The source option causes the system to copy the identified component from either the edit view or the current view, depending on which option or combination of options you use, as follows:

- If you use only the source option, the current view is the source view.

- If you use the destination option in combination with the source option, the edit view is the source view.
- If you use the destination and current options in combination with the source option, the current view is the source view.

If there are any conflicts between component names in the edit view, then the copy fails and no changes occur to the edit view.

Copying and renaming a component parameter for copy Prov

You can specify that the provisioning system copy a component into the edit view and rename it using the destination option:

```
-destination(<component_name>)
```

This option is valid only if you use the source option.

Copying from the current view parameter for copy Prov

You can specify that the provisioning system copy from the current view using the current option:

```
-current
```

The current option causes the system to change the default source view to the current view when copying a component.

This option is valid only if you use the source and destination options.

Nonrecursive parameter for copy Prov

You can specify that the provisioning system copy into the edit view a component, but not its subcomponents, using the nonrecursive option:

```
-nonrecursive
```

This option is valid only if you use the source option.

Examples for copy Prov

The following section contains examples that show how to use the copy Prov command:

- “Copying the current view into the edit view example” (page 115)

Copying the current view into the edit view example

You want to copy the current view into the edit view, replacing the edit view. You enter the following command:

```
copy Prov
```

You receive the following response:

```
Prov
ok
```

Copying a component into the edit view example

The edit view requires a *Vr* component. You want to copy the *Vr/1* component and its subcomponents from the current view into the edit view. You enter the following command:

```
copy -source(Vr/1) Prov
```

You receive the following response:

```
Prov
ok
```

To list all of the *Vr* components, you enter the following command:

```
list Vr/*
```

You receive following response:

```
Vr/1
ok
```

Copying and renaming a component within the edit view example

Use the copy Prov command to copy the *Vr/1* component from the edit view to a new component under a different name, also in the edit view. You enter the following command:

```
copy -source(Vr/1) -destination(Vr/2) Prov
```

You receive the following response:

```
Prov
ok
```

To list all of the *Vr* components, you enter the following command:

```
list Vr/*
```

You receive following response:

```
Vr/1
Vr/2
ok
```

You have now created the *Vr/2*, with the identical configuration as *Vr/1*. This procedure is useful if you provision a component in a certain manner and want to duplicate it exactly as another instance of the same component.

Copying a component from the current view to the edit view, and renaming it in the edit view example

Use the copy Prov command to copy the *Vr/1* component from the current view to a new component under a different name, in the edit view. You enter the following command:

```
copy -source(Vr/1) -current -destination(Vr/2) Prov
```

You receive the following response:

```
Prov
ok
```

To list all of the *Vr* components, you enter the following command:

```
list Vr/*
```

You receive following response:

```
Vr/2
ok
```

You have now created the *Vr/2* component, with the identical configuration as the *Vr/1* component in the current view.

Copying a component but not its subcomponents example

Use the copy Prov command to copy the *Vr/1* component, but not its subcomponents, from the current view to the edit view. You enter the following command:

```
copy -source(Vr/1) -nonrecursive Prov
```

You receive the following response:

```
Prov
ok
```

Note: The system automatically creates the required subcomponents with default provisioning data when you use this option.

Related information for copy Prov

See the following for information related to the copy Prov command:

- “Add command” (page 56)
- “Clear Prov command” (page 91)
- “Start Prov command” (page 287)

Delete command

Use the delete (del) command to remove a component (and its subcomponents) from the edit view. You can only use the delete command in provisioning mode.

The system also removes all provisioned links between the deleted component and any other component.

When you activate the edit view, Passport removes the deleted components and subcomponents from the running system. This process terminates the service provided by these components and their subcomponents.

> ***Note:*** You can use the copy Prov command to undo an accidental delete. See “Copy Prov command” (page 111).

For more information on the delete command, see

- “Properties for delete” (page 118)
- “Syntax for delete” (page 119)
- “Examples for delete” (page 119)
- “Related information for delete” (page 120)

Properties for delete

The following table describes the properties of the delete command.

Table 37
Properties of the delete command

Mode	Impact	Scope	Components
Provisioning	Configuration	Depends on component	All

Syntax for delete

The following table describes the syntax for the delete command.

Table 38
Syntax of the delete command

If you want to...	Use the following syntax:
Delete a component and all its associated subcomponents from the edit view	delete <component_name>

Examples for delete

The following examples show how to use the delete command:

- “Deleting a component example” (page 119)
- “Deleting a permanent component example” (page 119)

Deleting a component example

You want to delete a *LogicalProcessor* (*Lp*) component (*Lp/2* for this example). You enter the following command:

```
delete Lp/2
```

You receive the following response:

```
Lp/2
A total of 2 components have been deleted.
ok
```

Deleting a permanent component example

You want to delete the permanent *Shelf* component. You enter the following command:

```
delete Shelf
```

You receive the following response:

```
Shelf
The 'Delete' command is not allowed for this component.
command failed
```

The command failed because you cannot remove some required components using the delete command. To remove a required component, see "Clear Prov command" (page 91).

Related information for delete

See the following for information related to the delete command:

- "Add command" (page 56)
- "Clear Prov command" (page 91)
- "Copy Prov command" (page 111)
- "Start Prov command" (page 287)

Display command

The display (d) command lets you view the attributes of one or more components. You can display the operational and provisionable attributes of both the current and edit view, in either list or tabular format.

When you are in operational mode, the display command assumes you want to view operational attributes in the current view. When you are in provisioning mode, the display command assumes you want to view provisionable attributes in the edit view. The command uses these default associations, which are summarized in “Properties of the display command” (page 122), whenever possible. You can override the associations using command options.

By default, attributes are displayed in tabular format. If the number of attributes requested does not fit in a table, Passport only shows the most useful attributes. To see all the attributes, use the list format of display command.

The display command supports all command wildcarding capabilities. See “Command wildcarding” (page 39).

For more information on the display command, see

Properties for display

The display command works with all components. Some attributes require a higher impact than passive (defined by their read access value) to display. To find specific impact and scope information, see 241-5701-060 *Passport 7400, 15000, 20000 Components* or use the help command.

The following table describes the properties of the display command.

Table 39
Properties of the display command

Mode	Impact	Scope	Attribute	View
Operational	Passive	Depends on component	Operational	Current
Provisioning	Passive	Depends on component	Provisionable	Edit

Syntax for display

The following table describes the syntax for the display command.

Table 40
Syntax of the display command

If you want to...	Use the following syntax:
Display provisionable or operational attributes	display <component_name>
Display attribute values type	display <component_name>*
Display all top-level component types	display
Display operational attributes while in provisioning mode	display -operationalData<component_name>
Display provisionable attributes while in operational mode	display -provisionedData <component_name>
Display attribute values in the current view	display -current <component_name>

Parameters for display

The following items are the parameters of the display command:

Component parameter for display

If you only enter a component as a parameter to the display command, you get all attributes of the type and view associated with the current command mode (see "Properties of the display command" (page 122) for the associations). If, however, you specify an operational component while in provisioning mode, you get attributes from the current view. (Operational information does not exist in the edit view.) Passport informs you that it is displaying the current view and not the associated edit view.

You can use wildcarding to view attributes from more than one component. With type wildcarding, you can substitute the asterisk (*) wildcard character for the last component type-instance pair to view attributes from all subcomponents. The asterisk must be the last character of the command.

With instance wildcarding, you can substitute a wildcard (*) or a wildcard pattern for an instance value of a component. If you are displaying a component with multiple indexes, you can specify the value for a particular index, or you can wildcard one or more of its index values. Any unspecified index values will default to a wildcard (*).

Table 41
Components for the display command

Option	Description
<component_name>	Enter the name of the component.
<component_type>/*	Use an asterisk for an instance value to view attributes from all instances of a component type.
<component_name> *	Use an asterisk as the last component type-instance pair to view attributes from all subcomponents of a component. A table displays for each subcomponent.
<component_type>/ <index>, *, ...	Use an asterisk for one or more index values to view attributes from all instances of a multi-indexed component. At least the first index must be specified. Unspecified indices default to *.
<component_type>/* <component_type>/* ...	Use an asterisk for more than one instance value to view attributes from all instances of multiple components in the hierarchy. Only wildcarded instances appear as columns in the table.

Attribute and group parameters for display

You can specify exactly which attribute or group you want to view using the attribute and group parameters. When you specify an attribute or group, it displays even if its type conflicts with the current command mode. You can, for example, specify that you want to view a provisionable attribute while in operational mode. That attribute displays from the current view.

If, however, you specify an operational attribute while in provisioning mode, you get information from the current view since there are no operational attributes in the edit view. In this case, Passport informs you that it is displaying information from the current view and not the associated edit view.

You can specify attributes and groups in combination with instance wildcarding only.

Table 42
Attributes and groups for the display command

Option	Description
<attribute>	Enter the name of the attribute. You can enter a list of attributes or groups, separated by commas (,) to view many attributes.
<group>	Enter the name of the group. You can enter a list of attributes or groups, separated by commas (,) to view many attributes.

Override default parameters for display

You can override the default attribute type and view associations using the operationalData, provisionedData, and current options.

Table 43
Attribute type options for the display command

Option	Description
-operationalData (-o)	Use this option to view operational attributes while in provisioning mode. Attributes from the current view display since there are no operational attributes in the edit view.
-provisionedData (-p)	Use this option to view provisionable attributes while in operational mode.
-current (-c)	Use this option to view attributes in the current view when in provisioning mode.

Display format parameter for display

When you use wildcards to specify a component, the default format is tabular. The first row of the tabular output labels the components and attributes in the table. The component columns use double horizontal ruling (===) for their label, while attribute columns use single horizontal ruling (---). The remaining rows contain the attribute values for each component instance matched by wildcarding.

Passport shows as many attributes as fit within the screen width. If all the attributes do not fit in the table, you can change to list format using the noTabular option:

```
-noTabular
```

You can abbreviate the option as follows:

```
-nTabular
```

Simple attribute value parameters for display

Simple attribute value parameters consist of an expression that contains an attribute, a relational operator, and a value. A relational operator can be equal (=) or not equal (!=). Parentheses can be used, but are optional. See “Displaying components using simple filtering example” (page 132) for an example of simple attribute value parameters.

Complex attribute value parameters for display

Complex attribute value parameters are formed when simple attribute value parameters are joined using the following logical operators: AND and OR. A maximum of three simple expressions can be joined. Parentheses can be used, but are optional. See “Displaying components using complex filtering example” (page 133) for an example of complex attribute value parameters.

Applying both instance and attribute value parameters

You can combine both instance filtering and attribute value parameters to the same component instance. The format is:

```
<ComponentClass>/<instance>(attribute value)
```

Instance can be a wildcard (*) or a wildcard pattern. See “Displaying components using both instance and attribute value filtering” (page 133) for an example.

Examples for display

The following examples illustrate how to use the display command:

- “Displaying a component example” (page 127)
- “Displaying attributes and groups example” (page 128)
- “Displaying a wildcarded instance example” (page 128)

Displaying a component example

You are currently in operational command mode and you want to view the operational attributes of the shelf. You enter the following command:

```
display Shelf
```

You receive the following information from the current view:

```
Shelf
   busOperatingMode = dualBus
   hardwareFailures =
   numberOfSlots    = 16
   revertibleTimerCountdown = 0 seconds
```

You are now in provisioning mode and have been provisioning logical processor 4. You want to view the provisionable attributes of this logical processor, so you enter the following command:

```
display Lp/4
```

You receive the following information from the edit view:

```
Lp/4
   mainCard = Shelf Card/4
   spareCard =
   logicalProcessorType = Sw Lpt/ATM
   linkToApplications   =
   customerIdentifier   = 0
```

Displaying attributes and groups example

You are in operational mode and you want to view the *activeCard* attribute for logical processor 0. You enter the following command:

```
display Lp/0 activeCard
```

You receive the following information from the current view:

```
Lp/0
   activeCard = Shelf Card/0
```

You now want to also see, along with the *activeCard* attribute, all the attributes in the Provisioned group (*mainCard*, *spareCard*, *logicalProcessorType*, and *linkToApplications*). You enter the following command:

```
display Lp/0 activeCard, Provisioned
```

You receive the following information from the current view:

```
Lp/0
   activeCard = Shelf Card/0
   mainCard   = Shelf Card/0
   spareCard  = Shelf Card/15
   logicalProcessorType = Sw Lpt/CP
   linkToApplications   =
```

Displaying a wildcarded instance example

You are currently in operational command mode and you want to view the operational attributes of all the collectors, which are represented by instances of the *Collector* component. You enter the following command:

```
display Collector/*
```

You receive the following information from the current view:

```
Col/*
+=====+----------+----------+----------
| Col |currentQue|recordsRx |recordsDis
|     |  ueSize  |          |  carded
+=====+----------+----------+----------
|acc  |         0|         0|         0
|ala  |         0|        32|         0
|log  |         0|         0|         0
```

```
|deb  |         0|         0|         0
|scn  |         0|         0|         0
|trap |         0|         0|         0
|stats|         0|         0|         0
|rts  |         0|         0|         0
```

You now want to view the *recordsRx* attribute of all *Agent* components, which are subcomponents of *Collector*. You enter the following command:

```
display Collector/* Agent/* recordsRx
```

You receive the following information from the current view:

```
Col/* Ag/*
+=======+==+----------+
| Col   |Ag|recordsRx |
|       |  |          |
+=======+==+----------+
|acc    | 0|         0|
|acc    | 1|         0|
|ala    | 0|        32|
|ala    | 1|         0|
|log    | 0|         2|
|deb    | 0|         0|
|deb    | 1|         0|
|scn    | 0|         0|
|scn    | 1|         0|
|trap   | 0|         0|
|trap   | 1|         0|
|stats  | 0|         0|
|stats  | 1|         0|
|rts    | 0|         0|
|rts    | 1|         0|
```

Displaying a wildcarded type example

You are in provisioning mode and you want to view the provisionable attributes of all user IDs, which are represented by *Userid* subcomponents of *AccessControl*. You enter the following command:

```
display AccessControl *
```

You receive the following information from the edit view:

```
Ac Userid/*
+========+----+------+------+-------+----------------
| Userid |cid |scope |impact| nmifs |      dir
|        |    |      |      |  t    |
|        |    |      |      |l e    |
|        |    |      |      |o l f  |
|        |    |      |      |c n m f|
|        |    |      |      |a e i t|
|        |    |      |      |l t p p|
+========+----+------+------+-------+----------------
|ADMIN   |   0|networ|system|* * * *|/u/admin
|USER1   |   0|networ|config|* * . *|/u/user1
|USER2   |   0|networ|config|* * . *|/u/user2
|USER3   |   0|networ|servic|* * . *|/u/user3
|USER4   |   0|networ|servic|* * . *|/u/user4
|USER5   |   0|networ|servic|* * . *|/u/user5
```

Overriding defaults example

You are in provisioning mode and you want to view the operational attributes of the *Shelf* component, which exist only in the current view. You enter the following command:

```
display -operationalData Shelf
```

You receive the following information from the current view:

```
Shelf
   busOperatingMode = dualBus
   hardwareFailures =
   numberOfSlots    = 16
   revertibleTimerCountdown = 0 seconds
```

While still in provisioning mode, you want to view the provisionable attributes of the *Shelf* component in the current view, not the edit view. You enter the following command:

```
display -current Shelf
```

You receive the following information from the current view:

```
Shelf
   commentText = "Lab 2, H16"
```

Now in operational mode, you want to view the provisionable attributes of the *Shelf* component. You enter the following command:

```
display -provisionedData Shelf
```

You receive the following information from the current view:

```
Shelf
   commentText = "Lab 2, H16"
```

Back in provisioning mode, you want to view the operational attribute *busOperatingMode* of *Shelf*, which exists only in the current view. You enter the following command:

```
display Shelf busOperatingMode
```

You receive the following information from the current view:

```
Shelf
   busOperatingMode = dualBus
   All information displayed is from the current view
```

Displaying in list format example

While in provisioning mode, you want to view attributes of all logical processors in a table format. You enter the following command:

```
display Lp/*
```

You receive the following information from the edit view:

```
Lp/*
 Use -noTabular to see hidden attributes: cid
 and linkToApplications
+==+----------------+--------------+-------------
|Lp|     main       |     spare    |    lpt
+==+----------------+--------------+--------
| 0|Shelf Card/0    |!             |Sw Lpt/CP
| 1|Shelf Card/1    |Shelf Card/2  |Sw Lpt/FP
```

You now want to see the same information in list format to see the hidden attributes. You enter the following command:

```
display -noTabular Lp/*
```

You receive the following information from the edit view:

```
Lp/0
   mainCard = Shelf Card/0
   spareCard =
   logicalProcessorType = Sw Lpt/CP
   linkToApplications   =
   customerIdentifier   = 0
Lp/1
   mainCard = Shelf Card/1
   spareCard = Shelf Card/2
   logicalProcessorType = Sw Lpt/FP
   linkToApplications   =
   customerIdentifier   = 0
```

Displaying a multi-indexed component example

In provisioning mode, you want to view the *preferredOver* attribute of all *Ip Static Route* components with a subnetwork mask of 255.255.255.0 for virtual router 1. The subnetwork mask is the second index of the *Route* component. You enter the following command:

```
display Vr/1 Ip Static Route/*,255.255.255.0,*
preferredOver
```

You receive the following information from the edit view:

```
Vr/1 Ip Static Route/*,255.255.255.0,*
+===============+===+------
|     addr      |tos| pref
+===============+===+------
|27.64.144.0    |  0|extOsp|
|27.104.160.0   |  0|extOsp|
|27.187.32.0    |  0|extOsp|
|27.208.128.0   |  0|extOsp|
```

Displaying components using simple filtering example

You are currently in operational command mode and you want to display the operational components of the cards on the shelf that are not CPs. You enter the following command:

```
display shelf card/(cardTyp7= CP)
```

You receive the following information from the current view:

```
Shelf Card/*
Use -noTabular to see the many hidden attributes.
+===+--------+-------+--------+---------+-------+---------+----+
Card|osiAdmin|osiOper|osiUsage|currentLP|Cause  |self     |sc  |
    |        |       |        |         |failure|TestFault|stat|
+===+--------+-------+--------+---------+-------+---------+----+
|1  |unlck   |dis    |idle    |wrongC   |none   |notApp   |none|
```

Displaying components using complex filtering example

You are currently in operational command mode and you want to display the operational components of the cards on the shelf that are not CPs and have an enabled operational state.You enter the following command:

```
display shelf card/(cardType != CP
AND(operationalState = enabled))
```

You receive the following information from the current view:

Displaying components using both instance and attribute value filtering

You are currently in operational command mode and you want to display the instances of the *Sts* component that have an operational state of disabled, for the instances of the *Laps* component that begin with 2. You enter the following command:

```
display Laps/2* Sts/*(operationalState = disabled)
```

You receive the following information from the current view:

```
Laps/3* Sts/*
Use -noTabular to see the many hidden attributes.
+=====+===+-----+----+-----+------+---+---+---+---+---+---+----------
|Laps |Sts|osiAd|osi0|osiUs|snmpOp|lop|ais|rfi|slm|txA|txR|  pefs
|     |   | min |per | age |erStat|   |   |   |   |is |di |
|     |   |     |    |     |  us  |   |   |   |   |is |di |
+=====+===+-----+----+-----+------+---+---+---+---+---+---+----------
| 200 |  0|unlck|dis |idle |down  |off|on |off|off|off|on |       0
```

Related information for display

See the following for information related to the display command:

- “Cancel command” (page 65)
- “List command” (page 159)
- “Set command” (page 272)
- “Start Prov command” (page 287)
- “Command wildcarding” (page 39)

End Prov command

The end Prov command lets you exit provisioning mode. After exiting provisioning mode, you are working with the current view, not the edit view.

When you exit provisioning view, your provisioning changes are not automatically saved to the filesystem. However, the edit view and the changes you made to it continue to exist in memory. If you enter provisioning mode again using the start Prov command, you will be able to see your changes.

If the node resets, all changes made in the edit view are lost unless you have saved the edit view. For information on saving the edit view, see “Save Prov command” (page 264).

For more information on the end Prov command, see

Properties for end Prov

The following table describes the properties of the end Prov command.

Table 44
Properties of the end Prov command

Mode	Impact	Scope	Component
Provisioning	Configuration	Application	ProvisioningSystem (Prov)

Syntax for end Prov

The following table describes the syntax for the end Prov command.

Table 45
Syntax of the end Prov command

If you want to...	Use the following syntax:
Exit the provisioning mode	end Prov

Exiting the provisioning mode example

You want to exit the provisioning mode. You enter the following command:

```
end Prov
```

You receive the following response:

```
Prov
ok
```

Note: After exiting provisioning mode, the system prompt changes from PROV> to >.

Related information for end Prov

See the following for information related to the end Prov command:

- “Save Prov command” (page 264)
- “Start Prov command” (page 287)

Find command

The find (f) command lets you return a list of all potentially available components that can be linked to a given component class. This verb is common to every Passport component.

When you are in provisioning mode, the edit view is used. When you are in operational mode, the current view is used.

The find command supports all command wildcarding capabilities. See “Command wildcarding” (page 39).

For more information on the find command, see

- “Properties for find” (page 137)
- “Syntax for find” (page 137)
- “Parameters for find” (page 138)
- “Examples for find” (page 140)

Properties for find

The find command works with all components. The following table describes the properties of the find command.

Table 46
Properties of the find command

Mode	Impact	Scope	Attribute	View
Operational	Passive	Depends on component	Operational	Current
Provisioning	Passive	Depends on component	Provisionable	Edit

Syntax for find

The following table describes the syntax for the find command. The -freelinks option is always required.

Table 47
Syntax of the find command

If you want to find components that are available to be linked to ...	Use the following syntax:
a given component type	find -freelinks(component_type)
the given component type from the given component name and its subcomponents	find -freelinks(component_type) <component_name>
the operational subcomponents	find -freelinks(component_type) -operationalData
the provisioned subcomponents	find -freelinks(component_type) -provisionedData
the given component type in the current view	find -current -freelinks(component_type)
the given component. When a component has more than one link attribute, you need to specify the link attribute	find -freelinks(component_type) -attribute(attribute)
the given component without searching the subtree(s)	find -freelinks(component_type) -nonrecursive

Parameters for find

The following items are the parameters of the find command:

- “Component parameter for find” (page 139)

Component parameter for find

If you only enter a component as a parameter to the find command, all components are searched. If you enter a component name as a parameter, it acts as a filter. By default the search is recursive, so the target component actually indicates where the search should begin.

You can use wildcarding to search components using the find command. With type wildcarding, you can substitute a wildcard (*) to search all instances of the component.

With instance wildcarding, you can substitute a wildcard (*) for an instance value to search all instances of a component type, or you can substitute a wildcard pattern to search selected instances.

Override default parameters for find

You can override the default attribute type and view associations using the following options.

Table 48
Options for the find command

Option	Description
-operationalData (-o)	Use this option to search operational components while in provisioning mode.
-provisionedData (-p)	Use this option to search provisioned components while in operational mode.
-current (-c)	Use this option to search in the current view when in provisioning mode.
-freelinks (-f)	Required option with the find command to indicate that the find verb should return those components that are available to be linked to the given component or component class.

(Sheet 1 of 2)

Table 48 (continued)
Options for the find command

Option	Description
-attribute (-a)	Use this option if the component or component class provided by the -freelinks option has more than one link attribute.
-nonrecursive (-n)	Use this option to indicate that the subcomponents should not be recursively searched.

(Sheet 2 of 2)

Examples for find

Search and return those components that are available to be linked to the component class *AtmIf*:

```
find -freelinks(AtmIf)
```

You receive the following type of response:

```
Lp/2 Sonet/1 Sts/0

Lp/3 Sonet/1 Sts/0

Laps/201 Sts/0

Lp/4 V35/1

ok
```

Search and return those components that are available to be linked to AtmIf at and under lp/2.

```
find -freelinks(AtmIf) lp/2
```

You receive the following type of response:

```
Lp/2 Sonet/1 Sts/0

ok
```

Search and return those components that are available to be linked to the component class LogicalProcessor. You need to specify the link attribute as LogicalProcessor has more than one settable link attribute.

```
find -freelinks(LP) -attribute(Lpt)
```

You receive the following type of response:

```
Sw Lpt/CP

ok
```

Related information for find

See the following for information related to the find command:

- “Display command” (page 121)
- “List command” (page 159)
- “Command wildcarding” (page 39)

Format Fs Disk

The format Fs Disk command deletes all files and directories on a disk. This command also allows you to specify the volume name of the newly formatted disk.

Only format the standby disk. If you want to format the active disk, use the switchover Lp command to make the active disk the standby disk.

Before formatting a disk you must lock it using the lock command on its *Fs Disk* component. After locking a the disk on a two-disk node, the disks lose synchronization. While you are formatting a disk you cannot test or synchronize the disk.

By default, the format Fs Disk command will format a disk to its maximum physical capacity. To support older control processors with smaller disks, you can format a disk to 256 Mbyte even though it has a larger physical capacity.

CAUTION
Possible loss of important files
The format command erases all files on the disk and resets the disk volume name. All information stored on the disk is lost.

For more information on the format Fs Disk command, see

- "Properties for format Fs Disk" (page 143)
- "Syntax for format Fs Disk" (page 143)
- "Parameters for format Fs Disk" (page 143)
- "Examples for format Fs Disk" (page 144)
- "Related information for format Fs Disk" (page 145)

Properties for format Fs Disk

The following table describes the properties of the format Fs Disk command.

Table 49
Properties of the format Fs Disk command

Mode	Impact	Scope	Component
Operational or Provisioning	Passive	Device	FileSystem (Fs)

Syntax for format Fs Disk

The following table describes the syntax for the format Fs Disk command.

Table 50
Syntax of the format command

If you want to...	Use the following syntax:
Format a disk	format -volumename(<name>) Fs Disk/<number>
Format a disk to be compatible with 256-Mbyte disks	format -volumename(<name>) -backward Fs Disk/ <number>

Parameters for format Fs Disk

The format Fs Disk command has the following parameters:

- “Disk parameter for format Fs Disk” (page 143)
- “New volume name parameter for format Fs Disk” (page 144)
- “Format to smaller size parameter for format Fs Disk” (page 144)

Disk parameter for format Fs Disk

Specify which disk you want to format using the *Disk* component, which is a subcomponent of *Fs*:

```
Disk/<number>
```

Enter the number of the disk you want to format. The disk number corresponds to the slot number of the control processor that holds the disk. You can list the disk numbers using the following command:

```
list FileSystem Disk/*
```

You must lock the *Disk* component before formatting the disk.

New volume name parameter for format Fs Disk

Specify the name of the newly formatted disk using the volumename option:

```
-volumename(<name>)
```

Enter a name for the formatted disk. Enclose the name in quotation marks to prevent Passport from misinterpreting its characters. The name, which is case sensitive, can be up to 11 characters long.

If you do not specify a volume name, the command uses the node name (*nodeName* attribute of the *Module* component).

If you want the standby disk to automatically synchronize with the active disk at start-up, use the same name as the active disk. The *volumeName* attribute of the *Fs Disk* component indicates the name of the disk.

Format to smaller size parameter for format Fs Disk

You can format a large disk to 256 Mbyte to maintain compatibility with older control processors using the backward option:

```
-backward
```

Examples for format Fs Disk

The following examples illustrate how to use the format Fs Disk command:

- “Formatting a disk to its maximum size example” (page 145)
- “Formatting a disk to a smaller size example” (page 145)

Formatting a disk to its maximum size example

You want to format the standby disk on a 5-slot Passport switch to its maximum size. The active disk is in slot 0 and the standby disk is in slot 4. You begin by locking the standby disk using the following command:

```
lock Fs Disk/4
```

You then format the disk, giving it the name Node12, using the following command:

```
format -volumename(“Node12”) Fs Disk/4
```

You then unlock the standby disk and then synchronize it to the active disk.

Formatting a disk to a smaller size example

To maintain backward compatability, you want to format the standby disk on a 16-slot Passport switch to 256 Mbyte even though its maximum size 810 Mbyte. The the standby disk is in slot 15. You begin by locking the standby disk using the following command:

```
lock Fs Disk/15
```

You then format the disk, giving it the name Node10, using the following command:

```
format -volumename(“Node10”) -backward Fs Disk/15
```

You then unlock the standby disk and then synchronize it to the active disk.

Related information for format Fs Disk

See the following for information related to the format Fs Disk command:

- “Lock command” (page 177)
- “Switchover Lp command” (page 320)
- “Synchronize Fs command” (page 327)
- “Unlock command” (page 360)

Help command

The help (h) command lets you view information about components, groups, attributes, and verbs. The help command also lets you view component hierarchies.

For more information on the help command, see

Properties for help

The following table describes the properties of the help command.

Table 51
Properties of the help command

Mode	Impact	Scope	Components
Operational or Provisioning	Passive	Not applicable	All components regardless of scope

Syntax for help

The following table describes the syntax for the help command.

Table 52
Syntax of the help command

If you want to...	Use the following syntax:
List all top-level components and verbs that do not require a component	help
Get help on a component	help <component_class>
Get help on a group	help <component_class> <group>
Get help on an attribute	help <component_class> <attribute>
Get help on a verb	help -verb(<verb>) <component_class>
List the complete component hierarchy for the node	help -subcomponents
List a hierarchy for a component	help -subcomponents <component_class>

Parameters for help

The help command has the following parameters:

- “Component parameter for help” (page 148)
- “Attribute and group parameters for help” (page 149)
- “Verb parameter for help” (page 150)
- “Component hierarchy parameter for help” (page 151)

Component parameter for help

You specify the component for which you need help by using its component class:

```
<component_class>
```

A component class is a sequence of component types without their instance values. For more information on component class, refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview*.

"Help information for a component" (page 148) describes the information the help command gives you when you request help on a component. If you do not enter a component class, the help command gives you the following information:

- name of the node
- syntax of the help command
- list of top-level components
- list of verbs that do not require components

Table 53
Help information for a component

Response field	Description
Component	Indicates the full name, abbreviated name (in parentheses), and possible instance values (range or list)
Instantiation	Indicates whether the component is required, optional, dynamic (automatically created based on system conditions), or dynamic-optional (automatically created or manually added)
Scope	Indicates the scope required to access or change the component. Possible values are application, device, or network
Subcomponent Types	Lists all the possible subcomponents of the component
(Sheet 1 of 2)	

Table 53 (continued)
Help information for a component

Response field	Description
Provisionable Group	Lists the provisionable groups of the component. Following each group is a list of its attributes.
Operational Group	Lists the operational groups of the component. Following each group is a list its attributes.
Verbs	Lists the verbs you can use with the component. The capital letters in the verb indicate the minimum number of characters you must use when abbreviating the verb.
(Sheet 2 of 2)	

Attribute and group parameters for help

You can get help on particular groups and attributes by specifying the attribute or group name after the component class. You can get help on multiple groups and attributes by entering a list of groups and attributes, separated by commas (,).

When you request help on a group, the help command indicates whether the group is operational or provisionable and lists all its attributes. “Help information for an attribute” (page 150) describes the information the help command gives you when you request help on an attribute.

Table 54
Attributes and groups for the help command

Option	Description
<attribute>	Enter the name of the attribute for which you need help.
<group>	Enter the name of the group for which you need help.

Table 55
Help information for an attribute

Field	Description
Attribute	Indicates the full name and abbreviated name (in parentheses)
Access	Indicates the required impact for reading and writing the attribute value
Criticality	Indicates one of the following consequences of changing the attribute: none, component (component out of service), process (components related to the process out of service), module (node restarts)
Type	Indicates the data type of the attribute
Values	Indicates the range of possible values for the attribute. Attributes that have a data type of signed integer, decimal number, fixed point decimal, or hexadecimal number can take either a range of numerical values or one or more named values. For example, you are setting a decimal number attribute that defines the maximum number of records held in a queue. You can set the attribute to a numerical value in the range of 1 to 200. Alternatively, you can set the attribute to the named value *unlimited*, which allows an unlimited number of records in the queue.
Default	Indicates the default value for the attribute. Passport automatically sets attributes to their default value.

Verb parameter for help

You can get help on a verb using the verb (v) option:

```
-verb(<verb>)
```

Enter the name of the verb. For most verbs you must also enter the component to which the verb applies. For those verbs that do not require components, you can enter the help command with only the verb option.

When you request help on a verb, the help commands indicates the required impact of the verb and its options. If an option takes a value, the help command also indicates its data type and restrictions.

Component hierarchy parameter for help

You can list a component hierarchy using the subcomponent (sub) option:

```
-subcomponent
```

If you want to view a hierarchy for all components available on the node, enter the help command with only the subcomponent option. If you want to see a hierarchy for a particular component, enter the component after the subcomponent option.

Examples for help

The following examples show how to use the help command:

- “Getting help on help example” (page 151)
- “Getting help on a component example” (page 152)
- “Getting help on groups and attributes example” (page 153)
- “Getting help on a verb example” (page 153)
- “Displaying a component hierarchy example” (page 154)

Getting help on help example

You want to display the node name, help command syntax, top-level components, and verbs that do not require components. You enter the following command:

```
help
```

You receive the following response:

```
EM/A567B7
 usage: Help [-verb(<verb>)] [-subcomponents]
        [<component class> [<attribute|group>
        [,<attribute|group>]]]
 Top-
Level Components NetworkManagementInterfaceSystem,
                      AccessControl,
                      Collector,
```

```
                                NumberingPlanIndicator,
                                ModuleData,
                                Shelf,
                                LogicalProcessor,
                                Software,
                                FileSystem,
                                Routing,
                                TransportResource,
                                ProvisioningSystem,
                                NetworkSynchronization,
                                VirtualRouter,
                                LanApplication,
                                Trunk,
                                DpnGateway,
                                FrameRelayUni,
                                AtmInterface,
                                AtmCallRouter,
                                AtmRouting
      Verbs                       CANCEL,LOGOUT,ME,Help,List
```

Getting help on a component example

You want information on the *FileSystem Disk* component. You enter the following command:

```
help FileSystem Disk
```

You receive the following response:

```
Fs Disk
   Component            Disk/Decimal([0-15])
   Instantiation        dynamic
   Scope                device
   Subcomponent Types   Test
   Operational Group    OsiState
      Attributes        adminState,operationalState,
                        usageState
   Operational Group    Operational
      Attributes        volumeName,capacity,
                        unformattedCapacity,freeSpace,
                        badBlocksPercentage
   Verbs                FORMAT,Display,Help,List,LOCK,
                        Set,Unlock
```

Getting help on groups and attributes example

You want information on the Provisioned group and the *runningTime* attribute of the *Lp DS1* component. You enter the following command:

```
help Lp DS1 Provisioned, runningTime
```

You receive the following response:

```
Lp DS1
  Group            Provisioned (Prov)
     Properties    Provisionable
     Attributes    lineType,zeroCoding,clockingSource
                   raiAlarmType,lineLength
  Attribute        runningTime
     Access        Read: passive   Write: not allowed
     Criticality   none
     Type          Decimal
     Values        [0-4294967295]
     Default       0
```

Getting help on a verb example

You want information on the save verb of the *Prov* component. You enter the following command:

```
help -verb(save) Prov
```

You receive the following response:

```
Prov
   Verb               Save
      Impact          configuration
      Options
      -current,
      -file(Ascii String(1 to 31 ASCII characters)),
      -component{c}(Component name([<component_type>
      [/<instance_value>]]...)),
      -ascii,
      -portable
```

Displaying a component hierarchy example

You want to view the component hierarchy for the *Software* component. You enter the following command:

```
help -subcomponent Software
```

You receive the following response:

```
Sw
   Download (Dld)
   ApplicationVersion (Av)
      Feature (Feat)
   LogicalProcessorType (Lpt)
```

Related information for help

See the following for information related to the help command:

- “Display command” (page 121)
- “List command” (page 159)
- “Me command” (page 186)

Install command

The install command lets you load another version of firmware onto a Passport 15000 or 20000 fabric card's writable bank and activate the new firmware. The active firmware controls the operation of the fabric card.

See 241-5701-600 *Passport 7400, 15000, 20000 Configuration Guide* for a definition of a fabric card's fixed bank and writable bank.

Changing a fabric card's active firmware should not affect switch operation. However, it is best to do this procedure when the switch is most idle.

All fabric cards in a Passport 15000 or 20000 shelf should be operating with the same version of firmware. An alarm is generated when fabric cards are each running a different version of firmware.

To change a fabric card's firmware, following the instructions in 241-5701-272 *Passport 7400, 15000, 20000 Software Upgrade*.

For more information on the install command, see

- "Properties for install" (page 155)
- "Syntax for install" (page 156)
- "Parameters for install" (page 156)
- "Examples for install" (page 157)
- "Related information for install" (page 158)

Properties for install

The following table describes the properties of the install command.

Note: The install command must always include the file option.

Table 56
Properties of the install command

Mode	Impact	Scope	Component
Operational	Service	Device	Shelf FabricCard

Syntax for install

The following table describes the syntax for the install command.

Table 57
Syntax of the install command

If you want to...	Use the following syntax:
Install new firmware for a fabric card	install -file(<firm_version>) Shelf FabricCard/<n>

Parameters for install

There are two types of parameters for the install command:

- “Firmware file name parameter for install” (page 156)
- “Fabric card identifier parameter for install” (page 156)

Firmware file name parameter for install

You must specify the name of the file containing the new version of firmware with the file option:

```
-file(<firm_version>)
```

Fabric card identifier parameter for install

You must specify the instance of the fabric card component where the new firmware is needed:

```
FabricCard/<n>
```

Examples for install

The following example shows how to use the install command.

You want to upgrade the firmware for both fabric cards on your switch. The new firmware is in a file called CA0127A which has already been downloaded from the software distribution site.

You lock the fabric card by entering:

```
lock Shelf FabricCard/x
```

You install the new firmware by entering:

```
install -file(CA0127A) Shelf FabricCard/x
```

You wait a few minutes for the new firmware to be installed.

You verify that the fabric card operates correctly with the new firmware by entering:

```
start Shelf FabricCard/x test
```

The test results show that the fabric card is operating correctly.

You unlock the fabric card by entering:

```
unlock Shelf FabricCard/x
```

You repeat all of the above steps for FabricCard/y.

Related information for install

See the following for information related to the install command:

- “Set command” (page 272)
- “Start Sw Dld command” (page 291)
- “Stop Sw Dld command” (page 303)
- “Start <hardware> Test command” (page 282)
- “Lock command” (page 177)
- “Unlock command” (page 360)

List command

The list (l) command lets you view the subcomponents of a component. You can list both operational and provisionable components in the current and edit views. Operational components only contain operational attributes. Provisionable components can contain both provisionable and operational attributes.

When you are in operational mode, the list command assumes you want to view operational and provisionable components in the current view. When you are in provisioning mode, the list command assumes you want to view provisionable components in the edit view. Unless you specify otherwise, the list command uses these default associations, which are summarized in “Properties of the list command” (page 160).

The list command supports all command wildcarding capabilities. See “Command wildcarding” (page 39).

For more information on the list command, see

Properties for list

The following table describes the properties of the list command.

Table 58
Properties of the list command

Mode	Impact	Scope	Component	View
Operational	Passive	Depends on component	All operational and all provisionable	Current
Provisioning	Passive	Depends on component	All provisionable	Edit

Syntax for list

The following table describes the syntax for the list command.

Table 59
Syntax of the list command

If you want to...	Use the following syntax:
List subcomponent types of a component	list <component_name>
List instances of a component type	list <component_type>/*
List all subcomponents of a component	list <component_name> *
List all top-level component types	list
List operational subcomponents in the current view	list -operationalComponents <component_name>
List provisionable subcomponents	list -provisionedComponents <component_name>
List subcomponents in the current view	list -current <component_name>

Parameters for list

There are four types of parameters for the list command:

- “Component parameter for list” (page 161)
- “Listing using default overrides example” (page 162)
- “Simple attribute value parameters for list” (page 162)
- “Complex attribute value parameters for list” (page 163)

Component parameter for list

The list command displays components of the type (operational or provisionable) and from the view (current or edit) associated with the current command mode (see “Properties of the list command” (page 160) for the associations). However, if you specify an operational component while in provisioning mode, you will get operational components from the current view. (Operational information does not exist in the edit view.) Passport informs you that it is displaying information from the current view and not the associated edit view.

You can substitute a wildcard (*) or a wildcard pattern to view instances of a component or subcomponent. Substitute a wildcard (*) character for the component type instance to view all subcomponents instances. Use a wildcard for a component instance value or an index value in a multi-indexed component to view instances of a component type. You can view multiple components in the hierarchy by substituting a wildcard for more than one instance value.

Using the list without a component gives you all top-level component types.

Table 60
Components for the list command

Option	Description
<component_name>	Enter the name of the component.
<component_type>/*	Use an asterisk for an instance value to view all instances of a component type.
(Sheet 1 of 2)	

Table 60 (continued)
Components for the list command

Option	Description
<component_name> *	Use an asterisk as the last component type-instance pair to view all subcomponent instances.
<component_type>/<index>, *, ...	Use an asterisk for one or more index value to view instances of a multi-indexed component.
<component_type>/* <component_type>/* ...	Use an asterisk for more than one instance value to view instances of multiple components in the hierarchy.
(Sheet 2 of 2)	

Listing using default overrides example

You can override the default associations using the provisionedComponents, operationalComponents, and current options.

Table 61
Component kind options for the list command

Option	Description
-provisioned Components	Use this option to list provisioned subcomponents.
-operational Components	Use this option to list operational subcomponents in the current view.
-current	Use this option to list subcomponents or component instances from the current view when you are in provisioning mode.

Simple attribute value parameters for list

Simple attribute value parameters consist of an expression that contains an attribute, a relational operator, and a value. A relational operator can be equal (=) or not equal (!=). Parentheses can be used, but are optional. See “Listing components using simple filtering example” (page 166) for an example of a simple attribute value parameter.

Complex attribute value parameters for list

Complex attribute value parameters are formed when simple attribute value parameters are joined using the following logical operators: AND and OR. A maximum of three simple expressions can be joined. Parentheses can be used, but are optional. See “Listing components using complex filtering example” (page 167) for an example of a simple attribute value parameter.

Applying both instance and attribute value parameters

You can combine both instance filtering and attribute value parameters to the same component instance. The format is:

```
<ComponentClass>/<instance>(attribute value)
```

Instance can be a wildcard (*) or a wildcard pattern. See “Listing components using both instance and value-based filtering” (page 167) for an example.

Examples for list

The following examples show how to use the list command:

- “Listing components example” (page 163)
- “Overriding defaults example” (page 164)
- “Listing with multiple wildcards example” (page 166)
- “Listing components using simple filtering example” (page 166)
- “Listing components using complex filtering example” (page 167)
- “Listing components using both instance and value-based filtering” (page 167)

Listing components example

You want to list all logical processors, which are represented by instances of the *Lp* component. You enter the following command:

```
list Lp/*
```

You receive the following response:

```
Lp/0
Lp/1
Lp/4
Lp/5
```

You then want to view the subcomponent types of logical processor 1 (*Lp/1*). You enter the following command:

```
list Lp/1
```

You receive the following response:

```
Lp/1 V35/*
Lp/1 Eng
```

Now you want to view all the subcomponents of logical processor 1 (*Lp/1*). You enter the following command:

```
list Lp/1 *
```

You receive the following response:

```
Lp/1 V35/0
Lp/1 V35/1
Lp/1 V35/2
Lp/1 V35/3
Lp/1 V35/4
Lp/1 V35/5
Lp/1 V35/6
Lp/1 V35/7
Lp/1 Eng
```

Now you want to list all the *View* subcomponents of *Vr/1 Snmp* with an index of 1. You enter the following command:

```
list Vr/1 Snmp View/1,*
```

You receive the following response:

```
Vr/1 Snmp View/1,1
Vr/1 Snmp View/1,2
Vr/1 Snmp View/1,3
```

Overriding defaults example

While in operational mode, you want view all top-level provisioned component types. You enter the following command:

```
list -provisionedComponents
```

You receive the following response:

```
Col/*
Sw
Shelf
Lp/*
Rtg
Mod
Ac
Npi/*
```

Now while in provisioning mode, you want view the provisioned subcomponents of the *AccessControl* component in the current view. You enter the following command:

```
list -current Ac *
```

You receive the following response:

```
Ac Userid/ADMIN
Ac Userid/USER1
Ac Userid/USER2
Ac Userid/USER3
Ac Userid/USER4
Ac Userid/USER5
```

Still in provisioning mode, you want to view all the top-level operational component types in the current view. You enter the following command:

```
list -operationalComponents
```

You receive the following information from the current view:

```
Nmis
Trm
Prov
Fs
```

Finally, while still in provisioning mode, you want to view the subcomponent types of the *Nmis* component. *Nmis* is an operational component that only exists in the current view. You enter the following command:

```
list Nmis
```

You receive the following information from the current view:

```
Nmis Local
Nmis Telnet
Nmis Fmip
Nmis Ftp
   All information displayed is from the current view.
```

Listing with multiple wildcards example

You want to list all V.35 ports provisioned on all logical processors. Instances of the *V35* component represent V.35 ports. You enter the following command:

```
list Lp/* V35/*
```

You receive the following response:

```
Lp/1 V35/0
Lp/1 V35/1
Lp/1 V35/2
Lp/2 V35/0
Lp/2 V35/1
Lp/2 V35/2
Lp/5 V35/6
Lp/6 V35/3
```

Listing components using simple filtering example

You are currently in operational command mode and you want to list the operational components of the cards on the shelf that are not CPs. You enter the following command:

```
list shelf card/(cardType != CP)
```

You receive the following information from the current view:

```
Shelf Card/1
Shelf Card/2
Shelf Card/3
Shelf Card/4
```

Listing components using complex filtering example

You are currently in operational command mode and you want to list the operational components of the cards on the shelf that are not CPs and have an enabled operational state. You enter the following command:

```
list shelf card/(cardType != CP AND (operationalState
= enabled))
```

You receive the following information from the current view:

```
Shelf Card/1
Shelf Card/3
Shelf Card/4
```

Listing components using both instance and value-based filtering

You are currently in operational command mode and you want to list all full provisioning views with a version of CD02A. You enter the following command:

```
List Prov View/*.full.*(version = CD02A)
```

You receive the following information from the current view:

```
Prov View/abc_example.full.001
Prov View/abc_example.full.002
Prov View/abc_example.full.003
```

Related information for list

See the following for information related to the display command:

- “Add command” (page 56)
- “Cancel command” (page 65)
- “Delete command” (page 118)
- “Display command” (page 121)
- “Start Prov command” (page 287)
- “Command wildcarding” (page 39)

ListFile Fs command

The listFile Fs (ls Fs) command lets you list the contents (files and directories) of a directory. You can get information about a directory itself (not its contents), and you can list only those files or directories that are newer than a file or directory you specify.

For each file or directory you list, the listFile Fs command displays the following information:

- type (file or directory)
- protected (yes or no)
- size in bytes
- date and time of the last modification
- name

For more information on the listFile command, see

- “Properties for listFile Fs” (page 168)
- “Syntax for listFile Fs” (page 169)
- “Parameters for listFile Fs” (page 169)
- “Examples for listFile Fs” (page 170)
- “Related information for listFile Fs” (page 172)

Properties for listFile Fs

The following table describes the properties of the listFile Fs command.

Table 62
Properties of the listFile Fs command

Mode	Impact	Scope	Component
Operational or Provisioning	Passive	Device	FileSystem (Fs)

Syntax for listFile Fs

The following table describes the syntax for the listFile Fs command.

Table 63
Syntax of the listFile Fs command

If you want to...	Use the following syntax:
List the contents of the current working directory	listFile Fs
List the contents of a specific directory	listFile -path(<path>) Fs
List information about the current directory itself	listFile -status Fs
List information about a specific directory	listFile -path(<directorypath>) -status Fs
List files and directories that are newer than specific file	listFile -newer(<path>) Fs

Parameters for listFile Fs

There are three parameters for the listFile command:

- “Specific directory or file parameter for listFile Fs” (page 169)
- “Directory status parameter for listFile Fs” (page 170)
- “Newer than a file or directory parameter for listFile Fs” (page 170)

Specific directory or file parameter for listFile Fs

You can list the contents of a specific directory or file using the path option:

```
-path(<path>)
```

Enter the absolute or relative path for a directory or a particular file you want to list. Enclose the path in double quotation marks. For information on path syntax, refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview*.

If you do not enter a path option, the listFile command lists the current working directory. If you enter a file, the command only lists information about that file.

Directory status parameter for listFile Fs

Instead of listing the contents of a directory, you can list information about the directory itself using the status option:

```
-status
```

Use this option to list information about the current working directory or a specific directory.

Newer than a file or directory parameter for listFile Fs

You can list only those files and directories that are newer than a specific file or directory using the newer option:

```
-newer(<path>)
```

Enter the absolute or relative path to a file or directory that you want to use for a newer comparison. Passport only lists those files and directories that are newer than the file or directory you specify. For information on path syntax, refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview*.

Examples for listFile Fs

The following examples illustrate how to use the listFile Fs command:

- "Listing the contents of a directory example" (page 170)
- "Listing information about a directory example" (page 171)
- "Listing newer files and directories example" (page 172)

Listing the contents of a directory example

Your current working directory is the root directory (/) and you want to list its contents. You enter the following command:

```
listFile Fs
```

You receive the following response:

```
Fs
   Type Prot Size       Date       Time     Name
   dir  no   0          1998-06-05 09:13:52 tmp
   dir  no   0          1998-05-04 09:29:44 system
   dir  no   0          1998-05-
05 09:36:36 provisioning
   dir  no   0          1998-05-04 09:29:44 software
   dir  no   0          1998-05-04 09:34:02 spooled
```

You now want to list the contents of the software directory. You enter the following command:

```
listFile -path(“software”) Fs
```

You receive the following response:

```
Fs
   Type Prot Size       Date       Time     Name
   dir  no   0          1998-05-04 09:29:44 .
   dir  no   0          1998-06-05 09:13:36 ..
   dir  no   0          1998-05-04 09:29:44 base
   dir  no   0          1998-05-04 09:29:44 trunks
   dir  no   0          1998-05-
04 09:29:44 frameRelay
   dir  no   0          1998-05-
04 09:29:44 networking
```

Listing information about a directory example

Your current working directory is the root directory (/) and you want to list information about it and not its contents. You enter the following command:

```
listFile -status Fs
```

You receive the following response:

```
Fs
   Type Prot Size       Date       Time     Name
   dir  no   0          1998-06-05 09:13:36 /
```

Listing newer files and directories example

Your current working directory is /provisioning and you want to list all files and directories in it newer than the file vr.full.001. You enter the following command:

```
listFile -newer("vr.full.001") Fs
```

You receive the following response:

```
Fs
   Type Prot Size       Date       Time     Name
   dir  no   0          1998-05-
12 09:35:06 vr.full.003
   dir  no   0          1998-05-
29 08:05:42 steve.full.004
```

Related information for listFile Fs

See the following for information related to the listFile Fs command:

- "Chdir Fs command" (page 68)
- "WorkingDir Fs command" (page 366)

Load Prov command

Use the load Prov command to load a saved view into the edit view.

When you issue the load Prov command, the system checks if the software required for the saved view is running on the node. If not, the load Prov command fails, and on-screen messages identify which software features you need to load. You can override this behavior using the force option.

For more information on the load Prov command, see

Properties for load Prov

The following table describes the properties of the load Prov command.

Table 64
Properties of the load Prov command

Mode	Impact	Scope	Components
Provisioning	Configuration	Application	ProvisioningSystem (Prov)

Syntax for load Prov

The following table describes the syntax for the load Prov command.

Table 65
Syntax of the load Prov command

If you want to...	Use the following syntax:
Load a saved view file from the disk to the edit view	load -file(<file_name>) Prov
Load a saved view file from the disk to the edit view, but skip the check for required software	load -force -file(<file_name>) Prov

Parameters for load Prov

There are two types of parameters for the load Prov command:

- "Loading a specific saved view parameter for load Prov" (page 174)
- "Skipping the check for required software parameter for load Prov" (page 175)

Loading a specific saved view parameter for load Prov

You can specify that the provisioning system load a specific file into the edit view using the file option:

```
-file(<filename>)
```

The file option loads the specified saved view file from the disk into the edit view. The file option identifies the saved view using the three-part view filename. Refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview* for information on saved view names.

Skipping the check for required software parameter for load Prov

You can specify that the provisioning system load the specified saved view file and skip the check for required software using the force option:

```
-force
```

The force option overrides the semantic check for required software. The system loads the saved view regardless of the software running on the node.

The force option also causes the provisioning system to continue to process the entire view rather than stopping after five errors, as would normally occur. (See “Check Prov command” (page 71) for a detailed description of how the semantic check works.)

Examples for load Prov

The following examples show how to use the load Prov command:

- “Loading a saved view into the edit view example” (page 175)
- “Loading a saved view into the edit view, but skipping the system check for required software example” (page 176)

Loading a saved view into the edit view example

You want to load the saved view ABCD1.full.001 into the edit view. The proper software for the saved view is currently running on the node. You enter the following command:

```
load -file(ABCD1.full.001) Prov
```

You receive the following response:

```
Prov
Loaded or applied file "AB0045.full.001”
ok
```

If you incorrectly enter the filename, you receive the following response:

```
Prov
   The specified file does not exist.
command failed
```

Loading a saved view into the edit view, but skipping the system check for required software example

You want to the saved view AXL_test.full.004 into the edit view, but you want to skip the system check for the proper software because the node is missing some of the proper software. If you try to load the saved view without the force option, you receive a number of error messages ending with the following response:

```
Too many errors loading file AXL_test.full.004.
command failed
```

In this situation, the system detects the errors, and aborts the command.

To force the node to ignore the error messages and load the view, you enter the following command:

```
apply -force -file(AXL_test.full.004) Prov
```

You receive a number of error messages, ending with the following response:

```
Loaded or applied file AXL_test.full.004.
```

If the required software for particular components is missing, those components will be missing from the edit view when the loading completes.

Related information for load Prov

See the following for information related to the load Prov command:

- “Activate Prov command” (page 48)
- “Check Prov command” (page 71)
- “ReloadCp Lp command” (page 229)
- “Start Prov command” (page 287)
- “Stop Prov command” (page 300)

Lock command

The lock command lets you prevent a component from providing service.

The lock command causes a specific component to enter the shutting down OSI administrative state. While shutting down, the component continues to provide services to existing clients, but does not accept any new requests for service. In some cases, locking a component shuts down all its subcomponents. When the component no longer providing service, it enters the locked OSI administrative state.

Some components (like *Bus*) do not enter the shutting down state. They move directly from the unlocked to the locked state. With components that do enter the shutting down state, you can force the component to skip the shutting down state and move immediately to the locked state. When you force a component to lock immediately, all services on the component terminate.

When you force a *Shelf Card* component to lock, Passport 7400 cards behave differently than Passport 15000 and 20000 cards. For Passport 7400, the *lock -force* command causes the card to become locked and disabled, then it is reset, and finally it becomes locked and enabled. For Passport 15000 or 20000, the *lock -force* command only causes the card to become locked and disabled. The Passport 15000 or 20000 card does not reset. Its OSI operational state becomes *enabled* only after it is unlocked.

When you lock a *Trunk* component or an *oamEnet* component, it automatically returns to an unlocked state after five minutes of being in a locked state. If necessary, you can permanently lock a *Trunk* component or an *oamEnet* component.

Before you lock a component, you should verify its OSI administrative state by displaying the *adminState* attribute of the component.

For more information on the lock command, see

- “Properties for lock” (page 178)
- “Syntax for lock” (page 178)
- “Parameters for lock” (page 179)

- “Examples for lock” (page 179)
- “Related information for lock” (page 181)

Properties for lock

You cannot lock some components. The following table describes the properties of the lock command.

Table 66
Properties of the lock command

Mode	Impact	Scope	Components
Provisioning	Service	Depends on component	All
Provisioning	Service	Network	Trunk (Trk)

Syntax for lock

The following table describes the syntax for the lock command.

Table 67
Syntax of the lock command

If you want to...	Use the following syntax:
Lock a component	lock <component_name>
Immediately lock an component, skipping the shuttingDown state	lock -force <component_name>
Lock a trunk until you unlock it	lock -forever Trunk/<n>
Lock the CP’s OAM Ethernet port until you unlock it	lock -forever lp/0 oamEnet/0

Parameters for lock

There are two parameters for the lock command:

- “Immediately locking a component parameter for lock” (page 179)
- “Locking a trunk permanently parameter for lock” (page 179)

Immediately locking a component parameter for lock

You can lock immediately by using the force option:

```
-force
```

When you use the force option, the component skips the shutting down stage. All service on the component immediately ends and the component moves to the locked state. The abbreviation for the force option is -f.

Locking a trunk permanently parameter for lock

When you lock a *Trunk* or an *oamEnet* component, it automatically unlocks after five minutes. You can lock a *Trunk* component or an *oamEnet* component and have it remained locked using the forever option:

```
-forever
```

You can only use the forever option with the *Trunk* component or the *oamEnet* component. When you use this option, the component does not unlock after five minutes. To unlock the *Trunk* component or the *oamEnet* component, use the unlock command.

Examples for lock

The following examples show you how to use the lock command:

- “Locking a component example” (page 180)
- “Immediately locking a component example for lock” (page 180)
- “Permanently locking a Trunk component example for lock” (page 181)
- “Permanently locking an oamEnet component example for lock” (page 181)

Locking a component example

You want to lock the Lp/1 component. Before locking the component, you check its OSI administrative state using the following command:

```
display Lp/1 adminState
```

You receive the following response:

```
Lp/1
   adminState = unlocked
```

Since Lp/1 is unlocked, you lock it using the following command:

```
lock Lp/1
```

Passport issues an alarm that indicates Lp/1 is shutting down. The Lp/1 component continues to provide existing service, but it will not accept any new requests for service. You verify the administrative state of Lp/1 using the following command:

```
display Lp/1 adminState
```

You receive the following response:

```
Lp/1
   adminState = shuttingDown
```

When all service on Lp/1 ends (for example, when the its associated processor card restarts), its administrative state changes to locked.

Immediately locking a component example for lock

You want to immediately lock the *Lp/1* component, skipping the shutting down administrative state. Before locking the component, you check its OSI administrative state using the following command:

```
display Lp/1 adminState
```

You receive the following response:

```
Lp/1
   adminState = unlocked
```

Since Lp/1 is unlocked, you lock it immediately using the following command:

```
lock -force Lp/1
```

Passport ends all service on Lp/1 by restarting its associated processor card. Since Lp/1 is locked, the processor card does not load its software. The LED status display on the processor card slowly pulses red. You verify the administrative state of Lp/1 using the following command:

```
display Lp/1 adminState
```

You receive the following response:

```
Lp/1
   adminState = locked
```

Permanently locking a Trunk component example for lock

You want to permanently lock the *Trunk/0* component. You enter the following command:

```
lock -forever Trunk/0
```

The *Trunk* component moves to the shutting down state and then to the locked state. The *Trunk* component remains locked until you unlock it.

Permanently locking an oamEnet component example for lock

You want to permanently lock the *oamEnet* component. You enter the following command:

```
lock -forever lp/0 oamEnet/0
```

The *oamEnet* component moves to the shutting down state and then to the locked state. The *oamEnet* component remains locked until you unlock it.

Related information for lock

See the following for information related to the lock command:

- “Display command” (page 121)
- “Unlock command” (page 360)

Logout command

The logout (logoff) command ends your local or telnet session.

If you were in provisioning mode when you entered the logout command, Passport runs the end Prov command. It does not modify or save the edit view.

On a local terminal, the login prompt appears after the session ends. With a telnet session, the logout command closes the telnet connection. You have to re-establish a telnet connection to log back onto the node.

The quit command is identical to the logout command. These two commands, along with their abbreviations, give you four ways to end your session:

- logout
- logoff
- quit
- exit

For more information on the logout command, see

- "Properties for logout" (page 182)
- "Syntax for logout" (page 183)
- "Ending your session example" (page 183)
- "Related information for logout" (page 183)

Properties for logout

The following table describes the properties of the logout command.

Table 68
Properties of the logout command

Mode	Minimum Impact	Component
Operational or Provisioning	Passive	Does not affect components

Syntax for logout

The following table describes the syntax for the logout command.

Table 69
Syntax of the logout command

If you want to...	Use the following syntax:
End your local or telnet session	logout logoff

Ending your session example

You are logged onto a Passport node on the first telnet session (Nmis Telnet Session/1), and you want to end your session. You enter the following command:

```
logout
```

You receive the following response:

```
Userid admin logged out as EM/NODEA Nmis Telnet
Session/1 at 21:07:33
Connection closed by foreign host.
```

Related information for logout

See the following for information related to the logout command:

- “Clear Nmis <interface> Session command” (page 87)
- “Me command” (page 186)
- “Quit command” (page 224)
- “Telnet Vr command” (page 330)

MakeDirectory Fs command

The makeDirectory Fs (mkdir Fs) command lets you create new directories on the Passport file system.

For more information on the makeDirectory Fs command, see

Properties for makeDirectory Fs

The following table describes the properties of the makeDirectory Fs command.

Table 70
Properties of the makeDirectory Fs command

Mode	Impact	Scope	Components
Operational or Provisioning	Configuration	Device	FileSystem (Fs)

Syntax for makeDirectory Fs

The following table describes the syntax for the makeDirectory Fs command.

Table 71
Syntax of the makeDirectory Fs command

If you want to...	Use the following syntax:
Make a new directory	makeDirectory -path(<directorypath>) Fs

New directory parameter for makeDirectory Fs

You specify the new directory using the path option:

```
-path(<directorypath>)
```

Enter the absolute or relative path for the new directory. Enclose the path in double quotation marks. If the parent directory in the path for the new directory does not exist, the command fails. For information on path syntax, refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview*.

You must enter the path option.

Creating a new directory example

Your current working directory is /provisioning. You want to create a new directory in the /tmp directory called test. You enter the following command:

```
makeDirectory -path("/tmp/test") Fs
```

You check to make sure the directory was created using the following command:

```
listFile -path("/tmp/test") -status Fs
```

You receive the following response:

```
Fs
   Type Prot Size       Date       Time     Name
   dir  no   0          1998-06-09 12:41:28 test
```

Related information for makeDirectory Fs

See the following for information related to the makeDirectory Fs command:

- "Chdir Fs command" (page 68)
- "Copy Fs command" (page 106)
- "ListFile Fs command" (page 168)
- "Remove Fs command" (page 232)
- "WorkingDir Fs command" (page 366)

Me command

The me command displays information about your current session and the access capabilities of your userid.

The session information includes the current node name (represented by the instance value of the *EM* component), network management interface, and session number.

For more information on the me command, see

- “Properties for me” (page 186)
- “Syntax for me” (page 187)
- “Displaying information about current session example” (page 187)
- “Related information for me” (page 187)

Properties for me

The following table describes the properties of the me command.

Table 72
Properties of the me command

Mode	Minimum Impact	Component
Operational or Provisioning	Passive	Does not affect components

Syntax for me

The following table describes the syntax for the me command.

Table 73
Syntax of the me command

If you want to...	Use the following syntax:
Display information about your current session	me

Displaying information about current session example

You are logged onto a Passport node and you want to find out information about your session and the access capabilities of your user ID. You enter the following command:

```
me
```

You receive the following response:

```
EM/NODEA Nmis Local Session/1; userid = operator1
Impact = configuration; Scope = network; CustomerId = 0
Mode = oper
```

Related information for me

See the following for information related to the me command:

- “Clear Nmis <interface> Session command” (page 87)
- “Logout command” (page 182)

Move Fs command

The move (mv Fs) command lets you move and rename files and directories.

The move Fs command does not overwrite existing files. If you specify an existing file as a destination, the command fails. If you want to overwrite a file, you must first remove it using the remove Fs command.

When using this command, avoid moving files into the root directory (/). Reserve the root directory for directories only.

For more information on the move Fs command, see

- "Properties for move Fs" (page 188)
- "Source and destination parameters for move Fs" (page 189)
- "Examples for move Fs" (page 190)
- "Related information for move Fs" (page 191)

Properties for move Fs

The following table describes the properties of the move Fs command.

Table 74
Properties of the move Fs command

Mode	Impact	Scope	Components
Operational or Provisioning	Configuration	Device	FileSystem (Fs)

Syntax for move Fs

The following table describes the syntax for the move Fs command.

Table 75
Syntax of the move Fs command

If you want to...	Use the following syntax:
Move a file into a new directory	move -source(<filepath>) -destination(<directorypath>) Fs
Rename a file	move -source (<filepath>) -destination(<newfilepath>) Fs
Move a directory into a directory	move -source(<directorypath>) -destination(<directorypath>) Fs
Rename a directory	move -source(<directorypath>) -destination(<newdirectorypath>) Fs

Source and destination parameters for move Fs

Specify which file or directory you want to copy or rename using the source option. Specify where you want to move the file or directory or its new name using the destination option.

If the destination file or directory does not exist, the move Fs command renames the source. You can move and rename a file at the same time by specifying a destination path with a new file name in a different directory.

If the destination is an existing directory, the move Fs command moves the source file or directory into it. If the destination is an existing file, the command fails.

Table 76
Source and destination options for the move Fs command

Option	Description
-source (<filepath>) -source (<directorypath>)	Enter the absolute or relative path to the file or directory you want to move or rename. You must enter this option.
-destination (<filepath>) -destination (<directorypath>)	Enter the absolute or relative path to the new location or new name of the file or directory. You must enter this option.

Examples for move Fs

The following examples illustrate how to use the move Fs command:

- “Moving a file and directory example” (page 190)
- “Renaming a file example” (page 190)

Moving a file and directory example

Your current working directory is /spooled. You want to move the file account.001 in the current working directory into the /tmp directory. You enter the following command:

```
move -source(“account.001”) -destination(“/tmp”) Fs
```

You now want to move the oldaccount directory, which is in your current working directory, into the /tmp directory. You enter the following command:

```
move -source(“oldaccount”) -destination(“/tmp”) Fs
```

The oldaccount directory is now a subdirectory of /tmp.

Renaming a file example

You want to rename the file account.001 in your current working directory to acc.old. A file by that name does not exist in your current working directory. You enter the following command:

```
move -source(“account.001”) -destination(“acc.old”) Fs
```

You now decide to move the acc.old file into the /tmp directory while changing its name to acc.backup. You enter the following command:

```
move -source("acc.old") -destination("/tmp/
acc.backup") Fs
```

Related information for move Fs

See the following for information related to the move Fs command:

- "Copy Fs command" (page 106)
- "ListFile Fs command" (page 168)
- "MakeDirectory Fs command" (page 184)
- "Remove Fs command" (page 232)
- "WorkingDir Fs command" (page 366)

Newfile Col Sp command

The newfile Col Sp command lets you open a new spooling file for data collection.

Spooling files contain data (alarms, accounting, statistics, state change notifications, log, and debug) collected by the data collection system. You can transfer these files from the Passport node for analysis or billing purposes.

The command closes the currently open spooler file (accepting new data records), moves it to the closed directory, and opens a new file. The command fails if you have not turned spooling on for the data type (using the *spooling* attribute of the *Collector Spooler* component).

For more information on the newfile Col Sp command, see

- "Properties for newfile Col Sp" (page 192)
- "Syntax for newfile Col Sp" (page 193)
- "Data type parameter for newfile Col Sp" (page 193)
- "Creating a new spooling file example" (page 194)

Properties for newfile Col Sp

The following table describes the properties of the newfile Col Sp command:

Table 77
Properties of the newfile Col Sp command

Mode	Impact	Scope	Components
Operational or Provisioning	Service	Device	Collector Spooler (Col Sp)

Syntax for newfile Col Sp

The following table describes the syntax for the newfile Col Sp command.

Table 78
Syntax of the newfile Col Sp command

If you want to...	Use the following syntax:
Create a new spooling file	newfile Col/<data_type> Sp

Data type parameter for newfile Col Sp

You specify the data type for which you want to create a new spooling file using the instance value of the *Collector* (*Col*) component:

```
Col/<data_type> Sp
```

Enter one of the following data types:

- accounting (acc)
- alarm (ala)
- debug
- log
- scn
- stats

Note: The rtstats and traps data types cannot be spooled. Therefore, the rtstats and traps data types do not apply to the Col Sp command.

For more information on these data types, see 241-5701-611 *Passport 7400, 15000, 20000 Data Collection Guide*.

Creating a new spooling file example

You want to create a new spooling file for the accounting data type. You enter the following command:

```
newfile Col/accounting Sp
```

Passport closes the current spooling file, moves it to the closed directory, and opens a new spooling file.

Optimize command

The optimize command lets you to manually trigger an optimization pass when using the PNNI local and global rerouting feature. The optimize command can be issued at the module (node), interface, or connection level.

If you use the optimize command against the

- *ARtg Pnni Reroute* component
  - — you will trigger a module-wide (nodal) optimization pass
- *AtmIf*/<n> <*interface_type*> *Reroute* component
  - — you will trigger a path optimization pass at that ATM interface
- *AtmIf*/<n> *Vpt*/<*m*> <*interface_type*> *Reroute* component
  - — you will trigger a path optimization pass at a virtual path terminator (VPT)
- *AtmIf*/<n> *vcc*<*vpi.vci*> *Source RerouteOverride* component
  - — you will trigger a path optimization pass at that ATM vcc connection
- *AtmIf*/<n> *vpc*/<*vpi*> *Source RerouteOverride* component
  - — you will trigger a path optimization pass at that ATM vpc connection
- *AtmIf*/<n> *vpt*/<*m*> *vcc*<*vci*>*Source RerouteOverride* component
  - — you will trigger a path optimization pass at that ATM vpt vcc connection

Where:

- <m> is the virtual path terminator number or instance
- <n> is the ATM interface number or instance
- <interface_type> is either Uni, Iisp, Aini, or Pnni

The module optimization pass can be cancelled by using the option *-cancel*.

Note 1: A path optimization will only move connections from an incumbent route if the new route possesses a better path optimization metric. For more information, see 241-5701-702 *Passport 7400, 15000, 20000 ATM Routing and Signaling Fundamentals*.

Note 2: If a local connection recovery occurs in a downstream local rerouting domain, you can issue *optimize artg pnni* (if the component is provisioned) or *optimize atmif/x UNI/IISP/AINI/PNNI reroute* at the local rerouting node. For more information, see 241-5701-702 *Passport 7400, 15000, 20000 ATM Routing and Signaling Fundamentals*.

For more information on the optimize command, see

- "Syntax for optimize" (page 196)
- "Examples for optimize" (page 198)
- "Related information for optimize" (page 200)

Syntax for optimize

The recommended syntax for the optimize command works only with the default settings for the PNNI local and global feature. The following tables describe the syntax for the optimize command:

- "Syntax of the optimize command for the module" (page 196)
- "Syntax of the optimize command for the interface" (page 197)
- "Syntax of the optimize command for the connection" (page 198)

Table 79
Syntax of the optimize command for the module

If you want to...	Use the following syntax:
Manually trigger the module (nodal) optimization mechanism.	optimize ARtg Pnni Reroute
Manually cancel the module optimization pass.	optimize -cancel ARtg Pnni Reroute

Table 80
Syntax of the optimize command for the interface

If you want to...	Use the following syntax:
Manually trigger the path optimization mechanism for connections on an interface that has been recovered for path optimization.	optimize AtmIf/<n> <interface_type> Reroute or optimize AtmIf/<n> Vpt/<x> <interface_type> Reroute or optimize FrAtm<n> Reroute or optimize aal1Ces/<n> Reroute where <n> is the interface number <interface_type> is one of the following interface types: Uni, Aini, Iisp, or Pnni
Manually trigger the path optimization mechanism for all connections on the interface regardless of whether they are eligible for path optimization or not, including those that have been recovered when the optimize command was last run.	optimize -all AtmIf/<n> <interface_type> Reroute or optimize -all AtmIf/<n> Vpt/<x> <interface_type> Reroute where <n> is the interface number <interface_type> is one of the following interface types: Uni, Aini, Iisp, or Pnni

Table 81
Syntax of the optimize command for the connection

If you want to...	Use the following syntax:
Manually trigger the path optimization mechanism for connections that have been recovered for path optimization.	optimize AtmIf/<n> <connection_type> Src RerouteOv where <connection_type> is vcc/x.y, vpc/y, or vpt/x vcc/y or optimize FrAtm/<n> Dlci/<x> siwf Spvc RerouteOv

Examples for optimize

The following examples illustrate how to use the optimize command:

- "Triggering a module optimization pass" (page 198)
- "Cancelling a module optimization pass" (page 199)
- "Optimizing connections that have been recovered on an ATM interface" (page 199)
- "Optimizing all connections on an ATM interface" (page 200)

Triggering a module optimization pass

You want to manually trigger a module (nodal) optimization pass against the *ARtg Pnni Reroute* component. You enter the following command:

```
optimize ARtg Pnni Reroute
```

When the command completes successfully, you receive the following response:

```
Module optimization pass completed.
```

If a module optimization pass is in progress when you use the optimize command, you receive the following response:

```
A module optimization pass is already in progress.
```

Cancelling a module optimization pass

You want to cancel the module (nodal) optimization pass that is currently in progress. You enter the following command:

```
optimize -cancel ARtg Pnni Reroute
```

When the command completes successfully, you receive the following response:

```
The current module optimization pass has been
cancelled.
```

If there is a cancel command already in progress, you receive the following response:

```
A cancel of the module optimization pass is in
progress.
```

If there is no module optimization pass in progress, you receive the following response:

```
There is no module optimization pass currently in
progress.
```

Optimizing connections that have been recovered on an ATM interface

You want to manually trigger the path optimization mechanism against the *AtmIf <interface_type> Reroute* component for connections that have been recovered for path optimization and are eligible for path optimization. These connections are those that are included in the attribute *eligibleRecoveredConnections*. You enter the following command:

```
optimize AtmIf/<AtmIf> <interface_type> Reroute
```

When the command completes successfully, you receive the following response:

```
Interface optimization pass completed.
```

If an optimization pass on the ATM interface is in progress when you use the optimize command, you receive the following response:

```
An interface optimization pass is already in progress.
```

If a module (nodal) optimization pass is in progress when you use the optimize command, you receive the following response:

```
A module optimization pass is already in progress.
```

Optimizing all connections on an ATM interface

You want to manually trigger the path optimization mechanism against the *AtmIf <interface_type> Reroute* component for all connections on the interface regardless of whether they are eligible for path optimization or not., including those that have been recovered when the optimize command was last run. You enter the following command:

```
optimize -all AtmIf/<AtmIf> [Vpt/Vpi] <interface_type>
Reroute
```

where:
<interface_type> is Uni, Iisp, Aini, or Pnni

When the command completes successfully, you receive the following response:

```
Interface optimization pass completed.
```

If an optimization pass on the ATM interface is in progress when you use the optimize command, you receive the following response:

```
An interface optimization pass is already in progress.
```

If a module (nodal) optimization pass is in progress when you use the optimize command, you receive the following response:

```
A module optimization pass is already in progress.
```

Related information for optimize

See the following for information related to the optimize command:

- “Display command” (page 121)
- “List command” (page 159)

Ping <connection> Vc command

The ping <connection> Vc command gives you information about the paths between the end points of a frame relay virtual circuit.

The command sends one or more ping packets to the destination node of a specific frame relay virtual circuit. The destination node modifies the ping packet and sends it back to the source node. The ping <connection> Vc command uses this response to determine path information. The command waits up to 15 seconds for responses from the ping packets it sends.

The command returns either a single path to the destination node or the amount of time it takes for a packet to travel to the destination node and back again (round trip delay). You can adjust the size of the ping packet for both the trip to the destination node and the trip back to the source node.

The path taken to the destination node by the ping packet is the same as the path taken by frame relay data packets.

For more information on the ping <connection> Vc command, see

- “Properties for ping <connection> Vc” (page 202)
- “Syntax for ping <connection> Vc” (page 202)
- “Parameters for ping <connection> Vc” (page 203)
- “Examples for ping <connection> Vc” (page 205)
- “Related information for ping <connection> Vc” (page 206)

Properties for ping <connection> Vc

The following table describes the properties of the ping <connection> Vc command.

Table 82
Properties of the ping <connection> Vc command

Mode	Impact	Scope	Components
Operational or Provisioning	Passive	Application	FrameRelayUni DataLinkConnectionIdentifier Vc (FrUni Dlci Vc) FrameRelayNni DataLinkConnectionIdentifier Vc (FrNni Dlci Vc) IpInterfaceOverFrameRelay LogicalChannelNumber Vc (IpiFr Lcn Vc) RemoteServerAgent Connection Vc (Rsa Connection Vc)

Syntax for ping <connection> Vc

The following table describes the syntax for the ping <connection> Vc command.

Table 83
Syntax of the ping <connection> Vc command

If you want to...	Use the following syntax:
Find a single path to a node	ping <connection> Vc
Determine the round trip delay to a node	ping -roundTripDelay <connection> Vc
Determine the round trip delay using specific packet sizes	ping -roundTripDelay -sendPktDataSize (<size>) -returnPktDataSize (<size>) <connection> Vc

Parameters for ping <connection> Vc

There are two types of parameters for the ping <connection> Vc command:

- "Virtual circuit parameters for ping <connection> Vc" (page 203)
- "Round trip delay parameters for ping <connection> Vc" (page 204)

Virtual circuit parameters for ping <connection> Vc

You identify which virtual circuit you want path information about using the components of the ping <connection> Vc.

The command reports the routing identifier (RID), the module identifier (MID), and the process identifier (PID) number of each node on the path between the source and destination node. The PID number only applies to DPN-100 nodes. It indicates the process number of the trunk that received the ping packet. On a Passport node, the PID number is always zero.

Table 84
Components of the ping <connection> Vc command

Component	Notes
<connection>	Enter the type of connection for the virtual circuit. The following connection types are available: • FrUni/<n> Dlci/<n> • FrNni/<n> Dlci/<n> • IpiFr Lcn/<n> • Rsa/<n> Connection/<n> where <n> is the instance number of the service and the instance number of the connection
Vc	Always enter this component after the <connection> components.

Round trip delay parameters for ping <connection> Vc

When you use the roundTripDelay option, the command sends five ping packets to the destination node. It records the amount of time it takes to receive a reply back from the destination node for each packet. It then reports the minimum, maximum, and average round trip delay. The command does not report the path the ping packets took to the node.

You can adjust the amount of data the ping packets carry using the sendPktDataSize and returnPktDataSize options. The amount of data the packet carries affects the round trip delay. In most cases, larger packets have longer round trip delay times. Although you can enter zero bytes for both the sendPktDataSize and returnPktDataSize options, the ping packet always uses 12 data bytes to carry ping control information. Since the ping packet contains a 13-byte header, the smallest possible ping packet is 25 bytes.

The default amount of data for both the send and return packets is 128 bytes. If you enter the command using a *FrUni Dlci* or *FrNni Dlci* component and its transfer priority is 11, the default amount of data is 64 bytes.

If you enter a data packet size that exceeds the capacity of a some facility along the path, that facility discards the ping packet.

Table 85
Round trip delay parameters of the ping <connection> Vc command

Option	Notes
-roundTripDelay -rtd	Use this option to get round trip delay information. The command ignores this option you also select allPaths option.
-sendPktDataSize (<size>)	Enter the number of bytes of data to send from source to destination node. You can enter a value between 0 and 4095. This option has an effect only if you also specify the roundTripDelay option.
-returnPktDataSize (<size>)	Enter the number of bytes of data to send from the destination to the source node. You can enter a value between 0 and 4095. This option has an effect only if you also specify the roundTripDelay option.

Examples for ping <connection> Vc

The following examples show how to use the ping <connection> Vc command:

Determining a single path example

You want to find the path to the destination of the virtual circuit on frame relay UNI service 21 with a data link connection identifier of 103. You enter the following command:

```
ping FrUni/21 Dlci/103 Vc
```

You receive the following response:

```
FrUni/21 Dlci/103 Vc
  Destination Rid = 53, Mid = 1121. VC path taken is:
  Rid = 53    Mid = 1121    Pid = 0
```

The destination node is directly connected to your node, so there is only one node in the path.

Determining round trip delay example

You want to determine how long it takes a packet to travel to the destination of a virtual circuit (FrUni/44 Dlci/16 Vc) and back again. You want to send 256 data bytes to the destination node and you want to return 512 bytes back to the source node. You enter the following command:

```
ping -roundTripDelay -sendPktDataSize(256) -
returnPktDataSize(512) FrUni/44 Dlci/16 Vc
```

You receive the following response:

```
FrUni/44 Dlci/16 Vc
   sentPktDataSize = 256 bytes
   returnedPktDataSize = 512 bytes
   pktEmissionPriority = normal
   numRtdPktSent       = 5
   numRtdPktReceived   = 5
```

```
averageRtd          = 6 msec
minimumRtd          = 5 msec
maximumRtd          = 7 msec
```

Related information for ping <connection> Vc

See the following for information related to the ping <connection> Vc command:

- “Ping Rtg Dpn <node> command” (page 207)

Ping Rtg Dpn <node> command

The ping Rtg Dpn <node> command gives you information about the paths between network nodes that are using the dynamic packet routing system (DPRS).

The command sends one or more ping packets to a node specified using either its routing identifier (RID), its module identifier (MID), or both its RID and MID. The destination node modifies the ping packet and sends it back to the source node. The ping <dpn> command uses this response to determine path information.

The command returns a single path to the destination node, all the paths to the destination node, or the amount of time it takes for a packet to travel to the destination node and back again (round trip delay). You can adjust the path the ping packet selects by setting its sensitivity, reliability, and priority.

For more information on the ping Rtg Dpn <node> command, see

- “Properties for ping Rtg Dpn <node>” (page 208)
- “Syntax for ping Rtg Dpn <node>” (page 208)
- “Parameters for ping Rtg Dpn <node>” (page 209)
- “Examples for ping Rtg Dpn <node>” (page 212)
- “Related information for ping Rtg Dpn <node>” (page 214)

Properties for ping Rtg Dpn <node>

The following table describes the properties of the ping Rtg Dpn <node> command.

Table 86
Properties of the ping Rtg Dpn <node> command

Mode	Impact	Scope	Components
Operational	Passive	Network	Routing DpnAddressPlan ReachableRidInfo (Rtg Dpn Rid) Routing DpnAddressPlan ReachableMidInfo (Rtg Dpn Mid) Routing DpnAddressPlan ReachableRidInfo ReachableMidInfo (Rtg Dpn Rid Mid)

Syntax for ping Rtg Dpn <node>

The following table describes the syntax for the ping Rtg Dpn <node> command.

Table 87
Syntax of the ping Rtg Dpn <node> command

If you want to...	Use the following syntax:
Find a single path to a node	ping Rtg Dpn Rid/<rid> Mid/<mid>
Find a single path to a node with path selection	ping -sensitivity(delay | throughput) -reliability(high | normal) -priority(high | normal) Rtg Dpn Rid/<rid> Mid/<mid>
Find all paths to a node	ping -allPaths Rtg Dpn Rid/<rid> Mid/<mid>
Find all paths to a node with path selection	ping -sensitivity(delay | throughput) -reliability(high | normal) -priority(high | normal) -allPaths Rtg Dpn Rid/<rid> Mid/<mid>
(Sheet 1 of 2)	

Table 87 (continued)
Syntax of the ping Rtg Dpn <node> command

If you want to...	Use the following syntax:
Determine the round trip delay to a node	ping -roundTripDelay Rtg Dpn Rid/<rid> Mid/<mid>
Determine the round trip delay using specific packet sizes	ping -roundTripDelay -sendPktDataSize (<size>) -returnPktDataSize (<size>) Rtg Dpn Rid/<rid> Mid/<mid>
(Sheet 2 of 2)	

Parameters for ping Rtg Dpn <node>

There are four types of parameters for the ping Rtg Dpn <node> command:

- "Node identifier parameters for ping Rtg Dpn <node>" (page 209)
- "Path selection parameters for ping Rtg Dpn <node>" (page 210)
- "Round trip delay parameters for ping Rtg Dpn <node>" (page 211)
- "All paths parameter for ping Rtg Dpn <node>" (page 212)

Node identifier parameters for ping Rtg Dpn <node>

You identify which node you want path information about using the components of the ping Rtg Dpn <node> command. Specify a node by its MID, RID, or both.

With only the node identified, the command sends a single packet carrying 128 bytes of data to the destination node and waits for a reply. If it does not receive a reply in 15 seconds, it will time out. A timeout usually indicates there is a problem with the destination node or congestion on the network.

The command reports the RID, the MID, and the process identifier (PID) number of each node on the path between the source and destination node. The PID number only applies to DPN-100 nodes. It indicates the process number of the trunk that received the ping packet. On a Passport node, the PID number is always zero.

Table 88
Components of the ping Rtg Dpn <node> command

Component	Notes
Rtg Dpn	Always enter these two components.
Rid/<rid> Mid/<mid> Rid/<rid> Mid/<mid>	Specify the address of the destination node. Enter the RID, the MID, or both the RID and MID for the node. If you enter only a MID, the command only finds paths within your own RID. If you enter only a RID, the command finds paths to any node within that RID.

Path selection parameters for ping Rtg Dpn <node>

You can adjust how a ping packet selects its path by setting its reliability, sensitivity, and priority. You can use the path section parameters whether you are finding a single path, all paths, or determining the round trip delay.

Table 89
Path selection parameters of the ping Rtg Dpn <node> command

Option	Notes
-sensitivity(delay | throughput)	Enter *delay* to select the path with the shortest delay time. Enter *throughput* to select the path with the highest bandwidth. The default is throughput sensitivity.
-reliability(high | normal)	Enter *high* to select an overflow path if the primary path is congested. Enter *normal* to discard the ping packet if the path is congested. The default is high reliability.
-priority(high | normal)	Enter *high* to reduce the likelihood of the ping packet being discarded. The default is normal priority.

Round trip delay parameters for ping Rtg Dpn <node>

When you use the roundTripDelay option, the command sends five ping packets to the destination node. It records the amount of time it takes to receive a reply back from the destination node for each packet. It then reports the minimum, maximum, and average round trip delay. The command does not report the path the ping packets took to the node.

You can adjust the amount of data the ping packets carry using the sendPktDataSize and returnPktDataSize options. The amount of data the packet carries will affect the round trip delay. In most cases, larger packets will have longer round trip delay times. Although you can enter zero bytes for both the sendPktDataSize and returnPktDataSize options, the ping packet always uses 12 data bytes to carry ping control information. Since the ping packet contains a 13-byte header, the smallest possible ping packet is 25 bytes.

The default amount of data for both the send and return packets is 128 bytes.

Table 90
Round trip delay parameters of the ping Rtg Dpn <node> command

Option	Notes
-roundTripDelay -rtd	Use this option to get round trip delay information. The command ignores this option if you also select allPaths option.
-sendPktDataSize (<size>)	Enter the number of bytes of data to send from source to destination node. You can enter a value between 0 and 4095. This option has an effect only if you also specify the roundTripDelay option.
-returnPktDataSize (<size>)	Enter the number of bytes of data to send from the destination to the source node. You can enter a value between 0 and 4095. This option has an effect only if you also specify the roundTripDelay option.

All paths parameter for ping Rtg Dpn <node>

You can find many paths to the destination node using the allPaths option:

```
-allPaths
```

When you use the allPaths option, the command sends 16 ping packets to the destination node. It reports each unique path and the number of packets that took that path. The command waits up to 15 seconds for responses from all the packets.

Examples for ping Rtg Dpn <node>

The following examples show how to use the ping <dpn> command:

- "Determining a single path example" (page 212)
- "Determining all paths example" (page 213)
- "Determining round trip delay example" (page 213)

Determining a single path example

You want to find the path with the highest throughput to a DPN access module (AM) at RID 4 and MID 478. There are two Passport nodes between the node to which you are logged on (the source node) and the destination node. You enter the following command:

```
ping Rtg Dpn Rid/4 Mid/478
```

You receive the following response:

```
Rtg Dpn Rid/4 Mid/478
    Rid = 4    Mid = 1161    Pid = 0
    Rid = 4    Mid = 1160    Pid = 0
    Rid = 4    Mid = 478     Pid = 659459
```

Pid is the process identifier of the trunk process that received the packet on each RM or AM. This information is not available on Passport nodes and is set to 0.

Determining all paths example

You want to find all the paths to a destination node with RID 84 and MID 1103. There are three unique paths between the source node and the destination node. You enter the following command:

```
ping -allPaths Rtg Dpn Rid/84 Mid/1103
```

You receive the following response:

```
Rtg Dpn Rid/84 Mid/1103
    16 packets were sent, 16 packets were returned.
    Path 1 used 4 time(s):
    Rid = 84    Mid = 1115    Pid = 0
    Rid = 84    Mid = 1122    Pid = 0
    Rid = 84    Mid = 1103    Pid = 0
    Path 2 used 4 time(s):
    Rid = 84    Mid = 1115    Pid = 0
    Rid = 84    Mid = 1109    Pid = 0
    Rid = 84    Mid = 1103    Pid = 0
    Path 3 used 8 time(s):
    Rid = 84    Mid = 1377    Pid = 0
    Rid = 84    Mid = 1109    Pid = 0
    Rid = 84    Mid = 1103    Pid = 0
```

Determining round trip delay example

You want to determine how long it takes a packet to travel to a node (RID 84, MID 1103) and back again. You want to send 30 data bytes to the destination node and you want it to return 20 bytes back to the source node. You enter the following command:

```
ping -roundTripDelay -sendPktDataSize(30)
-returnPktDataSize(20) Rtg Dpn Rid/84 Mid/1103
```

You receive the following response:

```
Rtg Dpn Rid/84 Mid/1103
    sendPktDataSize     = 30 bytes
    returnPktDataSize   = 20 bytes
    pktEmissionPriority = normal
    numRTDPktSent       = 5
    numRTDPktReceived   = 5
```

```
averageRTD          = 11 msec
minimumRTD          = 11 msec
maximumRTD          = 13 msec
```

Related information for ping Rtg Dpn <node>

See the following for information related to the ping Rtg Dpn <node> command:

- “Ping <connection> Vc command” (page 201)

Protect Fs command

The protect Fs command protects a file from deletion or modification.

When a file is protected, you cannot delete it using the remove Fs command. To delete a protected file, you must first remove its protection using the unprotect Fs command. You can tell if a file is protected or not using the listFile Fs command.

You cannot protect a directory. When you copy a protected file, the new file is not protected.

For more information on the protect Fs command, see

Properties for protect Fs

The following table describes the properties of the protect Fs command.

Table 91
Properties of the protect Fs command

Mode	Impact	Scope	Component
Operational or Provisioning	Configuration	Device	FileSystem (Fs)

Syntax for protect Fs

The following table describes the syntax for the protect Fs command.

Table 92
Syntax of the protect Fs command

If you want to...	Use the following syntax:
Protect a file from deletion	protect -path(filepath) Fs

File to protect parameter for protect Fs

You specify which file you want to prevent from being deleted or modified using the path option:

```
-path(<filepath>)
```

Enter the absolute or relative path for the file you want to prevent from being deleted or modified. Enclose the path in double quotation marks. For information on path syntax, see “Entering commands” (page 36).

You must enter the path option.

Protecting a file example

You want to prevent the alarm.001 file in the /tmp directory from being deleted or modified. Your current working directory is root (/). You enter the following command:

```
protect -path(“/tmp/alarm.001”) Fs
```

You use the following listFile Fs commands to verify that the file was protected:

```
listFile -path(“/tmp/alarm.001”) Fs
```

You receive the following response:

```
Fs
   Type Prot Size    Date       Time     Name
   file yes  9547    1998-05-05 10:12:38  alarm.001
```

The *yes* in the Prot column indicates that the file is protected.

Related information for protect Fs

See the following for information related to the protect Fs command:

- “ListFile Fs command” (page 168)
- “Remove Fs command” (page 232)
- “Unprotect Fs command” (page 363)

ProtectionLockout Aps command on Passport 7400

The protectionLockout Aps (protLock Aps) command forces a link protected by automatic protection switching (APS) to stop using the protection line.

An APS-protected link has two lines: working and protection. The working line normally carries the data. The protection line acts as a backup to the working line. The line that is currently carrying the data is called the active line. When no manual overrides are in effect, the APS system automatically determines which line is active.

If the protection line is active when you enter this command, APS switches the data to the working line (making it active), even if the working line is degraded or has failed. If you have set the APS system to bidirectional mode and the far end of the link has already locked the protection line, the protectionLockout Aps command is ignored.

You can release the lock on the protection line using the clear Aps command. You can switch the active line between the working and protection line using the switch Aps command.

For more information on the protectionLockout Aps command, see

- “Properties for protectionLockout Aps” (page 219)
- “Syntax for protectionLockout Aps” (page 219)
- “Component instance parameter for protectionLockout Aps” (page 219)
- “Examples for protectionLockout Aps” (page 220)
- “Related information for protectionLockout Aps” (page 220)

Properties for protectionLockout Aps

The following table describes the properties of the protectionLockout Aps command.

Table 93
Properties of the protectionLockout Aps command

Mode	Impact	Scope	Component
Operational or Provisioning	Service	Device	AutomaticProtectionSwitching (Aps)

Syntax for protectionLockout Aps

The following table describes the syntax for the protectionLockout Aps command.

Table 94
Syntax of the protectionLockout Aps command

If you want to...	Use the following syntax:
Prevent APS from using the protection line	protectionLockout Aps/<n>

Component instance parameter for protectionLockout Aps

An Aps component instance represents a link that is protected by APS. The Aps component manages both the working and protection lines. To lock the protection line on a particular link, you specify the instance number of the Aps component:

```
Aps/<n>
```

The instance value can be any integer between 0 and 15999.

Examples for protectionLockout Aps

The following examples show how to use the protectionLockout Aps command:

- “Locking the protection line in unidirectional mode example” (page 220)
- “Locking the protection line in bidirectional mode example” (page 220)

Locking the protection line in unidirectional mode example

You have an APS-protected link (represented by Aps/4) that you want to prevent from using the protection line. APS is currently in unidirectional mode. You enter the following command:

```
protectionLockout Aps/4
```

APS stops using the protection line as the active line on the near end of the link. However, the far end of the link, which operates independently in unidirectional mode, can still use its protection line for the active line.

Locking the protection line in bidirectional mode example

You have an APS-protected link (represented by Aps/4) that you want to prevent from using the protection line. APS is currently in bidirectional mode. You enter the following command:

```
protectionLockout Aps/4
```

APS stops using the protection line as the active line on the near end and the far end of the link.

Related information for protectionLockout Aps

See the following for information related to the protectionLockout Aps command:

- “Clear Aps command on Passport 7400” (page 83)
- “Switch Aps command on Passport 7400” (page 306)

ProtectionLockout Laps command on Passport 15000 and 20000

The protectionLockout Laps (protLock Laps) command forces a link protected by line automatic protection switching (Line APS) to stop using the protection line.

An Line APS-protected link has two lines: working and protection. The working line normally carries the data. The protection line acts as a backup to the working line. The line that is currently carrying the data is called the active line. When no manual overrides are in effect, the Line APS system automatically determines which line is active.

If the protection line is active when you enter this command, Line APS switches the data to the working line (making it active), even if the working line is degraded or has failed. If you have set the Line APS system to bidirectional mode and the far end of the link has already locked the protection line, the protectionLockout Laps command is ignored.

You can release the lock on the protection line using the clear Laps command. You can switch the active line between the working and protection line using the switch Laps command.

For more information on the protectionLockout Laps command, see

- “Properties for protectionLockout Laps” (page 222)
- “Syntax for protectionLockout Laps” (page 222)
- “Component instance parameter for protectionLockout Laps” (page 222)
- “Examples for protectionLockout Laps” (page 223)
- “Related information for protectionLockout Laps” (page 223)

Properties for protectionLockout Laps

The following table describes the properties of the protectionLockout Laps command.

Table 95
Properties of the protectionLockout Laps command

Mode	Impact	Scope	Component
Operational or Provisioning	Service	Device	LineAutomaticProtection Switching (Laps)

Syntax for protectionLockout Laps

The following table describes the syntax for the protectionLockout Laps command.

Table 96
Syntax of the protectionLockout Laps command

If you want to...	Use the following syntax:
Prevent LAPS from using the protection line	protectionLockout Laps/<n>

Component instance parameter for protectionLockout Laps

A Laps component instance represents a link that is protected by LAPS. The Laps component manages both the working and protection lines. To lock the protection line on a particular link, you specify the instance number of the Laps component:

```
Laps/<n>
```

The instance value can be any integer between 0 and 15999.

Examples for protectionLockout Laps

The following examples show how to use the protectionLockout Laps command:

- “Locking the protection line in unidirectional mode example” (page 223)
- “Locking the protection line in bidirectional mode example” (page 223)

Locking the protection line in unidirectional mode example

You have an LAPS-protected link (represented by Laps/4) that you want to prevent from using the protection line. LAPS is currently in unidirectional mode. You enter the following command:

```
protectionLockout Laps/4
```

LAPS stops using the protection line as the active line on the near end of the link. However, the far end of the link, which operates independently in unidirectional mode, can still use its protection line for the active line.

Locking the protection line in bidirectional mode example

You have an LAPS-protected link (represented by Laps/4) that you want to prevent from using the protection line. LAPS is currently in bidirectional mode. You enter the following command:

```
protectionLockout Laps/4
```

LAPS stops using the protection line as the active line on the near end and the far end of the link.

Related information for protectionLockout Laps

See the following for information related to the protectionLockout Laps command:

- “Clear Laps command on Passport 15000 and 20000” (page 85)
- “Switch Laps command on Passport 15000 and 20000” (page 313)

Quit command

The quit (exit) command ends your local or telnet session. It is identical to the logout command.

The logout and quit commands, along with their abbreviations, give you four ways to end your session:

- logout
- logoff
- quit
- exit

See “Logout command” (page 182) for more information.

Reconnect command

The reconnect command enables you to release the current connection and try to re-establish the connection using one of the following parameters:

- The primary path parameter releases the current connection and tries to reconnect using the primary path. This is the default parameter.
- The alternate path parameter releases the current connection and tries to reconnect using the alternate path. If the alternate path has not been provisioned, the reconnect command will fail and the call will be released.
- The automatic route selection parameter releases the current connection and tries to reconnect using automatic route selection, for example, a normal Dijkstra algorithm used for SVC path selection. If the *automaticFallback* attribute is disabled, the reconnect command will fail and the call will be released.

  Note: In the case where the connection is unable to re-establish itself on the desired path, the retry sequence is activated in the following order: primary path, followed by alternate path if provisioned, and ending with a normal retry, if the *automaticFallback* attribute is set.

For more information on the reconnect command, see

- “Properties for reconnect” (page 225)
- “Syntax for reconnect” (page 227)
- “Parameters for reconnect” (page 227)
- “Examples for reconnect” (page 228)

Properties for reconnect

The following table describes the properties of the reconnect command:

Table 97
Properties of the reconnect command

Mode	Impact	Scope	Component
Operational and Provisioning	Service	Connection	Manual Designated Transit List (MDTL)

Syntax for reconnect

The following table describes the syntax for the reconnect command.

Table 98
Syntax of the reconnect command

If you want to...	Use the following syntax:
Release the current connection and re-establish the connection	reconnect [-primary/-pri] | [-alternate/-alter] | [-automatic/-auto]

Parameters for reconnect

The reconnect command has the following options:

- “Primary path for reconnect” (page 227)
- “Alternate path for reconnect” (page 227)
- “Automatic route selection for reconnect” (page 227)

Primary path for reconnect

You specify which connection to release and reconnect with using the primary path option. This is the default behavior:

```
-primary
```

Alternate path for reconnect

You specify the alternate path as the reconnect path using the alternate path option:

```
-alternate
```

Automatic route selection for reconnect

You specify the automatic route to perform the reconnect using the automatic option:

```
-automatic
```

Examples for reconnect

The following examples illustrate how to use the reconnect command:

- “Reconnect to primary path example” (page 228)
- “Reconnect to alternate path example” (page 228)
- “Reconnect to automatic route selection example” (page 228)

Reconnect to primary path example

You want to reconnect using the primary path parameter. You enter the following command:

```
reconnect -pri atmif/10 vcc/0.100 src mdtl
```

Reconnect to alternate path example

You want to reconnect using the alternate path parameter. You enter the following command:

```
reconnect -alter atmif/10 vcc/0.100 src mdtl
```

Reconnect to automatic route selection example

You want to reconnect using the automatic route selection parameter. You enter the following command:

```
reconnect -auto atmif/10 vcc/0.100 src mdtl
```

ReloadCp Lp command

The reloadCp Lp (reload Lp) command lets you reload the control processor (CP) using a saved provisioning view.

By default, the reloadCp Lp command will only reload the control processor if there is an available standby control processor. To minimize CP down time, the standby CP loads the saved provisioning view and then becomes the active CP. The original active CP then loads the saved provisioning view and becomes the standby CP. If you have only a single control processor, you must use the force option to reload it.

Reloading the CP can cause some or all function processors to reload. Depending on the software to be loaded, standby function processors may be reloaded and become active as the standby CP becomes active.

For more information on the reloadCp Lp command, see

Properties for reloadCp Lp

The following table describes the properties of the reloadCp Lp command:

Table 99
Properties of the reloadcp command

Mode	Impact	Scope	Components
Operational and Provisioning	Service	Device	LogicalProcessor (Lp)

Syntax for reloadCp Lp

The following table describes the syntax for the reloadCp Lp command.

Table 100
Syntax of the reloadCp Lp command

If you want to...	Use the following syntax:
Reload the control processor using a saved provisioning view	reloadCp -file(<viewname>) Lp/0
Reload the control processor when the node has a single CP	reloadcp -file(<viewname>) -force Lp/0

Parameters for reloadCp Lp

The reloadCp Lp command has the following options:

- "Saved view parameter for reloadCp Lp" (page 230)
- "Single-CP node force parameter for reloadCp Lp" (page 230)

Saved view parameter for reloadCp Lp

You specify which saved view that you want to reload on the control processor using the file option:

```
-file(<viewname>)
```

Enter the name of the saved view in the option. For information on the naming of saved views, refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview*.

Single-CP node force parameter for reloadCp Lp

You can force the reloadCp Lp command to work on a single-CP node using the force option:

```
-force
```

This option has no effect on a dual-CP node.

Examples for reloadCp Lp

The following examples illustrate how to use the reloadCp Lp command:

- “Reloading with a saved view on a dual-CP node example” (page 231)
- “Reloading with a saved view on a single-CP node example” (page 231)

Reloading with a saved view on a dual-CP node example

You want to reload a control processor on a dual-CP node using a saved view. You enter the following command:

```
reloadCp -file(ABCD1.full.001) Lp/0
```

The standby CP loads the new view and becomes the active CP. The active CP then loads the new view and becomes the standby CP.

Reloading with a saved view on a single-CP node example

You want to reload a control processor on a single-CP node using a saved view. You enter the following command:

```
reloadCp -file(ABCD1.full.001) -force Lp/0
```

The CP restarts and loads the saved view. Service on the node is interrupted during the reload.

Related information for reloadCp Lp

See the following for information related to the reloadCp Lp command:

- “Activate Prov command” (page 48)
- “Apply Prov command” (page 60)
- “Load Prov command” (page 173)
- “Reset <processor> command” (page 244)
- “Restart <processor> command” (page 253)

Remove Fs command

The remove Fs command lets you delete files and directories. If you remove a directory, you delete all its contents, including all its subdirectories.

If a file is protected from deletion, you have to turn its protection off using the unprotect Fs command before removing it. You can force the remove Fs command to delete protected files using the force option.

CAUTION
Possible system failure
Do not remove the directories in the root directory (/). These directories contain necessary system data and software. If you remove the /tmp directory the system fails.

For more information on the remove Fs command, see

- "Properties for remove Fs" (page 232)
- "Syntax for remove Fs" (page 233)
- "Parameters for remove Fs" (page 233)
- "Examples for remove Fs" (page 234)
- "Related information for remove Fs" (page 235)

Properties for remove Fs

The following table describes the properties of the remove Fs command.

Table 101
Properties of the remove Fs command

Mode	Impact	Scope	Components
Operational or Provisioning	Configuration	Device	FileSystem (Fs)

Syntax for remove Fs

The following table describes the syntax for the remove Fs command.

Table 102
Syntax of the remove Fs command

If you want to...	Use the following syntax:
Remove a file	remove -path(<filepath>) Fs
Remove a directory and all its contents	remove -path(<directorypath>) -recursive Fs
Remove a protected file	remove -path(<filepath>) -force -recursive Fs
Remove a directory and all its contents including protected files	remove -path(<directorypath>) -recursive -force Fs

Parameters for remove Fs

There are three parameters for the remove Fs command:

- “File or directory to remove parameter” (page 233)
- “Remove subdirectories parameter” (page 234)
- “Remove protected files parameter” (page 234)

File or directory to remove parameter

You specify which file or directory you want to remove using the path option:

```
-path(<filepath>)
```

```
-path(<directorypath>)
```

Enter the path to the file or directory you want to remove. If you specify a directory, you must also enter the recursive option. For information on path syntax, refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview*.

If you try to remove a protected file, the command fails. If you remove a directory that contains protected files, the command does not delete the protected files or directories that contain them. It does delete all other files and directories.

You must specify this option.

Remove subdirectories parameter

You can remove the contents of all subdirectories using the recursive option:

```
-recursive
```

To use this option, you must specify a directory in the path option.

Whenever deleting a directory, you must specify this option even if the directory contains no subdirectories.

Remove protected files parameter

You can remove files even if they are protected using the force option:

```
-force
```

Examples for remove Fs

The following examples illustrate how to use the remove Fs command:

- “Removing a file and directory example” (page 234)
- “Removing a protected file example” (page 235)

Removing a file and directory example

You want to delete the file account.05051334.010 in your current working directory. You enter the following command:

```
remove -path(“account.05051334.010”) Fs
```

You now want to remove a copy of the closed subdirectory you created in the /tmp directory. You enter the following command:

```
remove -path(“/tmp/closed”) Fs
```

The command removes the closed subdirectory along with all its subdirectories.

Removing a protected file example

You have a test directory in the /tmp directory containing the protected file test.info. Your current working directory is /tmp. You want to remove the test.info file. You enter the following command:

```
remove -path(“test/test.info”) -force Fs
```

You now want to remove the entire test directory. It still contains a number of protected files. You enter the following command:

```
remove -path(“test”) -recursive -force Fs
```

Related information for remove Fs

See the following for information related to the remove Fs command:

- “Chdir Fs command” (page 68)
- “ListFile Fs command” (page 168)
- “Move Fs command” (page 188)
- “Protect Fs command” (page 215)
- “Unprotect Fs command” (page 363)

Remove Sw Av command

The remove Sw Av command lets you remove an unused software application version from the file system. When you remove an application version, you also remove all of its associated patches.

The remove Sw Av command considers a software application version unused if it is not in the application version list (the *avList* attribute of the *Software* component) of any semantically checked view. A semantically checked view can include the current view, the edit view, and saved views.

If a semantically checked view uses the application version, the remove Sw Av command fails. The command also fails if a start Sw Dld, tidy Sw, check Prov, or another remove Sw Av command is in progress.

You can remove all unused software applications using the tidy Sw command.

For more information on the remove Sw Av command, see

- "Properties for remove Sw Av" (page 237)
- "Syntax for remove Sw Av" (page 237)
- "Application version parameter for remove Sw Av" (page 237)
- "Removing an unused application version example" (page 237)
- "Related information for remove Sw Av" (page 238)

Properties for remove Sw Av

The following table describes the properties of the remove Sw Av command.

Table 103
Properties of the remove Sw Av command

Mode	Impact	Scope	Components
Operational or Provisioning	Configuration	Device	Software ApplicationVersion (Sw Av)

Syntax for remove Sw Av

The following table describes the syntax for the remove Sw Av command.

Table 104
Syntax of the remove Sw Av command

If you want to...	Use the following syntax:
Remove an unused software application version	remove Sw Av/<application_version>

Application version parameter for remove Sw Av

You specify which application version you want to remove using the instance value of the *ApplicationVersion* (*Av*) component:

```
Sw Av/<application_version>
```

Enter the name of the application.

Removing an unused application version example

You want to remove the unused software application, ip_AA08A and all its associated patches. You enter the following command:

```
remove Sw Av/ip_AA08A
```

You receive the following response:

```
Sw Av/IP_AA08
   The following AV(s) have been removed:
       ip_AA008
```

Related information for remove Sw Av

See the following for information related to the remove Sw Av command:

- “Start Sw Dld command” (page 291)
- “Stop Sw Dld command” (page 303)
- “Tidy Sw command” (page 341)

Replay aaserv command

Use the replay aaserv command to cause all active alarms contained in the active alarm list (AAList) to be delivered to the session (for example, Nmis Fmip Session/<x>) that initiates the replay command.

CAUTION
Preside MDM applicability only
This command is only applicable internally on a Preside Multiservice Data Manager.

For more information on the replay aaserv command, see

- “Properties for replay aaserv” (page 239)
- “Syntax for replay aaserv” (page 240)
- “Related information for replay aaserv” (page 240)

Properties for replay aaserv

The following table describes the properties of the replay aaserv command:

Table 105
Properties of the replay aaserv command

Mode	Impact	Scope	Component
Operational	Passive	Device	ActiveAlarmServices

Syntax for replay aaserv

The following table describes the syntax for the replay aaserv command.

Table 106
Syntax of the replay aaserv command

If you want to...	Use the following syntax:
Forward all alarms from the AAList to your Preside Multiservice Data Manager	replay aaserv
Note: The replay aaserv command is only supported when used with a Preside Multiservice Data Manager. It should not be manually invoked by an operator. Refer to 241-6001-303 *Preside MDM Administrator Guide* for details on working with the AAList through Preside Multiservice Data Manager.	

When initiated, the replay aaserv command displays a summary of all active alarms from the AAList, as well as a response stating that all active alarms have been successfully replayed.

Note: The maximum number of concurrent replay aaserv commands allowed to run on a Passport is 8. From any given session however, the maximum number of concurrent replay aaserv commands allowed to run is 1.

Related information for replay aaserv

See the following for information related to the replay aaserv command:

- “Clear -notificationId(<id>) -comment(<text>) aaserv command” (page 79)

Reroute command

The reroute command lets you see if a better route exists for a soft PVC or soft PVP and establishes that better route if one exists. Invoking the reroute command will activate the path optimization even though the connection may not have been recovered.

Note that the reroute command will fail if there is another optimization pass already in progress on the module (node) or at the EBR interface. This command cannot be cancelled or deferred.

For more information on the reroute command, see

- "Properties for reroute" (page 241)
- "Syntax for reroute" (page 242)
- "Examples for reroute" (page 242)
- "Related information for reroute" (page 243)

Properties for reroute

The following table describes the properties of the reroute command.

Table 107
Properties of the reroute command

Mode	Impact	Scope	Component
Operational	Configuration	Device	AtmIf Vcc|Vpc|Vpt/Vcc Src EbrOv, RGty Call Seg LCo

Syntax for reroute

The following table describes the syntax for the reroute command.

Table 108
Syntax of the reroute command

If you want to...	Use the following syntax:
See if a better route exists for a connection and if one exists, establish the better route	reroute AtmIf/<n> Vcc/<vpi.vci> Src EbrOv or reroute AtmIf/<n> Vpc/<vpi> Src EbrOv or reroute AtmIf/<n> Vpt/<vpi> Vcc/<vci> Src EbrOv
Force calls (on an outbound gateway) to reroute to a better gateway link or gateway node	reroute RGty/<x> Call/<y> Seg/toDestination LCo
Force calls (on an inbound gateway) to reroute over a better path to a gateway or destination node	reroute RGty/<x> Call/<y> Seg/toDestination LCo

Examples for reroute

The following examples illustrate how to use the reroute command:

- “ATM interface example” (page 242)
- “Routing gateway example” (page 243)

ATM interface example

You want to see if a better route exists for a connection and, if one exists, establish the better route. You enter the following command:

```
reroute AtmIf/<AtmIf> Vcc/<vpi.vci>|Vpc/<vpi>|
Vpt/<vpi> Vcc/vpi> Src EbrOv
```

When the command completes successfully, you receive the following response:

```
Path optimization completed.
```

If a path optimization is in progress when you use the optimize command, you receive the following response:

```
A path optimization pass is already in progress at this
interface.
```

Routing gateway example

You want to reroute an individual logical connection between routing gateways. You enter the following command on the outbound gateway node:

```
reroute routingGateway/<x> Call/<y> Segment/
toDestination LogicalConnection
```

The scope of this command is limited to the individual call segment between the two gateway nodes. Unlike the optimize command, which only reroutes a call segment if a better path exists, the reroute command forces the call segment to reroute. The best currently available path will be taken; however, there is no guarantee it will be better than the former path.

Related information for reroute

See the following for information related to the reroute command:

- “Display command” (page 121)
- “List command” (page 159)
- “Optimize command” (page 195)

Reset <processor> command

The reset <processor> command resets the hardware, runs the self tests, and reloads the software for one or more processor cards. You can issue the command against a processor card, a logical processor (LP), or the entire shelf.

When a control processor resets, it reloads the software specified in its committed provisioning view. When a function processor resets, it reloads the software specified for it in the current view. You cannot reset a control processor while a disk synchronization is in progress.

CAUTION
Data loss may occur
When you reset a processor card it is temporarily unable to provide service. During the time the processor is resetting, data can be lost.

CAUTION
Loss of stable SVC connections may occur
When you reset a processor card, a loss of SVC connections may occur. Although stable SVC calls should remain active in a redundant processor configuration, exercise caution when issuing this command.

In some cases, a reset <processor> command causes a switchover from the active processor card to the standby processor card. A switchover can occur for both spared function processors and spared control processors. Special restrictions apply when causing a CP switchover. Before executing the *reset Lp*/0 command, the value of the *revertibleTimerCountdown* attribute, under the Shelf component, must be zero. See 241-5701-600 *Passport 7400, 15000, 20000 Configuration Guide* for information.

An alternative to resetting a processor is to restart it using the restart <processor> command. When you restart a processor, the hardware resets but the software only reinitializes. Although the restart <processor> command does not completely reset the processor card, it brings it back into service more quickly. However, data loss can still occur.

For more information on the reset <processor> command, see

Properties for reset <processor>

The following table describes the properties of the reset <processor> command.

Table 109
Properties of the reset <processor> command

Mode	Impact	Scope	Components
Operational and Provisioning	Service	Device	Shelf Shelf Card LogicalProcessor (Lp)

Syntax for reset <processor>

The following table describes the syntax for the reset <processor> command.

CAUTION
Data loss may occur
When you reset a processor card it is temporarily unable to provide service. During the time the processor is resetting, data can be lost.

CAUTION
Loss of stable SVC connections may occur
When you reset a processor card, a loss of SVC connections may occur. Although stable SVC calls should remain active in a redundant processor configuration, exercise caution when issuing this command.

Table 110
Syntax of the reset <processor> command

If you want to...	Use the following syntax:
Reset a particular processor card	reset Shelf Card/<n>
Reset the active processor card of a logical processor	reset Lp/<n>
Reset all processor cards on the shelf	reset Shelf

Processor identifier parameters for reset <processor>

You identify which processor or processors you want to reset using the components of the command.

If you issue the command against a *Shelf Card* component, it resets that particular processor card. If you issue the command against a *LogicalProcessor* (*Lp*) component, it resets the active processor card for that logical processor. The standby processor card for the logical processor (if present) does not reset. In most cases, a reset command causes a switchover from the active to standby processor.

If you issue the reset command against the *Shelf* component, it resets all the processor cards on the shelf.

CAUTION
Data loss may occur
When you reset a processor card it is temporarily unable to provide service. During the time the processor is resetting, data can be lost.

CAUTION
Loss of stable SVC connections may occur
When you reset a processor card, a loss of SVC connections may occur. Although stable SVC calls should remain active in a redundant processor configuration, exercise caution when issuing this command.

Table 111
Components for the reset <processor> command

Component	Description
Shelf Card/<n>	To reset a particular processor card, enter the slot number of the card.
Lp/<n>	To reset the active processor card of a logical processor, enter the number of the logical processor.
Shelf	To reset all the processors on the shelf, use the *Shelf* component.

Examples for reset <processor>

CAUTION
Data loss may occur
When you reset a processor card it is temporarily unable to provide service. During the time the processor is resetting, data can be lost.

CAUTION
Loss of stable SVC connections may occur
When you reset a processor card, a loss of SVC connections may occur. Although stable SVC calls should remain active in a redundant processor configuration, exercise caution when issuing this command.

The following examples show how to use the reset <processor> command:

- "Resetting a processor card example" (page 248)
- "Resetting a logical processor example" (page 248)
- "Resetting all processor cards example" (page 248)

Resetting a processor card example

You want to reset the processor card in slot 2. You enter the following command:

```
reset Shelf Card/2
```

The processor card in slot 2 resets.

Resetting a logical processor example

You want to reset logical processor 4. You enter the following command:

```
reset Lp/4
```

The active processor card for the logical processor resets.

Resetting all processor cards example

You want to reset the processor cards on the shelf. You enter the following command:

```
reset Shelf
```

All processor cards reset.

Related information for reset <processor>

See the following for information related to the reset <processor> command:

- “ReloadCp Lp command” (page 229)
- “Restart <processor> command” (page 253)
- “Switchover Lp command” (page 320)

Restart command

The restart command is only used when operating the Passport Packet Voice Gateways (PVG) application with ATM provisioned switched virtual connections (provisioned SVCs) and switched permanent virtual connections (SPVCs).

The restart command is used to restart the automatic connection procedure. Typically, this is done after the automatic connection procedure has been abandoned after reaching the maximum number of retries allowed. However, the restart command may be used before this maximum is reached.

If the restart command is used on an existing ATM connection, the restart command clears the existing connection and attempts to make a new connection using the current active address.

For more information on the restart command, see

- “Properties for restart” (page 251)
- “Syntax for restart” (page 251)
- “Examples for restart” (page 252)

You can issue the restart command only against the components listed in “Properties for restart” (page 251).

Properties for restart

The following table describes the properties of the restart command.

Table 112
Properties of the restart command

Mode	Impact	Scope	Components
Operational	Service	Device	Nsta/<n> Vgs AtmTConn/<m> Aap
			Nsta/<n> Vgs AtmTConn/<m> SpvcAp
			Nsta/<n> Vgs IpMConn/<m> Aap
			Nsta/<n> Vgs IpMConn/<m> SpvcAp
			Nsta/<n> Vgs Ctrl/<m> Aap
			Nsta/<n> Vgs Ctrl/<m> SpvcAp
			Nsta/<n> Vgs Conn/<m> Aap
			Nsta/<n> Vgs Conn/<m> SpvcAp

Syntax for restart

The following table describes the syntax for the restart command.

Table 113
Syntax of the restart command

If you want to...	Use the following syntax:
Restart the automatic connection procedure for a trunk using a provisioned SVC active access point.	restart Nsta/<n> Vgs <trunktype>/<m> Aap where <trunktype> can be AtmTConn, IpMConn, Conn, or Ctrl
Restart the automatic connection procedure for a trunk using an SPVC access point.	restart Nsta/<n> Vgs <trunktype>/<m> SpcvAp where <trunktype> can be AtmTConn, IpMConn, Conn, or Ctrl

Examples for restart

The following examples show how to use the restart command:

- "Restarting a trunk with bearer traffic using a provisioned SVC active access point" (page 252)
- "Restarting a signaling trunk using an SPVC access point" (page 252)

Restarting a trunk with bearer traffic using a provisioned SVC active access point

An ATM trunk carrying bearer traffic has been using the PVG retry process to re-establish a provisioned switched virtual connection (provisioned SVC) to an ATM address. The maximum number of retries were attempted, but all were unsuccessful. You have corrected the problem in the ATM network that was preventing the provisioned SVC from being established. You want to restart the automatic connection procedure. You enter the following:

```
restart Nsta/<n> Vgs AtmTConn/<m> Aap
```

The provisioned SVC is re-established.

Restarting a signaling trunk using an SPVC access point

An signaling trunk has been using the PVG retry process to re-establish a switched permanent virtual connection (SPVC) to an ATM address. The maximum number of retries were attempted, but all were unsuccessful. You have added the ATM address of a backup switch or have corrected the initial problem and want to re-establish the trunk. You want to restart the automatic connection procedure to connect to the backup switch. You enter the following:

```
restart Nsta/<n> Vgs Ctrl/<m> SpvcAp
```

The SPVC is re-established to the backup switch.

Restart <processor> command

The restart <processor> command resets the hardware (except for memory), runs the self tests, and reinitializes the software of a processor card. You can issue the command against a processor card or a logical processor (LP).

For the restart <processor> command to work on a control processor, the control processor must be running a committed view. If it is running an uncommitted view, the control processor resets instead of restarting. You cannot restart a control processor while a disk synchronization is in progress.

CAUTION
Data loss may occur
When you restart a processor it is temporarily unable to provide service. During the time the processor is restarting, data can be lost.

CAUTION
Loss of stable SVC connections may occur
When you restart a processor card, a loss of SVC connections may occur. Although stable SVC calls should remain active in a redundant processor configuration, exercise caution when issuing this command.

In some cases, a restart <processor> command causes a switchover from the active processor card to the standby processor card. A switchover can occur for both spared function processors and spared control processors. If you cause the switchover of control processors twice within 10 minutes, Passport automatically performs a shelf reset. Passport considers a second CP switchover within 10 minutes to be an indication of a more serious fault.

An alternative to restarting a processor is to reset it using the reset <processor> command. When you reset a processor, the hardware resets and the software reloads. The reset <processor> command provides a complete initialization of a processor card, but it takes longer to bring the processor card back into service.

For more information on the restart <processor> command, see

Properties for restart <processor>

The following table describes the properties of the restart <processor> command.

Table 114
Properties of the restart <processor> command

Mode	Impact	Scope	Components
Operational and Provisioning	Service	Device	Shelf Card LogicalProcessor (Lp)

Syntax for restart <processor>

The following table describes the syntax for the restart <processor> command.

CAUTION
Data loss may occur
When you restart a processor it is temporarily unable to provide service. During the time the processor is restarting, data can be lost.

CAUTION
Loss of stable SVC connections may occur
When you restart a processor card, a loss of SVC connections may occur. Although stable SVC calls should remain active in a redundant processor configuration, exercise caution when issuing this command.

Table 115
Syntax of the restart <processor> command

If you want to...	Use the following syntax:
Restart a processor card	restart Shelf Card/<n>
Restart the active processor card of a logical processor	restart Lp/<n>

Processor identifier parameters for restart <processor>

You identify which processor you want to restart using the components of the command.

If you issue the command against a *Shelf Card* component, it restarts that particular processor card. If you issue the command against a *LogicalProcessor* (*Lp*) component, it restarts the active processor card for that logical processor. The standby processor card for the logical processor (if present) does not restart. In most cases, a restart command causes a switchover from the active to standby processor.

CAUTION
Data loss may occur
When you restart a processor it is temporarily unable to provide service. During the time the processor is restarting, data can be lost.

CAUTION
Loss of stable SVC connections may occur
When you restart a processor card, a loss of SVC connections may occur. Although stable SVC calls should remain active in a redundant processor configuration, exercise caution when issuing this command.

Table 116
Components for the restart <processor> command

Component	Description
Shelf Card/<n>	To restart a particular processor card, enter the slot number of the card.
Lp/<n>	To restart the active processor card of an LP, enter the number of the LP.

Examples for restart <processor>

CAUTION
Data loss may occur
When you restart a processor it is temporarily unable to provide service. During the time the processor is restarting, data can be lost.

CAUTION
Loss of stable SVC connections may occur
When you restart a processor card, a loss of SVC connections may occur. Although stable SVC calls should remain active in a redundant processor configuration, exercise caution when issuing this command.

The following examples show how to use the restart <processor> command:

- “Restarting a processor card example” (page 257)
- “Restarting a logical processor example” (page 257)

Restarting a processor card example

You want to restart the processor card in slot 1. You enter the following:

```
restart Shelf Card/1
```

The processor card in slot 1 restarts.

Restarting a logical processor example

You want to restart logical processor 4. You enter the following command:

```
restart Lp/4
```

The active processor card for the logical processor restarts.

Related information for restart <processor>

See the following for information related to the restart <processor> command:

- “ReloadCp Lp command” (page 229)
- “Reset <processor> command” (page 244)
- “Switchover Lp command” (page 320)

Restore Prov command

Use the restore Prov command to reactivate the journaled current view if a switch resets to the committed view. The attribute *restorePossible* is set to yes in the case where the restore Prov command can be issued to restore the journaled current view.

CAUTION
If you do not confirm the restore Prov command (see "Confirm Prov command" (page 100)) within 20 minutes of issuing it, a system restart occurs and the system reactivates the last committed view as the current view (see "Commit Prov command" (page 96)). A service disruption occurs while the restart is in progress.

Note: Restoring a view can delete certain components in the current view. Some components may temporarily go out of service if the changes are critical to the provisioned data.

For more information on the restore Prov command, see

- "Properties for restore Prov" (page 258)
- "Syntax for restore Prov" (page 259)
- "Examples for restore Prov" (page 259)
- "Related information for restore Prov" (page 260)

Properties for restore Prov

The following table describes the properties of the restore Prov command.

Table 117
Properties of the restore Prov command

Mode	Impact	Scope	Component
Operational or Provisioning	Service	Application	ProvisioningSystem (Prov)

Syntax for restore Prov

The following table describes the syntax for the restore Prov command.

Table 118
Syntax of the restore Prov command

If you want to...	Use the following syntax:
reactivate the current journal view after a CP-switchover or switch reset	restore Prov

Examples for restore Prov

The following examples show how to use the restore Prov command:

- “Verifying that a restore is possible example” (page 259)
- “Restoring the current journaled view example” (page 260)

Verifying that a restore is possible example

You want to verify that a restore is possible. You enter the following command:

```
display Prov restorePossible
```

You receive the following response:

```
Prov
   restorePossible = yes
```

```
ok                          <timestamp>
```

Restoring the current journaled view example

You want to restore the current journaled view, then issue the restore prov command. You enter the following command:

```
restore Prov
```

You receive the following response:

```
Prov
    You have 20 minutes in which to confirm the
restore.
ok                          <timestamp>
```

You must perform a confirm Prov after the restore finishes. Refer to the “Confirm Prov command” (page 100).

Related information for restore Prov

See the following items for information related to the restore Prov command:

- “Activate Prov command” (page 48)
- “Commit Prov command” (page 96)
- “Confirm Prov command” (page 100)

Run Shelf Test command

The run Shelf Test command lets you manually test the bus clock source. The test verifies that all operational processor cards can receive signals from the clock source.

CAUTION
Risk of data loss
The bus clock source test can cause minor data loss on the bus. Only run this test when bus utilization is low.

Passport can automatically test the bus clock source. If you disable automatic bus clock source testing, you should manually test the bus clock source using the run Shelf Test command at least once a month. You can view information on the current status of the bus clock source by displaying the *clockSourceStatus* attribute of the *Shelf Bus* components.

To test the bus clock source, you must set the *type* attribute of the *Shelf Test* component to *busClock*. After performing the test using the run Shelf Test command, you view the test results by displaying the *busClockTestResult* attribute of the *Shelf Test* component.

For more information on the run Shelf Test command, see

- "Properties for run Shelf Test" (page 262)
- "Syntax for run Shelf Test" (page 262)
- "Running a bus clock source test example" (page 262)
- "Related information for run Shelf Test" (page 263)

Properties for run Shelf Test

The following table describes the properties of the run Shelf Test command.

Table 119
Properties of the run Shelf Test command

Mode	Impact	Scope	Components
Operational	Service	Device	Shelf Test

Syntax for run Shelf Test

The following table describes the syntax for the run Shelf Test command.

Table 120
Syntax of the run Shelf Test command

If you want to...	Use the following syntax:
Perform a bus clock source test	run Shelf Test

Running a bus clock source test example

You currently have automatic bus clock source testing disabled. You want to manually test the bus clock source. Before running the test, you ensure that the test type is correctly set using the following command:

```
set Shelf Test type busClock
```

You then run the test using the following command:

```
run Shelf Test
```

After the test stops, you display the results of the test using the following command:

```
display Shelf Test busClockTestResult
```

If all operational processor cards can receive signals from the clock source, you receive the following response:

```
Shelf Test
    busClockTestResult = pass
```

Related information for run Shelf Test

See the following for information related to the run Shelf Test command:

- “Start <hardware> Test command” (page 282)
- “Stop <hardware> Test command” (page 296)

Save Prov command

Use the save Prov command to store all or part of either a current view or an edit view in a saved view.

Saved views can be either full or partial:

- A full view contains all the configuration data from an edit view or a current view. The full view is useful if you wish to save a particular view for use at a later date, or if you want to save a particular node configuration and apply it to other nodes.
- A partial view contains only the configuration data for a particular component from an edit view. Partial views are useful for saving component provisioning that you wish to use on other nodes. With the partial view, you only have to provision a component once, save it as a partial view, and then copy that partial view onto as many nodes as you wish.

The provisioning system can also store a saved view in one of several formats.

- The portable format includes all the configuration data necessary to run on a view node.
- The ASCII format allows non-Passport systems to read a saved view.
- The ASCII flat format enables you to save a Passport provisioning file on the disk in a format that you can FTP back to a workstation for editing purposes.
- The commit format is a specialized version of the current view that allows for fast activation (see “Commit Prov command” (page 96)).
- The delta format is a view that includes only the changes made between the edit view and the current view. An edit view in delta format requires that the current view is in portable format.
- The partial format includes all the information about a component.

For more detailed information on full and partial saved views and saved view formats, see 241-5701-045 *Passport 7400, 15000, 20000 Management System User Interface Guide*.

Passport automatically increments the version number every time you save a version of a view. If you save a view using a new name or type, the system sets the version number to 001.

For more information on the save Prov command, see

- “Properties for save Prov” (page 265)
- “Syntax for save Prov” (page 266)
- “Parameters for save Prov” (page 267)
- “Examples for save Prov” (page 269)
- “Related information for save Prov” (page 271)

Properties for save Prov

The following table describes the properties of the save Prov command.

Table 121
Properties of the save Prov command

Mode	Impact	Scope	Components
Provisioning	Configuration	Application	ProvisioningSystem (Prov)

Syntax for save Prov

The following table describes the syntax for the save Prov command.

Table 122
Syntax of the save Prov command

If you want to...	Use the following syntax:
Save the provisioning data of the edit view, and have the system assign a default file name	save Prov ***Note:*** Using the save Prov command without options causes the system to save the edit view in a delta format. If the current view is not in portable format, the system first saves the current view in portable format then saves the edit view in a delta format.
Save the provisioning data of the current view, and have the system assign a default file name	save -current Prov
Save the provisioning data of the edit view (default) or the current view (using the current option), to a specific file	save -file(<file_name>) Prov save -current -file(<file_name>) Prov
Save the provisioning data of the edit view (default) or the current view (using the current option) in ascii format	save -ascii Prov save -current -ascii Prov
(Sheet 1 of 2)	

Table 122 (continued)
Syntax of the save Prov command

If you want to...	Use the following syntax:
Save the provisioning data of the edit view (default) or the current view (using the current option) in ascii flat format	save -asciiFlat Prov save -current -asciiFlat Prov
Save from the edit view (default) or the current view (using the current option), to a specific file, the provisioning data of a component	save -file(<file_name>) -component(<component_name>) Prov save -current -file(<file_name>) -component(<component_name>) Prov
Save a complete file to disk	save -file(<file_name>) -portable Prov
(Sheet 2 of 2)	

Parameters for save Prov

If you don't use any options with the save Prov command, the system saves the edit view using the last loaded, saved, or committed view name, and increments the number by 1.

There are six types of parameters for the save Prov command:

- "Save the current view parameter for save Prov" (page 268)
- "Save a view with a specific filename parameter for save Prov" (page 268)
- "Save a view in ASCII format parameter for save Prov" (page 268)
- "Save the provisioning data for a component parameter for save Prov" (page 268)
- "Save a view in portable format parameter for save Prov" (page 269)

Save the current view parameter for save Prov

You can specify that the provisioning system save the provisioning data of the current view using the current option:

```
-current
```

If this option is not used, the edit view is saved.

Save a view with a specific filename parameter for save Prov

You can specify that the provisioning system save a view to a specific filename using the file option:

```
-file(<file_name>)
```

If the filename you specify already exists, the provisioning system will increment the version number automatically. If the filename you specify does not exist, the provisioning system will create it for you, and give it the version number 001.

The file option identifies the saved view using only the name portion of the three-part filename. For details on filename syntax, refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview*.

The name must be between 1 and 31 characters and can only contain letters, numbers, and underscore characters.

Save a view in ASCII format parameter for save Prov

You can specify that the provisioning system save a view in ASCII format using the ascii option:

```
-ascii
```

You can use the ASCII format to export a view to a non-Passport application.

Save the provisioning data for a component parameter for save Prov

You can specify that the provisioning system save the provisioning data for a component in a partial saved view using the component option:

```
-component(<component_name>)
```

Use the component option to store a specific component and its subcomponent hierarchy in a partial saved view.

You must use the file option with this option.

Save a view in portable format parameter for save Prov

You can specify that the provisioning system save a view in portable format using the portable option:

```
-portable
```

Use this option to create a view that the system on another Passport node can reuse.

Examples for save Prov

The following examples show how to use the save Prov command:

- "Saving a view example" (page 269)
- "Saving a view with a specific filename example" (page 270)
- "Saving component provisioning data example" (page 270)
- "Saving a view in ASCII format example" (page 270)
- "Saving a view in portable format example" (page 271)

Saving a view example

You want to save the current view and have the system assign a default file name. You enter the following command:

```
save -current Prov
```

You receive the following response:

```
Prov
   The provisioning data is saved in file
   BE0019A.full.008.
```

Saving a view with a specific filename example

You want to save the provisioning data of either the edit view or current view to a specific filename. For this example, the assumption is that you want to save the current view to the file test1. You enter the following command:

```
save -current -file(test1) Prov
```

You receive the following response:

```
Prov
   Saving the current view into test1.full.001 (with
   portable formats) ...
   The provisioning data is saved in file
   test1.full.001.
```

Saving component provisioning data example

You want to store a component into a partial saved view. For this example, the assumption is that you want to save the *FrUni/1* component from the current view to the file FR_test. You enter the following command:

```
save -current -file(FR_test) -component(FrUni/1) Prov
```

You receive the following response:

```
Prov
   Saving the current view into FR_test.part.001 (with
   part formats) ...
   The provisioning data is saved in file
   AXL_test.part.001.
```

Saving a view in ASCII format example

You want to store the current view in ASCII format. For this example, the assumption is that you want to save the current view in ASCII format to the file current1_ascii. You enter the following command:

```
save -current -ascii -file(current_ascii) Prov
```

You receive the following response:

```
Prov
   Saving the current view into current_ascii.full.001
   (with ascii formats) ...
   The provisioning data is saved in file
   current1_ascii.full.001.
```

Saving a view in portable format example

You want to save a complete provisioning view to disk. For this example, the assumption is that you want to save the file BE0019A in a portable format. You enter the following command:

```
save -file(BE0019A) -portable Prov
```

You receive the following response:

```
Prov
   Saving the edit view into BE0019A.full.009 (with
   portable formats) ...
   The provisioning data is saved in file
   BE0019A.full.009.
```

Related information for save Prov

See the following for information related to the save Prov command:

- “Activate Prov command” (page 48)
- “Apply Prov command” (page 60)
- “Load Prov command” (page 173)
- “ReloadCp Lp command” (page 229)
- “Stop Prov command” (page 300)

Set command

The set (s) command allows you to change the value of an attribute.

You can only set provisionable attributes while you are in provisioning mode. All changes you make within provisioning mode affect the edit view. You cannot set the provisionable attributes of the current view.

You can set some operational attributes, such as test setup attributes. You can set these operational attributes in both operational and provisioning mode. The changes only affect the current view and are lost when the node restarts.

When setting an attribute's value, you need to know the data type of the attribute. A data type specifies the format of the value or values an attribute can assume. For detailed information on data types, see 241-5701-060 *Passport 7400, 15000, 20000 Components*.

Some attributes can have multiple values. You can add (turn on) and delete (turn off) values for a multiple-value attribute using the set command.

For more information on the set command, see

- "Properties for set" (page 272)
- "Syntax for set" (page 273)
- "Parameters for set" (page 274)
- "Examples for set" (page 278)
- "Related information for set" (page 281)

Properties for set

The set command works with all writable attributes (defined by their write access value). Some attributes require a higher impact than service. An attribute's component defines its required scope. To find specific write access, impact, and scope information for an attribute, see 241-5701-060 *Passport 7400, 15000, 20000 Components* or use the help command.

The following table describes the properties of the set command.

Table 123
Properties of the set command

Mode	Impact	Scope	Attribute	View
Operational	Service	Depends on component	Operational	Current
Provisioning	Service	Depends on component	Provisionable and Operational	Edit and Current

Syntax for set

The following table describes the syntax for the set command.

Table 124
Syntax of the set command

If you want to...	Use the following syntax:
Set the value of a single-value attribute	set <component> <attribute> <value>
Set values for a multiple-value attribute	set <component> <attribute> <value> [<value>]...
Set the value for more than one attribute of a component	set <component> <attribute> <value> [, <attribute> <values>]...
Delete the value of a single-value attribute	set <component> <attribute> set <component> <attribute> !
(Sheet 1 of 2)	

Table 124 (continued)
Syntax of the set command

If you want to...	Use the following syntax:
Delete (turn off) all values for a multiple-value attribute	set <component> <attribute> !
Delete (turn off) a value for a multiple-value attribute	set <component> <attribute> ~<value>
(Sheet 2 of 2)	

Parameters for set

The following are the parameters for the set command:

- "Component parameter for set" (page 274)
- "Data types for set" (page 274)
- "Attribute and value parameters for set" (page 276)
- "Delete value parameters for set" (page 277)

Component parameter for set

You specify which component's attributes you want to set using the component name:

<component>

Data types for set

An attribute has a data type. The data type defines the acceptable values for the attribute. There are two kinds of data types: single value and multiple value. Single-value data types accept only one value for the attribute. Multiple-value data types accept more than one value for the attribute.

Multiple-value data types are based on single-value data types. For example, the set of ... multiple-value data type can be based on the decimal, enumeration, or hexadecimal number single-value data types. An attribute with a set of decimal data type accepts a set of decimal values.

"Help information for an attribute" (page 150) and "Multiple-value data types" (page 276) describe the possible data types. For more information on data types, see 241-5701-060 *Passport 7400, 15000, 20000 Components*.

The data types of signed integer, decimal number, fixed point decimal, and hexadecimal number take either a range of numerical values or one or more named values.

For example, you are setting a decimal number attribute that defines the maximum number of records held in a queue. You can set the attribute to a numerical value in the range of 1 to 200. Alternatively, you can set the attribute to the named value *unlimited*, which allows an unlimited number of records in the queue.

Table 125
Single-value data types

Data type	Description	Example
Singed integer	A positive or negative integer	-100, 0, 100
Integer sequence	A sequence of integers separated by periods	63.1023, 65535.255
Decimal number	A positive integer	0, 1, 100
Fixed point decimal	A number with a decimal point	1.2, 4.56
Hexadecimal number	A number in hexadecimal format	00, B6A1
Hexadecimal string	A hexadecimal number stored in a string	00, B6A1
Dashed hexadecimal string	A hexadecimal number with dashes between each pair of digits stored in a string	3A, 3A-FC
IP address	An IP address	47.208.133.201
Enumeration	A set of values represented by a symbolic name	E1, DS1
(Sheet 1 of 2)		

Table 125 (continued)
Single-value data types

Data type	Description	Example
String	An ASCII string	view.full.001, system
Extended string	A string with an escape mechanism for non-ASCII characters	R\e9sum\e9 (Résumé)
BCD	A binary-coded decimal	374984736253
Component name	A component name	Shelf Bus/x
Link	A component name that forms a bi-directional link	Shelf Card/1
Time	A date and time	1998-08-24 15:17:44
(Sheet 2 of 2)		

Table 126
Multiple-value data types

Data type	Description	Example
Set of ...	A collection of values from a fixed range	alarm, ~debug, ~log, scn
List of ...	An unrestricted collection of values	base_AA0042, networking_AA042, etc.
Vector of ...	An indexed collection of values	lowPriority 8100 highPriority 65
Array of ...	A double-indexed collection of values	packetSize 16 throughputClass 0 8

Attribute and value parameters for set

In general, you specify an attribute name followed by its value.

If the attribute accepts multiple values, you can specify more than one value after the attributes name. For some multiple-value data types (for example, List of...), setting a value adds it to the attribute. For other data types (for example, Set of...), setting a value turns it on. You can delete or turn off values using the exclamation mark (!) and tilde (~) characters. See “Delete value parameters for set” (page 277).

You can set multiple attributes of a component by specifying each attribute-value assignment separated by a comma (,). Passport attempts all attribute assignments even if some of the assignments fail.

Table 127
Attribute and value parameters of the set command

Parameter	Notes
<attribute> <value>	Use this parameter to set a single-value attribute.
<attribute> <value> [<value>]...	Use this parameter to set a multiple-value attribute. Passport either adds the value to the attribute or turns the value on.
<attribute> <value> [, <attribute> <values>]...	Set multiple attributes of a component by specifying attribute-value assignments separated by commas (,).

Delete value parameters for set

Some attributes data types (for example, Set of ... and List of ...) accept multiple values. Multiple-value attributes support two special characters: exclamation mark (!), and tilde (~). The ! character deletes or turns off (depending on the data type of the attribute) all values for the attribute. The ~ character deletes or turns off (depending on the data type of the attribute) a particular value of a multi-value attribute. The ~ character must immediately precede the value you want to delete.

You can combine the delete characters with other value assignments in a single set command when setting multiple-value attributes.

You can also use the exclamation mark (!) to delete the value of single-value attribute. Alternatively, you can delete the value of a single-value attribute by not including a value in the set command. For example, you can delete the value of the *spareCard* attribute of Lp/1 using the following command:

```
set Lp/1 spareCard
```

Table 128
Attribute and value parameters of the set command

Parameter	Notes
!	Deletes or turns off all values for a multiple-value attribute. Deletes the value of a single-value attribute.
~<value>	Deletes or turns off a particular value for a multiple-value attribute.

Examples for set

The following example shows how to use the set command:

- "Setting a single-value attribute example" (page 278)
- "Setting a multiple-value attribute example" (page 279)
- "Setting multiple attributes example" (page 279)
- "Deleting an attribute value example" (page 280)

Setting a single-value attribute example

You want to set the *cardType* attribute of the *Shelf Card/1* component to V35. You enter the following command:

```
set Shelf Card/1 cardType V35
```

You display the attribute using the following command:

```
display Shelf Card/1 cardType
```

You receive the following response:

```
Shelf Card/1
   cardType = V35
```

Setting a multiple-value attribute example

You want to add a value to the multiple-value *featureList* attribute of the *Sw Lpt/TRUNK* component. You display its current values using the following command:

```
display Sw Lpt/TRUNK featureList
```

You receive the following response:

```
Sw Lpt/TRUNK
  featureList = unackTrunks,porsTrunks
```

You want to add the value atmTrunks. You enter the following command:

```
set Sw Lpt/TRUNK featureList atmTrunks
```

You display the new values of the attribute using the following command:

```
display Sw Lpt/TRUNK featureList
```

You receive the following response:

```
Sw Lpt/TRUNK
  featureList = unackTrunks,porsTrunks,atmTrunks
```

Setting multiple attributes example

You want to set the *mainCard* and *spareCard* attributes of the *Lp/1* component to Shelf Card/1 and Shelf Card/2, respectively. You enter the following command:

```
set Lp/1 mainCard Shelf Card/1, spareCard Shelf Card/2
```

You display these attributes using the following command:

```
display Lp/1 mainCard, spareCard
```

You receive the following response:

```
Lp/1
    mainCard = Shelf Card/1
    spareCard = Shelf Card/2
```

Deleting an attribute value example

You want to replace all the values for the *avList* attribute (*Sw* component) with base_AA01. You enter the following command:

```
set Sw avList ! base_AA01
```

You display the value of the attribute using the following command:

```
display Sw avList
```

You receive the following response:

```
Sw
   avList = base_AA01
```

You now want to turn off the alarm (ala) and turn on the log value of the *dataStreams* attribute of the *Nmis Local Session/1* component. You enter the following command to display the current values of the attribute:

```
display Nmis Local Session/1 dataStreams
```

You receive the following response:

```
Nmis Local Session/1
    dataStreams = ala ~deb ~log ~scn ~rts
```

The values preceded by ~ are turned off. You enter the following command to turn off the alarm and turn on the log value:

```
set Nmis Local Session/1 dataStreams ~alarm log
```

You display the attribute again using the following command:

```
display Nmis Local Session/1 dataStreams
```

You receive the following response:

```
Nmis Local Session/1
    dataStreams = ~ala ~deb log ~scn ~rts
```

Because you have removed a spare card for Lp/3, you now want to delete the value of the single-value attribute *spareCard*. You enter the following command:

```
set Lp/3 spareCard !
```

Related information for set

See the following for information related to the set command:

- “Add command” (page 56)
- “Delete command” (page 118)
- “Display command” (page 121)
- “Help command” (page 146)

Start <hardware> Test command

Use the start <hardware> Test command to start a system test (using the *Test* component) on any of the following hardware components:

- a card
- a bus
- a fabric card
- a port/channel
- the Passport filesystem disk

Note: You must lock a bus, a fabric card, a port/channel, or the Passport filesystem disk before testing it (see “Lock command” (page 177)). You do not need to lock a card before testing it.

You can run a test when you add new hardware or commission new services, as part of a scheduled maintenance program, or to diagnose hardware-related problems.

Before you can run a hardware test, you must define the test by setting a number of attributes of the *Test* component. For details on cards, buses, and disks, see 241-5701-600 *Passport 7400, 15000, 20000 Configuration Guide*. For details on ports and channels, see 241-5701-615 *Passport 7400, 15000, 20000 FP Configuration Reference*.

For more information on the start <hardware> Test command, see

Properties for start <hardware> Test

The following table describes the properties of the start <hardware> Test command.

Table 129
Properties of the start <hardware> Test command

Mode	Impact	Scope	Components
Operational or Provisioning	Service	Device	Shelf Card Shelf Bus LogicalProcessor (Lp) AutomaticProtectionSwitching (Aps) LineAutomaticProtection Switching (Laps)
Operational or Provisioning	Configuration	Device	FileSystem (Fs) Disk

Syntax for start <hardware> Test

The following table describes the syntax for the start <hardware> test command.

Table 130
Syntax of the start <hardware> Test command

If you want to...	Use the following syntax:
Start a test of the specified hardware component	start <hardware> Test Where the <hardware> is the Passport component that controls the piece of hardware you wish to test. These components are shown in “Testable hardware and components for the start <hardware> Test command” (page 284).

Testable hardware components parameter for start <hardware> Test

The <hardware> parameter identifies the hardware component that you wish to test. The following table describes the <hardware> option for the start <hardware> Test command.

Table 131
Testable hardware and components for the start <hardware> Test command

Hardware type	Component
Card	Shelf Card/<n>
Bus	Shelf Bus/x or Shelf Bus/y
Fabric card	Shelf FabricCard/x or Shelf FabricCard/y
Ports and channels	Lp/<n> V35/<n>
	Lp/<n> X21/<n>
	Lp/<n> DS1/<n>
	Lp/<n> DS1/<n> Chan
	Lp/<n> DS3/<n>
	Lp/<n> DS3/<n> DS1/<n>
	Lp/<n> DS3/<n> DS1/<n> Chan
	Lp/<n> E1/<n>
	Lp/<n> E1/<n> Chan
	Lp/<n> E3/<n>
	Lp/<n> Enet/<n>
	Lp/<n> Sonet/<n>
	Lp/<n> Sdh/<n>
	Lp/<n> JT2/<n>
	Lp/<n> Hssi/0
	Lp/<n> IlsFwdr/<n>

(Sheet 1 of 2)

Table 131 (continued)
Testable hardware and components for the start <hardware> Test command

Hardware type	Component
	Lp/<n> Ima/<n>
	Lp/<n> Vsp
	Lp/0 oamEnet/0
	Aps/<n>
	Laps/<n>
Disk	Fs Disk

(Sheet 2 of 2)

Starting a hardware test example

You want to test the card in slot 4 (Card/4). You define the test by setting the attributes of the Setup group of the *Test* component. For this example, you want to test card 4 using card 1 as the target card. You review the test setting using the following command:

```
display Shelf Card/4 Test Setup
```

You receive the following response:

```
Shelf Card/4 Test
   targetCard        = 1
   frmTypes          = loading verification
   frmPriorities     = lowPriority ~highPriority
   frmSize           = lowPriority : 64 bytes
                       highPriority : 56 bytes
   frmPatternType    = ccitt32kBitPattern
   customizedPattern = 55555555 Hex
   duration          = 1 minutes
```

You now want to start the test. You enter the following command:

```
start Shelf Card/4 Test
```

You receive the following response:

```
Shelf Card/4 Test
   Test started.
```

After the duration of the test, you receive the following response:

```
Shelf Card/4 Test
   Test stopped.
```

You now want to view the results of the test. You enter the following command:

```
display Shelf Card/4 Test Results
```

You receive the following response:

```
Shelf Card/4 Test
   elapsedTime = 1 minutes
   timeRemaining = 0 minutes
   causeOfTermination = testTimeExpired
   loadingFrmData      =               priority
                                 results| low high
                                ----------+ --- ----
                                framesSent| 612    0
                                framesLost|   0    0
   verificationFrmData =               priority
                                 results| low high
                               -----------+ --- ----
                               framesTested| 203    0
                                  framesBad|   0    0
```

Related information for start <hardware> Test

See the following for information related to the start <hardware> Test command:

- “Lock command” (page 177)
- “Stop <hardware> Test command” (page 296)

Start Prov command

Use the start Prov (st Prov) command to enter provisioning mode.

Only one operator can be in provisioning mode at any given time for a particular node. The system rejects a start Prov command if another operator attempts to enter provisioning mode. However, an operator with systemAdministration impact can force another operator out of provisioning mode by issuing the start Prov command with the force option.

For more information on the start Prov command, see

Properties for start Prov

The following table describes the properties of the start Prov command.

Table 132
Properties of the start Prov command

Mode	Impact	Scope	Components
Operational or Provisioning	Configuration	Application	ProvisioningSystem (Prov)

Syntax for start Prov

The following table describes the syntax for the start Prov command.

Table 133
Syntax of the start Prov command

If you want to...	Use the following syntax:
Enter provisioning mode	start Prov
Force another operator out of provisioning mode	start -force Prov

Remove another operator from provisioning mode parameter for start Prov

You can specify that the provisioning system remove another operator from a provisioning session, using the force option.

A system administrator uses the force option to take over another operator's provisioning session. The contents of the edit view are preserved, and any outstanding commands issued by the previous operator run to completion (that is, in progress commands are not aborted).

The system generates an alarm to inform the other operator that he or she has been forced out.

Examples for start Prov

The following examples show how to use the start Prov command:

- "Starting a provisioning session example" (page 288)
- "Taking over another operator's provisioning session example" (page 289)

Starting a provisioning session example

You want to enter provisioning mode. You enter the following command:

```
start Prov
```

You receive one of the following responses:

```
Prov
   The edit view is identical to the current view.
```

OR

```
Prov
   The edit view differs from the current view.
   Changed xx provisioning attribute(s) in the edit
   view.
```

Note: When entering provisioning mode, the system prompt changes from > to PROV> .

Taking over another operator's provisioning session example

You are the system administrator and you want to enter a provisioning session currently in use by another operator. You enter the following command:

```
start -force Prov
```

You receive one of the following responses:

```
Prov
   The edit view is identical to the current view.
```

OR

```
Prov
   The edit view differs from the current view.
   Changed xx provisioning attribute(s) in the edit
   view.
```

You and the other operator receive the following alarm to indicate that you have forced the provisioning operator out of the provisioning mode:

```
   Prov;
MSG warning operator operationalCondition     70000012
   ADMIN: unlocked     OPER: enabled      USAGE: busy
   AVAIL:               PROC: reporting     CNTRL:
   ALARM:             STBY: notSet       UNKNW: false
```

```
        Id: 0F000022  Rel:
        Com: The provisioning operator has been forced off
             by the System Administrator.

        Int: 15/1/3/63507;casResolverVerbProvStart.cc;195;
             p4.0d.27

Prov
```

Related information for start Prov

See the following for information related to the start Prov command:

- “End Prov command” (page 135)

Start Sw Dld command

The start Sw Dld command lets you begin downloading software applications from the software distribution site (SDS) to your Passport node. It will also let you download patches to a software application.

The SDS is a UNIX workstation that stores the complete set of Passport software. The command connects to the SDS using file transfer protocol (FTP) and downloads software applications you have specified onto the Passport file system. Note that secure FTP communications should be used for downloading software. Please refer to the Preside MDM Security User Guide 241-6001-040 for details. To start a download you must know the IP address and have a userid and a password for the SDS.

Before running the start Sw Dld command, you must set the *avListToDownload* (*dldList*) attribute of the *Sw Dld* component to the software application versions you want to download. You should also set the processor types for the application using the *processorTargets* attribute. You can specify i960, PowerPC (ppc), or both. Downloading files to support a processor type that you are not using on the node wastes disk space.

> ***Note:*** You cannot download files to support the PowerPC (ppc) processor type on nodes that have only a 256-Mbyte disk on their control processors.

Passport automatically downloads patches associated with the application versions it is downloading. If any patches for the application version are not available on the node, Passport includes them in the download. To acquire a new patch to an application version already on the node, download the application version again.

You cannot start the software download process if a remove Sw, tidy Sw, or check Prov command is currently running on the node. Once you have started downloading, you can stop the process using the stop Sw Dld command. You can monitor the progress of the download by displaying the attributes of the *Sw Dld* component.

For more information on the start Sw Dld command, see

- “Properties for start Sw Dld” (page 292)
- “Syntax for start Sw Dld” (page 292)
- “Starting an application version download example” (page 293)
- “Related information for start Sw Dld” (page 295)

Properties for start Sw Dld

The following table describes the properties of the start Sw Dld command.

Table 134
Properties of the start Sw Dld command

Mode	Impact	Scope	Components
Operational or Provisioning	Configuration	Device	Software Download (Sw Dld)

Syntax for start Sw Dld

The following table describes the syntax for the start Sw Dld command.

Table 135
Syntax of the start Sw Dld command

If you want to...	Use the following syntax:
Start a software download	start -host(<IPaddress>) -userId(<userID>) -password(<password>) Sw Dld

Parameters for start Sw Dld

There are two types of parameters for the start Sw Dld command:

- “SDS IP address parameter” (page 293)
- “Log on parameters” (page 293)

SDS IP address parameter

You must identify the software distribution site by its IP address using the host option:

```
-host(<IPaddress>)
```

You must enter the host option.

Log on parameters

To access the SDS site you must provide a valid user ID and password using the userID and password options. The user ID must be allowed FTP access to the site with read permission on all software files.

Table 136
Options for the start Prov command

Option	Description
-user(<userID>) -u(<userID>)	Enter a user ID for the software distribution site.
-password (<password>) -p(<password>)	Enter the password for the user ID you entered in the userID option.

Examples for start Sw Dld

The following examples show how to use the start Sw Dld command:

- “Starting an application version download example” (page 293)
- “Starting a patch download example” (page 294)

Starting an application version download example

You want to download the application versions frameRelay_AQ0123A and trunks_AQ0123A to your node. You set the *avListToDownload* attribute using the following command:

```
set Sw Dld avListToDownload ! frameRelay_AQ0123A
trunks_AQ0123A
```

Your node only has i960 processors on its function processors, so you set the *processorTargets* attribute using the following command:

```
set Sw Dld processorTargets i960 ~ppc
```

Your software distribution site has an IP address of 10.236.0.21. You can access the site using the user ID admin with a password of yu5u67. You enter the following command to begin the software download process:

```
start -host(10.236.0.21) -user(admin) -
password(yu5u67) Sw Dld
```

Passport begins to download the application versions and patches to those application version stored on the SDS. To monitor the progress of the download, you display the attributes of the *Sw Dld* component using the following command:

```
display Sw Dld
```

You receive the following response:

```
Sw Dld
   avBeingDownloaded  = trunks_AQ0123A
   status             = inProgress
   filesToTransfer    = 1
   processorTargets   = i960
   avListToDownload   = trunks_AQ0123A
   downloadedAvList     = frameRelay_AQ0123A
```

Starting a patch download example

You want to download any new patches to the application version base_AQ0123A available on the SDS. You set the *avListToDownload* attribute using the following command:

```
set Sw Dld avListToDownload base_AQ0123A
```

Your node only has i960 processors on its function processors, so you set the *processorTargets* attribute using the following command:

```
set Sw Dld processorTargets i960 ~ppc
```

Your software distribution site has an IP address of 10.236.0.21. You can access the site using the user ID admin with a password of yu5u67. You enter the following command to begin the software download process:

```
start -host(10.236.0.21) -user(admin) -
password(yu5u67) Sw Dld
```

Passport begins to download new patches to the base_AQ0123A application version. When the download is complete, you list all the patches now available for the base_AQ0123A application version using the following command:

```
list Sw Av/base_AQ0123A Patch/*
```

You receive the following response:

```
Sw Av/BASE_AQ0123A Patch/BASE001A
Sw Av/BASE_AQ0123A Patch/BASE002A
```

Related information for start Sw Dld

See the following for information related to the start Sw command:

- “Display command” (page 121)
- “Remove Sw Av command” (page 236)
- “Set command” (page 272)
- “Stop Sw Dld command” (page 303)
- “Tidy Sw command” (page 341)

Stop <hardware> Test command

Use the stop <hardware> Test command to stop a system test (see “Start <hardware> Test command” (page 282)) on any of the following hardware components:

- a card
- a bus
- a fabric card
- a port/channel
- the Passport filesystem disk

Note: You must lock a bus, a fabric card, a port/channel, or the Passport filesystem disk before testing it (see “Lock command” (page 177)). You do not need to lock a card before testing it.

Before you can run a hardware test, you must define the test by setting a number of attributes of the *Test* component. See “Start <hardware> Test command” (page 282).

For more information on the stop <hardware> Test command, see

- “Properties for stop <hardware> Test” (page 297)
- “Syntax for stop <hardware> Test” (page 297)
- “Testable hardware components parameter for stop <hardware> Test” (page 298)
- “Stopping a hardware test example” (page 299)
- “Related information for stop <hardware> Test” (page 299)

Properties for stop <hardware> Test

The following table describes the properties of the stop <hardware> Test command.

Table 137
Properties of the stop <hardware> Test command

Mode	Impact	Scope	Components
Operational or Provisioning	Service	Device	Shelf Card Shelf Bus LogicalProcessor (Lp) AutomaticProtectionSwitching (Aps) LineAutomaticProtection Switching (Laps)
Operational or Provisioning	Configuration	Device	FileSystem (Fs)

Syntax for stop <hardware> Test

The following table describes the syntax for the stop <hardware> Test command.

Table 138
Syntax of the stop <hardware> Test command

If you want to...	Use the following syntax:
Stop a test of the specified hardware component	stop <hardware> Test where <hardware> is the Passport component that controls the piece of hardware on which you are currently running a test. These components are shown in “Testable hardware and components for the stop <hardware> Test command” (page 298).

Testable hardware components parameter for stop <hardware> Test

The <hardware> parameter identifies the hardware component that the system is currently testing. The following table describes the <hardware> option for the stop <hardware> Test command.

Table 139
Testable hardware and components for the stop <hardware> Test command

Hardware type	Component
Card	Shelf Card/<n>
Bus	Shelf Bus/x or Shelf Bus/y
Fabric Card	Shelf FabricCard/x or Shelf FabricCard/y
Ports and channels	Lp/<n> V35/<n>
	Lp/<n> X21/<n>
	Lp/<n> DS1/<n>
	Lp/<n> DS1/<n> Chan
	Lp/<n> DS3/<n>
	Lp/<n> DS3/<n> DS1/<n>
	Lp/<n> DS3/<n> DS1/<n> Chan
	Lp/<n> E1/<n>
	Lp/<n> E1/<n> Chan
	Lp/<n> E3/<n>
	Lp/<n> Tr/<n>
	Lp/<n> Enet/<n>
	Lp/<n> Sonet/<n>
	Lp/<n> Sdh/<n>
	Lp/<n> JT2/<n>
	Lp/<n> Hssi/0

(Sheet 1 of 2)

Table 139 (continued)
Testable hardware and components for the stop <hardware> Test command

Hardware type	Component
	Lp/<n> IlsFwdr/<n>
	Lp/<n> Ima/<n>
	Lp/<n> Vsp
	Aps/<n>
	Laps/<n>
Disk	Fs Disk
(Sheet 2 of 2)	

Stopping a hardware test example

You want to stop a test that is in progress on Card/4. You enter the following command:

```
stop Shelf Card/4 Test
```

You receive the following response:

```
Shelf Card/4 Test
   Test stopped.
```

Related information for stop <hardware> Test

See the following for information related to the stop <hardware> Test command:

- “Lock command” (page 177)
- “Start <hardware> Test command” (page 282)

Stop Prov command

Use the stop Prov command to abort a provisioning command of the following types:

- activate Prov
- apply Prov
- check Prov
- load Prov
- save Prov

The stop Prov command is useful if you wish to abort a long-running provisioning command.

For more information on the stop Prov command, see

- “Properties for stop Prov” (page 300)
- “Syntax for stop Prov” (page 301)
- “Stopping a provisioning system command example” (page 301)
- “Related information for stop Prov” (page 302)

Properties for stop Prov

The following table describes the properties of the stop Prov command.

Table 140
Properties of the stop Prov command

Mode	Impact	Scope	Components
Provisioning	Configuration	Application	ProvisioningSystem(Prov)

Syntax for stop Prov

The following table describes the syntax for the stop Prov command.

Table 141
Syntax of the stop Prov command

If you want to...	Use the following syntax:
Abort a provisioning system command	stop Prov

Stopping a provisioning system command example

You want to stop a long-running provisioning system command. You enter the following command:

```
stop Prov
```

You receive the following response:

```
Prov
ok
```

Note: The system finishes the phase being executed by the in-progress command before it aborts the command. This means that the in-progress command can still take a few seconds to terminate after you issue stop Prov.

When the in-progress command terminates, you receive the following response:

```
command cancelled
```

Related information for stop Prov

See the following for information related to the stop Prov command:

- “Apply Prov command” (page 60)
- “Check Prov command” (page 71)
- “Load Prov command” (page 173)
- “Save Prov command” (page 264)

Stop Sw Dld command

The stop Sw Dld command lets you stop downloading software applications and patches from the software distribution site (SDS) to your Passport node.

The stop Sw Dld stops the running download without completing the application or patch it is currently downloading. You cannot use a partially downloaded application or patch. If you restart the download, Passport continues downloading where it left off.

After stopping a software download, you cannot immediately start another download. You have to wait for your Passport node to fully disconnect from the software distribution site.

You can monitor the progress of a download by displaying the attributes of the *Sw Dld* component. While the node is disconnecting from the SDS, the *status* attribute has a value of stopping.

For more information on the stop Sw Dld command, see

- "Properties for stop Sw Dld" (page 303)
- "Syntax for stop Sw Dld" (page 304)
- "Stopping a software download example" (page 304)
- "Related information for stop Sw Dld" (page 305)

Properties for stop Sw Dld

The following table describes the properties of the stop Sw Dld command.

Table 142
Properties of the stop Sw Dld command

Mode	Impact	Scope	Components
Operational or Provisioning	Configuration	Device	Software Download (Sw Dld)

Syntax for stop Sw Dld

The following table describes the syntax for the stop Sw Dld command.

Table 143
Syntax of the stop Sw Dld command

If you want to...	Use the following syntax:
Stop a software download	stop Sw Dld

Stopping a software download example

You have started downloading software applications and patches. To check the progress of the download, you display the attributes of the *Sw Dld* component using the following command:

```
display Sw Dld
```

You receive the following response:

```
Sw Dld
   avBeingDownloaded  = trunks_AQ0123A
   status             = inProgress
   filesToTransfer    = 1
   processorTargets   = i960
   avListToDownload   = trunks_AQ0123A
   downloadedAvList     = frameRelay_AQ0123A
```

You decide to stop the download using the following command:

```
stop Sw Dld
```

You immediately check the status of the download using the following command:

```
display Sw Dld status
```

You receive the following response:

```
Sw Dld
   status = stopping
```

Related information for stop Sw Dld

See the following for information related to the stop Sw command:

- “Display command” (page 121)
- “Remove Sw Av command” (page 236)
- “Start Sw Dld command” (page 291)
- “Tidy Sw command” (page 341)

Switch Aps command on Passport 7400

The switch Aps command allows you to manually switch between the working and protection line on an APS-protected link.

An APS-protected link has two lines: working and protection. The working line normally carries the data. The protection line acts as a backup to the working line. The line that is currently carrying the data is called the active line. Unless manually overridden, the APS system automatically determines the active line.

APS can ignore the switch Aps command depending on the state (request) of the near and far end of the link and the mode (unidirectional or bidirectional). You can determine the state of the link by displaying the following attributes of the Aps component:

- nearEndRequest
- nearEndRequestChannel
- farEndRequest
- farEndRequestChannel

The manual switch request created by the switch Aps command remains in effect until it is removed using the clear Aps command or it is overridden by a signal degradation, signal failure, or a lock to the protection line using the protectionLockout Aps command.

For more information on the switch Aps command, see

- “Properties for switch Aps” (page 307)
- “Syntax for switch Aps” (page 307)
- “Parameters for switch Aps” (page 308)
- “Examples for switch Aps” (page 309)
- “Related information for switch Aps” (page 312)

Properties for switch Aps

The following table describes the properties of the switch Aps command.

Table 144
Properties of the switch Aps command

Mode	Impact	Scope	Component
Operational or Provisioning	Service	Device	AutomaticProtectionSwitching (Aps)

Syntax for switch Aps

The following table describes the syntax for the switch Aps command.

Table 145
Syntax of the switch Aps command

If you want to...	Use the following syntax:
Switch from the working to the protection line	switch -workingToProtection Aps/<n>
Switch from the protection to the working line	switch -protectionToWorking Aps/<n>
Force a switch from working to protection line even if the protection line has a degraded or failed signal	switch -force -workingToProtection Aps/<n>
Force a switch from protection to working line even if the working line has a degraded or failed signal	switch -force -protectionToWorking Aps/<n>

Parameters for switch Aps

The following are the parameters for the switch Aps command:

- "Component instance parameter for switch Aps" (page 308)
- "Working to protection parameter for switch Aps" (page 308)
- "Protection to working for switch Aps" (page 309)
- "Force parameter for switch Aps" (page 309)

Component instance parameter for switch Aps

An Aps component instance represents a link that is protected by APS. The Aps component manages both the working and protection lines. To switch between the lines, you specify the instance number of the Aps component:

```
Aps/<n>
```

The instance value can be any integer between 0 and 15999.

Working to protection parameter for switch Aps

To switch from the working line to the protection line, use the workingToProtection option:

```
-workingToProtection
```

APS ignores a switch Aps command using this option in the following situations:

- the protection line is locked at the near end of the link (unidirectional and bidirectional mode) or the far end of the link (bidirectional mode only)
- the protection line is experiencing signal degradation or failure at the near end of the link (unidirectional and bidirectional mode) or the far end of the link (bidirectional mode only)

You can force the switch to occur when the protection line is experiencing signal degradation using the force option.

Protection to working for switch Aps

To switch from the protection line to the working line, use the protectionToWorking option:

```
-protectionToWorking
```

APS ignores a switch Aps command using this option in the following situation:

- the working line is experiencing signal degradation or failure at the near end of the link (unidirectional and bidirectional mode) or the far end of the link (bidirectional mode only)

You can force the switch to occur when the working line is experiencing signal degradation or signal failure using the force option.

Force parameter for switch Aps

You can force a switch to the working or protection line using the force option:

```
-force
```

You can force a switch to the working line when it is experiencing signal degradation or signal failure. You can only force a switch to the protection line when it has signal degradation, not when it has signal failure. You cannot force a switch to the protection line when it has been locked using the protectionLockout Aps command.

Examples for switch Aps

The following example shows how to use the switch Aps command:

- “Switching from working to protection line example” (page 310)
- “Switching from protection to working line example” (page 310)
- “Forcing a switch example” (page 311)

Switching from working to protection line example

You want to switch the active line from the working to the protection line on an APS-protected link (represented by Aps/1). You check the state of the link using the following command:

```
display Aps/1 nearEndRequest, nearEndRequestChannel,
farEndRequest, farEndRequestChannel
```

You receive the following response:

```
Aps/1
   nearEndRequest        = noRequest
  nearEndRequestChannel = working
  farEndRequest         = noRequest
  farEndRequestChannel    = working
```

Since there are no outstanding requests on neither the near or far end of the link, you enter the following command:

```
switch -workingToProtection Aps/1
```

The protection line becomes the active line at both ends of the link. You again check the state of the link using the following command:

```
display Aps/1 nearEndRequest, nearEndRequestChannel,
farEndRequest, farEndRequestChannel
```

You receive the following response:

```
Aps/1
  nearEndRequest        = manualSwitch
  nearEndRequestChannel = protection
  farEndRequest         = reverseRequest
  farEndRequestChannel    = protection
```

Switching from protection to working line example

You previously switched to the protection line on a link protected by APS (represented by Aps/1) in bidirectional mode. You now want to switch back to the working line. You enter the following command:

```
switch -protectionToWorking Aps/1
```

The working line becomes the active line at both ends of the link. You check the state of the link using the following command:

```
display Aps/1 mode, nearEndRequest,
nearEndRequestChannel, farEndRequest,
farEndRequestChannel
```

You receive the following response:

```
Lp/1
  mode                  = bidirectional
  nearEndRequest        = manualSwitch
  nearEndRequestChannel = working
  farEndRequest         = reverseRequest
  farEndRequestChannel    = working
```

Forcing a switch example

An APS-protected link (represented by Aps/1) in bidirectional mode has automatically switched from the working to the protection line because it has detected signal degradation on the working line. However, you want to switch back to the working line. Since the working line is experiencing signal degradation, you must for the switch using the following command:

```
switch -force -workingToProtection Aps/4
```

The working line becomes the active line at both ends of the link. You check the state of the link using the following command:

```
display Aps/1 mode, nearEndRequest,
nearEndRequestChannel, farEndRequest,
farEndRequestChannel
```

You receive the following response:

```
Lp/1
  mode                  = bidirectional
  nearEndRequest        = forcedSwitch
  nearEndRequestChannel = working
  farEndRequest         = reverseRequest
  farEndRequestChannel    = working
```

Related information for switch Aps

See the following for information related to the switch Aps command:

- “Clear Aps command on Passport 7400” (page 83)
- “ProtectionLockout Aps command on Passport 7400” (page 218)

Switch Laps command on Passport 15000 and 20000

The switch Laps command allows you to manually switch between the working and protection line on an line APS-protected link.

A line APS-protected link has two lines: working and protection. The working line normally carries the data. The protection line acts as a backup to the working line. The line that is currently carrying the data is called the active line. Unless manually overridden, the line APS system automatically determines the active line.

Line APS can ignore the switch Laps command depending on the state (request) of the near and far end of the link and the mode (unidirectional or bidirectional). You can determine the state of the link by displaying the following attributes of the Laps component:

- nearEndRequest
- nearEndRequestChannel
- farEndRequest
- farEndRequestChannel

The manual switch request created by the switch Laps command remains in effect until it is removed using the clear Laps command or it is overridden by a signal degradation, signal failure, or a lock to the protection line using the protectionLockout Laps command.

For more information on the switch Laps command, see

- “Properties for switch Laps” (page 314)
- “Syntax for switch Laps” (page 314)
- “Parameters for switch Laps” (page 315)
- “Examples for switch Laps” (page 316)
- “Related information for switch Laps” (page 319)

Properties for switch Laps

The following table describes the properties of the switch Laps command.

Table 146
Properties of the switch Laps command

Mode	Impact	Scope	Component
Operational or Provisioning	Service	Device	LineAutomaticProtection Switching (Laps)

Syntax for switch Laps

The following table describes the syntax for the switch Laps command.

Table 147
Syntax of the switch Laps command

If you want to...	Use the following syntax:
Switch from the working to the protection line	switch -workingToProtection Laps/<n>
Switch from the protection to the working line	switch -protectionToWorking Laps/<n>
Force a switch from working to protection line even if the protection line has a degraded or failed signal	switch -force -workingToProtection Laps/<n>
Force a switch from protection to working line even if the working line has a degraded or failed signal	switch -force -protectionToWorking Laps/<n>

Parameters for switch Laps

The following are the parameters for the switch Laps command:

Component instance parameter for switch Laps

An Laps component instance represents a link that is protected by Line APS. The Laps component manages both the working and protection lines. To switch between the lines, you specify the instance number of the Laps component:

```
Laps/<n>
```

The instance value can be any integer between 0 and 15999.

Working to protection parameter for switch Laps

To switch from the working line to the protection line, use the workingToProtection option:

```
-workingToProtection
```

LAPS ignores a switch Laps command using this option in the following situations:

- the protection line is locked at the near end of the link (unidirectional and bidirectional mode) or the far end of the link (bidirectional mode only)
- the protection line is experiencing signal degradation or failure at the near end of the link (unidirectional and bidirectional mode) or the far end of the link (bidirectional mode only)

You can force the switch to occur when the protection line is experiencing signal degradation using the force option.

Protection to working for switch Laps

To switch from the protection line to the working line, use the protectionToWorking option:

```
-protectionToWorking
```

Line APS ignores a switch Laps command using this option in the following situation:

- the working line is experiencing signal degradation or failure at the near end of the link (unidirectional and bidirectional mode) or the far end of the link (bidirectional mode only)

You can force the switch to occur when the working line is experiencing signal degradation or signal failure using the force option.

Force parameter for switch Laps

You can force a switch to the working or protection line using the force option:

```
-force
```

You can force a switch to the working line when it is experiencing signal degradation or signal failure. You can only force a switch to the protection line when it has signal degradation, not when it has signal failure. You cannot force a switch to the protection line when it has been locked using the protectionLockout Laps command.

Examples for switch Laps

The following example shows how to use the switch Laps command:

- “Switching from working to protection line example” (page 317)
- “Switching from protection to working line example” (page 317)
- “Forcing a switch example” (page 318)

Switching from working to protection line example

You want to switch the active line from the working to the protection line on a line APS-protected link (represented by Laps/1). You check the state of the link using the following command:

```
display Laps/1 nearEndRequest, nearEndRequestChannel,
farEndRequest, farEndRequestChannel
```

You receive the following response:

```
Laps/1
   nearEndRequest        = noRequest
  nearEndRequestChannel = working
  farEndRequest         = noRequest
  farEndRequestChannel    = working
```

Since there are no outstanding requests on neither the near or far end of the link, you enter the following command:

```
switch -workingToProtection Laps/1
```

The protection line becomes the active line at both ends of the link. You again check the state of the link using the following command:

```
display Laps/1 nearEndRequest, nearEndRequestChannel,
farEndRequest, farEndRequestChannel
```

You receive the following response:

```
Laps/1
  nearEndRequest        = manualSwitch
  nearEndRequestChannel = protection
  farEndRequest         = reverseRequest
  farEndRequestChannel    = protection
```

Switching from protection to working line example

You previously switched to the protection line on a link protected by line APS (represented by Laps/1) in bidirectional mode. You now want to switch back to the working line. You enter the following command:

```
switch -protectionToWorking Laps/1
```

The working line becomes the active line at both ends of the link. You check the state of the link using the following command:

```
display Laps/1 mode, nearEndRequest,
nearEndRequestChannel, farEndRequest,
farEndRequestChannel
```

You receive the following response:

```
Lp/1
  mode                  = bidirectional
  nearEndRequest        = manualSwitch
  nearEndRequestChannel = working
  farEndRequest         = reverseRequest
  farEndRequestChannel    = working
```

Forcing a switch example

A line APS-protected link (represented by Laps/1) in bidirectional mode has automatically switched from the working to the protection line because it has detected signal degradation on the working line. However, you want to switch back to the working line. Since the working line is experiencing signal degradation, you must for the switch using the following command:

```
switch -force -workingToProtection Laps/4
```

The working line becomes the active line at both ends of the link. You check the state of the link using the following command:

```
display Laps/1 mode, nearEndRequest,
nearEndRequestChannel, farEndRequest,
farEndRequestChannel
```

You receive the following response:

```
Lp/1
  mode                  = bidirectional
  nearEndRequest        = forcedSwitch
  nearEndRequestChannel = working
  farEndRequest         = reverseRequest
  farEndRequestChannel    = working
```

Related information for switch Laps

See the following for information related to the switch Laps command:

- "Clear Laps command on Passport 15000 and 20000" (page 85)
- "ProtectionLockout Laps command on Passport 15000 and 20000" (page 221)

Switchover Lp command

The switchover Lp command allows you to manually switch control between the active and standby processor cards of an LP.

You can schedule the switchover to occur at a future time. You can also cancel a scheduled switchover. You cannot schedule a switchover for the control processor (Lp/0).

If you have switched control from a main to the spare card to replace a defective main card, you should switch back to the main card once you have replaced the defective hardware. Switching back is particularly important when you are using one-for-*n* sparing. In this configuration, a single spare card protects multiple main cards. When the spare card is providing service for one of the main cards, all other main cards are unprotected.

Switching from the active to the standby control processor can cause Passport to restart the function processors of the node. If you have hot standby enabled, some function processors continue running (and providing service) during a control processor switchover. For more information on switching between active and standby control processors, see 241-5701-600 *Passport 7400, 15000, 20000 Configuration Guide*.

> ***Note:*** Do not perform a control processor switchover more than once every 10 minutes. Passport considers a second CP switchover within 10 minutes to be an indication of a more serious fault and will perform a shelf reset. Before executing the *switchover Lp* command, the value of the *revertibleTimerCountdown* attribute, under the Shelf component, must be zero.

You cannot manually switch control from the active to standby control processor if any of the following conditions are true:

- The disks on the two control processors are not synchronized.
- A software upgrade is in progress.
- The standby control processor is in the process of loading the committed view.

You can switch control from the active to the standby control process or when the current view is not the committed view if you use the -force option.

For more information on the switchover Lp command, see

Properties for switchover Lp

The following table describes the properties of the switchover Lp command.

Table 148
Properties of the switchover Lp command

Mode	Impact	Scope	Component
Operational or Provisioning	Service	Device	LogicalProcessor (Lp)

Syntax for switchover Lp

The following table describes the syntax for the switchover Lp command.

Table 149
Syntax of the switchover Lp command

If you want to...	Use the following syntax:
Immediately switch from the active to standby processor card	switchover Lp/<n>
Force a function processor switchover when the standby is not ready, or force a control processor switchover if the current view is not the committed view	switchover -force Lp/<n>
Schedule a future switchover from the active to standby processor card	switchover -time(<date_time>) Lp/<n>
Cancel a scheduled switchover	switchover -cancel Lp/<n>

Parameters for switchover Lp

The following are the parameters for the switchover Lp command:

- "Logical processor parameter for switchover Lp" (page 322)
- "Force parameter for switchover Lp" (page 323)
- "Scheduled time parameter for switchover Lp" (page 323)
- "Cancel scheduled time parameter for switchover Lp" (page 324)

Logical processor parameter for switchover Lp

You specify which logical processor you want to switchover using the instance of the *LogicalProcessor* (*Lp*) component:

Lp/<n>

Enter the number of the logical processor you want to switchover.

Force parameter for switchover Lp

You can force a logical processor switchover using the -force option:

-force

The -force option works differently depending on whether it is used on a function processor or a control processor.

Normally, you cannot use the switchover Lp command to switch from an active to a standby function processor if the standby is not ready to take over. A standby function processor is not ready to take over when it is in a degraded state. (For example, a function processor is degraded when there are software errors on the card, or the connections on the active card are not properly duplicated on the standby.) You can use the -force option to cause a switch to the standby even if the standby is not ready. If the degradation on the standby function processor is severe enough, failure can occur after the switchover.

Normally, you cannot use the switchover Lp command to switch from an active to a standby control processor if the current provisioning view is not the committed view. However, you can use the -force option to force a switch to the standby control processor under these conditions. After the switchover, the system reverts to the committed provisioning view. All FPs that are not running the committed provisioning view are restarted. Therefore, it is possible that service outages may occur after a forced switchover.

Scheduled time parameter for switchover Lp

You can schedule the switchover for a future time using the time option:

-time(<date_time>)

Enter the date and time for the scheduled switchover using the following format:

yyyy-mm-dd hh:mm

You can view the current time by displaying the *moduleTime* attribute of the *Time* component. You can view the scheduled time of a switchover by displaying the *scheduledSwitchover* attribute of an *Lp* component. If this attribute has a value of 0000-00-00 00:00, there is no scheduled switchover.

If you have scheduled a switchover and a switchover occurs (either manual or automatic) before the scheduled time, Passport cancels the scheduled switchover.

You can force scheduled switchover to begin immediately by entering the switchover Lp command with no options. To change a scheduled switchover, you must cancel it and then set a new a new date and time.

You cannot schedule a switchover between the active and standby control processor (Lp/0).

Cancel scheduled time parameter for switchover Lp

You can cancel a scheduled switchover using the cancel option:

```
-cancel
```

Examples for switchover Lp

The following example shows how to use the switchover command:

- "Immediately switching between active and standby function processor example" (page 325)
- "Immediately switching between active and standby control processor example" (page 325)
- "Forcing a switchover between active and standby function processor example" (page 325)
- "Forcing a switchover between active and standby control processor example" (page 326)
- "Scheduling a switchover example" (page 326)
- "Canceling a scheduled switchover example" (page 326)

Immediately switching between active and standby function processor example

You want to switch control from the active to the standby processor card for logical processor 1 (Lp/1). You enter the following command:

```
switchover Lp/1
```

The standby processor card becomes the active card. The active processor card restarts and becomes the standby processor card.

Immediately switching between active and standby control processor example

You want to switch control from the active control processor to the standby control processor. Since the control processor is always Lp/0, you enter the following command:

```
switchover Lp/0
```

The standby control processor becomes active. The active control processor restarts and becomes the standby processor card.

If you have hot standby enabled, function processors that support hot standby continue running uninterrupted during the switchover. Preside Multiservice Data Manager connections to the node are unavailable during the time it takes to complete the switchover.

If you have hot standby disabled, all function processors restart during a control processor switchover. Preside Multiservice Data Manager connections are unavailable during the time it takes to complete the switchover.

Forcing a switchover between active and standby function processor example

You want to force a switchover from the active to the standby processor card for logical processor 1 (Lp/1), which is in a degraded state due to software errors. You enter the following command:

```
switchover -force Lp/1
```

The standby processor card becomes the active card. The active processor card restarts and becomes the standby processor card.

Forcing a switchover between active and standby control processor example

You want to force a switchover from the active control processor to the standby control processor when the current provisioning view is not the committed view. Since the control processor is always Lp/0, you enter the following command:

```
switchover -force Lp/0
```

The standby control processor becomes the active control processor. The active processor card restarts and becomes the standby processor card. The system retains the current view.

Scheduling a switchover example

You want to schedule a switchover for logical processor 1 (Lp/1). You enter the following command in operational mode:

```
switchover -time(1998-10-03 00:00) Lp/1
```

When the node time (*moduleTime* attribute of the *Time* component) reaches the scheduled time, the active and standby processor cards of the logical processor switchover.

Canceling a scheduled switchover example

You want to cancel a scheduled switchover for logical processor 1 (Lp/1). You enter the following command:

```
switchover -cancel Lp/1
```

Related information for switchover Lp

See the following for information related to the switchover Lp command:

- “ReloadCp Lp command” (page 229)
- “Reset <processor> command” (page 244)
- “Restart <processor> command” (page 253)
- “Synchronize Fs command” (page 327)

Synchronize Fs command

The synchronize Fs command lets you manually synchronize the standby disk to the active disk. At start-up time, Passport automatically synchronizes the active and standby disks if they both have the same volume name.

While the disks are synchronizing, you cannot format or test either the active or standby disk. You also cannot switchover the active and standby control processors. If the capacity of the standby disk is smaller than the space used on the active disk, the command fails.

After synchronization is complete, the active and the standby disks are identical: they contain the same files and directories and have the same volume name. Once synchronized, Passport automatically mirrors changes made on the active disk to the standby disk.

The *syncStatus* attribute of the *FileSystem* component indicates if the disks are synchronized, not synchronized, or in the process of synchronization. The *syncProgress* attribute (also of the *FileSystem* component) indicates the percentage of the standby disk synchronized to the active disk.

Depending on the amount of data stored on the active disk and the difference between the two disks, the synchronization process can take several hours. Passport will not synchronize a full disk (a disk with no free space).

For more information on the synchronize Fs command, see

- “Properties for synchronize Fs” (page 328)
- “Syntax for synchronize Fs” (page 328)
- “Synchronizing disks example” (page 328)
- “Related information for synchronize Fs” (page 329)

Properties for synchronize Fs

The following table describes the properties of the synchronize Fs command.

Table 150
Properties of the synchronize Fs command

Mode	Impact	Scope	Component
Operational or Provisioning	Configuration	Device	FileSystem (Fs)

Syntax for synchronize Fs

The following table describes the syntax for the synchronize Fs command.

Table 151
Syntax of the synchronize Fs command

If you want to...	Use the following syntax:
Synchronize the standby disk to the active disk	synchronize Fs

Synchronizing disks example

At start-up the disks did not automatically synchronize. You now want to manually synchronize the standby disk to the active disk. You enter the following command:

```
synchronize Fs
```

The synchronization process begins. You check its progress using the following command:

```
display Fs syncProgress
```

You receive the following response:

```
Fs
   syncProgress = 72 %
```

Related information for synchronize Fs

See the following for information related to the synchronize Fs command:

- “Format Fs Disk” (page 142)
- “Switchover Lp command” (page 320)

Telnet Vr command

The telnet Vr command lets you connect to another Passport node so you can manage that node remotely. The remote node must be accessible through a virtual router. The telnet Vr command also lets you connect to any accessible device that complies with the basic telnet protocol specification (RFC 854).

You can only use the telnet Vr command if you are logged on to a Passport using the telnet network management interface. You cannot use the telnet Vr command from the local terminal or the Preside Multiservice Data Manager Command Console.

When you connect to another Passport node, the connection is transparent. All command responses and alarms appear as if you had directly connected to the node. You can use the me command to determine the name of the node to which you are currently connected.

The telnet Vr command includes a telnet command mode where you can enter commands to display and set information about the telnet connection.

You can end a telnet connection to another Passport node using the logout or quit commands or their abbreviations of logoff or exit. You can end a telnet connection to a device supporting the basic telnet protocol specification using the quit or exit command.

For more information on the telnet Vr command, see

- “Properties for telnet Vr” (page 331)
- “Syntax for telnet Vr” (page 331)
- “Parameters for telnet Vr” (page 331)
- “Telnet command mode” (page 332)
- “Examples for telnet Vr” (page 333)
- “Related information for telnet Vr” (page 334)

Properties for telnet Vr

Along with the normal impact and scope restrictions, your user ID must also have permission for outgoing telnet access.

The following table describes the properties of the telnet Vr command.

Table 152
Properties of the telnet Vr command

Mode	Impact	Scope	Components
Operational	Service	Application	VirtualRouter (Vr)

Syntax for telnet Vr

The following table describes the syntax for the telnet Vr command.

Table 153
Syntax of the telnet Vr command

If you want to...	Use the following syntax:
Connect to a Passport node through a virtual router	telnet -ipAddress(<address>) Vr/<n>

Parameters for telnet Vr

The following are the parameters for the telnet Vr command:

- “Virtual router parameter for telnet Vr” (page 332)
- “IP address parameter for telnet Vr” (page 332)

Virtual router parameter for telnet Vr

To establish a telnet connection to a remote device, that device must be accessible through (within the address space of) a management or customer virtual router on your node. You specify the virtual router using the instance value of a *VirtualRouter* (*Vr*) component:

```
Vr/<n>
```

IP address parameter for telnet Vr

You specify the IP address of the remote device you want to connect to using the ipAddress option:

```
-ipAddress(<address>)
```

Enter the IP address using decimal dot notation. The IP address must be with in the address space of the management or virtual router you specify using the *Vr* component.

Telnet command mode

The telnet Vr command supports a command mode where you can enter a number of commands to display information about your telnet connection.

You enter into command mode by pressing the escape character after having established a telnet connection. By default, the escape character is Control-].

When you are in command mode, the command prompt changes to the node name followed by *:telnet*>. For example, if you are currently logged onto the node named NODEA and you press the escape character, the command prompt changes to the following:

```
NODEA:telnet>
```

You exit command mode by successfully entering a command, or by pressing the Enter key.

The following table describes the commands available in telnet command mode.

Table 154
Commands available in telnet command mode

Command	Description
?	Displays help information, including a list of all commands.
close	Closes the telnet connection and exits.
display	Displays operating parameters of the connection, including character mappings.
quit, exit, logout, logoff	Closes the telnet connection and exits.
set	Sets operating parameters of the connection. You can see a list of parameters you can set by following the set command with a question mark (?): set ? You can set the escape character using the following syntax: set escape <char> where: <char> is the escape character.
status	Displays the status of the connection, including the IP address of the remote device and the current escape character.

Examples for telnet Vr

You are currently logged onto a Passport node using the telnet network management interface. You want to connect to another Passport node accessible through the management virtual router (Vr/1). The IP address of the Passport node is 10.12.142.8. You enter the following command:

```
telnet -ipAddress(10.12.142.8) Vr/1
```

You receive the following response:

```
Trying 10.12.142.8...
Connected to 10.12.142.8.
Escape character is '^]'.

Enter login:
```

You can now log into the Passport node.

Related information for telnet Vr

See the following for information related to the telnet Vr command:

- "Clear Nmis <interface> Session command" (page 87)
- "Logout command" (page 182)
- "Me command" (page 186)
- "Quit command" (page 224)

Tidy Prov command

Use the tidy Prov command to remove saved provisioning views from the system disk.

Do not use tidy Prov to remove old views immediately after performing a software upgrade. If problems arise as a result of the upgrade, you will need an old view so you can downgrade to the previous software version.

When the provisioning system encounters a file system error, it cancels the tidy Prov command and provides details on the nature of the error.

The provisioning system has built-in safeguards to prevent you from accidentally removing any of the following views:

- the committed view
- the most recent full view used by one of the activate Prov, commit Prov, load Prov, or save Prov commands
- the current view, if it exists as a saved view (that is, the saved view containing the data in the current view)
- saved views that are read-only
- non-provisioning files or directories that do not have the standard <name>.<type>.<number> format
- saved views in portable format that support dependent saved views in delta format (that is, any view that is a base for another view)

For more information on the tidy Prov command, see

- “Properties for tidy Prov” (page 336)
- “Syntax for tidy Prov” (page 336)
- “Parameters for tidy Prov” (page 337)
- “Examples for tidy Prov” (page 338)
- “Related information for tidy Prov” (page 340)

Properties for tidy Prov

The following table describes the properties of the tidy Prov command.

Table 155
Properties of the tidy Prov command

Mode	Impact	Scope	Components
Operational or Provisioning	Configuration	Application	ProvisioningSystem (Prov)

Syntax for tidy Prov

You must use at least one option when issuing the tidy Prov command.

The following table describes the syntax for the tidy Prov command.

Table 156
Syntax of the tidy Prov command

If you want to...	Use the following syntax:
Remove all views	tidy -all Prov
Remove all but the listed view(s)	tidy -keep(<file_name> <file_name>... <file_name>) Prov
Remove only the listed view(s)	tidy -remove(<file_name> <file_name>... <file_name>) Prov
Remove all views that match a filename, except for the listed view(s)	tidy -remove(<file_name> -keep(<file_name> <file_name>... <file_name>) Prov
Remove all views created during a specified period of time	tidy -fromDate(<date>) -toDate(<date>) Prov ***Note:*** The format for <date> must be yyyy-mm-dd.

Parameters for tidy Prov

There are five types of parameters for the tidy Prov command:

- “Remove all saved views parameter for tidy Prov” (page 337)
- “Remove saved views by date parameter for tidy Prov” (page 337)
- “Remove all but certain saved views parameter for tidy Prov” (page 337)
- “Remove only certain saved views parameter for tidy Prov” (page 338)

Remove all saved views parameter for tidy Prov

You can specify that the provisioning system remove all saved views from the disk using the all option:

```
-all
```

The all option removes all saved views (except those views that cannot be removed).

You cannot use the all option with the remove and date options.

Remove saved views by date parameter for tidy Prov

You can specify that the provisioning system remove all saved views that fall within a range of dates using the fromDate and toDate options:

```
-fromDate(<date>)
```

```
-toDate(<date>)
```

The option value <date> must be in the form yyyy-mm-dd (for example, 1998-06-23).

Remove all but certain saved views parameter for tidy Prov

You can specify that the provisioning system remove all saved views but those you wish to keep using the keep option:

```
-keep(<file_name>)
```

The keep option removes all saved views except those specified in the keep list. The keep option takes precedence over all other tidy options.

If you specify the keep option alone, then the provisioning system removes all views except those specified in the keep list.

If you specify either the remove or the date options with the keep option, then the system removes only those views that are not on the keep list and that the remove or date options specify.

The <file_name> option value identifies the saved view using the three-part view filename. When you specify more than one view, separate each filename by a space. Refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview* for information on filename format.

Remove only certain saved views parameter for tidy Prov

You can specify that the provisioning system remove only the saved views that you specify, using the remove option:

```
-remove(<file_name>)
```

The <file_name> value identifies the saved view using the three-part view filename. Where you specify more than one view, separate each filename by a space. Refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview* for information on filename format.

Examples for tidy Prov

The following examples show how to use the tidy Prov command:

- “Removing all saved views example” (page 338)
- “Removing saved views from a specific period example” (page 339)
- “Removing all but the specified saved views example” (page 339)
- “Removing all provisioning views that match a specific filename example” (page 340)
- “Removing a single saved view example” (page 340)

Removing all saved views example

You want to remove all saved views. You enter the following command:

```
tidy -all Prov
```

You receive a response similar to the following:

```
Prov
   Non-provisioning file cloneprov will be kept.
   Committed file BE0019A.full.008 will be kept.
   Base file BE0019A.full.006 will be kept.
   34 file(s) deleted.
```

Removing saved views from a specific period example

You want to remove saved views with date stamps that fall within a defined period of time. You enter the following command:

```
tidy -fromDate(199x-06-01) -toDate(199x-06-31) Prov
```

You receive a response similar to the following:

```
Prov
Last used file BE0019A.full.010 will be kept.
Non-provisioning file cloneprov will be kept.
5 file(s) deleted.
```

Removing all but the specified saved views example

You want to remove all saved views except for those views specified in the keep list. You enter the following command:

```
tidy -keep(TEST1.full.009 TEST1.full.004) Prov
```

You receive a response similar to the following:

```
Prov
   Current view file test3.full.001 will be kept.
   Last used file test3.full.010 will be kept.
   Committed file BE0019A.full.008 will be kept.
   Base file BE0019A.full.006 will be kept.
   7 file(s) deleted.
```

Removing all provisioning views that match a specific filename example

You want to remove all provisioning views that match a specific filename, except specified saved views (in this example, TEST2.full.012 and TEST2.full.020). You enter the following command:

```
tidy -remove(TEST2) -keep(TEST2.full.012
TEST2.full.020) Prov
```

You receive a response similar to the following:

```
Prov
Last used file test2.full.022 will be kept.
Base file test2.full.001 will be kept.
5 file(s) deleted.
```

In this example, the last view used (test2.full.022) is part of the set defined by the -remove option. Since it is the last file, you cannot delete it. Also, test2.full.001 is the base for another view, and so you cannot delete it.

Removing a single saved view example

You want to remove a single saved view. Use the remove option and specify the view using a complete filename. You enter the following command:

```
tidy -remove(AB05sn.full.049) Prov
```

You receive a response similar to the following:

```
Prov
1 file(s) deleted
```

Related information for tidy Prov

See the following for information related to the tidy Prov command:

- “Save Prov command” (page 264)
- “Start Prov command” (page 287)

Tidy Sw command

The tidy Sw command lets you remove all unused software application versions and patches from the file system. Before removing any software, you can have the tidy Sw command list the applications that it will remove.

The tidy Sw command considers a software application version unused if it is not in the application version list (the *avList* attribute of the *Software* component) of any semantically checked view, including the current view, the edit view, and saved views. Whenever the tidy Sw command removes an application version, it also removes all patches associated with that application version.

The command fails if a start Sw Dld, remove Sw Av, check Prov, or another tidy Sw command is in progress.

You can remove a particular unused software application using the remove Sw Av command. You can remove unused provisioning files using the tidy Prov command.

For more information on the tidy Sw command, see

- "Properties for tidy Sw" (page 342)
- "Properties for tidy Sw" (page 342)
- "Applications to remove query parameter" (page 342)
- "Deleting all unused software example" (page 343)
- "Related information for tidy Sw" (page 343)

Properties for tidy Sw

The following table describes the properties of the tidy Sw command:

Table 157
Properties of the tidy Sw command

Mode	Impact	Scope	Components
Operational or Provisioning	Configuration	Device	Software (Sw)

Syntax for tidy Sw

The following table describes the syntax for the tidy Sw command:

Table 158
Syntax of the tidy Sw command

If you want to...	Use the following syntax:
Determine which application versions the tidy Sw command will remove	tidy -query Sw
Remove all unused software application versions	tidy Sw

Applications to remove query parameter

You can list the unused software application versions on the node using the query option:

`-query`

The application versions listed using the query option are the ones the tidy Sw command will remove when you issue the command without any options. When you use the query option, the tidy Sw command does not remove any software.

You should always using the tidy Sw command with the query option before issuing it with no options. This practice allows you to verify the application versions the command will delete before it deletes them.

Deleting all unused software example

You want to remove all unused application versions and their associated patches. You list the unused application versions using the following command:

```
tidy -query Sw
```

You receive the following response:

```
Sw
    The following AV(s) would be removed if the Tidy
command was issued without the query option:
       ip_AA008      vns_AA008
```

You want all the application versions on the list deleted, so you enter the following command:

```
tidy Sw
```

You receive the following response:

```
Sw
   The following AV(s) have been removed:
       ip_AA008     vns_AA008
```

Related information for tidy Sw

See the following for information related to the tidy Sw command:

- “Remove Sw Av command” (page 236)
- “Start Sw Dld command” (page 291)
- “Stop Sw Dld command” (page 303)

Touch Fs command

The touch Fs command lets you set the modification date and time of a file or directory. You can set the modification date and time to the current date and time or to a specific date and time.

If the specified file or directory does not exist, the command creates a new empty file (0 bytes).

For more information on the touch command, see

Properties for touch Fs

The following table describes the properties of the touch command.

Table 159
Properties of the touch Fs command

Mode	Impact	Scope	Component
Operational or Provisioning	Configuration	Device	FileSystem (Fs)

Syntax for touch Fs

The following table describes the syntax for the touch Fs command.

Table 160
Syntax of the touch Fs command

If you want to...	Use the following syntax:
Set the modification date and time of a file or directory to the current date and time	touch -path(<path>) Fs
Set the modification date and time of a file or directory to a specific date and time	touch -path(<path>) -date(<datetime>) Fs

Parameters for touch Fs

There are two parameters for the touch Fs command:

- "File or directory to update parameter for touch Fs" (page 345)
- "Date and time parameter for touch Fs" (page 346)

File or directory to update parameter for touch Fs

You specify which file or directory for which you want to set the modification date and time using the path option:

```
-path(<path>)
```

Enter the absolute or relative path for the file or directory for which you want to update the modification date and time. Enclose the path in double quotation marks. For information on path syntax, refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview*.

You must enter this option.

Date and time parameter for touch Fs

You can specify a modification date and time for the file or directory using the date option:

```
-date(<datetime>)
```

Enter the new modification date and time using the format yyyy-mm-dd hh:mm:ss.

If you do not enter a date option, the modification date and time is set to the current node date and time.

Examples for touch

The following examples illustrate how to use the touch command:

- “Setting to current date and time example” (page 346)
- “Setting to specific date and time example” (page 346)

Setting to current date and time example

You want to set the modification date and time of the info.txt file in your current working directory to the current date and time. You enter the following command:

```
touch -path(“info.txt”) Fs
```

You check the modification date and time using the listFile Fs command.

Setting to specific date and time example

You want to set the modification date and time of the info.txt file in your current working directory to exactly 9:00 PM on January 1, 2001. You enter the following command:

```
touch -path(“info.txt”) -date(2001-01-01 09:00:00) Fs
```

You check the modification date and time using the following listFile Fs command:

```
listFile -path(“info.txt”) Fs
```

You receive the following response:

```
Fs
   Type Prot Size       Date       Time     Name
   file no   0          2001-01-01 09:00:00 info.txt
```

Related information for touch Fs

See the following for information related to the workingDir Fs command:

- “ListFile Fs command” (page 168)
- File and directory syntax. Refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview* for details.

Trace command

The trace command is used to initiate a path or connection trace. The path trace traces new point-to-point connections in the process of being established. The connection trace collects information on existing point-to-point connections that have already been established.

You need to set up the parameters for the path trace test connection that are listed in the 241-5701-715 *Passport 7400, 15000, 20000 ATM Monitoring and Troubleshooting Guide* prior to performing the trace command.

For more information on the trace command, see

- "Path trace test connection" (page 348)
- "Path trace filters" (page 351)
- "Connection trace command" (page 356)

Path trace test connection

Path trace test connections can be initiated at the trace source node towards a called party number. A test connection will setup an SPVC from the trace source to the trace destination and will always be released once it reaches the trace destination.

For more information on path trace test connections, see

- "Properties for path trace test connection" (page 348)
- "Syntax for path trace test connection" (page 349)
- "Parameters for path trace test connection" (page 350)
- "Example for path trace test connection" (page 350)

Properties for path trace test connection

The following table displays the properties of the trace command for the path trace test connection.

Table 161
Properties of the trace command

Mode	Impact	Scope	Component
Operational or Provisioning	Configuration	Device	PathTraceTest Connection

Syntax for path trace test connection

The following table describes the syntax for the path trace test connection.

Table 162
Syntax of the path trace test connection

If you want to...	Use the following syntax:
trace a path	trace atmif/* uni/iisp/aini/pnni pathTraceTestConnection
set the path trace test connection destination address	set atmif/* uni/iisp/aini/pnni pathTraceTestConnection testCalledAddress <40 hex digits>
set the path trace test connection vpi.vci	set atmif/* uni/iisp/aini/pnni pathTraceTestConnection testCalledVpiVci <[0-4095].[0-65535]>
(Sheet 1 of 2)	

Table 162 (continued)
Syntax of the path trace test connection

If you want to...	Use the following syntax:
set other path trace test connection attributes	set atmif/* uni/iisp/aini/pnni pathTraceTestConnection <operational_attributes> <operational_attribute_value> where: <operational_attributes> can be traceCrankback, txTrafficDescType, txTrafficDescParm, rxTrafficDescType, rxTrafficDescParm, atmServiceCategory, fwdQosClass, fwdQosParameters, bwdQosClass, bwdQosParameters, bearerClassBbc, transferCapabilityBbc, clippingBbc, bestEffort
trace a path trace test connection	trace atmif/* uni/iisp/aini/pnni pathTraceTestConnection
(Sheet 2 of 2)	

Parameters for path trace test connection

For the path trace, there are no options that need to be specified.

Example for path trace test connection

The verb trace is used to initiate the path trace. For the path trace, there are no options that need to be specified. For a successful path trace test connection, the response will be displayed on the operator console:

```
trace atmif/20 uni PathTraceTestConnection

AtmIf/20 Uni Ptt
Released with :
causeCode = 31
diagCode  = ""
Trace status = Completed normally
Src port Id = 1310721
Src port name = EM/PARIS Atmif/20

Node Id = 60A039123456789123456789124401002O1B124C0060
```

```
Node name = PARIS
Port Id = 1376257
Port name = EM/PARIS Atmif/21

Node Id = 60A039123456789123456789125501009OCF42E80060
Node name = TOKYO
Port Id = 655361
Port name = EM/TOKYO Atmif/10
ok              2001-11-05 15:11:30.29
```

A failed path trace test connection will display the cause code as follows:

```
> trace atmif/20 uni PathTraceTestConnection

AtmIf/20 Uni Ptt
Connection failed or not enough resources.
Cause code= 3
command failed 2001-11-05 08:56:16.88
```

Only one path test connection query is supported per signaling interface at a time. If the trace verb is issued before the previous trace command has responded, the following response will be returned to the later trace command:

```
Trace Command is in progress.
```

If the trace verb is issued and no reply is returned before the timer expires as predefined in the signaling specification, the following response is displayed:

```
The Trace Command has timed out.
```

If the path trace test connection cannot be routed to the called party, the following response is displayed:

```
No route found.
```

Path trace filters

Path trace filters on certain interfaces are configured at the trace source node and are used to define the criteria for which a connection will be traced. The filter criteria includes: called party number, calling party number, and connection type. When a filter is provisioned on a certain interface, all connections going through that interface, which satisfy all defined filtering criteria, will be traced. This is used for connections that will remain up.

For more information on path trace test connections, see

- “Properties for path trace filter” (page 352)
- “Syntax for path trace filter” (page 352)
- “Example for path trace filter” (page 354)
- “Records for path trace filter” (page 356)

Properties for path trace filter

The following table displays the properties of the trace command for the path trace filter:

Table 163
Properties of the trace command for path trace filter

Mode	Impact	Scope	Component
Operational or Provisioning	Configuration	Device	PathTraceFilter
Provisioning	must be added to set the maxTrace-Records	Device	Ov
Provisioning	Must be set to a number between 1 and 200 to ensure the path trace will work properly	Device	maxTraceRecords

Syntax for path trace filter

The following table describes the syntax for the path trace filter:

Table 164
Syntax of the trace command for path trace filter

If you want to...	Use the following syntax:
Set the maximum trace records used by the filter	add Lp/n Eng atmpathTrace Ov set Lp/n Eng atmpathTrace Ov MaxTraceRecords >1 to 200
Add a path trace filter	add atmif/* uni/iisp/aini/pnni ptf
Set the parameters for the path trace filter	set atmif/* uni/iisp/aini/pnni ptf connectionType <svc spvc svp spvp> set atmif/20 uni/iisp/aini/pnni ptf connectionConfiguration <ptp ptmp> set atmif/20 uni/iisp/aini/pnni ptf serviceCategory <cbr rtVbr nrtVbr ubr> set atmif/20 uni/iisp/aini/pnni ptf traceTimeout <1-70560> set atmif/20 uni/iisp/aini/pnni ptf maxTraceRecords <1-200>
Start the path trace filter	start atmif/* uni/iisp/aini/pnni ptf
Stop the path trace filter	stop atmif/* uni/iisp/aini/pnni ptf
Clear the path trace filter	clear atmif/* uni/iisp/aini/pnni ptf
Display the records for successful calls being established	display atmif/* vcc/* PathTraceRecord
Display the record for the successful call at each hop	display atmif/* vcc/* PathTraceRecord hop/*

(Sheet 1 of 2)

Table 164 (continued)
Syntax of the trace command for path trace filter

If you want to...	Use the following syntax:
Display the records for unsuccessful calls being established	display atmif/* uni/iisp/aini/pnni ptf PathTraceRecord
Display the record for the unsuccessful call at each hop	display atmif/* uni/iisp/aini/pnni ptf PathTraceRecord hop/*
(Sheet 2 of 2)	

Example for path trace filter

Before a filter can be started, the user must first add the *Lp/n Eng atmpathTrace Ov* component and then set the *MaxTraceRecords* to a value other than zero (the maximum value of 10 is selected in this example) as follows:

```
> add lp/2 eng atmPathTrace Ov

Lp/2 Eng PTrace Ov
The following components have been created:
Lp/2 Eng PTrace
Lp/2 Eng PTrace Ov
ok              2001-11-04 17:18:49.47


> set Lp/2 Eng PTrace Ov maxTraceRecords 10
Lp/2 Eng PTrace Ov
ok              2001-11-04 17:19:33.69
```

In order to use the path trace filter command, the user must first add it under the uni/iisp/aini/pnni interface. The filter and its attributes can then be modified using operational commands.

```
> add atmif/20 uni ptf
AtmIf/20 Uni Ptf
The following components have been created:
AtmIf/20 Uni Ptf
ok              2001-11-04 17:22:02.50
```

Several verbs are provided for this component allowing the user to START, STOP and CLEAR the records of a filter. The verb START starts the *traceTimeout* timer associated with the filter. If the verb START is run successfully, a response with the amount of records available is provided to the user as follows:

```
> start atmif/20 uni ptf

AtmIf/20 Uni Ptf
There are 10 records available for storage.
ok             2001-11-04 17:22:26.36
```

The verb STOP stops the *traceTimeout* timer associated with the filter. The records associated with the filter are not removed.

```
> stop atmif/20 uni ptf

AtmIf/20 Uni Ptf

ok             2001-11-04 17:24:07.21
```

The verb CLEAR removes all records associated with a stopped filter.

```
> clear atmif/20 uni ptf
AtmIf/20 Uni Ptf

ok             2001-11-04 17:24:14.77
```

The user is not allowed to change the criteria of a running filter. In order to make changes, the filter must first be stopped. The following command is used to display the path trace filter components that can be set:

```
> d -o atmif/20 uni ptf

AtmIf/20 Uni Ptf
traceInfo = crankBack connId callRef
calledPartyPrefix = ""
callingPartyPrefix = ""
connectionType     = svc spvc svp spvp
connectionConfiguration = ptp ptmp
serviceCategory         = cbr rtVbr nrtVbr ubr
traceTimeout            = 60 minutes
maxTraceRecords         = 2
state                   = active
```

```
recordCount             = 0
matchedConnections      = 0
elapsedTime             = 0 minutes
timeRemaining           = 60 minutes
ok             2001-11-04 17:23:20.75
```

Records for path trace filter

The trace information for a successful connection is stored under the *atmif/m vpc/, atmif/n vcc/x.y or atmif/m vpt/x vcc/y* component. The last unsuccessful trace connection is stored under the *atmif/n UNI/IISP/AINI/PNNI PTraceFilter* component.

The number of records that can be stored on a per card basis can be overridden, from its default value by changing the provisionable attribute *maxRecords* under the *lp/m eng atmpathTrace Ov* component. When the maximum number of records has been reached, each filter will overwrite its oldest record if it needs to store a newer record.

In the case where multiple filters are provisioned on the same Lp, each filter has a *maxRecords* attribute associated with the filter. This will allow the operator to manage the number of records allocated to each filter. But if the sum of the filter *maxRecords* attribute is greater than the *maxRecords* attribute of the *Lp Eng atmpathTrace* component, then each filter may not get the requested number of records.

The maximum number of concurrent requests that can be handled can be overridden from its default value by changing the provisionable attribute *maxConcurrentRequests* under the l*p/m eng atmpathTrace ov* component.

Connection trace command

The connection trace collects information on existing point-to-point connections that have already been established. For more information on the connection trace command, see the following sections:

- “Properties for connection trace” (page 357)
- “Syntax for connection trace” (page 357)
- “Parameters for connection trace” (page 357)
- “Examples for connection trace” (page 358)

Properties for connection trace

The following table “Properties of the trace command” (page 349) describes the properties of the trace command:

Table 165
Properties of the trace command

Mode	Impact	Scope	Component
Operational or Provisioning	Configuration	Device	atmif/m vcc/x.y atmif/m vpc/x atmif/m vpt/x vcc/y

Syntax for connection trace

The following table “Syntax of the connection trace command” (page 357) describes the syntax for the connection trace command:

Table 166
Syntax of the connection trace command

If you want to...	Use the following syntax:
trace a connection	trace [-s -v -a]

Parameters for connection trace

For the connection trace, there are several options outlined in the following list:

- “Indicates usage of the new trace command” (page 357)
- “Indicates inclusion of vpi and vci values” (page 358)
- “Indicates inclusion of call reference values” (page 358)

Indicates usage of the new trace command

You can specify that the new trace procedure be used instead of the Passport proprietary OAM based trace procedure by using the -s option:

-s

Note: The -s option must be present for connection trace, otherwise, the trace defaults to the OAM-based trace.

Indicates inclusion of vpi and vci values

You can specify whether the trace includes VPI and VCI values using the -v option:

-v

Note: The -s option must be present if either one of the -v and/or -a options are present. Otherwise, the trace defaults to the OAM-based trace.

Indicates inclusion of call reference values

You can specify whether the trace includes call reference values using the -a option:

-a

Note: The -s option must be present if either one of the -v and/or -a options are present. Otherwise, the trace defaults to the OAM-based trace.

Examples for connection trace

For a successful connection test, you receive the following response:

trace -s -v -a atmif/20 vcc/0.100

```
AtmIf/20 Vcc/0.100
Switch side trace result:
Trace status = Completed normally
Src port Id = 1310721
Src port name = EM/LUCIELUCIE Atmif/20

Vpi.Vci = 0.100
Call reference = 7FFFFF
Node Id = 60A039123456789123456789124401002O1B124C0060
Node name = LUCIELUCIE
Port Id = 1441793
Port name = EM/LUCIELUCIE Atmif/22
```

```
Vpi.Vci = 0.560
Call reference = 01
Node Id = 60A039123456789123456789125501009OCF42E80060
Node name = QUIGONQUIGON
Port Id = 655361
Port name = EM/QUIGONQUIGON Atmif/10
Destination vpi.vci= 0.100
Destination call reference = 7FFFFF
ok              2001-11-04 18:20:27.68
```

Only one connection trace query is supported per signaling component at a time. If the trace verb is issued before the previous trace command has responded, the following response will be returned to the later trace command:

```
Trace Command is in progress.
```

If the trace verb is issued and no reply is returned before the timer has expired, the following response is displayed:

```
The Trace Command has timed out.
```

If the trace verb is issued to a *vpc/x, vcc/x.y, or a vpt/x vcc/y* component that is not active or was not created by signaling, the following response is displayed:

```
Connection is not active, or trace is not applicable.
```

Related information for path filter trace

See the following for information related to the path filter trace command:

- “Start Prov command” (page 287)
- “Stop Prov command” (page 300)
- “Clear Prov command” (page 91)

Unlock command

The unlock command lets you unlock a component you have previously locked using the lock command. By locking a component, you prevent it from providing service. By unlocking a locked component, you allow the component to provide service again.

Before you unlock a component, you should verify its OSI administrative state by displaying the *adminState* attribute of the component.

For more information on the unlock command, see

- "Properties for unlock" (page 361)
- "Syntax for unlock" (page 361)
- "Unlocking a locked component example" (page 361)
- "Related information for unlock" (page 362)

Properties for unlock

The following table describes the properties of the unlock command.

Table 167
Properties of the unlock command

Mode	Impact	Scope	Components
Provisioning	Service	Device	Many

Syntax for unlock

The following table describes the syntax for the unlock command.

Table 168
Syntax of the unlock command

If you want to...	Use the following syntax:
Unlock a component	unlock <component_name>

Unlocking a locked component example

You want to unlock the *Lp/1* component. Before unlocking the component, you check its OSI administrative state using the following command:

```
display Lp/1 adminState
```

You receive the following response:

```
Lp/1
   adminState = locked
```

Since Lp/1 is locked, you unlock it using the following command:

```
unlock Lp/1
```

Passport issues an alarm that indicates Lp/1 is now unlocked. You verify the administrative state of Lp/1 using the following command:

```
display Lp/1 adminState
```

You receive the following response:

```
Lp/1
   adminState = unlocked
```

Related information for unlock

See the following for information related to the unlock command:

- “Lock command” (page 177)

Unprotect Fs command

The unprotect Fs command removes the deletion and modification protection from a file.

When a file is protected, you cannot delete it using the remove Fs command. To delete a protected file, you must first remove its protection using the unprotect Fs command. You can tell if a file is protected or not using the listFile Fs command.

You cannot protect or unprotect a directory.

For more information on the unprotect Fs command, see

Properties for unprotect Fs

The following table describes the properties of the unprotect Fs command:

Table 169
Properties of the unprotect Fs command

Mode	Impact	Scope	Component
Operational or Provisioning	Configuration	Device	FileSystem (Fs)

Syntax for unprotect Fs

The following table describes the syntax for the unprotect Fs command.

Table 170
Syntax of the unprotect Fs command

If you want to...	Use the following syntax:
Remove deletion protection a file	unprotect -path(filepath) Fs

File to unprotect parameter for unprotect Fs

You specify the file for which you want to remove deletion and modification protection using the path option:

```
-path(<filepath>)
```

Enter the absolute or relative path for the file for which you want to remove deletion and modification protection. Enclose the path in double quotation marks. For information on path syntax, refer to 241-5701-030 *Passport 7400, 15000, 20000 Overview*.

You must enter the path option.

Unprotecting a file example

There is a protected file in the /tmp directory called alarm.001 that you want to delete. Before deleting it, you must remove its deletion and modification protection. Your current working directory is root (/). You enter the following command:

```
unprotect -path(“/tmp/alarm.001”) Fs
```

You use the following listFile Fs commands to verify that the file was unprotected:

```
listFile -path(“/tmp/alarm.001”) Fs
```

You receive the following response:

```
Fs
   Type Prot Size    Date       Time      Name
   file no   9547    1998-05-05 10:12:38  alarm.001
```

The *no* in the Prot column indicates that the file is unprotected.

Related information for unprotect Fs

See the following for information related to the unprotect Fs command:

- “ListFile Fs command” (page 168)
- “Protect Fs command” (page 215)
- “Remove Fs command” (page 232)

WorkingDir Fs command

The workingDir Fs (pwd Fs) command displays your current working directory.

For more information on the workingDir Fs command, see

Properties for workingDir Fs

The following table describes the properties of the workingDir command.

Table 171
Properties of the workingDir command

Mode	Impact	Scope	Component
Operational or Provisioning	Passive	Device	FileSystem (Fs)

Syntax for workingDir Fs

The following table describes the syntax for the workingDir Fs command.

Table 172
Syntax of the workingDir Fs command

If you want to...	Use the following syntax:
Display your current working directory	workingDir Fs

Displaying the current working directory example

Your current working directory is /spooled/closed. You enter the following command:

```
workingDir Fs
```

You receive the following response:

```
Fs
   /spooled/closed
```

Related information for workingDir Fs

See the following for information related to the workingDir Fs command:

- “Chdir Fs command” (page 68)
- “ListFile Fs command” (page 168)

Passport 7400, 15000, 20000
Commands

Release 5.2

Publication: 241-5701-050
Document status: Standard
Document version: 5.2S2
Document date: December 2003
Printed in Canada