VYATTA, INC. | Vyatta System

WAN Interfaces

REFERENCE GUIDE

Serial Interfaces
DSL Interfaces
Wireless Modem Interfaces

Vyatta
Suite 200
1301 Shoreway Road
Belmont, CA 94002
vyatta.com
650 413 7200
1 888 VYATTA 1 (US and Canada)

ISSUE DATE: February 2009

DOCUMENT REVISION. VC5 v03

RELEASED WITH: VC5.0.2

PART NO. A0-0112-10-0002

Table of Contents

Quick Reference to Commands

Use this section to help you quickly locate a command.

Quick List of Examples

Use this list to help you locate examples you'd like to try or look at.

Preface

This guide explains how to configure and use interfaces for the wide area network (WAN). It describes the available commands and provides configuration examples.

This preface provides information about using this guide. The following topics are covered:

- Intended Audience
- Organization of This Guide
- Document Conventions
- Vyatta Publications

Intended Audience

This guide is intended for experienced system and network administrators. Depending on the functionality to be used, readers should have specific knowledge in the following areas:

- Networking and data communications
- TCP/IP protocols
- General router configuration
- Routing protocols
- Network administration
- Network security

Organization of This Guide

This guide has the following aid to help you find the information you are looking for:

- **Quick Reference to Commands**

  Use this section to help you quickly locate a command.

- **Quick List of Examples**

  Use this list to help you locate examples you'd like to try or look at.

This guide has the following chapters and appendixes:

Chapter	Description	Page
Chapter 1: Serial Interfaces	This chapter explains how to work with serial interfaces on the Vyatta system.	1
Chapter 2: DSL Interfaces	This chapter explains how to use Digital Subscriber Line (DSL) interfaces on the Vyatta system. Currently the Vyatta system supports Asymmetrical DSL (ADSL) interfaces only.	82
Chapter 3: Wireless Modem Interfaces	This chapter explains how to work with wireless modems on the Vyatta system.	98
Glossary of Acronyms		122

Document Conventions

This guide contains advisory paragraphs and uses typographic conventions.

Advisory Paragraphs

This guide uses the following advisory paragraphs:

Warnings alert you to situations that may pose a threat to personal safety, as in the following example:

WARNING *Risk of injury. Switch off power at the main breaker before attempting to connect the remote cable to the service power at the utility box.*

Cautions alert you to situations that might cause harm to your system or damage to equipment, or that may affect service, as in the following example:

CAUTION *Risk of loss of service. Restarting a running system will interrupt service.*

Notes provide information you might need to avoid problems or configuration errors:

NOTE *You must create and configure network interfaces before enabling them for routing protocols.*

Typographic Conventions

This document uses the following typographic conventions:

`Courier`	Examples, command-line output, and representations of configuration nodes.
`boldface Courier`	In an example, your input: something you type at a command line.
boldface	In-line commands, keywords, and file names .
italics	Arguments and variables, where you supply a value.
<key>	A key on your keyboard. Combinations of keys are joined by plus signs (“+”). An example is <Ctrl>+<Alt>+<Del>.
[*arg1* | *arg2*]	Enumerated options for completing a syntax. An example is [enable | disable].

num1–numN	A inclusive range of numbers. An example is 1–65535, which means 1 through 65535.
arg1..argN	A range of enumerated values. An example is eth0..eth3, which means eth0, eth1, eth2, and eth3.
arg [*arg* ...] *arg*,[*arg*,...]	A value that can optionally represent a list of elements (a space-separated list in the first case, and a comma-separated list in the second case).

Vyatta Publications

More information about the Vyatta system is available in the Vyatta technical library, and on www.vyatta.com and www.vyatta.org.

Full product documentation is provided in the Vyatta technical library. To see what documentation is available for your release, see the *Vyatta Documentation Map*. This guide is posted with every release of Vyatta software and provides a great starting point for finding what you need.

Chapter 1: Serial Interfaces

This chapter explains how to work with serial interfaces on the Vyatta system.

This chapter presents the following topics:

- Serial Interface Configuration
- Testing Serial Interfaces
- Serial Interface Commands

Serial Interface Configuration

This section presents the following topics:

- Virtual Interfaces on Serial Interfaces
- Enabling Interfaces
- Viewing Available Serial Interfaces

Virtual Interfaces on Serial Interfaces

The Vyatta system distinguishes between physical interfaces (*interfaces*), and logical interfaces (*virtual interfaces*, or *vifs*).

Every physical network device in the system is considered to be an "interface." An example of a interface is a physical port on a serial card. Every serial interface has zero or more corresponding vifs.

On serial interfaces, physical line characteristics are specific for the interface, but encapsulation (Cisco HDLC, Frame Relay, or Point-to-Point Protocol) is specified for vifs.

Unlike Ethernet interfaces, a physical serial interface cannot directly have a configured IP address. Instead, the IP address must be assigned to the vif.

Note that each serial vif can support exactly one IP address.

Enabling Interfaces

The Vyatta system will automatically discover any available physical serial interfaces on startup. Before you can apply any configuration to a serial interface, a vif must be "created" for the interface and its encapsulation specified in the configuration tree.

For serial interfaces, physical line characteristics are applied to the interface as a whole. Encapsulation characteristics are applied to the vif, as shown in the configuration hierarchy below:

```
interfaces {
   serial wan0 {
      ppp {
         vif 1 {
         }
      }
   }
}
```

The current implementation supports Cisco HDLC, Frame Relay, and Point-to-Point Protocol encapsulation.

- Cisco HDLC and point-to-point interfaces support only one vif, and this vif must have the identifier “1”.
- The identifier for Frame Relay vifs is the DLCI number. This can range from 16 through 991.
- Currently, any vif on a serial interface can support exactly one IP address.

Viewing Available Serial Interfaces

You can only configure interfaces that actually are available to the operating system on the hardware you are using.

To view all the interfaces known to the operating system, use the **show interfaces system** command in operational mode, as shown in Example 1-1:

Example 1-1 Viewing available system interfaces

```
vyatta@vyatta> show interfaces system
```

Testing Serial Interfaces

This section presents the following topics:

- Serial Loopbacks
- Serial Line Testing Strategy

If a problem occurs on a serial line it is useful to be able to systematically test it in order to isolate the location of the problem. In general, the problem must be:

- On the local device,
- On the remote device, or
- On the line that connects the two devices.

One of the main tools for isolating problems on serial lines is through the use of a loopback.

Serial Loopbacks

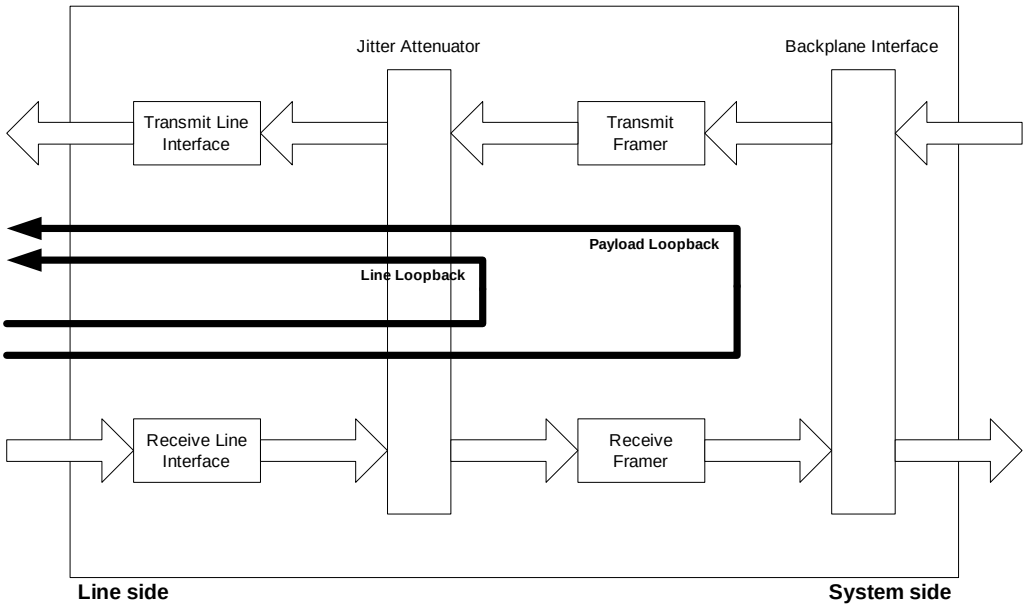
Serial loopbacks operate by configuring the card to return data it receives back to its source. Serial loopbacks can either be *line-facing* (that is, they return data received from the T1/E1/T3/E3 line back to the line), or *system-facing* (that is, they return data received from the system back to the system). The loopbacks are provided at various points in the card in order to diagnose problems on the line or on the card itself.

Serial loopbacks are built into the serial card. The exact loopbacks available depend the chipset of the card. The Vyatta system supports Sangoma serial cards, which may use one of a number of chipsets; the loopback options on the Sangoma card you are using depend on the chipset on the card.

The Vyatta system auto-detects the chipset on your Sangoma card: the CLI command completion mechanism displays all the options, and only the options, supported by the chipset on your card. Likewise, the CLI will only accept options supported by the chipset on your card.

Figure 1-1 provides a generalized block diagram of the chipset on a serial card, showing the line-facing loopbacks within the chipset.

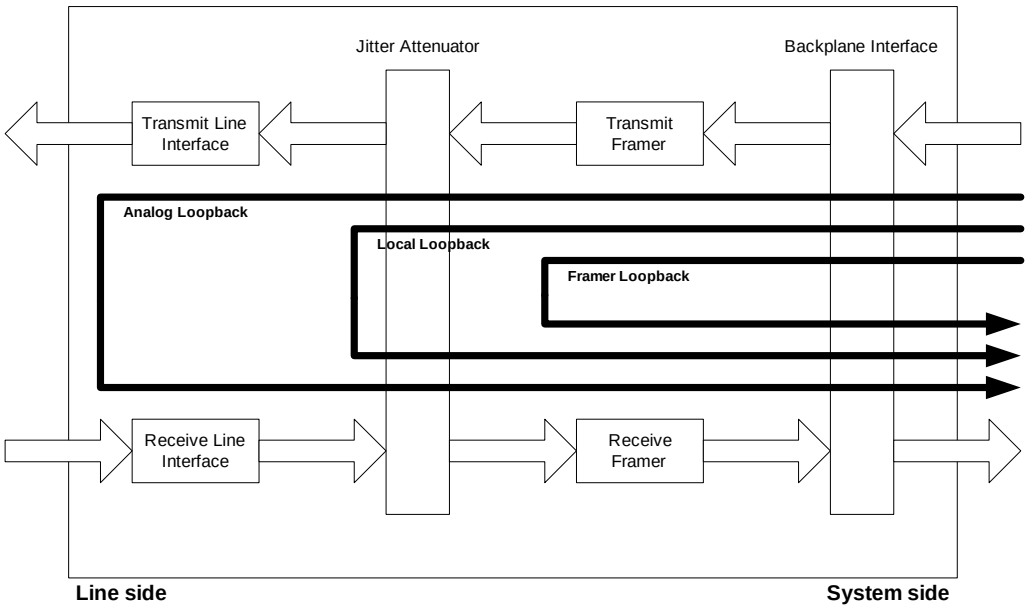
Figure 1-1 Line-facing loopbacks



NOTE *Receive lines on the system side are ignored during line-facing loopbacks.*

Figure 1-2 provides a generalized block diagram of the chipset on a serial card, showing the system-facing loopbacks within the chipset.

Figure 1-2 System-facing loopbacks



NOTE *Transmit lines on the line side are ignored during system-facing loopbacks.*

Table 1-1 summarizes the loopback options available on the various Sangoma T1/E1 and T3/E3 card chipsets. It also shows, in parenthses, the names of the loopbacks used by the chipset manufacturers - which differ, in some cases, to the names used in the Vyatta CLI.

Table 1-1 Loopback options available on Sangoma card chipsets

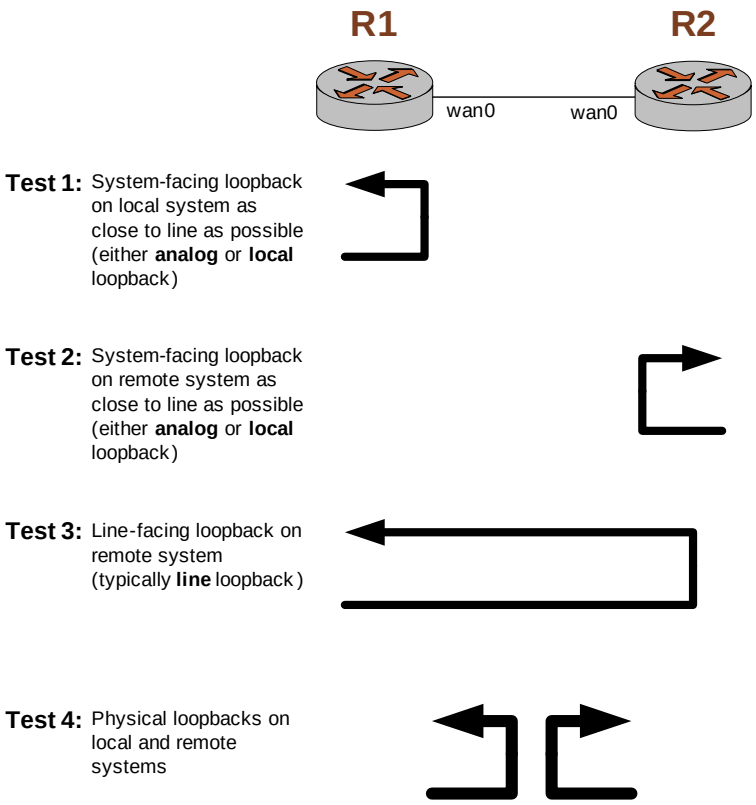
Option	Maxim	PMC-Sierra	Exar
line	Yes (RLB)	Yes (Line)	Yes (Remote)
payload	Yes (PLB)	Yes (Payload)	-
analog	Yes (ALB)	-	Yes (Analog)
local	Yes (LLB)	Yes (Diagnostic)	Yes (Digital)
framer	Yes (FLB)	-	-

NOTE *On Sangoma cards with the PMC-Sierra chipset all ports must be configured with the same line type (e.g. all T1 or all E1).*

Serial Line Testing Strategy

Figure 1-3 shows two Vyatta devices, R1 and R2, connected via a serial line. The tests that follow describe the steps that could be performed to diagnose a communication problem between the two devices.

Figure 1-3 Sample scernario for loopback testing



The general strategy is to sequentially test for problems in the local system, in the remote system, and in the circuit connecting the two. To do this, the serial loopback operational commands are used. These are the **loopback down** command (see page 72) command, tthe **loopback test** command (see page 74) and tthe **loopback up** command (see page 76).

NOTE *For examples showing successful and unsucessful tests, refer to the* ***loopback test*** *command (see page 74).*

Test 1: Test the Local System

The first step is to determine whether there is a problem on the WAN card on the local system. To do this, a loopback is defined as close to the line as possible and then run the loopback test.

Example 1-2 Testing the local WAN interface

Step	Command
Specify an analog loopback on the local system.	`vyatta@R1> loopback up wan0 analog`
Send test data through the loopback.	`vyatta@R1> loopback test wan0`
Turn off the analog loopback on the local system once the test is complete.	`vyatta@R1> loopback down wan0 analog`

A failure of this test indicates a problem on the local WAN interface. If the test is successful, proceed to Test 2.

Test 2: Test the Remote System

The second step is to determine if there is a problem on the WAN card in the remote system. To do this we set up a loopback as close to the line as possible and then run the loopback test on the remote system.

Example 1-3 Testing the remote WAN interface

Step	Command
Specify an analog loopback on the remote system.	`vyatta@R2> loopback up wan0 analog`
Send test data through the loopback.	`vyatta@R2> loopback test wan0`

Example 1-3 Testing the remote WAN interface

Turn off the analog loopback on the remote system once the test is complete.	`vyatta@R2> loopback down wan0 analog`

A test failure indicates a problem on the remote WAN interface. If the test is successful, proceed to Test 3.

Test 3: Test the Circuit

The third step is to confirm a problem with the circuit between the two devices. To do this, a line-facing loopback is defined on the remote system and then the loopback test run from the local system.

Example 1-4 Setting up the **remote** system for a remote loopback

Step	Command
Specify a line-facing loopback on the **remote** system.	`vyatta@R2> loopback up wan0 line`

Example 1-5 Testing the remote loopback from the **local** system

Step	Command
Send test data through the remote loopback from the **local** system.	`vyatta@R1> loopback test wan0`

Example 1-6 Turning off remote loopback on the **remote** system

Step	Command
Turn off the line loopback on the **remote** system once the test is complete.	`vyatta@R2> loopback down wan0 line`

A successful test indicates that all of the components are working properly. A test failure indicates that there is a problem on the physical interface of either the local or the remote WAN interface. In this case, proceed to Test 4.

Test 4: Test the Physical Interfaces

The fourth step is to confirm a problem on the physical interface of the WAN card on either the local or the remote system. You can create the physical loopback in a number of ways:

- Using a physical loopback plug
- Using a loopback at a patch panel
- Using a loopback at the circuit provider.

In each case, the idea is to provide a loopback that is external to the WAN card and then run the loopback test on the system with the physical loopback installed. Example 1-7 shows a physical loopback test run on the local system.

Example 1-7 Testing a physical loopback on the local system

Step	Command
Send test data through the physical loopback on the local system.	`vyatta@R1> loopback test wan0`

If the test is successful on the local system, run the same test on the remote system. If the test on the remote system is also successful, this indicates a problem with the circuit between the two devices (and therefore a problem for the circuit provider to resolve), since all other components have been tested.

Serial Interface Commands

This chapter contains the following commands.

Configuration Commands	
Serial Interface	
interfaces serial <wanx>	Specifies basic serial interface configuration, including Layer 2 encapsulation characteristics.
interfaces serial <wanx> description <desc>	Specifies a description for a serial interface.
interfaces serial <wanx> encapsulation <type>	Sets the encapsulation type for a serial interface.
E1 Options	
interfaces serial <wanx> e1-options	Specifies the physical line characteristics for an E1 serial interface.
interfaces serial <wanx> e1-options clock <type>	Sets the timing source for an E1 serial interface.
interfaces serial <wanx> e1-options framing <type>	Sets the framing for an E1 serial interface.
interfaces serial <wanx> e1-options mru <mru>	Specifies the Maximum Receive Unit (MRU) size for an E1 serial interface.
interfaces serial <wanx> e1-options mtu <mtu>	Specifies the Maximum Transmit Unit (MTU) size for an E1 serial interface.
interfaces serial <wanx> e1-options timeslots	Defines timeslots for a 32-channel channelized E1 line.
E3 Options	
interfaces serial <wanx> e3-options	Specifies the physical line characteristics for an E3 serial interface.
interfaces serial <wanx> e3-options clock <type>	Specifies the timing source for an E3 serial interface.
interfaces serial <wanx> e3-options framing <type>	Specifies the framing type for an E3 serial interface.
interfaces serial <wanx> e3-options line-coding <type>	Specifies the line coding for an E3 serial interface.
Synchronous Serial	
interfaces serial <wanx> synch-options	Specifies the physical line characteristics for synchronous serial interfaces.

interfaces serial <wanx> synch-options baud-rate <rate>	Sets the bit rate for an internally clocked synchronous serial interface.
interfaces serial <wanx> synch-options clock <type>	Sets the timing source for a synchronous serial interface.
interfaces serial <wanx> synch-options connection <type>	Sets the connection type for a synchronous serial interface.
interfaces serial <wanx> synch-options line-coding <type>	Sets the line coding standard for a synchronous serial interface.
interfaces serial <wanx> synch-options line-idle <type>	Sets the idle line signalling for a synchronous serial interface.
T1 Options	
interfaces serial <wanx> t1-options	Specifies the physical line characteristics for T1 serial interfaces.
interfaces serial <wanx> t1-options clock <type>	Sets the timing source for a T1 serial interface.
interfaces serial <wanx> t1-options lbo <range>	Specifies the line build-out (LBO) range for a T1 serial interface.
interfaces serial <wanx> t1-options mru <mru>	Specify the Maximum Receive Unit (MRU) size for a T1 serial interface.
interfaces serial <wanx> t1-options mtu <mtu>	Specify the Maximum Transmit Unit (MTU) size for a T1 serial interface.
interfaces serial <wanx> t1-options timeslots	Defines timeslots for a 24-channel channelized T1 line.
T3 Options	
interfaces serial <wanx> t3-options	Specifies the physical line characteristics for a T3 serial interface.
interfaces serial <wanx> t3-options clock <type>	Specifies the timing source for the circuit.
interfaces serial <wanx> t3-options framing <type>	Specifies the framing type for a T3 serial interface.
interfaces serial <wanx> t3-options line-coding <type>	Specifies the line coding for a T3 serial interface.
Operational Commands	
clear interfaces serial	Clears counters for serial interfaces
show interfaces serial	Displays serial interface information.

Serial Loopback Commands	
loopback down	Deactivates loopbacks on a Sangoma T1/E1 or T3/E3 card.
loopback test	Starts a loopback diagnostic test on a Sangoma T1/E1 or T3/E3 card.
loopback up	Activates a loopback on a Sangoma T1/E1 or T3/E3 card.

Commands for using other system features with serial interfaces can be found in the following locations.

Related Commands Documented Elsewhere	
Firewall	Commands for configuring firewall on serial interfaces are described in the *Vyatta Security Reference Guide*.
OSPF	Commands for configuring the Open Shortest Path First routing protocol on serial interfaces are described in the *Vyatta OSPF Reference Guide*.
RIP	Commands for configuring the Routing Information Protocol on serial interfaces are described in the *Vyatta RIP Reference Guide*.
QoS	Commands for configuring quality of service on serial interfaces are described in the *Vyatta Policy and QoS Reference Guide*.
System interfaces	Commands for showing the physical interfaces available on your system are described in the Vyatta Basic System Reference Guide.
VRRP	Commands for configuring Virtual Router Redundancy Protocol on serial interfaces are described in the *Vyatta High Availability Reference Guide*.

clear interfaces serial

Clears counters for serial interfaces

Syntax

clear interfaces serial [*wanx* **counters** {**all** | **physical** | **cisco-hdlc** | **frame-relay** | **ppp**}]

Command Mode

Operational mode.

Parameters

wanx	The identifier of a configured serial interface.
all	Clears all counters for the specified serial interface.
physical	Clears counters related to the physical line settings for the specified interface.
cisco-hdlc	Clears counters related to Cisco HDLC settings for the specified interface.
frame-relay	Clears counters related to Frame Relay settings for the specified interface.
ppp	Clears counters related to Point-to-Point Protocol settings for the specified interface.

Usage Guidelines

Use this command to clear statistics for a specified serial interface.

When used with no option, this command clears all counters on all serial interfaces. When a protocol or interface is specified, this command clears the counters for the specified protocol on the specified interface.

Examples

Example 1-8 shows the result of the **clear interfaces serial** command used with no options.

Example 1-8 “clear interfaces serial”

```
vyatta@R1> clear interfaces serial
```

```
Communication statistics flushed
Operational statistics flushed
DSU/CSU Perfomance Monitoring counters were flushed.
Performance monitoring counters flushed
PPP statistics flushed
Communication statistics flushed
Operational statistics flushed
DSU/CSU Perfomance Monitoring counters were flushed.
Performance monitoring counters flushed
PPP statistics flushed
vyatta@R1>
```

Example 1-9 shows the result of the **clear interfaces serial** command use with the **wan0 counters cisco-hdlc** options.

Example 1-9 “clear interfaces serial wan0 counters cisco-hdlc”: Displaying the result of the clear command.

```
vyatta@R1> clear interfaces serial wan0 counters cisco-hdlc
DSU/CSU Perfomance Monitoring counters were flushed.
Performance monitoring counters flushed
   ------------------------------------
      wan0.1: SLARP STATISTICS
   ------------------------------------
 SLARP frame transmission/reception statistics
       SLARP request packets transmitted:   0
          SLARP request packets received:   0
         SLARP Reply packets transmitted:   0
            SLARP Reply packets received:   0
     SLARP keepalive packets transmitted:   0
        SLARP keepalive packets received:   0
Incoming SLARP Packets with format errors
                     Invalid SLARP Code:   0
                Replies with bad IP addr:   0
                Replies with bad netmask:   0
SLARP timeout/retry statistics
                  SLARP Request timeouts:   0
            keepalive reception timeouts:   0
Cisco Discovery Protocol frames
                              Transmitted:   0
                                 Received:   0
DSU/CSU Perfomance Monitoring counters were flushed.
vyatta@R1>
```

interfaces serial <wanx>

Specifies basic serial interface configuration, including Layer 2 encapsulation characteristics.

Syntax

set interfaces serial *wanx*

delete interfaces serial *wanx*

show interfaces serial *wanx*

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.

Default

None.

Usage Guidelines

Use this command to configure a serial interface. You can define multiple serial interfaces by creating multiple **serial** configuration nodes.

Note that you cannot use **set** to change the name of the serial interface. To change the name of a serial interface, you must **delete** the old **serial** configuration node and create a new one.

Use the **set** form of this command to create a serial interface, provided the interface physically exists on your system. To see the interfaces available to the system kernel, use the **show interfaces system** command, which is described in the Vyatta Basic System Reference Guide.

Use the **delete** form of this command to remove all configuration for a serial interface.

Use the **show** form of this command to view a serial interface configuration.

interfaces serial <wanx> description <desc>

Specifies a description for a serial interface.

Syntax

set interfaces serial *wanx* **description** *desc*

delete interfaces serial *wanx* **description**

show interfaces serial *wanx* **description**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      description text
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
desc	Optional. A brief description for the serial interface. If the description contains spaces, it must be enclosed in double quotes. By default, the system auto-detects the card type and indicates it in the description.

Default

None.

Usage Guidelines

Use this command to specify a description for the serial interface.

Use the **set** form of this command to set the description for the serial interface.

Use the **delete** form of this command to remove description configuration.

Use the **show** form of this command to view description configuration.

interfaces serial <wanx> e1-options

Specifies the physical line characteristics for an E1 serial interface.

Syntax

set interfaces serial *wanx* **e1-options**

delete interfaces serial *wanx* **e1-options**

show interfaces serial *wanx* **e1-options**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      e1-options {
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.

Default

None.

Usage Guidelines

Use this command to specify the physical line characteristics of traffic that will pass through this E1 serial interface.

Configuring this option designates this interface as an E1 interface for transmitting signals in European digital transmission (E1) format. The E1 signal format carries information at a rate of 2.048 Mbps and can carry 32 channels of 64 Kbps each.

Currently, only high-density bipolar of order 3 (**hdb3**) line encoding is supported.

NOTE *On Sangoma cards with the PMC-Sierra chipset all ports must be configured with the same line type (e.g. all T1 or all E1).*

Use the **set** form of this command to specify the physical line characteristics for E1 serial interfaces.

Use the **delete** form of this command to remove E1 configuration.

Use the **show** form of this command to view E1 configuration.

interfaces serial <wanx> e1-options clock <type>

Sets the timing source for an E1 serial interface.

Syntax

set interfaces serial *wanx* **e1-options clock** *type*

delete interfaces serial *wanx* **e1-options clock**

show interfaces serial *wanx* **e1-options clock**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      e1-options {
         clock [internal|external]
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
type	Optional. Sets the timing source for the circuit. Supported values are as follows: **internal**: The interface will use the internal clock. **external**: The interface will use the external DTE Tx and Rx clock. The default is **external**.

Default

None.

Usage Guidelines

Use this command to specify the clock source for an E1 circuit.

Use the **set** form of this command to set the E1 clock source.

Use the **delete** form of this command to restore the default E1 clock source.

Use the **show** form of this command to view E1 clock source configuration.

interfaces serial <wanx> e1-options framing <type>

Sets the framing for an E1 serial interface.

Syntax

set interfaces serial *wanx* **e1-options framing** *type*

delete interfaces serial *wanx* **e1-options framing**

show interfaces serial *wanx* **e1-options framing**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      e1-options {
         framing [g704|g704-no-crc4|unframed]
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
type	Sets the frame type for the interface. Supported values are as follows: **g704**: Uses the G.704 framing specification and sets the E1 frame type to use CRC4. **g704-no-crc**: Uses the G.704 framing specification and sets the E1 frame type not to use CRC4. **unframed**: Configures full-rate (2048 kbps) unchannelized E1 bandwidth for the line. E1 unframed signaling options are available only on the Sangoma A104 line card. The default is **g704**.

Default

The framing is according to the G.704 specification with CRC.

Usage Guidelines

Use this command to specify the framing for an E1 circuit.

Use the **set** form of this command to set the framing.

Use the **delete** form of this command to restore the default framing.

Use the **show** form of this command to view framing configuration.

interfaces serial <wanx> e1-options mru <mru>

Specifies the Maximum Receive Unit (MRU) size for an E1 serial interface.

Syntax

set interfaces serial *wanx* **e1-options mru** *mru*

delete interfaces serial *wanx* **e1-options mru**

show interfaces serial *wanx* **e1-options mru**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      e1-options {
         mru 8-8188
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
mru	Optional. Sets the Maximum Receive Unit (MRU). This is the maximum packet size that the interface is willing to receive. The range is 8 to 8188. The default is 1500. Note that for IPv6 connections, the MRU must be at least 1280.

Default

The MRU is 1500.

Usage Guidelines

Use this command to specify the Maximum Receive Unit. This is the maximum packet size the interface is willing to receive.

Use the **set** form of this command to set the MRU.

Use the **delete** form of this command to restore the default MRU.

Use the **show** form of this command to view MRU configuration.

interfaces serial <wanx> e1-options mtu <mtu>

Specifies the Maximum Transmit Unit (MTU) size for an E1 serial interface.

Syntax

set interfaces serial *wanx* **e1-options mtu** *mtu*

delete interfaces serial *wanx* **e1-options mtu**

show interfaces serial *wanx* **e1-options mtu**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      e1-options {
         mtu 8-8188
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
mtu	Optional. Sets the Maximum Transfer Unit (MTU), in octets, for the interface as a whole. This will apply to all vifs defined for the interface. When forwarding, IPv4 packets larger than the MTU will be fragmented unless the DF bit is set. In that case, the packets will be dropped and an ICMP “Packet too big” message is returned to the sender. The range is 8 to 8188. If not set, fragmentation will never be performed.

Default

Fragmentation is not performed

Usage Guidelines

Use this command to specify the Maximum Transfer Unit (MTU). This is the maximum packet size the interface will send.

Use the **set** form of this command to set the MTU.

Use the **delete** form of this command to restore the default MTU behavior.

Use the **show** form of this command to view MTU configuration.

interfaces serial <wanx> e1-options timeslots

Defines timeslots for a 32-channel channelized E1 line.

Syntax

set interfaces serial *wanx* **e1-options timeslots** {**start** *start* | **stop** *stop*}

delete interfaces serial *wanx* **e1-options timeslots** [**start | stop**]

show interfaces serial *wanx* **e1-options timeslots** [**start | stop**]

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      e1-options {
         timeslots {
            start 1-32
            stop 1-32
         }
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
start *start*	The first timeslot in the range. The range of values is 1 to 32, where the value of **start** must be less than the value of **stop**. The default is 1.
stop *stop*	The last timeslot in the range. The range of values is 1 to 32, where the value of **start** must be less than the value of **stop**. The default is 32.

Default

The line is not channelized.

Usage Guidelines

Use this command to configure a fraction of a 32-channel channelized E1 line. To do this, you assign a range of timeslots to the line.

Use the **set** form of this command to define timeslots for the line.

Use the **delete** form of this command to remove channelization configuration.

Use the **show** form of this command to view channelization configuration.

interfaces serial <wanx> e3-options

Specifies the physical line characteristics for an E3 serial interface.

Syntax

set interfaces serial *wanx* **e3-options**

delete interfaces serial *wanx* **e3-options**

show interfaces serial *wanx* **e3-options**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      e3-options {
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.

Default

None.

Usage Guidelines

Use this command to specify the physical line characteristics for E3 serial interfaces.

Use the **set** form of this command to set the physical line characteristics.

Use the **delete** form of this command to remove physical line configuration.

Use the **show** form of this command to view physical line configuration.

interfaces serial <wanx> e3-options clock <type>

Specifies the timing source for an E3 serial interface.

Syntax

set interfaces serial *wanx* **e3-options clock** {**internal** | **external**}

delete interfaces serial *wanx* **e3-options clock**

show interfaces serial *wanx* **e3-options clock**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      e3-options {
         clock [internal|external]
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
type	Optional. Sets the timing source for the circuit. Supported values are as follows: **internal**: The interface will use the internal clock. **external**: The interface will use the external DTE Tx and Rx clock. The default is external.

Default

The interface uses the external DTE Tx and Rx clock.

Usage Guidelines

Use this command to specify the timing source for the circuit.

Use the **set** form of this command to set the timing source.

Use the **delete** form of this command to restore the default timing source.

Use the **show** form of this command to view timing source configuration.

interfaces serial <wanx> e3-options framing <type>

Specifies the framing type for an E3 serial interface.

Syntax

set interfaces serial *wanx* **e3-options framing** {**g751** | **g832** | **unframed**}

delete interfaces serial *wanx* **e3-options framing**

show interfaces serial *wanx* **e3-options framing**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      e3-options {
         framing [g751|g832|unframed]
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
type	Optional. Sets the frame type for the interface. Supported values are as follows: **g751**: Sets the E3 frame type to be G.751-compliant. **g832**: Sets the E3 frame type to be G.832-compliant. **unframed**: Configures full-rate (34368 kbps) unchannelized E3 bandwidth for the line.

Default

The frame type is G.751-compliant.

Usage Guidelines

Use this command to specify the framing type for an E3 interface.

Use the **set** form of this command to set the framing type.

Use the **delete** form of this command to restore the default E3 framing.

Use the **show** form of this command to view E3 framing configuration.

interfaces serial <wanx> e3-options line-coding <type>

Specifies the line coding for an E3 serial interface.

Syntax

set interfaces serial *wanx* **e3-options line-coding** {**hdb3** | **ami**}

delete interfaces serial *wanx* **e3-options line-coding**

show interfaces serial *wanx* **e3-options line-coding**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      e3-options {
         line-coding [hdb3|ami]
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
type	Optional. Sets the line coding for the interface. Supported values are as follows: **hdb3**: Sets the E3 line coding to be HDB3 (High Density Bipolar of order 3) compliant. **ami**: Sets the E3 line coding to be AMI (Alternate Mask Inversion) compliant.

Default

HDB3 line coding is used.

Usage Guidelines

Use this command to specify the line coding type for the interface.

Use the **set** form of this command to set the line coding.

Use the **delete** form of this command to restore the default line coding.

Use the **show** form of this command to view line coding configuration.

interfaces serial <wanx> encapsulation <type>

Sets the encapsulation type for a serial interface.

Syntax

set interfaces serial *wanx* **encapsulation** *type*

delete interfaces serial *wanx* **encapsulation**

show interfaces serial *wanx* **encapsulation**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      encapsulation [ppp|cisco-hdlc|frame-relay]
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
type	Mandatory. Sets the encapsulation type for the interface. Supported values are as follows: **ppp**: Uses Point-to-Point Protocol (PPP) encapsulation for the interface. **cisco-hdlc**: Uses Cisco High-Level Data Link Control (Cisco HDLC) encapsulation on the interface. **frame-relay**: Uses Frame Relay encapsulation on the interface.

Default

None.

Usage Guidelines

Use this command to specify the encapsulation type for a serial interface.

Use the **set** form of this command to set the encapsulation type.

Use the **delete** form of this command to remove encapsulation type configuration.

Use the **show** form of this command to view encapsulation type configuration.

NOTE *Commands for configuring Cisco HDLC, Frame Relay, and Point-to-Point Protocol encapsulation are described in Vyatta Encapsulation and Tunnels Reference Guide.*

interfaces serial <wanx> synch-options

Specifies the physical line characteristics for synchronous serial interfaces.

Syntax

set interfaces serial *wanx* **synch-options**

delete interfaces serial *wanx* **synch-options**

show interfaces serial *wanx* **synch-options**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      synch-options {
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.

Default

None.

Usage Guidelines

Use this command to specify the physical line characteristics of traffic that will pass through a synchronous serial interface. Synchronous serial cards supported include the Sangoma A142 and A144. These cards interface to an external CSU/DSU.

Note that the **synch-options** commands will not work with cards that have an integrated CSU/DSU (e.g. Sangoma A101, A102, A104, A108, and A301). Similarly, the **t1-options**, **t3-options**, **e1-options**, and **e3-options** command will not work with cards that do not have an integrated CSU/DSU (e.g. Sangoma A142, and A144).

Use the **set** form of this command to set the physical line characteristics for a synchronous serial interfaces.

Use the **delete** form of this command to remove synchronous serial physical line configuration.

Use the **show** form of this command to view synchronous serail physical line configuration.

interfaces serial <wanx> synch-options baud-rate <rate>

Sets the bit rate for an internally clocked synchronous serial interface.

Syntax

set interfaces serial *wanx* **synch-options baud-rate** *rate*

delete interfaces serial *wanx* **synch-options baud-rate**

show interfaces serial *wanx* **synch-options baud-rate**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      synch-options {
         baud-rate 1-8000000
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
baud-rate	Optional. Sets the bit rate in bits per second for the circuit when the **clock** type is “internal”. It is ignored if the **clock** is set to “external”. The range is **1** to **8000000**. The default is **1546000** (i.e. T1).

Default

The default is **1546000**.

Usage Guidelines

Use this command to specify the bit rate for an internally clocked synchronous serial interface.

Use the **set** form of this command to set the bit rate.

Use the **delete** form of this command to restore the default bit rate.

Use the **show** form of this command to view bit rate configuration.

interfaces serial <wanx> synch-options clock <type>

Sets the timing source for a synchronous serial interface.

Syntax

set interfaces serial *wanx* **synch-options clock** *type*

delete interfaces serial *wanx* **synch-options clock**

show interfaces serial *wanx* **synch-options clock**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      synch-options {
         clock [internal|external]
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
type	Optional. Sets the timing source for the circuit. Supported values are as follows: **internal**: The interface will use the internal clock. The line speed will be determined by the **baud-rate** parameter. **external**: The interface will use the external DTE Tx and Rx clock. The default is **external**.

Default

The interface uses the external DTE Tx and Rx clock.

Usage Guidelines

Use this command to specify the clock source for a synchronous serial interface.

Use the **set** form of this command to set the clock source.

Use the **delete** form of this command to restore the default clock source.

Use the **show** form of this command to view clock source configuration.

interfaces serial <wanx> synch-options connection <type>

Sets the connection type for a synchronous serial interface.

Syntax

set interfaces serial *wanx* **synch-options connection** *type*

delete interfaces serial *wanx* **synch-options connection**

show interfaces serial *wanx* **synch-options connection**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      synch-options {
         connection [permanent|switched]
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
type	Optional. Sets the connection type for the circuit. Supported values are as follows: **permanent**: The interface supports (non-IP) permanent virtual circuits. **switched**: The interface supports (non-IP) switched virtual circuits. The default is **permanent;** this should be used for IP networks.

Default

The interface defaults to **permanent** and should not be changed.

Usage Guidelines

Use this command to specify the connection type for a synchronous serial interface. Note that this setting should not be changed from the default value for IP networks.

Use the **set** form of this command to set the connection type.

Use the **delete** form of this command to restore the default connection type.

Use the **show** form of this command to view connection type configuration.

interfaces serial <wanx> synch-options line-coding <type>

Sets the line coding standard for a synchronous serial interface.

Syntax

set interfaces serial *wanx* **synch-options line-coding** *type*

delete interfaces serial *wanx* **synch-options line-coding**

show interfaces serial *wanx* **synch-options line-coding**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      synch-options {
         line-coding [NRZ|NRZI]
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
type	Optional. Sets the line coding for the circuit. Supported values are as follows: **NRZ**: The interface will use No Return to Zero (NRZ) line coding. **NRZI**: The interface will use No Return to Zero Inverted (NRZI) line coding. The default is **NRZ**.

Default

The interface uses **NRZ** line coding by default.

Usage Guidelines

Use this command to specify the line coding for a synchronous serial interface.

Use the **set** form of this command to set the line coding.

Use the **delete** form of this command to restore the default line coding.

Use the **show** form of this command to view line coding configuration.

interfaces serial <wanx> synch-options line-idle <type>

Sets the idle line signalling for a synchronous serial interface.

Syntax

set interfaces serial *wanx* **synch-options line-idle** *type*

delete interfaces serial *wanx* **synch-options line-idle**

show interfaces serial *wanx* **synch-options line-idle**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      synch-options {
         line-idle [flag|mark]
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
type	Optional. Sets the idle line signalling for the circuit. Supported values are as follows: **flag**: An idle line should be signalled as flag (logical 0). **mark**: An idle line should be signalled as mark (logical 1). The default is **flag**.

Default

The interface uses **flag** signalling to signify an idle line by default.

Usage Guidelines

Use this command to specify the idle line signalling for a synchronous serial interface.

Use the **set** form of this command to set the idle line signalling.

Use the **delete** form of this command to restore the default idle line signalling.

Use the **show** form of this command to view idle line signalling configuration.

interfaces serial <wanx> t1-options

Specifies the physical line characteristics for T1 serial interfaces.

Syntax

set interfaces serial *wanx* **t1-options**

delete interfaces serial *wanx* **t1-options**

show interfaces serial *wanx* **t1-options**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      t1-options
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.

Default

None.

Usage Guidelines

Use this command to specify the physical line characteristics of traffic that will pass through a T1 serial interface.

Configuring this option designates this interface as a T1 interface for transmitting digital signals in the T-carrier system used in the United States, Japan, and Canada. The T1 signal format carries 24 pulse code modulation (PCM) signals using time-division multiplexing (TDM) at an overall rate of 1.544 Mbps.

Currently, only bipolar 8-zero line coding is supported.

NOTE *On Sangoma cards with the PMC-Sierra chipset all ports must be configured with the same line type (e.g. all T1 or all E1).*

Use the **set** form of this command to set the physical line characteristics for a T1 serial interfaces.

Use the **delete** form of this command to remove T1 physical line configuration.

Use the **show** form of this command to view T1 physical line configuration.

interfaces serial <wanx> t1-options clock <type>

Sets the timing source for a T1 serial interface.

Syntax

set interfaces serial *wanx* **t1-options clock** *type*

delete interfaces serial *wanx* **t1-options clock**

show interfaces serial *wanx* **t1-options clock**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      t1-options {
         clock [internal|external]
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
type	Optional. Sets the timing source for the circuit. Supported values are as follows: **internal**: The interface will use the internal clock. **external**: The interface will use the external DTE Tx and Rx clock. The default is **external**.

Default

The interface uses the external DTE Tx and Rx clock.

Usage Guidelines

Use this command to specify the clock source for a T1 serial interface.

Use the **set** form of this command to set the T1 clock source.

Use the **delete** form of this command to restore the default T1 clock source.

Use the **show** form of this command to view T1 clock source configuration.

interfaces serial <wanx> t1-options lbo <range>

Specifies the line build-out (LBO) range for a T1 serial interface.

Syntax

set interfaces serial *wanx* **t1-options lbo** *range*

delete interfaces serial *wanx* **t1-options lbo**

show interfaces serial *wanx* **t1-options lbo**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      t1-options {
         lbo [0-110ft|110-220ft|220-330ft|330-440ft|440-550ft]
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
range	Sets the maximum line build-out length. Supported values are as follows: **0–110ft**: The line will not exceed 110 feet in length. **110–220ft**: The line will be between 110 and 220 feet in length. **220–330ft**: The line will be between 220 and 330 feet in length. **330–440ft**: The line will be between 330 and 440 feet in length. **440–550ft**: The line will be between 440 and 550 feet in length. The default is **0-110ft**.

Default

The line build-out length is 0 to 110 feet.

Usage Guidelines

Use this command to specify the line build-out (LBO) range for the T1 line.

Use the **set** form of this command to set the LBO.

Use the **delete** form of this command to restore the default LBO.

Use the **show** form of this command to view LBO configuration.

interfaces serial <wanx> t1-options mru <mru>

Specify the Maximum Receive Unit (MRU) size for a T1 serial interface.

Syntax

set interfaces serial *wanx* **t1-options mru** *mru*

delete interfaces serial *wanx* **t1-options mru**

show interfaces serial *wanx* **t1-options mru**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      t1-options {
         mru 8-8188
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
mru	Optional. Sets the Maximum Receive Unit (MRU). This is the maximum packet size that the interface is willing to receive. The range is 8 to 8188. The default is 1500. Note that for IPv6 connections, the MRU must be at least 1280.

Default

The default is 1500.

Usage Guidelines

Use this command to specify the MRU for a T1 serial interface. This is the maximum packet size the interface is willing to receive.

Use the **set** form of this command to set the MRU.

Use the **delete** form of this command to remove MRU configuration.

Use the **show** form of this command to view MRU configuration.

interfaces serial <wanx> t1-options mtu <mtu>

Specify the Maximum Transmit Unit (MTU) size for a T1 serial interface.

Syntax

set interfaces serial *wanx* **t1-options mtu** *mtu*

delete interfaces serial *wanx* **t1-options mtu**

show interfaces serial *wanx* **t1-options mtu**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      t1-options {
         mtu 8-8188
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
mtu	Optional. Sets the Maximum Transfer Unit (MTU), in octets, for the interface as a whole. This will apply to all vifs defined for the interface. When forwarding, IPv4 packets larger than the MTU will be fragmented unless the DF bit is set. In that case, the packets will be dropped and an ICMP “Packet too big” message is returned to the sender. The range is 8 to 8188. If not set, fragmentation is never performed.

Default

Fragmentation is never performed.

Usage Guidelines

Use this command to specify the Maximum Transmit Unit (MTU) for a T1 serial interface. This is the maximum packet size the interface will send.

Use the **set** form of this command to set the MTU.

Use the **delete** form of this command to remove MTU configuration.

Use the **show** form of this command to view MTU configuration.

interfaces serial <wanx> t1-options timeslots

Defines timeslots for a 24-channel channelized T1 line.

Syntax

set interfaces serial *wanx* **t1-options timeslots** {**start** *start* | **stop** *stop*}

delete interfaces serial *wanx* **t1-options timeslots** [**start | stop**]

show interfaces serial *wanx* **t1-options timeslots** [**start | stop**]

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      t1-options {
         timeslots {
            start u32
            stop u32
         }
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
start *start*	The first timeslot in the range. The range of values is 1 to 32, where the value of **start** must be less than the value of **stop**. The default is 1.
stop *stop*	The last timeslot in the range. The range of values is 1 to 32, where the value of **start** must be less than the value of **stop**. The default is 32.

Default

T1 lines are not channelized.

Usage Guidelines

Use this command to configure channelization on a 24-channel T1 line. To do this, you assign a range of timeslots to the line.

Use the **set** form of this command to define timeslots for the line.

Use the **delete** form of this command to remove channelization configuration.

Use the **show** form of this command to view channelization configuration.

interfaces serial <wanx> t3-options

Specifies the physical line characteristics for a T3 serial interface.

Syntax

set interfaces serial *wanx* **t3-options**

delete interfaces serial *wanx* **t3-options**

show interfaces serial *wanx* **t3-options**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      t3-options {
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.

Default

None.

Usage Guidelines

Use this command to specify the physical line characteristics of traffic that will pass through this T3 serial interface.

Configuring this option designates this interface as a T3 interface for transmitting digital signals in the T-carrier system used in the United States, Japan, and Canada. The T3 signal format carries multiple T1 channels multiplexed, resulting in transmission rates of up to 44.736 Mbit/s.

Use the **set** form of this command to specify the physical line characteristics for the T3 interface.

Use the **delete** form of this command to remove T1 physical line configuration.

Use the **show** form of this command to view T1 physical line configuration.

interfaces serial <wanx> t3-options clock <type>

Specifies the timing source for the circuit.

Syntax

set interfaces serial *wanx* **t3-options clock** *type*

delete interfaces serial *wanx* **t3-options clock**

show interfaces serial *wanx* **t3-options clock**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      t3-options {
         clock [internal | external]
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
type	Optional. Sets the timing source for the circuit. Supported values are as follows: **internal**: The interface will use the internal clock. **external**: The interface will use the external DTE Tx and Rx clock. The default is **external**.

Default

The interface uses the external DTE Tx and Rx clock.

Usage Guidelines

Use this command to specify the timing source for a T3 serial interface.

Use the **set** form of this command to set the clock source.

Use the **delete** form of this command to remove clock source configuration.

Use the **show** form of this command to view clock source configuration.

interfaces serial <wanx> t3-options framing <type>

Specifies the framing type for a T3 serial interface.

Syntax

set interfaces serial *wanx* **t3-options framing** *type*

delete interfaces serial *wanx* **t3-options framing**

show interfaces serial *wanx* **t3-options framing**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      t3-options {
         framing [c-bit|ml3]
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
type	Optional. Sets the frame type for the interface. Supported values are as follows: **c-bit**: Sets the T3 frame type to C-bit parity **m13**: Sets the T3 frame type to M13. The default is **c-bit**.

Default

T3 interfaces use C-bit parity framing.

Usage Guidelines

Use this command to specify the framing type for a T3 serial interface.

Use the **set** form of this command to set the framing type.

Use the **delete** form of this command to remove framing type configuration.

Use the **show** form of this command to view framing type configuration.

interfaces serial <wanx> t3-options line-coding <type>

Specifies the line coding for a T3 serial interface.

Syntax

set interfaces serial *wanx* **t3-options line-coding** *type*

delete interfaces serial *wanx* **t3-options line-coding**

show interfaces serial *wanx* **t3-options line-coding**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   serial wan0..wan23 {
      t3-options {
         line-coding [ami|b3zs]
      }
   }
}
```

Parameters

wanx	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be **wan0** to **wan23**, depending on what serial interfaces that are actually available on the system.
type	Optional. Sets the T3 line coding. Supported values are as follows: **ami**: Sets the line coding to alternate mark inversion (AMI). **b3zs**: Sets the line coding to bipolar 3-zero substitution. The default is **b3zs**.

Default

T3 serial lines use bipolar 3-zero substitution line coding.

Usage Guidelines

Use this command to specify the line coding type for a T3 serial interface.

Use the **set** form of this command to set the line coding type.

Use the **delete** form of this command to remove line coding type configuration.

Use the **show** form of this command to view line coding type configuration.

loopback down

Deactivates loopbacks on a Sangoma T1/E1 or T3/E3 card.

Syntax

loopback down *wanx* {**all | analog | framer | line | local | payload**}

Command Mode

Operational mode.

Parameters

wanx	The name of a serial interface.
all	Attempts to deactivate all of the loopbacks supported by the card, whether or not they are active at the moment. Since de-activating each loopback may take several seconds, this command may take on the order of 15 seconds to complete.
analog	Deactivates the Line Interface Unit analog loopback.
framer	Deactivates the framer loopback.
line	Deactivates the Line Interface Unit line loopback.
local	Deactivates the Line Interface Unit local loopback.
payload	Deactivates the payload loopback.

Default

None.

Usage Guidelines

Use this command to deactivate individual loopbacks, or all loopbacks, on a Sangoma T1/E1 or T3/E3 card.

These commands may indicate a successful outcome (e.g. “loopback XXX deactivated”) even if the loopback in question was not active when the command was run. Also, each command may take several seconds to complete.

Like the “loopback up” command, the CLI completion for this command only displays the loopbacks supported by the chipset on the card used. The “all” option is supported by all chipsets.

Examples

Example 1-10 shows the first screen of output for **loopback down wanx local**.

Example 1-10 “loopback down wan0 local”: De-activating a local loopback on wan0

```
vyatta@R1> loopback down wan0 local

Diagnostic Digital Loopback mode is deactivated!

vyatta@R1>
```

loopback test

Starts a loopback diagnostic test on a Sangoma T1/E1 or T3/E3 card.

Syntax

loopback test *wanx*

Command Mode

Operational mode.

Parameters

wanx	The name of a serial interface.

Default

None.

Usage Guidelines

Use this command to send data traffic on the line and check to see if the data is returned. The test succeeds if the data sent on the line is returned. The test will not run unless the line is up.

This test is usually performed to verify “continuity” from the local WAN interface, out to the point where the circuit is looped back, and back to the WAN interface. The test is typically performed on a line that is looped at the far end of the circuit, though should succeed if one of the system-facing loopbacks on the local WAN interface is active.

Examples

Example 1-12 shows the first screen of output (successful) for **loopback test wanx**.

Example 1-11 “loopback test wan0”: Test a loopback on wan0 - successful

```
vyatta@R1> loopback test wan0

Starting Loop Test (press ctrl-c to exit)!

 Sep 10 04:34:39 | Test 0001 | Successful (Ok)!
 Sep 10 04:34:39 | Test 0002 | Successful (Ok)!
```

```
 Sep 10 04:34:40 | Test 0003 | Successful (Ok)!
 Sep 10 04:34:40 | Test 0004 | Successful (Ok)!
 Sep 10 04:34:41 | Test 0005 | Successful (Ok)!
 Sep 10 04:34:41 | Test 0006 | Successful (Ok)!
 Sep 10 04:34:42 | Test 0007 | Successful (Ok)!
vyatta@R1>
```

Example 1-12 shows the first screen of output (unsuccessful) for **loopback test wanx**.

Example 1-12 “loopback test wan0”: Test a loopback on wan0 - unsuccessful

```
vyatta@R1> loopback test wan0

Starting Loop Test (press ctrl-c to exit)!

 Sep 10 04:36:09 | Test 0001 | Timeout!
 Sep 10 04:36:09 | Test 0002 | Timeout!
 Sep 10 04:36:10 | Test 0003 | Timeout!
 Sep 10 04:36:10 | Test 0004 | Timeout!
 Sep 10 04:36:11 | Test 0005 | Timeout!
vyatta@R1>
```

loopback up

Activates a loopback on a Sangoma T1/E1 or T3/E3 card.

Syntax

loopback up *wanx* {**analog** | **framer** | **line** | **local** | **payload**}

Command Mode

Operational mode.

Parameters

wanx	The name of a serial interface.
analog	Activate the Line Interface Unit analog loopback. This loops data back to the system at the analog side of the Line Interface Unit within the Sangoma card. This parameter is available on Sangoma cards with Maxim and Exar chipsets. It is not available on cards with PMC-Sierra chipsets.
framer	Activate the framer loopback. This loops data back to the system at the framer subsystem within the Sangoma card. This parameter is available only on Sangoma cards with Maxim chipsets. It is not available on cards with PMC-Sierra or Exar chipsets.
line	Activate the Line Interface Unit line loopback. This loops data back to the line at the digital side of the Line Interface Unit. This parameter is available on Sangoma cards with Maxim, PMC-Sierra, and Exar chipsets.
local	Activate the Line Interface Unit local loopback. This loops data back to the system at the digital side of the Line Interface Unit. This parameter is available on Sangoma cards with Maxim, PMC-Sierra, and Exar chipsets.
payload	Activate the payload loopback. This loops data back to the line at the framer subsystem. This parameter is available only on Sangoma cards with Maxim and PMC-Sierra chipsets. It is not available on cards with Exar chipsets.

Default

None.

Usage Guidelines

Use this command to activate a loopback on a Sangoma T1/E1 or T3/E3 card. Once activated, a loopback will remain active until it is taken down with a “loopback down” command, or when the interface is reconfigured, or when the system is rebooted. The status (up or down) of loopbacks activated with this command can be displayed using the “physical” option of the **show interfaces serial** command (see page 78).

NOTE *More than one loopback can be active at the same time.*

Examples

Example 1-13 shows the first screen of output for **loopback up wanx local**.

Example 1-13 “loopback up wan0 local”: Activating a local loopback on wan0

```
vyatta@R1> loopback up wan0 local

Diagnostic Digital Loopback mode is activated!

vyatta@R1>
```

show interfaces serial

Displays serial interface information.

Syntax

show interfaces serial [*wanx* {**cisco-hdlc** | **loopback** | **frame-relay** [**pvc** | **pvc-list** [**active**]] | **physical** | **ppp** | **trace**}]

Command Mode

Operational mode.

Parameters

wanx	The name of a serial interface. If an interface is specified, you must also specify one of the **cisco-hdlc**, **frame-relay**, **physical**, **ppp,** or **trace** options.
cisco-hdlc	Shows Cisco HDLC information for the specified serial interface.
frame-relay	Shows Frame Relay information for the specified serial interface.
loopback	Shows loopback information for the specified serial interface.
pvc	Displays details for Frame Relay PVCs.
pvc-list	Lists of Frame Relay permanent virtual circuits (PVCs). When used with no option, displays all configured PVCs.
active	Lists only active Frame Relay PVCs.
physical	Shows physical device information for the specified serial interface.
ppp	Shows Point-to-Point protocol information for the specified serial interface.
trace	Outputs a trace of the raw frames incoming from, and outgoing to, the specified interface. This trace continues until <Ctrl>-c is pressed.

Default

Information is shown for all available serial interfaces.

Usage Guidelines

Use this command to view the operational status of a serial interface.

Examples

Example 1-14 shows the first screen of output for **show interfaces serial**.

Example 1-14 “show interfaces serial”: Displaying serial interface information

```
vyatta@R1> show interfaces serial
wan1: <POINTOPOINT,NOARP,UP,10000> mtu 1450 qdisc pfifo_fast
qlen 100
    link/ppp
RX:  bytes    packets     errors    dropped    overrun      mcast
       773         67          0          0          0          0
TX:  bytes    packets     errors    dropped    carrier collisions
       813         68          0          0          0          0

wan1.1: <POINTOPOINT,MULTICAST,NOARP,UP,10000> mtu 1340 qdisc
pfifo_fast qlen 3
    link/ppp
    inet 2.2.2.2 peer 1.1.1.1/32 scope global wan1.1
RX:  bytes    packets     errors    dropped    overrun      mcast
        72          5          0          0          0          0
TX:  bytes    packets     errors    dropped    carrier collisions
        78          5          0          0          0          0

wan0: <POINTOPOINT,NOARP,UP,10000> mtu 1450 qdisc pfifo_fast
qlen 100
    link/ppp
RX:  bytes    packets     errors    dropped    overrun      mcast
       813         68          0          0          0          0
TX:  bytes    packets     errors    dropped    carrier collisions
       773         67          0          0          0          0

wan0.1: <POINTOPOINT,MULTICAST,NOARP,UP,10000> mtu 1350 qdisc
pfifo_fast qlen 3
    link/ppp
    inet 1.1.1.1 peer 2.2.2.2/32 scope global wan0.1
RX:  bytes    packets     errors    dropped    overrun      mcast
        78          5          0          0          0          0
TX:  bytes    packets     errors    dropped    carrier collisions
        72          5          0          0          0          0
```

Example 1-15 shows the output for **show interfaces serial** *wanx* **ppp**.

Example 1-15 "show interfaces serial *wanx* ppp"

```
vyatta@ppp> show interfaces serial wan0 ppp
   ---------------------------------
      wan0: ROUTER UP TIME
   ---------------------------------
    Router UP Time:  14 minute(s),  6 seconds

PPP data:
IN.BYTES  :          0
IN.PACK   :          0
IN.VJCOMP :          0
IN.VJUNC  :          0
IN.VJERR  :          0
OUT.BYTES :          0
OUT.PACK  :          0
OUT.VJCOMP:          0
OUT.VJUNC :          0
OUT.NON-VJ:          0
```

Example 1-16 shows the output for **show interfaces serial** *wanx* **trace**.

NOTE *The output can be interrupted by pressing Ctrl-C*

Example 1-16 "show interfaces serial *wanx* trace"

```
vyatta@ppp> show interfaces serial wan0 trace
OUTGOING       Len=14  TimeStamp=56407   Aug 22 06:31:49 314767
[1/100s]
Raw (HEX)       00 01 03 08 00 75 95 01 01 01 03 02 A7 00

FR decode       DLCI=0 C/R=0 EA=0 FECN=0 BECN=0 DE=0 EA=1
                Signalling ANSI
                Link Verification Req   Sx=0xA7  Rx=0x00

OUTGOING       Len=13  TimeStamp=56407   Aug 22 06:31:49 314779
[1/100s]
Raw (HEX)       FC F1 03 09 00 75 01 01 01 03 02 A7 00

FR decode       DLCI=1023 C/R=0 EA=0 FECN=0 BECN=0 DE=0 EA=1
                Signalling ANSI
                Link Verification Req   Sx=0x00  Rx=0x01

OUTGOING       Len=13  TimeStamp=56408   Aug 22 06:31:49 315013
[1/100s]
```

```
Raw (HEX)       00 01 03 08 00 75 51 01 01 53 02 A7 00

FR decode       DLCI=0 C/R=0 EA=0 FECN=0 BECN=0 DE=0 EA=1
                Signalling ANSI
                Link Verification Req   Sx=0x00  Rx=0x00

INCOMING    Len=14  TimeStamp=56560   Aug 22 06:31:49 467620
[1/100s]
Raw (HEX)       00 01 03 08 00 75 95 01 01 01 03 02 A7 00

FR decode       DLCI=0 C/R=0 EA=0 FECN=0 BECN=0 DE=0 EA=1
                Signalling ANSI
                Link Verification Req   Sx=0xA7  Rx=0x00

INCOMING       Len=13  TimeStamp=56561   Aug 22 06:31:49 467999
[1/100s]

FR decode       DLCI=1023 C/R=0 EA=0 FECN=0 BECN=0 DE=0 EA=1
                Signalling ANSI
                Link Verification Req   Sx=0x00  Rx=0x00

INCOMING       Len=13  TimeStamp=56561   Aug 22 06:31:49 468379
[1/100s]
Raw (HEX)       00 01 03 08 00 75 51 01 01 53 02 A7 00

FR decode       DLCI=0 C/R=0 EA=0 FECN=0 BECN=0 DE=0 EA=1
                Signalling ANSI
                Link Verification Req   Sx=0x00  Rx=0x00
```

Chapter 2: DSL Interfaces

This chapter explains how to use Digital Subscriber Line (DSL) interfaces on the Vyatta system. Currently the Vyatta system supports Asymmetrical DSL (ADSL) interfaces only.

This chapter presents the following topics:

- DSL Configuration
- DSL Commands

DSL Configuration

This section presents the following topics:

- ADSL Interfaces Overview
- ADSL Configuration Example

ADSL Interfaces Overview

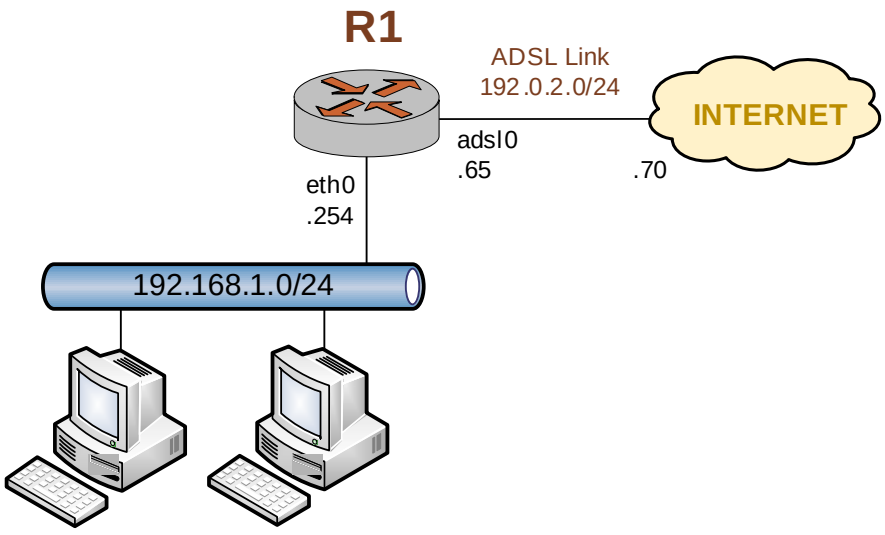
DSL (Digital Subscriber Loop) is a service that utilizes intelligent modulation techniques to convert traditional phone lines—Plain Old Telephone System (POTS) or Integrated Services Digital Network (ISDN)—into high-bandwidth conduits used for Internet access. Like dial-up, cable, wireless, and T1/E1, DSL is an access methodology rather than an end-to-end solution by itself.

Asymmetric DSL (ADSL) is a technology that provides higher bandwidth in one direction (typically downstream) than in the other.

ADSL Configuration Example

Figure 2-1 shows a typical ADSL configuration as an access protocol between a customer premesis and an Internet Service Provider. In this example, the ADSL interface is configured using Point-to-Point Protocol over Ethernet (PPPoE). PPPoE links typically include authentication, so a user ID and password are configured in this example.

Figure 2-1 Typical ADSL network configuration



With PPPoE encapsulation the local and remote IP addresses can be automatically negotiated instead of explicitly specified. This is the default: auto-negotiation is performed automatically if the addresses are not specified.

PPPoE encapsulation also allows for “on-demand” connection, in which the interface establishes the PPPoE connection when traffic is sent. On-demand connection is enabled using the **connect-on-demand** option.

Example 2-1 sets up a PPPoE encapsulation on interface adsl0. In this example:

- A Sangoma S518 ADSL NIC is connected to the interface.
- The interface has one PVC. The PVC identifier is automatically detected.
- The PPPoE unit number is 0.
- The local IP address is 192.0.2.65 (prefix-length 24). This is in the public IP range, since this interface will connect over the wide-area network.
- The IP address of the far end is 192.0.2.70. This is on the same network (prefix-length 24) as this interface.
- The user id is set to “customerA”.
- The password is set to “Aremotsuc”.

Tip: *Where public IP addresses would normally be used, the example uses RFC 3330 “TEST-NET” IP addresses (192.0.2.0/24)*

To create and configure this ADSL interface, perform the following steps in configuration mode:

Example 2-1 Creating and configuring an ADSL interface for PPPoE encapsulation

Step	Command
Specify that the system should auto-detect an identifier for the pvc.	`vyatta@R1# set interfaces adsl adsl0 pvc auto` `[edit]`
Set the line encapsulation to PPPoE using unit number 0.	`vyatta@R1# set interfaces adsl adsl0 pvc auto pppoe 0` `[edit]`
Assign the local IP address to the interface.	`vyatta@R1# set interfaces adsl adsl0 pvc auto pppoe 0 local-address 192.0.2.65` `[edit]`
Set the network mask (prefix length) for the interface.	`vyatta@R1# set interfaces adsl adsl0 pvc auto pppoe 0 prefix-length 24` `[edit]`
Set the IP address of the far end of the connection.	`vyatta@R1# set interfaces adsl adsl0 pvc auto pppoe 0 remote-address 192.0.2.70` `[edit]`

Example 2-1 Creating and configuring an ADSL interface for PPPoE encapsulation

Set the user id for the link.	`vyatta@R1# set interfaces adsl adsl0 pvc auto pppoe 0 user-id customerA` `[edit]`
Set the password for the link.	`vyatta@R1# set interfaces adsl adsl0 pvc auto pppoe 0 password Aremotsuc` `[edit]`
Commit the configuration.	`vyatta@R1# commit` `[edit]`

View the configuration.	`vyatta@R1# show interfaces adsl adsl0`
	`  pvc auto {`
	`  pppoe 0 {`
	`  local-address 192.0.2.65`
	`  prefix-length 24`
	`  remote-address 192.0.2.70`
	`  user-id customerA`
	`  password Aremotsuc`
	`  }`
	`  }`
	`vyatta@R1#`

DSL Commands

This chapter contains the following commands.

Configuration Commands	
DSL Interface Global Configuration	
interfaces adsl <adslx>	Defines an ADSL interface.
interfaces adsl <adslx> pvc <pvc-id>	Defines a permanent virtual circuit (PVC) on an ADSL interface.
interfaces adsl <adslx> watchdog <state>	Enables or disables the ADSL watchdog feature on the link.
Operational Commands	
show interfaces adsl <if-name>	Displays status of an ADSL interface.
show interfaces adsl <if-name> capture	Starts a low-level packet trace on an ADSL interface.
show interfaces adsl <if-name> queue	Displays queue information on an ADSL interface.
show interfaces adsl <if-name> status	Displays detailed status of an ADSL interface.

Commands for using other system features with DSL interfaces can be found in the following locations.

Related Commands Documented Elsewhere	
Firewall	Commands for configuring firewall on DSL interfaces are described in the *Vyatta Security Reference Guide*.
PPPoE	Commands for configuring Point-to-Point Protocol over Ethernet on DSL interfaces are described in the *Vyatta Encapsulation and Tunnels Reference Guide*.
OSPF	Commands for configuring the Open Shortest Path First routing protocol on DSL interfaces are described in the *Vyatta OSPF Reference Guide*.
RIP	Commands for configuring the Routing Information Protocol on DSL interfaces are described in the *Vyatta RIP Reference Guide*.
QoS	Commands for configuring quality of service on DSL interfaces are described in the *Vyatta Policy and QoS Reference Guide*.
System interfaces	Commands for showing the physical interfaces available on your system are described in the Vyatta Basic System Reference Guide.

VRRP	Commands for configuring Virtual Router Redundancy Protocol on DSL interfaces are described in the *Vyatta High Availability Reference Guide*.

interfaces adsl <adslx>

Defines an ADSL interface.

Syntax

set interfaces adsl *adslx*

delete interfaces adsl *adslx*

show interfaces adsl *adslx*

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   adsl adslx {
   }
}
```

Parameters

adslx	Mandatory. Multi-node. The identifier for the ADSL interface you are defining. This may be **adsl0** to **adsl***x*, depending on what physical ADSL ports are actually available on the system. The system automatically creates as many ADSL interface configuration nodes as there are physical ADSL ports on your system.

Default

Configuration nodes are created for all available physical ADSL ports on startup.

Usage Guidelines

Use this command to configure an ADSL interface.

You can use the **set** form of this command to create an ADSL interface, provided the interface physically exists on your system. However, the system automatically creates a configuration node for each system port, so you should not need to use the **set** form of this command to create an ADSL interface unless you have deleted it.

To see the interfaces available to the system kernel, use **show interfaces system** command.

Use the **delete** form of this command to remove all configuration for an ADSL interface. The system will create an empty configuration node for the interface the next time the system starts.

Use the **show** form of this command to view ADSL interface configuration.

interfaces adsl <adslx> pvc <pvc-id>

Defines a permanent virtual circuit (PVC) on an ADSL interface.

Syntax

set interfaces adsl *adslx* **pvc** *pvc-id*

delete interfaces adsl *adslx* **pvc** *pvc-id*

show interfaces adsl *adslx* **pvc** *pvc-id*

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   adsl adslx {
      pvc [0-255/0-65535|auto] {
      }
   }
}
```

Parameters

adslx	Mandatory. Multi-node. The identifier for the ADSL interface you are defining. This may be **adsl0** to **adsl***x*, depending on what physical ADSL ports are actually available on the system.
pvc-id	Mandatory. The identifier for the PVC. It can either be the *vpi*/*vci* pair or the keyword **auto**, where *vpi* is a Virtual Path Index from 0 to 255, *vci* is a Virtual Circuit Index from from 0 to 65535, and **auto** directs the system to detect the Virtual Path Index and Virtual Circuit Index automatically.

Default

None.

Usage Guidelines

Use this command to define a permanent virtual Circuit (PVC) on an ADSL interface.

At this time only a single PVC is supported. In addition, only a single encapsulation—either Classical IP over Asynchronous Transfer Mode (IPOA), Point-to-Point Protocol over Asynchronous Transfer Mode (PPPoA), or Point-to-Point Protocol over Ethernet (PPPoE)—is supported on the PVC.

Use the **set** form of this command to define a PVC.

Use the **delete** form of this command to remove all configuration for a PVC.

Use the **show** form of this command to view PVC configuration.

interfaces adsl <adslx> watchdog <state>

Enables or disables the ADSL watchdog feature on the link.

Syntax

set interfaces adsl *adslx* **watchdog** *state*

delete interfaces adsl *adslx* **watchdog**

show interfaces adsl *adslx* **watchdog**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   adsl adslx {
      watchdog [enable|disable]
         }
      }
   }
}
```

Parameters

adslx	Mandatory. Multi-node. The identifier for the ADSL interface you are defining. This may be **adsl0** to **adsl***x*, depending on what physical ADSL ports are actually available on the system.
state	Optional. Specifies whether the watchdog feature is to be used. Supported values are as follows: **enable**: The interface sends watchdog requests every 10 seconds. **disable**: The watchdog process is not performed. If disabled, the watchdog process will not be performed.

Default

The default is enabled.

Usage Guidelines

Use this command to enable or disable the ADSL watchdog feature.

When the watchdog feature is enabled, the system sends Asynchronous Transfer Mode Operation, Administration, and Maintenance (OAM) loopback packets every 10 seconds to confirm the connectivity of the PVC.

Use the **set** form of this command to enable or disable the watchdog feature.

Use the **delete** form of this command to remove the configuration and restore the default behavior.

Use the **show** form of this command to display ADSL watchdog configuration.

show interfaces adsl <if-name>

Displays status of an ADSL interface.

Syntax

show interfaces adsl *if-name*

Command Mode

Operational mode.

Parameters

if-name	Mandatory. The name of the interface. This can be the name of a PPPoA-, PPPoE-, or Classical IPOA- encapsulated DSL interface; that is the interface name can be **pppoa***x*, **pppoe***x*, or **adsl***x*.

Default

None.

Usage Guidelines

Use this command to display the status of an ADSL interface.

The status information for Classical IP over Asynchronous Transfer Mode (IPOA) encapsulation includes IP layer information (since the ADSL interface is used as an IP-layer interface), where Point-to-Point Protocol over Asynchronous Transfer Mode (PPPoA) and Point-to-Point Protocol over Ethernet (PPPoE) encapsulations do not.

show interfaces adsl <if-name> capture

Starts a low-level packet trace on an ADSL interface.

Syntax

show interfaces adsl *if-name* **capture**

Command Mode

Operational mode.

Parameters

if-name	Mandatory. The name of the interface. This can be the name of a PPPoA-, PPPoE-, or Classical IPOA- encapsulated DSL interface; that is the interface name can be **pppoa***x*, **pppoe***x*, or **adsl***x*.

Default

None.

Usage Guidelines

Use this command to view the packets send and received on the specified ADSL interface. The trace will continue until stopped.

show interfaces adsl <if-name> queue

Displays queue information on an ADSL interface.

Syntax

show interfaces adsl *if-name* **queue** [**class** | **filter**]

Command Mode

Operational mode.

Parameters

if-name	Mandatory. The name of the interface. This can be the name of a PPPoA-, PPPoE-, or Classical IPOA- encapsulated DSL interface; that is the interface name can be **pppoa***x*, **pppoe***x*, or **adsl***x*.
class	Optional. Displays the queue classes for a device.
filter	Optional. Displays the queue filters for a device.

Default

None.

Usage Guidelines

Use this command to display queue information on the specified ADSL interface.

show interfaces adsl <if-name> status

Displays detailed status of an ADSL interface.

Syntax

show interfaces adsl *if-name* **status**

Command Mode

Operational mode.

Parameters

if-name	Mandatory. The name of the interface. This can be the name of a PPPoA-, PPPoE-, or Classical IPOA- encapsulated DSL interface; that is the interface name can be **pppoa***x*, **pppoe***x*, or **adsl***x*.

Default

None.

Usage Guidelines

Use this command to display detailed physical and data link status and statistics for an ADSL interface.

This command displays status independent of the encapsulation used on the interface.

Chapter 3: Wireless Modem Interfaces

This chapter explains how to work with wireless modems on the Vyatta system.

This chapter presents the following topics:

- Wireless Modem Configuration
- Wireless Modem Interface Commands

Wireless Modem Configuration

The **wirelessmodem** interface provides access (through a wireless modem) to wireless networks provided by various cellular providers including AT&T and Verizon.

The Vyatta system interfaces with wireless modems via the **interfaces wirelessmodem** configuration commands.

For example, using a Sierra Wireless USB Connect 881 modem to access the AT&T network you would configure the system using the system defaults as follows:

Example 3-1 Sierra Wireless USB Connect 881 modem accessing the AT&T network

Step	Command
Specify a wirelessmodem configuration node.	`vyatta@R1# set interfaces wirelessmodem wlm0` `[edit]`
Commit the change	`vyatta@R1# commit` `[edit]`

Show the configuration.	`vyatta@R1# show interfaces wirelessmodem`
	`  wlm0 {`
	`  }`
	`[edit]`

In this case the default **network** (att) and default **device** (ttyUSB0) are used.

To use a UT Starcom (Pantech) 3G modem to access the Verizon network you would configure the system as follows:

Example 3-2 UT Starcom 3G modem accessing the Verizon network

Step	Command
Specify the Verizon chat script.	`vyatta@R1# set interfaces wirelessmodem wlm0 network verizon` `[edit]`
Specify the system device for the UT Starcom modem.	`vyatta@R1# set interfaces wirelessmodem wlm0 device ttyACM0` `[edit]`
Commit the change	`vyatta@R1# commit` `[edit]`

Example 3-2 UT Starcom 3G modem accessing the Verizon network

Show the configuration.	vyatta@R1# **show interfaces wirelessmodem**
	wlm0 {
	network verizon
	device ttyACM0
	}
	[edit]

Once the configuration is set up (as in either of the previous examples) the network is accessible.

It is possible to disconnect from the network using the **disconnect interface** *wlmx* command in operational mode as follows:

Example 3-3 Disconnecting from the network

Step	Command
Disconnecting from the network.	vyatta@R1> **disconnect interface wlm0**

Once disconnected the **connect interface** *wlmx* command can be used in operational mode to reconnect to the network as follows:

Example 3-4 Connecting to the network

Step	Command
Connect to the network.	vyatta@R1> **connect interface wlm0**

Wireless Modem Interface Commands

This chapter contains the following commands.

Configuration Commands	
interfaces wirelessmodem <wlmx>	Defines a wirelessmodem interface.
interfaces wirelessmodem <wlmx> backup	Specifies that a backup default route will be installed in the routing table.
interfaces wirelessmodem <wlmx> description <desc>	Specifies a description for a wirelessmodem interface.
interfaces wirelessmodem <wlmx> device <device>	Specifies the system device for the wireless modem.
interfaces wirelessmodem <wlmx> mtu <mtu>	Specifies the Maximum Transmit Unit (MTU) size for a wirelessmodem interface.
interfaces wirelessmodem <wlmx> network <scriptfile>	Specifies the chat script to use on a wirelessmodem interface.
interfaces wirelessmodem <wlmx> no-dns	Specifies that the network provider's DNS host should not be added to the local name resolution path.
interfaces wirelessmodem <wlmx> ondemand	Specifies that a connection will be re-established only when traffic is to be sent.
Operational Commands	
clear interfaces connection <wlmx>	Resets a PPP session on a wirelessmodem interface.
connect interface <wlmx>	Brings a wirelessmodem interface up.
disconnect interface <wlmx>	Brings a wirelessmodem interface down.
show interfaces wirelessmodem	Displays wirelessmodem interface information.

Commands for using other system features with wirelessmodem interfaces can be found in the following locations.

Related Commands Documented Elsewhere	
Firewall	Commands for configuring firewall on wirelessmodem interfaces are described in the *Vyatta Security Reference Guide*.
show system usb	.Displays information about peripherals connected to the USB bus. This command is described in the *Vyatta Basic System Reference Guide.*

clear interfaces connection <wlmx>

Resets a PPP session on a wirelessmodem interface.

Syntax

clear interfaces connection *wlmx*

Command Mode

Operational mode.

Parameters

wlmx	Mandatory. The interface to be operationally brought down and then up. The range is **wlm0** to **wlm999.**

Default

None.

Usage Guidelines

Use this command to operationally bring a Point-to-Point Protocol (PPP) session on a wirelessmodem interface down and then up.

connect interface <wlmx>

Brings a wirelessmodem interface up.

Syntax

connect interface *wlmx*

Command Mode

Operational mode.

Parameters

wlmx	Mandatory. The interface to be operationally brought down and then up. The range is **wlm0** to **wlm999.**

Default

None.

Usage Guidelines

Use this command to operationally bring a wirelessmodem interface up.

disconnect interface <wlmx>

Brings a wirelessmodem interface down.

Syntax

disconnect interface *wlmx*

Command Mode

Operational mode.

Parameters

wlmx	Mandatory. The interface to be operationally brought down and then up. The range is **wlm0** to **wlm999.**

Default

None.

Usage Guidelines

Use this command to operationally bring a wirelessmodem interface down.

interfaces wirelessmodem <wlmx>

Defines a wirelessmodem interface.

Syntax

set interfaces wirelessmodem *wlmx*

delete interfaces wirelessmodem *wlmx*

show interfaces wirelessmodem *wlmx*

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   wirelessmodem wlm0..wlm999 {
   }
}
```

Parameters

wlmx	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are defining. This may be **wlm0** to **wlm999**.

Default

None.

Usage Guidelines

Use this command to configure a wirelessmodem interface. You can define multiple wirelessmodem interfaces by creating multiple **wirelessmodem** configuration nodes. When a wirelessmodem interface is created a default route to the upstream provider is installed in the routing table with an administrative distance of 0, making it a primary default route. This behavior can be changed using the **interfaces wirelessmodem <wlmx> backup** command.

Note that you cannot use **set** to change the name of the wirelessmodem interface. To change the name of a wirelessmodem interface, you must **delete** the old **wirelessmodem** configuration node and create a new one.

Use the **set** form of this command to create a wirelessmodem interface. Once the interface is created its status can be viewed using the **show interfaces** command.

Use the **delete** form of this command to remove all configuration for a wirelessmodem interface.

Use the **show** form of this command to view a wirelessmodem interface configuration.

interfaces wirelessmodem <wlmx> backup

Specifies that a backup default route will be installed in the routing table.

Syntax

set interfaces wirelessmodem *wlmx* **backup** [**distance** *distance*]

delete interfaces wirelessmodem *wlmx* **backup** [**distance**]

show interfaces wirelessmodem *wlmx* **backup** [**distance**]

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   wirelessmodem wlm0..wlm999 {
      backup {
         distance u32
      }
   }
}
```

Parameters

wlmx	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are using. This may be **wlm0** to **wlm999**.
distance	Optional. The administrative distance on the default route. The default is 10.

Default

If the **backup** option is not used a default route is added to the routing table with an administrative distance of 0. In other words, the default route is a primary default route. If the **backup** option is used but the distance option is not set, the default administrative distance for the default route is 10.

Usage Guidelines

Use this command to add a backup default route to the upstream provider to the routing table. This is useful if you wish to use the wireless network as a backup rather than a primary access to the upstream peer. When this option is set, the default route received from the upstream peer is added as a backup default route with an administrative distance of 10 (this value can be modified using the **distance** option). When this is done, the wireless modem will be used only if the primary route fails.

Use the **set** form of this command to add a backup default route to the upstream provider to the routing table.

Use the **delete** form of this command to remove the backup default route.

Use the **show** form of this command to view the configuration.

interfaces wirelessmodem <wlmx> description <desc>

Specifies a description for a wirelessmodem interface.

Syntax

set interfaces wirelessmodem *wlmx* **description** *desc*

delete interfaces wirelessmodem *wlmx* **description**

show interfaces wirelessmodem *wlmx* **description**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   wirelessmodem wlm0..wlm999 {
      description text
   }
}
```

Parameters

wlmx	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are using. This may be **wlm0** to **wlm999**.
desc	Optional. A brief description for the wirelessmodem interface. If the description contains spaces, it must be enclosed in double quotes.

Default

None.

Usage Guidelines

Use this command to specify a description for the wirelessmodem interface.

Use the **set** form of this command to set the description for the wirelessmodem interface.

Use the **delete** form of this command to remove description configuration.

Use the **show** form of this command to view description configuration.

interfaces wirelessmodem <wlmx> device <device>

Specifies the system device for the wireless modem.

Syntax

set interfaces wirelessmodem *wlmx* **device** *device*

delete interfaces wirelessmodem *wlmx* **device**

show interfaces wirelessmodem *wlmx* **device**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   wirelessmodem wlm0..wlm999 {
      device text
   }
}
```

Parameters

wlmx	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are using. This may be **wlm0** to **wlm999**.
device	Optional. The system device used for the wireless modem. 3G modems typically use either **ttyUSB***x* or **ttyACM***x*. The default is **ttyUSB0**.

Default

The system device is **ttyUSB0**.

Usage Guidelines

Use this command to specify the system device for the wireless modem.

Use the **set** form of this command to set the system device for the wireless modem.

Use the **delete** form of this command to remove device configuration.

Use the **show** form of this command to view device configuration.

interfaces wirelessmodem <wlmx> mtu <mtu>

Specifies the Maximum Transmit Unit (MTU) size for a wirelessmodem interface.

Syntax

set interfaces wirelessmodem *wlmx* **mtu** *mtu*

delete interfaces wirelessmodem *wlmx* **mtu**

show interfaces wirelessmodem *wlmx* **mtu**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   wirelessmodem wlm0..wlm999 {
      mtu 1-1500
   }
}
```

Parameters

wlmx	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are using. This may be **wlm0** to **wlm999**.
mtu	Optional. Sets the Maximum Transfer Unit (MTU), in octets, for the interface. This value will be used unless the peer requests a smaller value via MRU (Maximum Receive Unit) negotiation. When forwarding, IPv4 packets larger than the MTU will be fragmented unless the DF bit is set. In that case, the packets will be dropped and an ICMP “Packet too big” message is returned to the sender. The range is 1 to 1500. If the DF flag is set, fragmentation will never be performed.

Default

The MTU is 1500. Fragmentation is not performed.

Usage Guidelines

Use this command to specify the Maximum Transfer Unit (MTU). This is the maximum packet size the interface will send.

Use the **set** form of this command to set the MTU.

Use the **delete** form of this command to restore the default MTU behavior.

Use the **show** form of this command to view MTU configuration.

interfaces wirelessmodem <wlmx> network <scriptfile>

Specifies the chat script to use on a wirelessmodem interface.

Syntax

set interfaces wirelessmodem *wlmx* **network** *scriptfile*

delete interfaces wirelessmodem *wlmx* **network**

show interfaces wirelessmodem *wlmx* **network**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   wirelessmodem wlm0..wlm999 {
      network [att|verison]
   }
}
```

Parameters

wlmx	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are using. This may be **wlm0** to **wlm999**.
scriptfile	Mandatory. Specifies the chat script file used for a specific network.

Default

The **att** chat script is used..

Usage Guidelines

Use this command to specify the chat script for a given network. The chat script is a sequence of AT commands sent to the modem. Each network requires a chat script. If your network does not work with one of the existing chat scripts you can create a custom script and place it in /opt/vyatta/share/ppp/network. The CLI will automatically recognize it as a valid configuration option.

Use the **set** form of this command to specify the chat script to use.

Use the **delete** form of this command to remove the chat script.

Use the **show** form of this command to view the configuration.

interfaces wirelessmodem <wlmx> no-dns

Specifies that the network provider's DNS host should not be added to the local name resolution path.

Syntax

set interfaces wirelessmodem *wlmx* **no-dns**

delete interfaces wirelessmodem *wlmx* **no-dns**

show interfaces wirelessmodem *wlmx* **no-dns**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   wirelessmodem wlm0..wlm999 {
      no-dns
   }
}
```

Parameters

wlmx	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are using. This may be **wlm0** to **wlm999**.

Default

Add the network provider's DNS hosts to the local name resolution path.

Usage Guidelines

Use this command to specify that the network provider's DNS hosts are not to be added to the local name resolution path.

Use the **set** form of this command to specify that the network provider's DNS hosts are not to be added to the local name resolution path.

Use the **delete** form of this command to restore the default.

Use the **show** form of this command to view the configuration.

interfaces wirelessmodem <wlmx> ondemand

Specifies that a connection will be re-established only when traffic is to be sent.

Syntax

set interfaces wirelessmodem *wlmx* **ondemand**

delete interfaces wirelessmodem *wlmx* **ondemand**

show interfaces wirelessmodem *wlmx* **ondemand**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
   wirelessmodem wlm0..wlm999 {
      ondemand
   }
}
```

Parameters

wlmx	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are using. This may be **wlm0** to **wlm999**.

Default

The modem will always try to maintain a connection.

Usage Guidelines

Use this command to specify that, after a modem drops a connection, the connection will be re-established only when traffic is to be sent over the link.

NOTE *This feature is not guaranteed to work as the 3G wireless modems were not designed to do this.*

Use the **set** form of this command to specify that a connection will be re-established only when traffic is to be sent.

Use the **delete** form of this command to restore the default.

Use the **show** form of this command to view the configuration.

show interfaces wirelessmodem

Displays wirelessmodem interface information.

Syntax

show interfaces wirelessmodem [*wlmx* [**debug** | **stats**]]

Command Mode

Operational mode.

Parameters

wlmx	The name of a wirelessmodem interface. This may be **wlm0** to **wlm999**.
debug	Shows the startup debug log for the specified wirelessmodem interface.
stats	Shows interface statistics for the specified wirelessmodem interface.

Default

Information is shown for all available wirelessmodem interfaces.

Usage Guidelines

Use this command to view the operational status of a wirelessmodem interface.

Examples

Example 3-6 shows output for **show interfaces** with the wlm0 interface visible at the bottom of the output.

Example 3-5 “show interfaces”: Displaying interface status

```
vyatta@R1> show interfaces
Interface    IP Address         State       Link   Description
eth0         10.1.0.175/24      up          up
eth1         -                  up          down
eth2         -                  up          down
eth3         -                  up          down
eth4         -                  up          down
eth5         -                  up          down
```

```
lo            127.0.0.1/8         up          up
lo            ::1/128             up          up
wlm0          166.129.139.21/32   up          up
```

Example 3-6 shows output for **show interfaces wirelessmodem** *wlmx*.

Example 3-6 "show interfaces wirelessmodem wlm0": Displaying wirelessmodem interface information

```
vyatta@R1> show interfaces wirelessmodem wlm0
wlm0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast state UNKNOWN qlen 100
    link/ppp
    inet 166.129.139.21 peer 10.64.64.64/32 scope global wlm0

   RX:  bytes    packets     errors    dropped    overrun      mcast
          94          7          0          0          0          0
   TX:  bytes    packets     errors    dropped    carrier collisions
         157          8          0          0          0          0
```

Example 3-7 shows output for **show interfaces wirelessmodem** *wlmx* **debug**.

NOTE *The output can be interrupted by pressing Ctrl-C*

Example 3-7 "show interfaces wirelessmodem wlm0 debug": Displaying debug information for the wirelessmodem interface

```
vyatta@R1> show interfaces wirelessmodem wlm0 debug
Serial connection established.
using channel 1
Using interface ppp0
Connect: ppp0 <--> /dev/ttyUSB0
sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x3092a865>
<pcomp> <accomp>] rcvd [LCP ConfReq id=0x0 <asyncmap 0x0> <auth
chap MD5> <magic 0x3ae155ee> <pcomp> <accomp>]
lcp_reqci: returning CONFNAK.
sent [LCP ConfNak id=0x0 <auth pap>]
rcvd [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0x3092a865>
<pcomp> <accomp>] rcvd [LCP ConfReq id=0x1 <asyncmap 0x0> <auth
pap> <magic 0x3ae155ee> <pcomp> <accomp>]
lcp_reqci: returning CONFACK.
sent [LCP ConfAck id=0x1 <asyncmap 0x0> <auth pap> <magic
0x3ae155ee> <pcomp> <accomp>] sent [PAP AuthReq id=0x1
user="saturn" password=<hidden>] rcvd [LCP DiscReq id=0x2
magic=0x3ae155ee] rcvd [PAP AuthAck id=0x1 ""] PAP
authentication succeeded sent [CCP ConfReq id=0x1 <deflate 15>
```

```
<deflate(old#) 15> <bsd v1 15>] sent [IPCP ConfReq id=0x1
<compress VJ 0f 01> <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns3
0.0.0.0>] rcvd [LCP ProtRej id=0x3 80 fd 01 01 00 0f 1a 04 78 00
18 04 78 00 15 03 2f] Protocol-Reject for 'Compression Control
Protocol' (0x80fd) received rcvd [IPCP ConfNak id=0x1 <ms-dns1
10.11.12.13> <ms-dns3 10.11.12.14>] sent [IPCP ConfReq id=0x2
<compress VJ 0f 01> <addr 0.0.0.0> <ms-dns1 10.11.12.13> <ms-dns3
10.11.12.14>] rcvd [IPCP ConfNak id=0x2 <ms-dns1 10.11.12.13>
<ms-dns3 10.11.12.14>] sent [IPCP ConfReq id=0x3 <compress VJ 0f
01> <addr 0.0.0.0> <ms-dns1 10.11.12.13> <ms-dns3 10.11.12.14>]
rcvd [IPCP ConfReq id=0x0]
ipcp: returning Configure-NAK
sent [IPCP ConfNak id=0x0 <addr 0.0.0.0>] rcvd [IPCP ConfRej
id=0x3 <compress VJ 0f 01>] sent [IPCP ConfReq id=0x4 <addr
0.0.0.0> <ms-dns1 10.11.12.13> <ms-dns3 10.11.12.14>] rcvd [IPCP
ConfReq id=0x1]
ipcp: returning Configure-ACK
sent [IPCP ConfAck id=0x1]
rcvd [IPCP ConfNak id=0x4 <addr 166.129.139.21> <ms-dns1
209.183.54.151> <ms-dns3 209.183.54.151>] sent [IPCP ConfReq
id=0x5 <addr 166.129.139.21> <ms-dns1 209.183.54.151> <ms-dns3
209.183.54.151>] rcvd [IPCP ConfAck id=0x5 <addr 166.129.139.21>
<ms-dns1 209.183.54.151> <ms-dns3 209.183.54.151>]
ipcp: up
Could not determine remote IP address: defaulting to 10.64.64.64
Cannot determine ethernet address for proxy ARP local  IP address
166.129.139.21 remote IP address 10.64.64.64
primary   DNS address 209.183.54.151
secondary DNS address 209.183.54.151
```

Example 3-8 shows the output for **show interfaces wirelessmodem** *wlmx* **stats**.

Example 3-8 “show interfaces wirelessmodem *wlmx* stats”: Displaying statistics for the wirelessmodem interface

```
vyatta@R1> show interfaces wirelessmodem wlm0 stats
IN   PACK VJCOMP  VJUNC  VJERR  |      OUT   PACK VJCOMP  VJUNC
NON-VJ
     94     7     0     0     0 |     157     8     0     0     8
```

Glossary of Acronyms

ACL	access control list
ADSL	Asymmetric Digital Subscriber Line
AS	autonomous system
ARP	Address Resolution Protocol
BGP	Border Gateway Protocol
BIOS	Basic Input Output System
BPDU	Bridge Protocol Data Unit
CA	certificate authority
CHAP	Challenge Handshake Authentication Protocol
CLI	command-line interface
DDNS	dynamic DNS
DHCP	Dynamic Host Configuration Protocol
DLCI	data-link connection identifier
DMI	desktop management interface
DMZ	demilitarized zone
DNS	Domain Name System
DSCP	Differentiated Services Code Point
DSL	Digital Subscriber Line
eBGP	external BGP
EGP	Exterior Gateway Protocol

ECMP	equal-cost multipath
ESP	Encapsulating Security Payload
FIB	Forwarding Information Base
FTP	File Transfer Protocol
GRE	Generic Routing Encapsulation
HDLC	High-Level Data Link Control
I/O	Input/Ouput
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers
IGP	Interior Gateway Protocol
IPS	Intrusion Protection System
IKE	Internet Key Exchange
IP	Internet Protocol
IPOA	IP over ATM
IPsec	IP security
IPv4	IP Version 4
IPv6	IP Version 6
ISP	Internet Service Provider
L2TP	Layer 2 Tunneling Protocol
LACP	Link Aggregation Control Protocol
LAN	local area network
MAC	medium access control
MIB	Management Information Base
MLPPP	multilink PPP
MRRU	maximum received reconstructed unit
MTU	maximum transmission unit

NAT	Network Address Translation
ND	Neighbor Discovery
NIC	network interface card
NTP	Network Time Protocol
OSPF	Open Shortest Path First
OSPFv2	OSPF Version 2
OSPFv3	OSPF Version 3
PAM	Pluggable Authentication Module
PAP	Password Authentication Protocol
PCI	peripheral component interconnect
PKI	Public Key Infrastructure
PPP	Point-to-Point Protocol
PPPoA	PPP over ATM
PPPoE	PPP over Ethernet
PPTP	Point-to-Point Tunneling Protocol
PVC	permanent virtual circuit
QoS	quality of service
RADIUS	Remote Authentication Dial-In User Service
RIB	Routing Information Base
RIP	Routing Information Protocol
RIPng	RIP next generation
Rx	receive
SNMP	Simple Network Management Protocol
SONET	Synchronous Optical Network
SSH	Secure Shell
STP	Spanning Tree Protocol
TACACS+	Terminal Access Controller Access Control System Plus

TCP	Transmission Control Protocol
ToS	Type of Service
Tx	transmit
UDP	User Datagram Protocol
vif	virtual interface
VLAN	virtual LAN
VPN	Virtual Private Network
VRRP	Virtual Router Redundancy Protocol
WAN	wide area network