XORP Command Line Interface User Guide
Part 1: Command Structure

Version 1.0

XORP Project
International Computer Science Institute
Berkeley, CA 94704, USA
feedback@xorp.org

July 8, 2004

1 Introduction

To interact with a XORP router using the command line interface (CLI), a user runs xorpsh. This allows configuration of the router and monitoring of the router state.

In this document we describe how to interact with xorpsh. This document does not provide specifics of how to configure BGP, PIM, SNMP and other processes - this will eventually be described in additional documents.

The user interface style is loosely modelled on that of a Juniper router. This manual and the xorpsh itself are works in progress, and so may change significantly in the future.

2 Running xorpsh

xorpsh provides an interactive command shell to a XORP user, similar in many ways to the role played by a Unix shell. In a production router, it is envisaged that xorpsh might be set up as a user's login shell - they would login to the router via ssh and be directly in the xorpsh environment. However, for research and development purposes, it makes more sense to login normally to the machine running the rtrmgr process, and to run xorpsh directly from the Unix command line.

xorpsh should normally be run as a regular user; it is neither necessary or desirable to run it as root. If the user is to be permitted to make changes to the running router configuration, they need to be in the Unix group `xorp`. The choice of GID for group xorp is not important.

xorpsh needs to be able to communicate with the rtrmgr using the local file system. If the rtrmgr cannot write files in /tmp that xorpsh can read, then xorpsh will not be able to authenticate the user to the rtrmgr.

Multiple users can run xorpsh simultaneously. There is some degree of configuration locking to prevent simultaneous changes to the router configuration, but currently this is fairly primitive.

3 Basic Commands

On starting xorpsh, you will be presented with a command line prompt:

```
Xorp>
```

You can exit xorpsh at any time by trying Control-d.
Typing "?" at the prompt will list the commands currently available to you:

```
Xorp> ?
Possible completions:
  configure   Switch to configuration mode
  help        Provide help with commands
  quit        Quit this command session
  show        Display information about the system
```

If you type the first letter or letters of a command, and hit `<Tab>`, then command completion will occur. At any time you can type "?" again to see further command completions. For example:

```
Xorp> config?
Possible completions:
  configure   Switch to configuration mode
Xorp> config
```

If the cursor is after the command, typing "?" will list the possible parameters for the command:

```
Xorp> configure ?
Possible completions:
  <[Enter]>   Execute this command
  exclusive   Switch to configuration mode, locking out other users
Xorp> configure
```

3.1 Command History and Command Line Editing

xorpsh supports emacs-style command history and editing of the text on the command line. The most important commands are:

- The **up-arrow** or **control-p** moves to the previous command in the history.
- The **down-arrow** or **control-n** moves to the next command in the history.
- The **left-arrow** or **control-b** moves back along the command line.
- The **right-arrow** or **control-f** move forward along the command line.
- **control-a** moves to the beginning of the command line.
- **control-e** moves to the end of the command line.
- **control-d** deletes the character directly under the cursor.
- **control-t** toggles (swaps) the character under the cursor with the character immediately preceding it.
- **control-space** marks the current cursor position.

- **control-w** deletes the text between the mark and the current cursor position, copying the deleted text to the cut buffer.
- **control-k** kills (deletes) from the cursor to the end of the command line, copying the deleted text to the cut buffer.
- **control-y** yanks (pastes) the text from the cut buffer, inserting it at the current cursor location.

4 Command Modes

xorpsh has two command modes:

Operational Mode, which allows interaction with the router to monitor it's operation and status.

Configuration Mode, which allows the user to view the configuration of the router, to change that configuration, and to load and save configurations to file.

Generally speaking, operational mode is considered to give non-privileged access; there should be nothing a user can type that would seriously impact the operation of the router. In contrast, configuration mode allows all aspects of router operation to be modified.

In the long run, xorpsh and the rtrmgr will probably come to support fine-grained access control, so that some users can be given permission to change only subsets of the router configuration. At the present time though, there is no fine-grained access control.

A user can only enter configuration mode if they are in the `xorp` Unix group.

5 Operational Mode

```
Xorp> ?
Possible completions:
  configure   Switch to configuration mode
  help        Provide help with commands
  quit        Quit this command session
  show        Display information about the system
```

The main commands in operational mode are:

configure: switches from operational mode to configuration mode.

help: provides online help.

quit: quit from xorpsh.

show: displays many aspects of the running state of the router.

5.1 Show Command

```
Xorp> show ?
Possible completions:
  <[Enter]>   Execute this command
  bgp         Display information about BGP
  host        Display information about the host
  igmp        Display information about IGMP
  interfaces  Show network interface information
  mfea        Display information about IPv4 MFEA
  mfea6       Display information about IPv6 MFEA
  mld         Display information about MLD
  pim         Display information about IPv4 PIM
  pim6        Display information about IPv6 PIM
  rip         Display information about RIP
  route       Show route table
Xorp> show
```

The *show* command is used to display many aspects of the running state of the router. We don't describe the sub-commands here, because they depend on the running state of the router. For example, only a router that is running BGP should provide `show bgp` commands[1].

As an example, we show the peers of a BGP router:

[1]Note that currently all possible commands are shown, even if the router is not running a particular protocol.

```
Xorp> **show bgp peers detail**
OK
Peer 1: local 192.150.187.108/179 remote 192.150.187.109/179
  Peer ID: 192.150.187.109
  Peer State: ESTABLISHED
  Admin State: START
  Negotiated BGP Version: 4
  Peer AS Number: 65000
  Updates Received: 5157, Updates Sent: 0
  Messages Received: 5159, Messages Sent: 1
  Time since last received update: 4 seconds
  Number of transitions to ESTABLISHED: 1
  Time since last entering ESTABLISHED state: 47 seconds
  Retry Interval: 120 seconds
  Hold Time: 90 seconds, Keep Alive Time: 30 seconds
  Configured Hold Time: 90 seconds, Configured Keep Alive Time: 30 seconds
  Minimum AS Origination Interval: 0 seconds
  Minimum Route Advertisement Interval: 0 seconds
```

6 Configuration Mode

```
Xorp> configure
Entering configuration mode.
There are no other users in configuration mode.

[edit]
XORP>
```

When in configuration mode, the command prompt changes to be all capitals. The command prompt is also usually preceded by a line indicating which part of the configuration tree is currently being edited.

```
[edit]
XORP> ?
Possible completions:
  create         Create a sub-element
  delete         Delete a configuration element
  edit           Edit a sub-element
  exit           Exit from this configuration level
  help           Provide help with commands
  load           Load configuration from a file
  quit           Quit from this level
  run            Run an operational-mode command
  save           Save configuration to a file
  set            Set the value of a parameter
  show           Show the value of a parameter
  top            Exit to top level of configuration
  up             Exit one level of configuration
XORP>
```

The router configuration has a tree form similar to the directory structure on a Unix filesystem. The current configuration or parts of the configuration can be shown with the *show* command:

```
[edit]
XORP> show interfaces
  interface rl0 {
    description: "control interface"
    vif rl0 {
      address 192.150.187.108 {
        prefix-length: 25
        broadcast: 192.150.187.255
        enabled: true
      }
      enabled: true
    }
    enabled: true
  }
  targetname: "fea"
```

6.1 Moving around the Configuration Tree

You can change the current location in the configuration tree using the *edit*, *exit*, *quit*, *top* and *up* commands.

- **edit** <***element name***>: Edit a sub-element
- **exit**: Exit from this configuration level, or if at top level, exit configuration mode.
- **quit**: Quit from this level
- **top**: Exit to top level of configuration
- **up**: Exit one level of configuration

For example:

```
[edit]
XORP> edit interfaces interface rl0 vif rl0

[edit interfaces interface rl0 vif rl0]
XORP> show
  address 192.150.187.108 {
    prefix-length: 25
    broadcast: 192.150.187.255
    enabled: true
  }
  enabled: true

[edit interfaces interface rl0 vif rl0]
XORP> up

[edit interfaces interface rl0]
XORP> top

[edit]
XORP>
```

6.2 Loading and Saving Configurations

On startup, the rtrmgr will read a configuration file. It will then start up and configure the various router components as specified in the configuration file.

The configuration file can be created externally, using a normal text editor, or it can be saved from the running router configuration. A configuration file can also be loaded into a running router, causing the previous running configuration to be discarded. The commands for this are:

- **save** <***filename***>: save the current configuration in the specified file.
- **load** <***filename***>: load the specified file, discarding the currently running configuration.

6.3 Setting Configuration Values

- **set** <***path to config***> <***value***>: set the value of the specified configuration node.

The *set* command is used to set or change the value of a configuration option. The change does not actually take effect immediately - the *commit* command must be used to apply this and any other uncommitted changes.

In the example below, the prefix length (netmask) of address 192.150.187.108 on vif rl0 is changed, but not yet committed. The ">" indicates parts of the configuration that have changed but not yet been committed.

```
[edit interfaces interface rl0]
XORP> show
  description: "control interface"
  vif rl0 {
    address 192.150.187.108 {
      prefix-length: 25
      broadcast: 192.150.187.255
      enabled: true
    }
    enabled: true
  }
  enabled: true

[edit interfaces interface rl0]
XORP> set vif rl0 address 192.150.187.108 prefix-length 24
OK

[edit interfaces interface rl0]
XORP> show
  description: "control interface"
  vif rl0 {
    address 192.150.187.108 {
>     prefix-length: 24
      broadcast: 192.150.187.255
      enabled: true
    }
    enabled: true
  }
```

6.4 Adding New Configuration

- **create** <***path to new config node***> : create new configuration node.
- **create** <***path to new config node***> { : create new configuration node and start editing it.

New configuration can be added by the *create* command. If we type *create* followed by the path to a new configuration node, the node will be created. All parameters within that node will be assigned their default values (if exist). After that the node can be edited with the *edit* command. If we type { after the path to the new configuration node, the node will be created, the default values will be assigned, and we can directly start editing that node. The user interface for this is currently rather primitive and doesn't permit the more free-form configuration allowed in configuration files.

For example, to configure a second vif on interface rl0:

```
[edit interfaces interface rl0]
XORP> show
  description: "control interface"
  vif rl0 {
    address 192.150.187.108 {
      prefix-length: 24
      broadcast: 192.150.187.255
      enabled: true
    }
    enabled: true
  }
  enabled: true

[edit interfaces interface rl0]
XORP> create vif rl0b {
    > address 10.0.0.1 {
    >   prefix-length 16
    >   broadcast 10.0.255.255
    >   enabled true
    >   }
    >   enabled true
    > }
OK. Use "commit" to apply these changes.

[edit interfaces interface rl0]
XORP> show
  description: "control interface"
  vif rl0 {
    address 192.150.187.108 {
      prefix-length: 24
      broadcast: 192.150.187.255
      enabled: true
    }
    enabled: true
  }
> vif rl0b {
>   address 10.0.0.1 {
>     prefix-length: 16
>     broadcast: 10.0.255.255
>     enabled: true
>   }
>   enabled: true
> }
  enabled: true
```

6.5 Deleting Parts of the Configuration

The *delete* command can be used to delete subtrees from the configuration. The deletion will be visible in the results of the *show* command, but will not actually take place until the changes are committed.

```
XORP> show interfaces interface rl0
  description: "control interface"
  vif rl0 {
    address 192.150.187.108 {
      prefix-length: 24
      broadcast: 192.150.187.255
      enabled: true
    }
    enabled: true
  }
  vif rl0b {
    address 10.0.0.1 {
      prefix-length: 16
      broadcast: 10.0.255.255
      enabled: true
    }
    enabled: true
  }
  enabled: true

[edit]
XORP> delete interfaces interface rl0 vif rl0b
Deleting:
  address 10.0.0.1 {
    prefix-length: 16
    broadcast: 10.0.255.255
    enabled: true
  }

OK

[edit]
XORP> show interfaces interface rl0
  description: "control interface"
  vif rl0 {
    address 192.150.187.108 {
      prefix-length: 24
      broadcast: 192.150.187.255
      enabled: true
    }
    enabled: true
  }
  enabled: true
```

6.6 Committing Changes

```
[edit interfaces interface rl0]
XORP> **commit**
OK
```

The *commit* command commits all the current configuration changes. This can take a number of seconds before the response is given.

If xorpsh was built with debugging enabled, the response can be considerably more verbose than shown above!

If two or more users are logged in using configuration mode, and one of them changes the configuration, the others will receive a warning:

```
[edit]
XORP>
The configuration had been changed by user mjh
XORP>
```

6.7 Discarding Changes

The user can discard a batch of changes by editing them back to their original configuration, or by using the *exit* command to leave configuration mode:

```
[edit]
XORP> exit
ERROR: There are uncommitted changes
Use "commit" to commit the changes, or "exit discard" to discard them

XORP> exit discard
Xorp>
```