

VYATTA, INC.



Vyatta System

# WAN Interfaces

## REFERENCE GUIDE

Serial Interfaces

Testing Serial Lines

DSL Interfaces

Wireless Modem Interfaces



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

## **COPYRIGHT**

Copyright © 2005–2012 Vyatta, Inc. All rights reserved.

Vyatta reserves the right to make changes to software, hardware, and documentation without notice. For the most recent version of documentation, visit the Vyatta web site at [vyatta.com](http://vyatta.com).

## **PROPRIETARY NOTICES**

Vyatta is a registered trademark of Vyatta, Inc.

Hyper-V is a registered trademark of Microsoft Corporation.

VMware, VMware ESX, and VMware server are trademarks of VMware, Inc.

XenServer, and XenCenter are trademarks of Citrix Systems, Inc.

All other trademarks are the property of their respective owners.

RELEASE DATE: October 2012

DOCUMENT REVISION. 6.5R1 v01

RELEASED WITH: 6.5R1

PART NO. A0-0212-10-0016

# Contents

<b>Quick List of Commands</b> .....	<b>vi</b>
<b>List of Examples</b> .....	<b>ix</b>
<b>Preface</b> .....	<b>xi</b>
Intended Audience .....	xii
Organization of This Guide .....	xii
Document Conventions .....	xiii
Vyatta Publications .....	xiii
<b>Chapter 1 Serial Interfaces</b> .....	<b>1</b>
Serial Interface Configuration .....	2
Virtual Interfaces on Serial Interfaces .....	2
Enabling Interfaces .....	2
Viewing Available Serial Interfaces .....	3
Serial Interface Commands .....	5
clear interfaces serial .....	8
interfaces serial <wanx> .....	11
interfaces serial <wanx> description <desc> .....	13
interfaces serial <wanx> e1-options .....	15
interfaces serial <wanx> e1-options clock <type> .....	17
interfaces serial <wanx> e1-options framing <type> .....	19
interfaces serial <wanx> e1-options timeslots .....	21
interfaces serial <wanx> e3-options .....	23
interfaces serial <wanx> e3-options clock <type> .....	24
interfaces serial <wanx> e3-options framing <type> .....	26
interfaces serial <wanx> e3-options line-coding <type> .....	28
interfaces serial <wanx> encapsulation <type> .....	30
interfaces serial <wanx> synch-options .....	32
interfaces serial <wanx> synch-options baud-rate <rate> .....	34
interfaces serial <wanx> synch-options clock <type> .....	36
interfaces serial <wanx> synch-options connection <type> .....	38
interfaces serial <wanx> synch-options line-coding <type> .....	40
interfaces serial <wanx> synch-options line-idle <type> .....	42
interfaces serial <wanx> t1-options .....	44
interfaces serial <wanx> t1-options clock <type> .....	46
interfaces serial <wanx> t1-options lbo <range> .....	48

interfaces serial <wanx> t1-options timeslots . . . . .	50
interfaces serial <wanx> t3-options . . . . .	52
interfaces serial <wanx> t3-options clock <type> . . . . .	54
interfaces serial <wanx> t3-options framing <type> . . . . .	56
interfaces serial <wanx> t3-options line-coding <type> . . . . .	58
monitor interfaces serial <wanx> traffic . . . . .	60
show interfaces serial . . . . .	62
show interfaces serial <wanx> . . . . .	64
show interfaces serial <wanx> bert-status . . . . .	65
show interfaces serial <wanx> cisco-hdlc . . . . .	67
show interfaces serial <wanx> frame-relay . . . . .	68
show interfaces serial <wanx> log . . . . .	69
show interfaces serial <wanx> loopback . . . . .	70
show interfaces serial <wanx> physical . . . . .	71
show interfaces serial <wanx> ppp . . . . .	72
show interfaces serial <wanx> queue . . . . .	74
show interfaces serial <wanx> trace . . . . .	75
<b>Chapter 2 Testing Serial Lines . . . . .</b>	<b>77</b>
Serial Line Testing Overview . . . . .	78
Serial Line Loopbacks . . . . .	78
Loopback Tests . . . . .	81
Bit Error Rate Tests . . . . .	83
Serial Line Testing Commands . . . . .	85
test-definition bert <test-name> . . . . .	87
test-definition bert <test-name> duration <duration> . . . . .	89
test-definition bert <test-name> err-insert-rate <rate> . . . . .	90
test-definition bert <test-name> loopup-code <code> . . . . .	92
test-definition bert <test-name> pattern <pattern-name> . . . . .	94
test-definition bert <test-name> pattern <pattern-name> alternating-word . . . . .	96
test-definition bert <test-name> pattern <pattern-name> repeating . . . . .	98
test interface <wanx> start-bert <test-name> . . . . .	100
test interface <wanx> stop-bert . . . . .	102
test loopback down . . . . .	103
test loopback start . . . . .	105
test loopback up . . . . .	107
<b>Chapter 3 DSL Interfaces . . . . .</b>	<b>109</b>
DSL Configuration . . . . .	110
ADSL Interfaces Overview . . . . .	110
ADSL Configuration Example . . . . .	110

---

DSL Commands .....	113
interfaces adsl <adslx> .....	115
interfaces adsl <adslx> pvc <pvc-id> .....	117
interfaces adsl <adslx> watchdog <state> .....	119
monitor interfaces adsl <if-name> traffic .....	121
show interfaces adsl <if-name> .....	122
show interfaces adsl <if-name> queue .....	123
show interfaces adsl <if-name> status .....	124
<b>Chapter 4 Wireless Modem Interfaces .....</b>	<b>125</b>
Wireless Modem Configuration .....	126
Wireless Modem Interface Commands .....	128
clear interfaces connection <wlmx> .....	130
connect interface <wlmx> .....	131
disconnect interface <wlmx> .....	132
interfaces wirelessmodem <wlmx> .....	133
interfaces wirelessmodem <wlmx> backup .....	135
interfaces wirelessmodem <wlmx> description <desc> .....	137
interfaces wirelessmodem <wlmx> device <device> .....	139
interfaces wirelessmodem <wlmx> ipv6 address .....	140
interfaces wirelessmodem <wlmx> ipv6 disable-forwarding .....	142
interfaces wirelessmodem <wlmx> ipv6 dup-addr-detect-transmits <num> .....	144
interfaces wirelessmodem <wlmx> ipv6 router-advert .....	146
interfaces wirelessmodem <wlmx> mtu <mtu> .....	151
interfaces wirelessmodem <wlmx> network <scriptfile> .....	153
interfaces wirelessmodem <wlmx> no-dns .....	155
interfaces wirelessmodem <wlmx> ondemand .....	156
show interfaces wirelessmodem .....	158
<b>Glossary of Acronyms .....</b>	<b>161</b>

# Quick List of Commands

Use this list to help you quickly locate commands.

clear interfaces connection <wlmx> .....	130
clear interfaces serial .....	8
connect interface <wlmx> .....	131
disconnect interface <wlmx> .....	132
interfaces adsl <adslx> pvc <pvc-id> .....	117
interfaces adsl <adslx> watchdog <state> .....	119
interfaces adsl <adslx> .....	115
interfaces serial <wanx> description <desc> .....	13
interfaces serial <wanx> e1-options clock <type> .....	17
interfaces serial <wanx> e1-options framing <type> .....	19
interfaces serial <wanx> e1-options timeslots .....	21
interfaces serial <wanx> e1-options .....	15
interfaces serial <wanx> e3-options clock <type> .....	24
interfaces serial <wanx> e3-options framing <type> .....	26
interfaces serial <wanx> e3-options line-coding <type> .....	28
interfaces serial <wanx> e3-options .....	23
interfaces serial <wanx> encapsulation <type> .....	30
interfaces serial <wanx> synch-options baud-rate <rate> .....	34
interfaces serial <wanx> synch-options clock <type> .....	36
interfaces serial <wanx> synch-options connection <type> .....	38
interfaces serial <wanx> synch-options line-coding <type> .....	40
interfaces serial <wanx> synch-options line-idle <type> .....	42
interfaces serial <wanx> synch-options .....	32
interfaces serial <wanx> t1-options clock <type> .....	46
interfaces serial <wanx> t1-options lbo <range> .....	48
interfaces serial <wanx> t1-options timeslots .....	50
interfaces serial <wanx> t1-options .....	44
interfaces serial <wanx> t3-options clock <type> .....	54
interfaces serial <wanx> t3-options framing <type> .....	56
interfaces serial <wanx> t3-options line-coding <type> .....	58
interfaces serial <wanx> t3-options .....	52
interfaces serial <wanx> .....	11

interfaces wirelessmodem <wlmx> backup .....	135
interfaces wirelessmodem <wlmx> description <desc>.....	137
interfaces wirelessmodem <wlmx> device <device> .....	139
interfaces wirelessmodem <wlmx> ipv6 address .....	140
interfaces wirelessmodem <wlmx> ipv6 disable-forwarding .....	142
interfaces wirelessmodem <wlmx> ipv6 dup-addr-detect-transmits <num> .....	144
interfaces wirelessmodem <wlmx> ipv6 router-advert .....	146
interfaces wirelessmodem <wlmx> mtu <mtu> .....	151
interfaces wirelessmodem <wlmx> network <scriptfile> .....	153
interfaces wirelessmodem <wlmx> no-dns .....	155
interfaces wirelessmodem <wlmx> ondemand .....	156
interfaces wirelessmodem <wlmx> .....	133
monitor interfaces adsl <if-name> traffic .....	121
monitor interfaces serial <wanx> traffic .....	60
show interfaces adsl <if-name> queue .....	123
show interfaces adsl <if-name> status .....	124
show interfaces adsl <if-name> .....	122
show interfaces serial <wanx> bert-status .....	65
show interfaces serial <wanx> cisco-hdlc .....	67
show interfaces serial <wanx> frame-relay.....	68
show interfaces serial <wanx> log .....	69
show interfaces serial <wanx> loopback.....	70
show interfaces serial <wanx> physical.....	71
show interfaces serial <wanx> ppp .....	72
show interfaces serial <wanx> queue .....	74
show interfaces serial <wanx> trace .....	75
show interfaces serial <wanx> .....	64
show interfaces serial .....	62
show interfaces wirelessmodem .....	158
test interface <wanx> start-bert <test-name> .....	100
test interface <wanx> stop-bert.....	102
test loopback down.....	103
test loopback start.....	105
test loopback up .....	107
test-definition bert <test-name> duration <duration> .....	89
test-definition bert <test-name> err-insert-rate <rate> .....	90

---

test-definition bert <test-name> loopup-code <code> .....	92
test-definition bert <test-name> pattern <pattern-name> alternating-word .....	96
test-definition bert <test-name> pattern <pattern-name> repeating .....	98
test-definition bert <test-name> pattern <pattern-name> .....	94
test-definition bert <test-name> .....	87



# List of Examples

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

Example 1-1 Viewing available system interfaces .....	3
Example 1-2 "clear interfaces serial" .....	9
Example 1-3 "clear interfaces serial wan0 counters cisco-hdlc": Displaying the result of the clear command. ....	9
Example 1-4 Displaying traffic data .....	60
Example 1-5 "show interfaces serial": Displaying serial interface information .....	62
Example 1-6 "show interfaces serial wan1": Displaying serial interface information .....	64
Example 1-7 "show interfaces serial wanx bert-status" .....	65
Example 1-8 "show interfaces serial wanx ppp" .....	72
Example 1-9 "show interfaces serial wanx trace" .....	75
Example 2-1 Testing the local WAN interface.....	82
Example 2-2 Testing the remote WAN interface .....	82
Example 2-3 Testing the circuit .....	82
Example 2-4 Testing a physical loopback on the local system.....	83
Example 2-5 Starting a BERT .....	101
Example 2-6 Stopping a BERT .....	102
Example 2-7 Deactivating a local loopback on wan0 .....	104
Example 2-8 Successful test of a loopback .....	105
Example 2-9 Unsuccessful loopback test .....	106
Example 2-10 Activating a local loopback .....	108
Example 3-1 Creating and configuring an ADSL interface for PPPoE encapsulation .....	112
Example 4-1 Sierra Wireless USB Connect 881 modem accessing the AT&T network .....	126
Example 4-2 UT Starcom 3G modem accessing the Verizon network .....	126
Example 4-3 Disconnecting from the network .....	127
Example 4-4 Connecting to the network.....	127
Example 4-5 "show interfaces": Displaying interface status .....	158
Example 4-6 "show interfaces wirelessmodem wlm0": Displaying wirelessmodem interface information .....	159
Example 4-7 "show interfaces wirelessmodem wlm0 debug": Displaying debug information for the wirelessmodem interface .....	159

Example 4-8 “show interfaces wirelessmodem wlmx statistics”: Displaying statistics for the wirelessmodem interface  
160

# Preface

This document describes the various deployment, installation, and upgrade options for Vyatta software.

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

- [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
- IP services

## Organization of This Guide

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

- [Quick List of Commands](#)  
Use this list to help you quickly locate commands.
- [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:

Chapter	Description	Page
<a href="#">Chapter 1: Serial Interfaces</a>	This chapter explains how to configure and monitor serial interfaces on the Vyatta system.	1
<a href="#">Chapter 2: Testing Serial Lines</a>	This chapter describes the tests available for serial interfaces on the Vyatta system.	77
<a href="#">Chapter 3: DSL Interfaces</a>	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.	109
<a href="#">Chapter 4: Wireless Modem Interfaces</a>	This chapter explains how to work with wireless modems on the Vyatta system.	125
<a href="#">Glossary of Acronyms</a>		161

# Document Conventions

This guide uses the following advisory paragraphs, as follows.



**WARNING** Warnings alert you to situations that may pose a threat to personal safety.



**CAUTION** Cautions alert you to situations that might cause harm to your system or damage to equipment, or that may affect service.

**NOTE** Notes provide information you might need to avoid problems or configuration errors.

This document uses the following typographic conventions.

Monospace	Examples, command-line output, and representations of configuration nodes.
<b>bold Monospace</b>	Your input: something you type at a command line.
<b>bold</b>	Commands, keywords, and file names, when mentioned inline.  Objects in the user interface, such as tabs, buttons, screens, and panes.
<i>italics</i>	An argument or variable where you supply a value.
<key>	A key on your keyboard, such as <Enter>. Combinations of keys are joined by plus signs (“+”), as in <Ctrl>+c.
[ key1   key2]	Enumerated options for completing a syntax. An example is [enable   disable].
<i>num1–numN</i>	A inclusive range of numbers. An example is 1–65535, which means 1 through 65535, inclusive.
<i>arg1..argN</i>	A range of enumerated values. An example is eth0..eth3, which means eth0, eth1, eth2, or eth3.
<i>arg[ arg...]</i> <i>arg[,arg...]</i>	A value that can optionally represent a list of elements (a space-separated list and a comma-separated list, respectively).

## Vyatta Publications

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

Additional information is available on [www.vyatta.com](http://www.vyatta.com) and [www.vyatta.org](http://www.vyatta.org).

# Chapter 1: Serial Interfaces

This chapter explains how to configure and monitor serial interfaces on the Vyatta system.



*This feature is available only in the Vyatta Subscription Edition.*

This chapter presents the following topics:

- [Serial Interface Configuration](#)
- [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 an 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
eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP qlen 1000
  link/ether 00:15:c5:fb:ba:e8 brd ff:ff:ff:ff:ff:ff
  inet 10.1.0.44/24 brd 10.1.0.255 scope global eth0
  inet6 fe80::215:c5ff:fe9b:bae8/64 scope link
      valid_lft forever preferred_lft forever

RX:  bytes   packets   errors   dropped   overrun   mcast
    18469     257         0         0         0         1
TX:  bytes   packets   errors   dropped   carrier   collisions
    17434     170         0         0         0         0

eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP qlen 1000
  link/ether 00:15:c5:fb:ba:e9 brd ff:ff:ff:ff:ff:ff
  inet6 fe80::215:c5ff:fe9b:bae9/64 scope link
      valid_lft forever preferred_lft forever

RX:  bytes   packets   errors   dropped   overrun   mcast
     0         0         0         0         0         0

```

```

TX: bytes    packets    errors    dropped    carrier collisions
     424         4         0         0         0         0

```

```

lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever

```

```

RX: bytes    packets    errors    dropped    overrun    mcast
     3600         60         0         0         0         0
TX: bytes    packets    errors    dropped    carrier collisions
     3600         60         0         0         0         0

```

```

wan0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 4112 qdisc pfifo_fast state UP
qlen 100
    link/ppp

```

```

RX: bytes    packets    errors    dropped    overrun    mcast
      0         0         0         5         0         0
TX: bytes    packets    errors    dropped    carrier collisions
      0         0         0         0         0         0

```

```

wan1: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 4112 qdisc pfifo_fast state UP
qlen 100
    link/ppp

```

```

RX: bytes    packets    errors    dropped    overrun    mcast
      0         0         0         5         0         0
TX: bytes    packets    errors    dropped    carrier collisions
      0         0         0         0         0         0

```

---

# Serial Interface Commands

This chapter contains the following commands.

Configuration Commands	
<b>Serial Interface</b>	
<code>interfaces serial &lt;wan&gt;</code>	Specifies basic serial interface configuration, including Layer 2 encapsulation characteristics.
<code>interfaces serial &lt;wan&gt; description &lt;desc&gt;</code>	Specifies a description for a serial interface.
<code>interfaces serial &lt;wan&gt; encapsulation &lt;type&gt;</code>	Sets the encapsulation type for a serial interface.
<b>E1 Options</b>	
<code>interfaces serial &lt;wan&gt; e1-options</code>	Specifies the physical line characteristics for an E1 serial interface.
<code>interfaces serial &lt;wan&gt; e1-options clock &lt;type&gt;</code>	Sets the timing source for an E1 serial interface.
<code>interfaces serial &lt;wan&gt; e1-options framing &lt;type&gt;</code>	Sets the framing for an E1 serial interface.
<code>interfaces serial &lt;wan&gt; e1-options timeslots</code>	Defines timeslots for a 32-channel channelized E1 line.
<b>E3 Options</b>	
<code>interfaces serial &lt;wan&gt; e3-options</code>	Specifies the physical line characteristics for an E3 serial interface.
<code>interfaces serial &lt;wan&gt; e3-options clock &lt;type&gt;</code>	Specifies the timing source for an E3 serial interface.
<code>interfaces serial &lt;wan&gt; e3-options framing &lt;type&gt;</code>	Specifies the framing type for an E3 serial interface.
<code>interfaces serial &lt;wan&gt; e3-options line-coding &lt;type&gt;</code>	Specifies the line coding for an E3 serial interface.
<b>Synchronous Serial</b>	
<code>interfaces serial &lt;wan&gt; synch-options</code>	Specifies the physical line characteristics for synchronous serial interfaces.
<code>interfaces serial &lt;wan&gt; synch-options baud-rate &lt;rate&gt;</code>	Sets the bit rate for an internally clocked synchronous serial interface.
<code>interfaces serial &lt;wan&gt; synch-options clock &lt;type&gt;</code>	Sets the timing source for a synchronous serial interface.

<code>interfaces serial &lt;wan&gt; synch-options connection &lt;type&gt;</code>	Sets the connection type for a synchronous serial interface.
<code>interfaces serial &lt;wan&gt; synch-options line-coding &lt;type&gt;</code>	Sets the line coding standard for a synchronous serial interface.
<code>interfaces serial &lt;wan&gt; synch-options line-idle &lt;type&gt;</code>	Sets the idle line signalling for a synchronous serial interface.
<b>T1 Options</b>	
<code>interfaces serial &lt;wan&gt; t1-options</code>	Specifies the physical line characteristics for T1 serial interfaces.
<code>interfaces serial &lt;wan&gt; t1-options clock &lt;type&gt;</code>	Sets the timing source for a T1 serial interface.
<code>interfaces serial &lt;wan&gt; t1-options lbo &lt;range&gt;</code>	Specifies the line build-out (LBO) range for a T1 serial interface.
<code>interfaces serial &lt;wan&gt; t1-options timeslots</code>	Defines timeslots for a 24-channel channelized T1 line.
<b>T3 Options</b>	
<code>interfaces serial &lt;wan&gt; t3-options</code>	Specifies the physical line characteristics for a T3 serial interface.
<code>interfaces serial &lt;wan&gt; t3-options clock &lt;type&gt;</code>	Specifies the timing source for the circuit.
<code>interfaces serial &lt;wan&gt; t3-options framing &lt;type&gt;</code>	Specifies the framing type for a T3 serial interface.
<code>interfaces serial &lt;wan&gt; t3-options line-coding &lt;type&gt;</code>	Specifies the line coding for a T3 serial interface.
<b>Operational Commands</b>	
<code>clear interfaces serial</code>	Clears counters for serial interfaces
<code>monitor interfaces serial &lt;wan&gt; traffic</code>	Displays traffic on a serial interface.
<code>show interfaces serial</code>	Displays statistics and status information for all serial interfaces.
<code>show interfaces serial &lt;wan&gt;</code>	Displays statistics and status information for a serial interface.
<code>show interfaces serial &lt;wan&gt; bert-status</code>	Displays a message indicating BERT progress for a serial interface.
<code>show interfaces serial &lt;wan&gt; cisco-hdlc</code>	Displays Cisco HDLC information for a serial interface.
<code>show interfaces serial &lt;wan&gt; frame-relay</code>	Displays Frame Relay information for a serial interface.

<code>show interfaces serial &lt;wan&gt; log</code>	Displays log information for a serial interface.
<code>show interfaces serial &lt;wan&gt; loopback</code>	Displays loopback information for a serial interface.
<code>show interfaces serial &lt;wan&gt; physical</code>	Displays physical device information for a serial interface.
<code>show interfaces serial &lt;wan&gt; ppp</code>	Displays Point-to-Point protocol information for a serial interface.
<code>show interfaces serial &lt;wan&gt; queue</code>	Displays queue information for a serial interface.
<code>show interfaces serial &lt;wan&gt; trace</code>	Displays trace information for a serial interface.

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

Related Commands Documented Elsewhere	
Serial Interface Testing	Commands for testing serial interfaces, including commands for configuring BERT tests, are described in " <a href="#">Chapter 2: Testing Serial Lines</a> ."
Firewall	Commands for configuring firewall on serial interfaces are described in the <i>Vyatta Firewall Reference Guide</i> .
OSPF	Commands for configuring the Open Shortest Path First routing protocol on serial interfaces are described in the <i>Vyatta OSPF Reference Guide</i> .
RIP	Commands for configuring the Routing Information Protocol on serial interfaces are described in the <i>Vyatta RIP Reference Guide</i> .
QoS	Commands for configuring quality of service on serial interfaces are described in the <i>Vyatta QoS Reference Guide</i> .
System interfaces	Commands for showing the physical interfaces available on your system are described in the <i>Vyatta Basic System Reference Guide</i> .

## 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

<i>wanx</i>	The identifier of a configured serial interface.
<b>all</b>	Clears all counters for the specified serial interface.
<b>physical</b>	Clears counters related to the physical line settings for the specified interface.
<b>cisco-hdlc</b>	Clears counters related to Cisco HDLC settings for the specified interface.
<b>frame-relay</b>	Clears counters related to Frame Relay settings for the specified interface.
<b>ppp</b>	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 1-2](#) shows the result of the **clear interfaces serial** command used with no options.

---

**Example 1-2 “clear interfaces serial”**


---

```

vyatta@R1> clear interfaces serial
Communication statistics flushed
Operational statistics flushed
DSU/CSU Performance Monitoring counters were flushed.
Performance monitoring counters flushed
PPP statistics flushed
Communication statistics flushed
Operational statistics flushed
DSU/CSU Performance Monitoring counters were flushed.
Performance monitoring counters flushed
PPP statistics flushed
vyatta@R1>

```

---

**Example 1-3** shows the result of the **clear interfaces serial wan0 counters cisco-hdlc** options.

**Example 1-3 “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 Performance 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 Performance 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 wanx {
    }
}
```

---

### Parameters

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , 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 wanx {
    description desc
  }
}
```

---

### Parameters

---

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , depending on what serial interfaces that are actually available on the system.
<i>desc</i>	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 wanx {
    e1-options {
    }
  }
}
```

---

### Parameters

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , 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 wanx {
    e1-options {
      clock type
    }
  }
}
```

---

### Parameters

---

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

---

---

### 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 wanx {
    e1-options {
      framing type
    }
  }
}
```

---

### Parameters

---

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , depending on what serial interfaces that are actually available on the system.
<i>type</i>	Sets the frame type for the interface. Supported values are as follows:  <b>g704</b> : Uses the G.704 framing specification and sets the E1 frame type to use CRC4.  <b>g704-no-crc</b> : Uses the G.704 framing specification and sets the E1 frame type not to use CRC4.  <b>unframed</b> : 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 <b>g704</b> .

---

---

### 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 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 wanx {
    e1-options {
      timeslots {
        start start
        stop stop
      }
    }
  }
}
```

---

### Parameters

---

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , depending on what serial interfaces that are actually available on the system.
<b>start</b> <i>start</i>	The first timeslot in the range. The range of values is 1 to 32, where the value of <b>start</b> must be less than the value of <b>stop</b> . The default is 1.
<b>stop</b> <i>stop</i>	The last timeslot in the range. The range of values is 1 to 32, where the value of <b>start</b> must be less than the value of <b>stop</b> . 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 wanx {
    e3-options {
    }
  }
}
```

---

### Parameters

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , 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 wanx {
    e3-options {
      clock [internal|external]
    }
  }
}
```

---

### Parameters

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , depending on what serial interfaces that are actually available on the system.
<i>type</i>	Optional. Sets the timing source for the circuit. Supported values are as follows: <b>internal:</b> The interface will use the internal clock. <b>external:</b> 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 wanx {
    e3-options {
      framing [g751|g832|unframed]
    }
  }
}
```

---

### Parameters

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , depending on what serial interfaces that are actually available on the system.
<i>type</i>	Optional. Sets the frame type for the interface. Supported values are as follows:  <b>g751</b> : Sets the E3 frame type to be G.751-compliant. <b>g832</b> : Sets the E3 frame type to be G.832-compliant. <b>unframed</b> : 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 wanx {
    e3-options {
      line-coding [hdb3|ami]
    }
  }
}
```

---

### Parameters

---

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , depending on what serial interfaces that are actually available on the system.
<i>type</i>	Optional. Sets the line coding for the interface. Supported values are as follows:  <b>hdb3</b> : Sets the E3 line coding to be HDB3 (High Density Bipolar of order 3) compliant.  <b>ami</b> : 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 wanx {
    encapsulation type
  }
}
```

---

### Parameters

---

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , depending on what serial interfaces that are actually available on the system.
<i>type</i>	Mandatory. Sets the encapsulation type for the interface. Supported values are as follows:  <b>ppp</b> : Uses Point-to-Point Protocol (PPP) encapsulation for the interface.  <b>cisco-hdlc</b> : Uses Cisco High-Level Data Link Control (Cisco HDLC) encapsulation on the interface.  <b>frame-relay</b> : 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 Encapsulations Reference Guide and the Vyatta PPP-Based Encapsulations 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 wanx {
    synch-options {
    }
  }
}
```

---

### Parameters

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , 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 serial 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 wanx {
    synch-options {
      baud-rate rate
    }
  }
}
```

---

### Parameters

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , depending on what serial interfaces that are actually available on the system.
<b>baud-rate</b>	Optional. Sets the bit rate in bits per second for the circuit when the <b>clock</b> type is “internal”. It is ignored if the <b>clock</b> 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 wanx {
    synch-options {
      clock type
    }
  }
}
```

---

### Parameters

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

---

### 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 wanx {
    synch-options {
      connection type
    }
  }
}
```

---

### Parameters

---

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , depending on what serial interfaces that are actually available on the system.
<i>type</i>	Optional. Sets the connection type for the circuit. Supported values are as follows:  <b>permanent:</b> The interface supports (non-IP) permanent virtual circuits.  <b>switched:</b> The interface supports (non-IP) switched virtual circuits.  The default is <b>permanent</b> ; 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 wanx {
    synch-options {
      line-coding type
    }
  }
}
```

---

### Parameters

---

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

---

---

### 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 wanx {
    synch-options {
      line-idle type
    }
  }
}
```

---

### Parameters

---

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

---

---

### 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 wanx {
    t1-options
  }
}
```

---

### Parameters

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , 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 wanx {
    t1-options {
      clock type
    }
  }
}
```

---

### Parameters

---

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

---

---

### 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 wanx {
    t1-options {
      lbo range
    }
  }
}
```

---

### Parameters

---

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

---

---

**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 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 wanx {
    t1-options {
      timeslots {
        start start
        stop stop
      }
    }
  }
}
```

---

### Parameters

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , depending on what serial interfaces that are actually available on the system.
<b>start</b> <i>start</i>	The first timeslot in the range. The range of values is 1 to 24, where the value of <b>start</b> must be less than the value of <b>stop</b> . The default is 1.
<b>stop</b> <i>stop</i>	The last timeslot in the range. The range of values is 1 to 24, where the value of <b>start</b> must be less than the value of <b>stop</b> . The default is 24.

---



---

### 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 wanx {
    t3-options {
    }
  }
}
```

---

### Parameters

<i>wanx</i>	Mandatory. Multi-node. The identifier for the serial interface you are defining. This may be <b>wan0</b> to <b>wan23</b> , 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 wanx {
    t3-options {
      clock type
    }
  }
}
```

---

### Parameters

---

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

---

---

### 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 wanx {
    t3-options {
      framing type
    }
  }
}
```

---

### Parameters

---

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

---

---

### 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 wanx {
    t3-options {
      line-coding type
    }
  }
}
```

---

### Parameters

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

---

### 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.

## monitor interfaces serial <wanx> traffic

Displays traffic on a serial interface.

### Syntax

```
monitor interfaces serial wanx traffic [not port port | port port]
```

### Command Mode

Operational mode.

### Parameters

<i>wanx</i>	The name of a serial interface.
not port <i>port</i>	Show captured traffic on all but this port.
port <i>port</i>	Show captured traffic on this port only.

### Default

Captured traffic for all ports on the specified interface is shown.

### Usage Guidelines

Use this command to view serial traffic on the specified interface. Type Ctrl-C to stop the output.

### Examples

[Example 1-4](#) shows traffic data on interface wan0.

#### Example 1-4 Displaying traffic data

```
vyatta@vyatta:~$ monitor interfaces serial wan0 traffic
Capturing traffic on wan0 ...
0.000000 fe80::ad08:8661:4d:b925 -> ff02::c      SSDP M-SEARCH * HTTP/1.1
0.000067 fe80::69ca:5c11:bcf6:29da -> ff02::c      SSDP M-SEARCH * HTTP/1.1
2.608804 fe80::8941:71ef:b55d:e348 -> ff02::1:2    DHCPv6 Solicit
3.010862 fe80::ad08:8661:4d:b925 -> ff02::c      SSDP M-SEARCH * HTTP/1.1
3.010901 fe80::69ca:5c11:bcf6:29da -> ff02::c      SSDP M-SEARCH * HTTP/1.1
```

```
4.568357 192.168.1.254 -> 238.255.255.251 SSDP NOTIFY * HTTP/1.1
4.568372 192.168.1.254 -> 238.255.255.251 SSDP NOTIFY * HTTP/1.1
...
```

---

# show interfaces serial

Displays statistics and status information for all serial interfaces.

---

## Syntax

```
show interfaces serial
```

---

## Command Mode

Operational mode.

---

## Parameters

None.

---

## Default

None.

---

## Usage Guidelines

Use this command to view the operational status of all serial interfaces.

---

## Examples

[Example 1-5](#) shows the first screen of output for `show interfaces serial`.

**NOTE** If a BERT is running there will be text indicating this within the output for the affected interface.

Example 1-5 “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
```

---

## show interfaces serial <wanx>

Displays statistics and status information for a serial interface.

---

### Syntax

```
show interfaces serial wanx
```

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>wanx</i>	The name of a serial interface.
-------------	---------------------------------

---

---

### Default

None.

---

### Usage Guidelines

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

---

### Examples

[Example 1-6](#) shows the first screen of output for **show interfaces serial wan1**.

**NOTE** If a BERT is running there will be text indicating this within the output for the affected interface.

Example 1-6 “show interfaces serial wan1”: Displaying serial interface information

---

```
vyatta@R1> show interfaces serial wan1
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
```

---

## show interfaces serial <wanx> bert-status

Displays a message indicating BERT progress for a serial interface.

---

### Syntax

```
show interfaces serial wanx bert-status
```

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>wanx</i>	The name of a serial interface.
-------------	---------------------------------

---

---

### Default

None.

---

### Usage Guidelines

Use this command to view a message indicating whether a BERT is in progress or not and, if so, information about the test. If a BERT is currently not running it prints the date and time of the most recent completion along with information about the test.

---

### Examples

[Example 1-7](#) shows the output for `show interfaces serial wanx bert-status` executed during a BERT.

Example 1-7 “show interfaces serial *wanx* bert-status”

---

```
vyatta@R1> show interfaces serial wan0 bert-status

*****
***          BERT Results          ***
*****
--- BERT test is running ---
--- Tx/Rx:
      Bit Counts          : 112169031
--- Errors:
      Bit Errors          : 0
```

---

```
        Errored Seconds      : 0
        Error Free Seconds   : 74
--- Statistics:
        Lock status          : ** IN-LOCK **
        Available Seconds    : 74
        Bit Error Rate       : 0.000000e+00
        % Error Free Seconds : 100.0%
```

---



## show interfaces serial <wanx> cisco-hdlc

Displays Cisco HDLC information for a serial interface.

---

### Syntax

```
show interfaces serial wanx cisco-hdlc
```

---

### Command Mode

Operational mode.

---

### Parameters

<i>wanx</i>	The name of a serial interface.
-------------	---------------------------------

---

### Default

None.

---

### Usage Guidelines

Use this command to view Cisco HDLC information for a serial interface.

## show interfaces serial <wanx> frame-relay

Displays Frame Relay information for a serial interface.

---

### Syntax

```
show interfaces serial wanx frame-relay
```

---

### Command Mode

Operational mode.

---

### Parameters

<i>wanx</i>	The name of a serial interface.
-------------	---------------------------------

---

### Default

None.

---

### Usage Guidelines

Use this command to view Frame Relay information for a serial interface.

## show interfaces serial <wanx> log

Displays log information for a serial interface.

---

### Syntax

```
show interfaces serial wanx log [tail]
```

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>wanx</i>	The name of a serial interface.
<b>tail</b>	Show log messages as they are added to the log file. Type Ctrl-C to stop the output.

---

---

### Default

None.

---

### Usage Guidelines

Use this command to view log information for a serial interface.

## show interfaces serial <wanx> loopback

Displays loopback information for a serial interface.

---

### Syntax

```
show interfaces serial wanx loopback
```

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>wanx</i>	The name of a serial interface.
-------------	---------------------------------

---

---

### Default

None.

---

### Usage Guidelines

Use this command to view loopback information for a serial interface.

## show interfaces serial <wanx> physical

Displays physical device information for a serial interface.

---

### Syntax

```
show interfaces serial wanx physical
```

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>wanx</i>	The name of a serial interface.
-------------	---------------------------------

---

---

### Default

None.

---

### Usage Guidelines

Use this command to view physical device information for a serial interface.

## show interfaces serial <wanx> ppp

Displays Point-to-Point protocol information for a serial interface.

---

### Syntax

```
show interfaces serial wanx ppp
```

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>wanx</i>	The name of a serial interface.
-------------	---------------------------------

---

---

### Default

None.

---

### Usage Guidelines

Use this command to view Point-to-Point protocol information for a serial interface.

---

### Examples

[Example 1-8](#) shows the output for `show interfaces serial wanx ppp`.

Example 1-8 “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
```

---

## show interfaces serial <wanx> queue

Displays queue information for a serial interface.

---

### Syntax

```
show interfaces serial wanx queue
```

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>wanx</i>	The name of a serial interface.
-------------	---------------------------------

---

---

### Default

None.

---

### Usage Guidelines

Use this command to view queue information for a serial interface.



## show interfaces serial <wanx> trace

Displays trace information for a serial interface.

### Syntax

```
show interfaces serial wanx trace
```

### Command Mode

Operational mode.

### Parameters

<i>wanx</i>	The name of a serial interface.
-------------	---------------------------------

### Default

None.

### Usage Guidelines

Use this command to view a trace of the raw frames incoming from, and outgoing to, the specified serial interface. This trace continues until <Ctrl>-c is pressed.

### Examples

[Example 1-9](#) shows the output for `show interfaces serial wanx trace`.

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

Example 1-9 “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: Testing Serial Lines

This chapter describes the tests available for serial interfaces on the Vyatta system.



*This feature is available only in the Vyatta Subscription Edition.*

This chapter presents the following topics:

- [Serial Line Testing Overview](#)
- [Serial Line Loopbacks](#)
- [Bit Error Rate Tests](#)
- [Serial Line Testing Commands](#)

## Serial Line Testing Overview

---

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.

The Vyatta system offers two mechanisms for isolating problems on serial lines:

- **Loopback tests.** “Loopback” tests use serial loopbacks to provide a basic connectivity test. Loopback tests are run using “loopback” operational commands.
- **Bit error rate tests (BERTs).** BERTs use serial loopbacks and a configured bit stream pattern provide a more extensive test of line quality. Configured BERT parameters can be saved as a named test; subsequently a saved BERT can be run using operational commands.

## Serial Line 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.

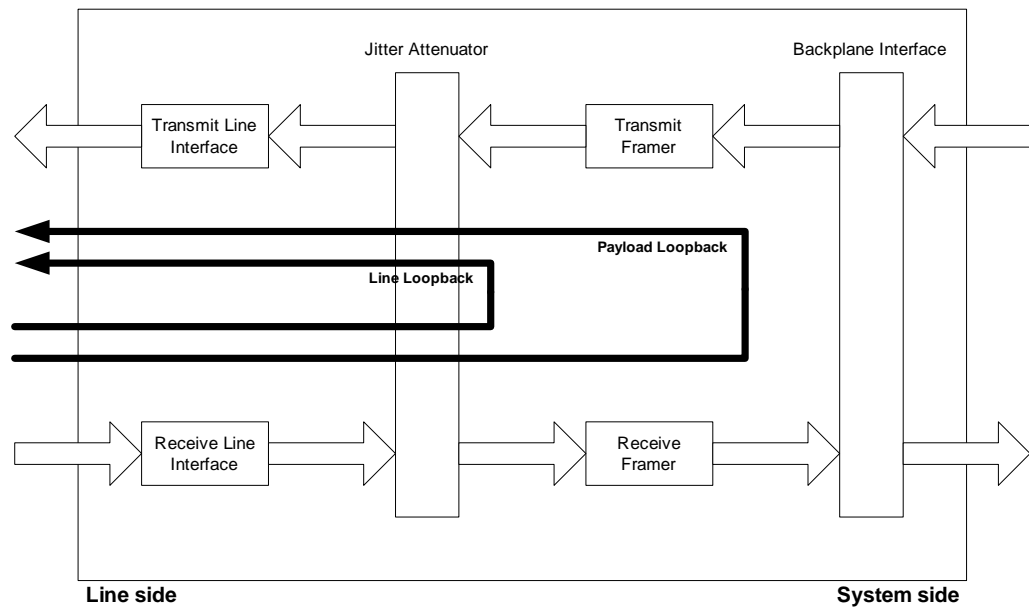
Serial loopbacks are built into the serial card. The exact loopbacks available depend on the chipset of the card. The Vyatta system supports Sangoma serial cards, which may use one of a number of chipsets; the loopback options on the Sangoma card you are using depend on the chipset on the card.

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.

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

[Figure 2-1](#) provides a generalized block diagram of the chipset on a serial card, showing the line-facing loopbacks within the chipset.

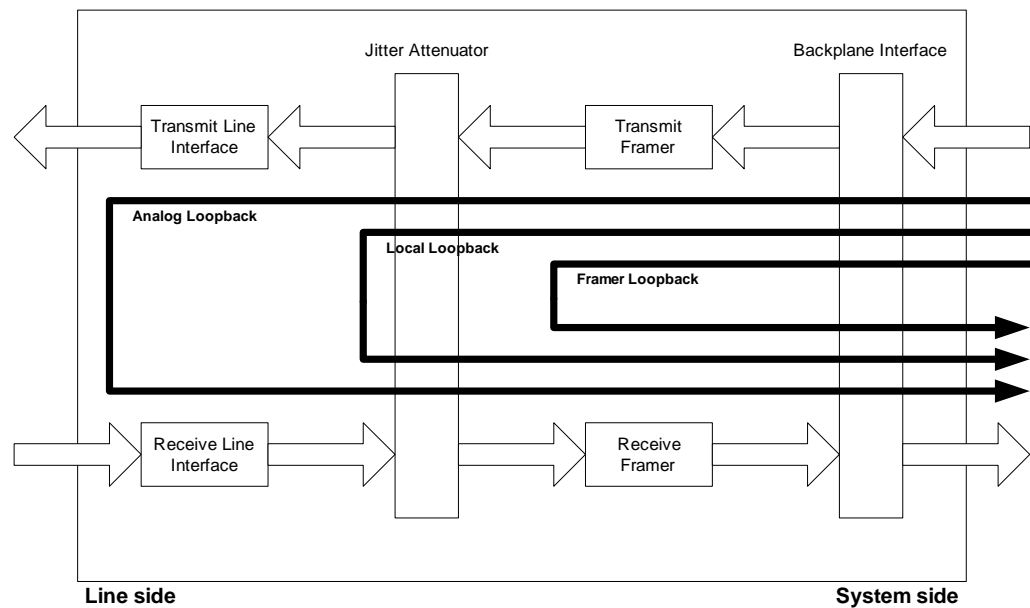
Figure 2-1 Line-facing loopbacks



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

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

Figure 2-2 System-facing loopbacks



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

Table 2-1 summarizes the loopback options available on the various Sangoma T1/E1 and T3/E3 card chipsets. It also shows, in parentheses, the names of the loopbacks used by the chipset manufacturers; these differ, in some cases, from the names used in the Vyatta CLI.

Table 2-1 Loopback options available on Sangoma card chipsets

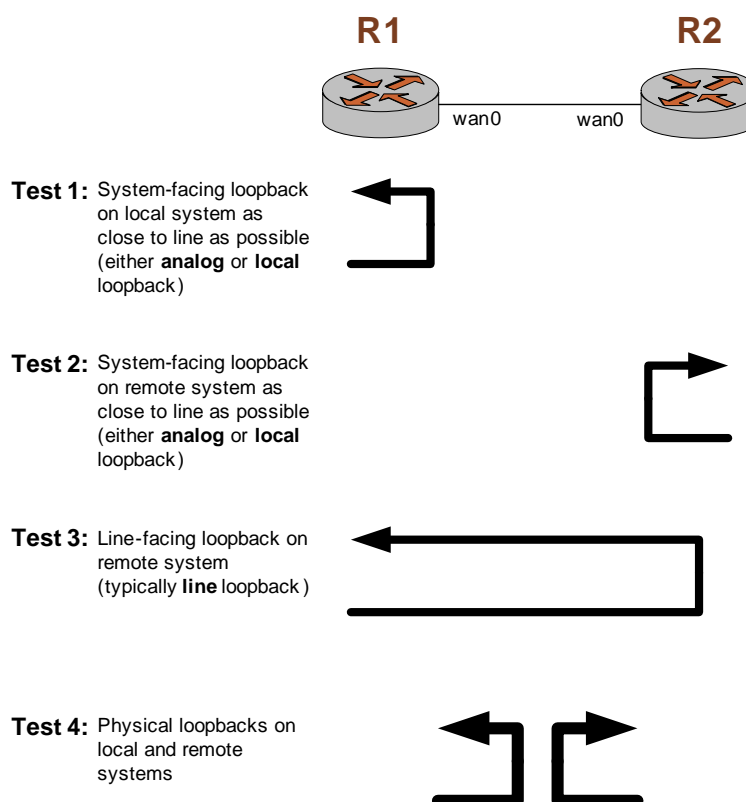
Option	Maxim	PMC-Sierra	Exar
line	Yes (RLB)	Yes (Line)	Yes (Remote)
payload	Yes (PLB)	Yes (Payload)	-
analog	Yes (ALB)	-	Yes (Analog)
local	Yes (LLB)	Yes (Diagnostic)	Yes (Digital)
framer	Yes (FLB)	-	-

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

## Loopback Tests

Figure 2-3 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 2-3 Sample scenario 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 **loopback** operational commands are used. Examples of successful and unsuccessful tests are shown in [test loopback start command](#).

### 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 (that is, a problem on the local WAN interface). To check this, a loopback is defined as close to the line as possible and the loopback test is run on the local system (R1 in the example), as shown in [Example 2-1](#).

## Example 2-1 Testing the local WAN interface

Step	Command
Activate an analog loopback on the local system.	<code>vyatta@R1:~\$ test loopback up wan0 analog</code>
Send test data through the loopback.	<code>vyatta@R1:~\$ test loopback start wan</code>
Deactivate the analog loopback on the local system once the test is complete.	<code>vyatta@R1:~\$ test loopback down wan0 analog</code>

If this test fails, there is a problem on the local WAN interface. If this test succeeds, the problem has not been isolated: proceed to Test 2.

**Test 2: Test the Remote System**

The second step is to determine whether there is a problem on the WAN card in the remote system (that is, on the remote WAN interface). To check this, a loopback is defined as close to the line as possible and the loopback test is run on the remote system (R2 in the example).

## Example 2-2 Testing the remote WAN interface

Step	Command
Activate an analog loopback on the remote system.	<code>vyatta@R2&gt; test loopback up wan0 analog</code>
Send test data through the loopback.	<code>vyatta@R2&gt; test loopback start wan0</code>
Deactivate the analog loopback on the remote system once the test is complete.	<code>vyatta@R2&gt; test loopback down wan0 analog</code>

If this test fails, there is a problem on the remote WAN interface. If this test succeeds, the problem must be in the circuit between the two interfaces: 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 2-3 Testing the circuit

Step	Command
Activate a line-facing loopback on the <b>remote</b> system.	<code>vyatta@R2&gt; test loopback up wan0 line</code>



**Example 2-3** Testing the circuit

Send test data through the remote loopback from the <b>local</b> system.	<code>vyatta@R1:~\$ test loopback start wan0</code>
Deactivate the line loopback on the <b>remote</b> system once the test is complete.	<code>vyatta@R2&gt; test loopback down wan0 line</code>

If this test succeeds, all of the components are working properly. If this test fails, there is a problem on the physical interface of either the local or the remote WAN interface: 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 2-4](#) shows a physical loopback test run on the local system.

**Example 2-4** Testing a physical loopback on the local system

Step	Command
Send test data through the physical loopback on the local system.	<code>vyatta@R1:~\$ test loopback start wan0</code>

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.

## Bit Error Rate Tests

A bit error rate test (BERT) is the standard procedure for troubleshooting T1/E1 and T3/E3 circuits and equipment. A BERT is accomplished by setting up a loopback at some point in the circuit, then sending a bit stream into the circuit and monitoring the returned bit stream. The bit stream returned should be identical to the one sent. A BERT counts the number of bits received in error and calculates a bit error rate (BER) figure.

The bit stream of the BERT involves a stress pattern; different stress patterns exercise the circuit in different ways. For example, the Quasi- Random Signal Source pattern can be used to test jitter (delay) in a circuit, while a pattern of all zeros is effective in discovering Alternate Mark Inversion options (a type of line encoding) have been misconfigured.

The Sangoma T1/E1 family of cards supported by the Vyatta system includes features that allow a BERT to be run directly from the Vyatta system. Although the Sangoma T3/E3 family of cards supports serial loopbacks, it does not provide BERT functionality.

For Sangoma T1/E1 cards, parameters can be defined for a BERT test and saved under a test name. Once defined, the test can be run by referencing the test and the WAN interface on which the test is to be run. Any number of BERTs can be defined. BERT parameters are defined in configuration mode; BERTs are run from operational mode.

**NOTE** For tests that use a hardware loopback plug, be sure to configure the serial interface for internal clock.

**NOTE** A test in which a requested loopup-code fails may cause the serial interface to become non-responsive. This can be fixed by either restarting the wanpipe service or restarting the system. To restart the wanpipe service enter the following command from the command line: **/etc/init.d/vyatta-wan restart**. To restart the system use the **reboot** command.

# Serial Line Testing Commands

This chapter contains the following commands.

Configuration Commands	
<code>test-definition bert &lt;test-name&gt;</code>	Defines a bit error rate test (BERT).
<code>test-definition bert &lt;test-name&gt; duration &lt;duration&gt;</code>	Specifies the length of time for which a BERT is to run.
<code>test-definition bert &lt;test-name&gt; err-insert-rate &lt;rate&gt;</code>	Specifies the rate at which single-bit errors are inserted into the bit stream.
<code>test-definition bert &lt;test-name&gt; loopup-code &lt;code&gt;</code>	Specifies the kind of loop-up code to send to the far end before a BERT.
<code>test-definition bert &lt;test-name&gt; pattern &lt;pattern-name&gt;</code>	Specifies the bit pattern to be transmitted during a BERT.
<code>test-definition bert &lt;test-name&gt; pattern &lt;pattern-name&gt; alternating-word</code>	Allows you to define an alternating and repeating bit pattern for a BERT.
<code>test-definition bert &lt;test-name&gt; pattern &lt;pattern-name&gt; repeating</code>	Allows you to define a repeating bit pattern for a BERT.
Operational Commands	
Serial Loopback	
<code>test loopback down</code>	Deactivates loopbacks on a Sangoma T1/E1 or T3/E3 card.
<code>test loopback start</code>	Starts a loopback test on a Sangoma T1/E1 or T3/E3 card.
<code>test loopback up</code>	Activates a loopback on a Sangoma T1/E1 or T3/E3 card.
BERTs	
<code>test interface &lt;wanx&gt; start-bert &lt;test-name&gt;</code>	Starts a saved bit error rate test.
<code>test interface &lt;wanx&gt; stop-bert</code>	Stops a bit error rate test that is in progress.

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

Related Commands Documented Elsewhere	
Serial Interfaces	Commands for configuring and monitoring serial interfaces, including commands showing the status of BERTs, are described in the “ <a href="#">Chapter 1: Serial Interfaces</a> .”
Firewall	Commands for configuring firewall on serial interfaces are described in the <i>Vyatta Firewall Reference Guide</i> .
OSPF	Commands for configuring the Open Shortest Path First routing protocol on serial interfaces are described in the <i>Vyatta OSPF Reference Guide</i> .
RIP	Commands for configuring the Routing Information Protocol on serial interfaces are described in the <i>Vyatta RIP Reference Guide</i> .
QoS	Commands for configuring quality of service on serial interfaces are described in the <i>Vyatta Policy and QoS Reference Guide</i> .
System interfaces	Commands for showing the physical interfaces available on your system are described in the <i>Vyatta Basic System Reference Guide</i> .
VRRP	Commands for configuring Virtual Router Redundancy Protocol on serial interfaces are described in the <i>Vyatta High Availability Reference Guide</i> .

## test-definition bert <test-name>

Defines a bit error rate test (BERT).

---

### Syntax

```
set test-definition bert test-name
delete test-definition bert test-name
show test-definition bert test-name
```

---

### Command Mode

Configuration mode.

---

### Configuration Statement

```
test-definition {
    bert test-name {
    }
}
```

---

### Parameters

---

<i>test-name</i>	Mandatory. Multi-node. The identifier for the BERT configuration that you are creating.  You can define any number of BERTs by creating multiple <b>test bert</b> configuration nodes.
------------------	--

---

---

### Default

None.

---

### Usage Guidelines

Use this command to define a named bit error rate test (BERT). BERT capability is currently available on Sangoma T1/E1 cards.

The saved test contains configuration parameters including the duration of the test, the bit stream pattern, the error insertion rate, and the method for activating and deactivate the serial loopbacks.

To run the saved BERT, use the [test interface <wanx> start-bert <test-name> command](#).

Use the **set** form of this command to create or modify a BERT.

Use the **delete** form of this command to remove a BERT.

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

## test-definition bert <test-name> duration <duration>

Specifies the length of time for which a BERT is to run.

---

### Syntax

```
set test-definition bert test-name duration duration
delete test-definition bert test-name duration
show test-definition bert test-name duration
```

---

### Command Mode

Configuration mode.

---

### Configuration Statement

```
test-definition {
    bert test-name {
        duration duration
    }
}
```

---

### Parameters

---

<i>test-name</i>	Mandatory. The name of a BERT.
<b>duration</b>	The duration of the BERT, in seconds. The range is 1 to 4294967295. The default is 30.

---

---

### Default

The test duration is 30 seconds.

---

### Usage Guidelines

Use this command to specify the length of time (in seconds) the BERT is to run.

Use the **set** form of this command to set the BERT duration.

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

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

## test-definition bert <test-name> err-insert-rate <rate>

Specifies the rate at which single-bit errors are inserted into the bit stream.

### Syntax

```
set test-definition bert test-name err-insert-rate [10e-1 | 10e-2 | 10e-3 | 10e-4 | 10e-5
| 10e-6 | 10e-7 | none]
delete test-definition bert test-name err-insert-rate
show test-definition bert test-name err-insert-rate
```

### Command Mode

Configuration mode.

### Configuration Statement

```
test-definition {
    bert test-name {
        err-insert-rate rate
    }
}
```

### Parameters

<i>test-name</i>	Mandatory. The name of a BERT.
10e-1	Inserts bit errors at a rate of 10 <sup>1</sup> . This is the equivalent of 1 bit error for every 10 bits transmitted.
10e-2	Inserts bit errors at a rate of 10 <sup>2</sup> . This is the equivalent of 1 bit error for every 100 bits transmitted.
10e-3	Inserts bit errors at a rate of 10 <sup>3</sup> (10 to the power of -3). This is the equivalent of 1 bit error for every 1,000 bits transmitted.
10e-4	Inserts bit errors at a rate of 10 <sup>4</sup> (10 to the power of -4). This is the equivalent of 1 bit error for every 10,000 bits transmitted.
10e-5	Inserts bit errors at a rate of 10 <sup>5</sup> (10 to the power of -5). This is the equivalent of 1 bit error for every 100,000 bits transmitted.
10e-6	Inserts bit errors at a rate of 10 <sup>6</sup> (10 to the power of -6). This is the equivalent of 1 bit error for every 1,000,000 bits transmitted.



---

<b>10e-7</b>	Insert bit errors at a rate of $10^{-7}$ (10 to the power of -7). This is the equivalent of 1 bit error for every 10,000,000 bits transmitted.
<b>none</b>	Do not insert bit errors.

---

---

### Default

No single-bit errors are inserted into the bit stream.

---

### Usage Guidelines

Use this command to specify the rate that single-bit errors are inserted into the bit stream during a bit rate error test (BERT).

Use the **set** form of this command to specify the rate that single-bit errors are inserted into the bit stream.

Use the **delete** form of this command to remove the error insertion rate configuration.

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

## test-definition bert <test-name> loopup-code <code>

Specifies the kind of loop-up code to send to the far end before a BERT.

---

### Syntax

```
set test-definition bert test-name loopup-code [line | payload | none]
delete test-definition bert test-name loopup-code
show test-definition bert test-name loopup-code
```

---

### Command Mode

Configuration mode.

---

### Configuration Statement

```
test-definition {
    bert test-name {
        loopup-code code
    }
}
```

---

### Parameters

<i>test-name</i>	Mandatory. The name of a BERT.
<b>line</b>	Sends a loop-up code activating a “line” loopback.
<b>payload</b>	Sends a loop-up code activating a “payload” loopback.
<b>none</b>	Does not send a loop-up code. This is the default.

---

### Default

No loop-up code is sent to the far end.

---

### Usage Guidelines

Use this command to specify the loop-up code that is sent to equipment at the far end of a circuit.

The loop-up code is sent to the far end before a BERT begins. The loop-up code instructs the far end to activate a loopback and what kind of loopback to activate.

When the system is configured to send a loop-up code, a corresponding loop-down code is sent to the far end at the end of the test to instruct it to deactivate the loopback.

Use the **set** form of this command to specify the loop-up code to be sent to the far end before a BERT begins.

Use the **delete** form of this command to restore default loop-up code configuration.

Use the **show** form of this command to view loop-up code configuration.

## test-definition bert <test-name> pattern <pattern-name>

Specifies the bit pattern to be transmitted during a BERT.

### Syntax

```
set test-definition bert test-name pattern [3-in-24 | all-ones | all-zeros | daly |
pseudo-2e7 | pseudo-2e9 | pseudo-2e11 | pseudo-2e15 | qrss]
```

```
delete test-definition bert test-name pattern
```

```
show test-definition bert test-name pattern
```

### Command Mode

Configuration mode.

### Configuration Statement

```
test-definition {
    bert test-name {
        pattern pattern name
    }
}
```

### Parameters

<i>test-name</i>	Mandatory. The name of a BERT.
<b>3-in-24</b>	Generates a repeating 24-bit pattern of 3 one-bits and 21 zero-bits.
<b>all-ones</b>	Generates an all-ones bit pattern.
<b>all-zeros</b>	Generates an all-zeros bit pattern.
<b>daly</b>	Generates a repeating pattern of 55 octets in a sequence that changes rapidly between ones and zeros.
<b>pseudo-2e7</b>	Generates a pseudo-random bit-pattern 2 <sup>7</sup> bits in length.
<b>pseudo-2e9</b>	Generates a pseudo-random bit-pattern 2 <sup>9</sup> bits in length.
<b>pseudo-2e11</b>	Generates a pseudo-random bit-pattern 2 <sup>11</sup> bits in length.
<b>pseudo-2e15</b>	Generates a pseudo-random bit-pattern 2 <sup>15</sup> bits in length.

---

<b>qrss</b>	Generates a quasi-random signal source (QRSS) pattern, which is a pseudorandom binary sequence. This sequence produces every combination of 20-bit words, repeats every 1,048,575 bits, and suppresses consecutive zeros to no more than 14.
-------------	--

---

---

### Default

A QRSS pattern is used.

---

### Usage Guidelines

Use this command to specify the bit pattern to be transmitted during a bit rate error test (BERT).

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

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

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

## test-definition bert <test-name> pattern <pattern-name> alternating-word

Allows you to define an alternating and repeating bit pattern for a BERT.

### Syntax

```
set test-definition bert test-name pattern alternating-word [first-word pattern1 |
second-word pattern2 | repeat-count count]
```

```
delete test-definition bert test-name pattern alternating-word [first-word |
second-word | repeat-count]
```

```
show test-definition bert test-name pattern alternating-word [first-word |
second-word | repeat-count]
```

### Command Mode

Configuration mode.

### Configuration Statement

```
test-definition {
  bert test-name {
    pattern pattern name {
      alternating-word {
        first-word pattern1
        second-word pattern2
        repeat-count count
      }
    }
  }
}
```

### Parameters

<i>test-name</i>	Mandatory. The name of a BERT.
<i>pattern1</i>	The first 16-bit word of the alternating pattern, specified in hexadecimal without a leading “0x”; for example, “2ea7.”
<i>pattern2</i>	The second 16-bit word of the alternating pattern, specified in hexadecimal without a leading “0x”; for example, “2ea7.”

---

<i>count</i>	The number of times each word is to be transmitted before sending the other word. For example, a repeat count of 1 means that the first word is sent once, then the second word is sent once, then the process is repeated. A repeat count of 2 means that the first word is sent twice, then the other word is sent twice, then the process is repeated. The range is 1 to 256. The default is 1.
--------------	--

---

---

### Default

When this option is not set, alternating word patterns are not used.

---

### Usage Guidelines

Use this command to define an alternating and repeating pattern of 16-bit words to be used as a bit pattern in a bit rate error test (BERT).

If an alternating word pattern is not specified, the BERT uses the pattern specified by the [test-definition bert <test-name> pattern <pattern-name> command](#).

Use the **set** form of this command to define an alternating word pattern.

Use the **delete** form of this command to remove alternating word pattern configuration.

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

## test-definition bert <test-name> pattern <pattern-name> repeating

Allows you to define a repeating bit pattern for a BERT.

### Syntax

```
set test-definition bert test-name pattern repeating [word pattern | length-in-bits length]
```

```
delete test-definition bert test-name pattern repeating [word | length-in-bits]
```

```
show test-definition bert test-name pattern repeating [word | length-in-bits]
```

### Command Mode

Configuration mode.

### Configuration Statement

```
test-definition {
    bert test-name {
        pattern pattern name {
            repeating {
                word pattern
                length-in-bits length
            }
        }
    }
}
```

### Parameters

<i>test-name</i>	Mandatory. The name of a BERT.
word <i>pattern</i>	A 32-bit pattern, specified in hexadecimal without a leading “0x”; for example, “2ea732a5.” If fewer than 8 hexadecimal digits are provided, the high-order (left-most) digits are padded with zeros.
length-in-bits <i>length</i>	The length of the bit pattern. This number of low-order bits from the specified pattern is repeated. The range is 1 to 32. The default is 32.



---

**Default**

None.

---

**Usage Guidelines**

Use this command to define a repeating pattern of up to 32 bits to be used as the bit pattern in a BERT.

Use the **set** form of this command to create or modify a repeating bit pattern.

Use the **delete** form of this command to remove repeating bit pattern.

Use the **show** form of this command to view repeating bit pattern.

## test interface <wanx> start-bert <test-name>

Starts a saved bit error rate test.

---

### Syntax

```
test interface wanx start-bert test-name [wait | watch]
```

---

### Command Mode

Operational mode.

---

### Parameters

<i>wanx</i>	The name of the physical-layer T1 serial interface on which the BERT is to run.
<i>test-name</i>	The name of the saved BERT to use.
<b>wait</b>	Directs the system to retain control until test completion and not to generate output during the test.
<b>watch</b>	Directs the system to retain control until test completion but generate periodic messages.

---

### Default

Control returns immediately to the user and the test proceeds in the background.

---

### Usage Guidelines

Use this command to start a bit error rate test (BERT) using the set of configuration parameters saved under the specified test name.

If the **wait** keyword is used, control is not returned to the user until after the test completes. In this case, no output is generated during the test.

If the **watch** keyword is used, control is still not returned to the user until after the test completes; however, the system prints periodic status messages about the test.

Regardless of whether keywords are used or not, the test can be terminated early by entering <Ctrl>+c or by issuing `test interface <wanx> stop-bert command`.

---

## Examples

[Example 2-5](#) shows the system messages generated when the TEST1 BERT is run on wan0.

### Example 2-5 Starting a BERT

---

```
vyatta@R1> test interface wan0 start-bert TEST1
```

```
Execute BERT Start command... Done!
```

```
Running test in background for 120 seconds...
```

```
vyatta@R1>
```

---

## test interface <wanx> stop-bert

Stops a bit error rate test that is in progress.

---

### Syntax

```
test interface wanx stop-bert
```

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>wanx</i>	The name of the physical-layer T1 serial interface on which the BERT is being run.
-------------	--

---

---

### Default

None.

---

### Usage Guidelines

Use this command to stop a bit error rate test (BERT) on the specified WAN interface.

If no test is in progress the system displays a message to that effect.

---

### Examples

[Example 2-6](#) shows the system messages generated when the BERT is running on wan0 is stopped.

#### Example 2-6 Stopping a BERT

---

```
vyatta@R1> test interface wan0 stop-bert
Stopping BERT test running on interface wan0
Killing BERT test monitor process (pid 15339) ...Done.
vyatta@R1>
```

---

## test loopback down

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

---

### Syntax

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

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>wanx</i>	The name of a serial interface.
<b>all</b>	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.
<b>analog</b>	Deactivates the Line Interface Unit analog loopback.
<b>framer</b>	Deactivates the framer loopback.
<b>line</b>	Deactivates the Line Interface Unit line loopback.
<b>local</b>	Deactivates the Line Interface Unit local loopback.
<b>payload</b>	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. Each command may take several seconds to complete.

Note that these tests may generate messages indicating a successful outcome (for example, “loopback XXX deactivated”) even if the loopback in question was not active when the command was run.

Note that when CLI completion is used for this command, the system displays only the loopbacks supported by the chipset on the card used. The **all** option is supported by all chipsets.

---

## Examples

[Example 2-7](#) shows the first screen of output for **test loopback down wanx local**.

Example 2-7 Deactivating a local loopback on wan0

---

```
vyatta@R1> test loopback down wan0 local

Diagnostic Digital Loopback mode is deactivated!

vyatta@R1>
```

---

## test loopback start

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

---

### Syntax

```
test loopback start wanx
```

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>wanx</i>	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.

**NOTE** For *local* and *framer* loopbacks, make sure the interface clocking is set to **internal** or the line will not come up due to lack of received clocking and the test will fail.

**NOTE** It may take up to 60 seconds after issuing a **test loopback up** command before the **test loopback start** command can be run successfully.

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 the test should succeed if one of the system-facing loopbacks on the local WAN interface is active.

---

### Examples

[Example 2-8](#) shows the first screen of results for a successful loopback test on wan0.

Example 2-8 Successful test of a loopback

---

```
vyatta@R1> test loopback start 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 2-9](#) shows the first screen of results for an unsuccessful loopback test on wan0.

#### Example 2-9 Unsuccessful loopback test

---

```
vyatta@R1> test loopback start 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>
```

---



## test loopback up

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

---

### Syntax

```
test loopback up wanx {analog | framer | line | local | payload}
```

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>wanx</i>	The name of a serial interface.
<b>analog</b>	Activate the Line Interface Unit analog loopback. This loopback 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.
<b>framer</b>	Activate the framer loopback. This loopback 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.
<b>line</b>	Activate the Line Interface Unit line loopback. This loopback 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.
<b>local</b>	Activate the Line Interface Unit local loopback. This loopback 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.
<b>payload</b>	Activate the payload loopback. This loopback 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 serial loopback on a Sangoma T1/E1 or T3/E3 card. More than one loopback can be active at a time.

Once activated, a loopback remains active until it is deactivated by using a **test 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 [monitor interfaces serial <wanx> traffic command](#).

---

### Examples

[Example 2-10](#) shows the system response to activating a local loopback.

Example 2-10 Activating a local loopback

---

```
vyatta@R1> test loopback up wan0 local

Diagnostic Digital Loopback mode is activated!

vyatta@R1>
```

---

## Chapter 3: 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 feature is available only in the Vyatta Subscription Edition.*

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

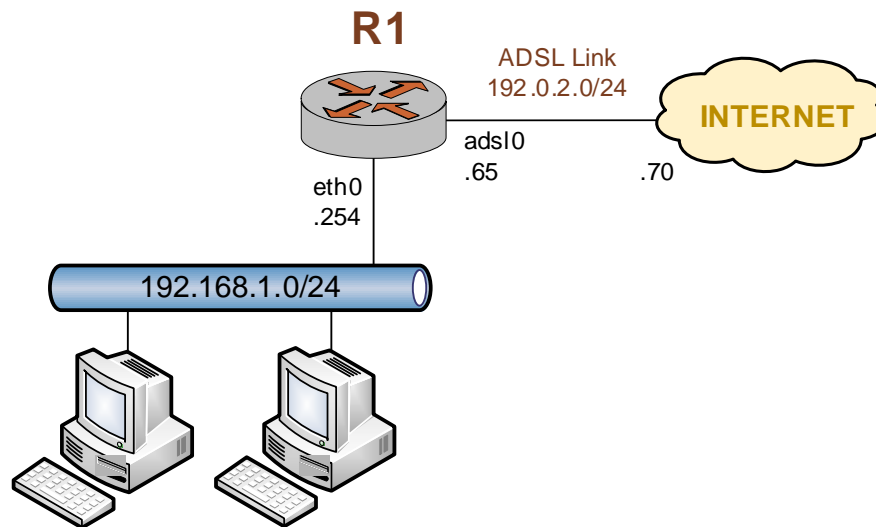
DSL (Digital Subscriber Loop) is a service that utilizes intelligent modulation techniques to convert traditional phone lines—Plain Old Telephone System (POTS) or Integrated Services Digital Network (ISDN)—into high-bandwidth conduits used for Internet access. Like dial-up, cable, wireless, and T1/E1, 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 3-1](#) shows a typical ADSL configuration as an access protocol between a customer premises 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 3-1 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 3-1** 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. 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 as this interface.
- The user id is set to “customerA”.
- The password is set to “Aremotsuc”.

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

*Tip: Where public IP addresses would normally be used, the example uses RFC 5735 “TEST-NET” IP addresses (192.0.2.0/24)*

## Example 3-1 Creating and configuring an ADSL interface for PPPoE encapsulation

Step	Command
Specify that the system should auto-detect an identifier for the pvc.	<pre>vyatta@R1# set interfaces adsl adsl0 pvc auto</pre>
Set the line encapsulation to PPPoE using unit number 0.	<pre>vyatta@R1# set interfaces adsl adsl0 pvc auto pppoe 0</pre>
Assign the local IP address to the interface.	<pre>vyatta@R1# set interfaces adsl adsl0 pvc auto pppoe 0 local-address 192.0.2.65</pre>
Set the IP address of the far end of the connection.	<pre>vyatta@R1# set interfaces adsl adsl0 pvc auto pppoe 0 remote-address 192.0.2.70</pre>
Set the user id for the link.	<pre>vyatta@R1# set interfaces adsl adsl0 pvc auto pppoe 0 user-id customerA</pre>
Set the password for the link.	<pre>vyatta@R1# set interfaces adsl adsl0 pvc auto pppoe 0 password Arehotsuc</pre>
Commit the configuration.	<pre>vyatta@R1# commit</pre>
View the configuration.	<pre>vyatta@R1# show interfaces adsl adsl0     pvc auto {         pppoe 0 {             local-address 192.0.2.65             remote-address 192.0.2.70             user-id customerA             password Arehotsuc         }     } vyatta@R1#</pre>

# DSL Commands

This chapter contains the following commands.

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

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

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



## 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

<i>adslx</i>	Mandatory. Multi-node. The identifier for the ADSL interface you are defining. This may be <b>adsl0</b> to <b>adslx</b> , 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 pvc-id {
    }
  }
}
```

---

### Parameters

---

<i>adslx</i>	Mandatory. Multi-node. The identifier for the ADSL interface you are defining. This may be <b>adsl0</b> to <b>adslx</b> , depending on what physical ADSL ports are actually available on the system.
<i>pvc-id</i>	Mandatory. The identifier for the PVC. It can either be the <i>vpi/vci</i> pair or the keyword <b>auto</b> , where <i>vpi</i> is a Virtual Path Index from 0 to 255, <i>vci</i> is a Virtual Circuit Index from from 0 to 65535, and <b>auto</b> directs the system to detect the Virtual Path Index and Virtual Circuit Index automatically.  Note that the <b>auto</b> option may not be able to detect VPI/VCI in all cases. Because of this the <b>auto</b> option should only be used in cases where VPI/VCI are not known.

---

---

### 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 state
  }
}
```

---

### Parameters

<i>adslx</i>	Mandatory. Multi-node. The identifier for the ADSL interface you are defining. This may be <b>adsl0</b> to <b>adslx</b> , depending on what physical ADSL ports are actually available on the system.
<i>state</i>	Optional. Specifies whether the watchdog feature is to be used. Supported values are as follows:  <b>enable:</b> The interface sends watchdog requests every 10 seconds.  <b>disable:</b> 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.

## monitor interfaces adsl <if-name> traffic

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

---

### Syntax

```
monitor interfaces adsl if-name traffic
```

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>if-name</i>	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 <b>pppoax</b> , <b>pppoex</b> , or <b>adslx</b> .
----------------	--

---

---

### 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>

Displays status of an ADSL interface.

---

### Syntax

```
show interfaces adsl if-name
```

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>if-name</i>	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 <b>pppoax</b> , <b>pppoex</b> , or <b>adslx</b> .
----------------	--

---

---

### 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> queue

Displays queue information on an ADSL interface.

---

### Syntax

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

---

### Command Mode

Operational mode.

---

### Parameters

<i>if-name</i>	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 <b>pppoax</b> , <b>pppoex</b> , or <b>adslx</b> .
<b>class</b>	Optional. Displays the queue classes for a device.
<b>filter</b>	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

---

<i>if-name</i>	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 <b>pppoax</b> , <b>pppoex</b> , or <b>adslx</b> .
----------------	--

---

---

### 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 4: Wireless Modem Interfaces

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



*This feature is available only in the Vyatta Subscription Edition.*

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 4-1 Sierra Wireless USB Connect 881 modem accessing the AT&T network

Step	Command
Specify a wirelessmodem configuration node.	<code>vyatta@R1# set interfaces wirelessmodem wlm0</code>
Commit the change	<code>vyatta@R1# commit</code>
Show the configuration.	<code>vyatta@R1# show interfaces wirelessmodem wlm0 { }</code>

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 4-2 UT Starcom 3G modem accessing the Verizon network

Step	Command
Specify the Verizon chat script.	<code>vyatta@R1# set interfaces wirelessmodem wlm0 network verizon</code>
Specify the system device for the UT Starcom modem.	<code>vyatta@R1# set interfaces wirelessmodem wlm0 device ttyACM0</code>
Commit the change	<code>vyatta@R1# commit</code>
Show the configuration.	<code>vyatta@R1# show interfaces wirelessmodem wlm0 { network verizon device ttyACM0 }</code>

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 *wlm<sub>x</sub>*** command in operational mode as follows:

Example 4-3 Disconnecting from the network

---

<b>Step</b>	<b>Command</b>
Disconnecting from the network.	<code>vyatta@R1:~\$ disconnect interface wlm0</code>

---

Once disconnected the **connect interface *wlm<sub>x</sub>*** command can be used in operational mode to reconnect to the network as follows:

Example 4-4 Connecting to the network

---

<b>Step</b>	<b>Command</b>
Connect to the network.	<code>vyatta@R1:~\$ connect interface wlm0</code>

---

# Wireless Modem Interface Commands

This chapter contains the following commands.

Configuration Commands	
<code>interfaces wirelessmodem &lt;wlmx&gt;</code>	Defines a wirelessmodem interface.
<code>interfaces wirelessmodem &lt;wlmx&gt; backup</code>	Specifies that a backup default route will be installed in the routing table.
<code>interfaces wirelessmodem &lt;wlmx&gt; description &lt;desc&gt;</code>	Specifies a description for a wirelessmodem interface.
<code>interfaces wirelessmodem &lt;wlmx&gt; device &lt;device&gt;</code>	Specifies the system device for the wireless modem.
<code>interfaces wirelessmodem &lt;wlmx&gt; ipv6 address</code>	Assigns an IPv6 address to a wireless modem interface.
<code>interfaces wirelessmodem &lt;wlmx&gt; ipv6 disable-forwarding</code>	Disables IPv6 forwarding on a wirelessmodem interface.
<code>interfaces wirelessmodem &lt;wlmx&gt; ipv6 dup-addr-detect-transmits &lt;num&gt;</code>	Specifies the number of times to transmit NS packets as part of the DAD process.
<code>interfaces wirelessmodem &lt;wlmx&gt; ipv6 router-advert</code>	Specifies the router advertisements to be sent from a wirelessmodem interface.
<code>interfaces wirelessmodem &lt;wlmx&gt; mtu &lt;mtu&gt;</code>	Specifies the Maximum Transmit Unit (MTU) size for a wirelessmodem interface.
<code>interfaces wirelessmodem &lt;wlmx&gt; network &lt;scriptfile&gt;</code>	Specifies the chat script to use on a wireless modem interface.
<code>interfaces wirelessmodem &lt;wlmx&gt; no-dns</code>	Specifies that the network provider's DNS host should not be added to the local name resolution path.
<code>interfaces wirelessmodem &lt;wlmx&gt; ondemand</code>	Specifies that a connection will be re-established only when traffic is to be sent.
Operational Commands	
<code>clear interfaces connection &lt;wlmx&gt;</code>	Resets a PPP session on a wirelessmodem interface.
<code>connect interface &lt;wlmx&gt;</code>	Brings a wirelessmodem interface up.
<code>disconnect interface &lt;wlmx&gt;</code>	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 <i>Vyatta Security Reference Guide</i> .
----------	---

---

show system usb	.Displays information about peripherals connected to the USB bus. This command is described in the <i>Vyatta Basic System Reference Guide</i> .
-----------------	---

---

## clear interfaces connection <wlmx>

Resets a PPP session on a wirelessmodem interface.

---

### Syntax

```
clear interfaces connection wlmx
```

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>wlmx</i>	Mandatory. The interface to be operationally brought down and then up. The range is <b>wlm0</b> to <b>wlm999</b> .
-------------	--

---

---

### 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

---

<i>wlmx</i>	Mandatory. The interface to be operationally brought up. The range is <b>wlm0</b> to <b>wlm999</b> .
-------------	--

---

---

### 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

---

<i>wlmx</i>	Mandatory. The interface to be operationally brought down. The range is <b>wlm0</b> to <b>wlm999</b> .
-------------	--

---

---

### 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 wlmx {
    }
}
```

---

### Parameters

<i>wlmx</i>	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are defining. This may be <b>wlm0</b> to <b>wlm999</b> .
-------------	--

---

### 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 wlmx {  
        backup {  
            distance distance  
        }  
    }  
}
```

---

### Parameters

<i>wlmx</i>	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are using. This may be <b>wlm0</b> to <b>wlm999</b> .
<i>distance</i>	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 wlmx {
        description desc
    }
}
```

---

### Parameters

<i>wlmx</i>	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are using. This may be <b>wlm0</b> to <b>wlm999</b> .
<i>desc</i>	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 wlmx {  
        device device  
    }  
}
```

---

### Parameters

<i>wlmx</i>	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are using. This may be <b>wlm0</b> to <b>wlm999</b> .
<i>device</i>	Optional. The system device used for the wireless modem. 3G modems typically use either <b>ttyUSBx</b> or <b>ttyACMx</b> . The default is <b>ttyUSB0</b> .

---

### 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> ipv6 address

Assigns an IPv6 address to a wireless modem interface.

### Syntax

```
set interfaces wirelessmodem wlmx ipv6 address [autoconf | eui64 ipv6prefix]
delete interfaces ethernet ethx ipv6 address [autoconf | eui64 ipv6prefix]
show interfaces ethernet ethx ipv6 address [autoconf | eui64]
```

### Command Mode

Configuration mode.

### Configuration Statement

```
interfaces {
  wirelessmodem wlmx {
    ipv6 {
      address {
        autoconf
        eui64 ipv6prefix
      }
    }
  }
}
```

### Parameters

<i>wlmx</i>	The interface identifier. Supported values are <b>wlm0</b> through <b>wlm23</b> .
<b>autoconf</b>	Generates an IPv6 address using the SLAAC protocol. Set this value if the interface is performing a “host” function rather than a “router” function. This value can be specified in addition to specifying static IPv6, static IPv4, or IPv4 DHCP addresses on the interface.
<i>ipv6prefix</i>	The 64-bit IPv6 address prefix used to configure an IPv6 address, in EUI-64 format. The system concatenates this prefix with a 64-bit EUI-64 value derived from the 48-bit MAC address of the interface.

---

### Default

None.

---

### Usage Guidelines

Use this command to assign an IPv6 address to a wireless modem interface.

You can use the **autoconf** keyword to direct the system to autoconfigure the address, using the SLAAC (Stateless Address Auto-Configuration) protocol defined in RFC 4862. Alternatively, you can provide an EUI-64 IPv6 address prefix so that the system constructs the IPv6 address.

If you want the system to use SLAAC to acquire addresses on this interface, then in addition setting this parameter, you must also disable IPv6 forwarding, either globally (using the [system ipv6 disable-forwarding](#) command) or specifically on this interface (using the [interfaces <interface> ipv6 disable-forwarding](#) command).

Use the **set** form of this command to specify an IPv6 address for the interface.

Use the **delete** form of this command to delete an IPv6 address from the interface.

Use the **show** form of this command to view IPv6 address configuration settings.

## interfaces wirelessmodem <wlmx> ipv6 disable-forwarding

Disables IPv6 forwarding on a wirelessmodem interface.

---

### Syntax

```
set interfaces wirelessmodem wlmx ipv6 disable-forwarding
delete interfaces wirelessmodem wlmx ipv6 disable-forwarding
show interfaces wirelessmodem wlmx ipv6 disable-forwarding
```

---

### Command Mode

Configuration mode.

---

### Configuration Statement

```
interfaces {
  wirelessmodem wlmx {
    ipv6 {
      disable-forwarding
    }
  }
}
```

---

### Parameters

<i>wlmx</i>	The interface identifier. Supported values are <b>wlm0</b> through <b>wlm23</b> .
-------------	---

---

### Default

IPv6 packets are forwarded.

---

### Usage Guidelines

Use this command to disable IPv6 packet forwarding on a wireless modem interface. You can also disable IPv6 forwarding globally (that is, for all interfaces) using the [system ipv6 disable-forwarding](#) command.

Use the **set** form of this command to disable IPv6 packet forwarding on an interface.

Use the **delete** form of this command to enable IPv6 packet forwarding on an interface.

Use the **show** form of this command to display IPv6 packet forwarding interface configuration.

## interfaces wirelessmodem <wlmx> ipv6 dup-addr-detect-transmits <num>

Specifies the number of times to transmit NS packets as part of the DAD process.

---

### Syntax

```
set interfaces wirelessmodem wlmx ipv6 dup-addr-detect-transmits num
delete interfaces wirelessmodem wlmx ipv6 dup-addr-detect-transmits
show interfaces wirelessmodem wlmx ipv6 dup-addr-detect-transmits
```

---

### Command Mode

Configuration mode.

---

### Configuration Statement

```
interfaces {
  wirelessmodem wLmx {
    ipv6 {
      dup-addr-detect-transmits num
    }
  }
}
```

---

### Parameters

<i>wlmx</i>	The interface identifier. Supported values are <b>wlm0</b> through <b>wlm23</b> .
<i>num</i>	The number of times to transmit NS packets as part of the DAD process. The default is 1.

---

### Default

One NS packet is transmitted as part of the DAD process.

---

### Usage Guidelines

Use this command to specify the number of times to transmit Neighbor Solicitation (NS) packets as part of the Duplicate Address Detection (DAD) process.

Use the **set** form of this command to specify the number of times to transmit Neighbor Solicitation (NS) packets as part of the Duplicate Address Detection (DAD) process.

Use the **delete** form of this command to delete the parameter from the interface and use the default value.

Use the **show** form of this command to view NS packet configuration for DAD.

## interfaces wirelessmodem <wlmx> ipv6 router-advert

Specifies the router advertisements to be sent from a wirelessmodem interface.

### Syntax

```
set interfaces wirelessmodem wlmx ipv6 router-advert [cur-hop-limit limit]
[default-lifetime lifetime] [default-preference preference] [link-mtu mtu]
[managed-flag state] [max-interval interval] [min-interval interval]
[other-config-flag state] [prefix ipv6net [autonomous-flag state | on-link-flag state
| preferred-lifetime lifetime | valid-lifetime lifetime]] [reachable-time time]
[retrans-timer time] [send-advert state]
```

```
delete interfaces wirelessmodem wlmx ipv6 router-advert [cur-hop-limit]
[default-lifetime] [default-preference] [link-mtu] [managed-flag] [max-interval]
[min-interval] [other-config-flag] [prefix ipv6net [autonomous-flag | on-link-flag
| preferred-lifetime | valid-lifetime ]] [reachable-time ] [retrans-timer [send-advert]
```

```
show interfaces wirelessmodem wlmx ipv6 router-advert
```

### Command Mode

Configuration mode.

### Configuration Statement

```
interfaces {
  wirelessmodem wlmx {
    ipv6 {
      router-advert {
        cur-hop-limit limit
        default-lifetime lifetime
        default-preference preference
        link-mtu mtu
        managed-flag state
        max-interval interval
        min-interval interval
        other-config-flag state
        prefix ipv6net {
          autonomous-flag state
          on-link-flag state
          preferred-lifetime lifetime
          valid-lifetime lifetime
        }
        reachable-time time
      }
    }
  }
}
```



```

        retrans-timer time
        send-advert state
    }
}
}
}

```

## Parameters

<i>wlmx</i>	The interface identifier. Supported values are <b>wlm0</b> through <b>wlm23</b> .
<b>cur-hop-limit</b> <i>limit</i>	Specifies the Hop Count field of the IP header for outgoing (unicast) IP packets. This value is placed in the Hop Count field of the IP header for outgoing (unicast) IP packets. The range is 0 to 255. The default is 64. A value of 0 means unspecified by the router.
<b>default-lifetime</b> <i>lifetime</i>	Specifies the lifetime, in seconds, associated with the default router. Supported values are 0, which indicates that the router is not a default router, and the range from the value configured for the <b>max-interval</b> option to 9000 (18.2 hours). If not configured, the value for this timer is three times <b>max-interval</b> .
<b>default-preference</b> <i>preference</i>	The preference associated with the default router. Supported values are as follows: <b>low</b> : The default router is low preference. <b>medium</b> : The default router is medium preference. <b>high</b> : The default router is high preference. The default is <b>medium</b> .
<b>link-mtu</b> <i>mtu</i>	The MTU value to be advertised for the link. The range of values is 0, or 1280 to the maximum MTU for the type of link, as defined in RFC 2464. The default is 0, which means the MTU is not specified in the router advertisement message. That is because it is expected that the MTU will be configured directly on the interface itself and not for routing advertisements. You can configure this option in cases where the link MTU is not well known.  If the value set here does not match the MTU configured on the interface, the system issues a warning but does not fail.

<b>managed-flag</b> <i>state</i>	Whether to use the administered protocol for address autoconfiguration. Supported values are as follows: <b>true:</b> Hosts use the administered (stateful) protocol for address autoconfiguration in addition to any addresses autoconfigured using stateless address autoconfiguration. <b>false:</b> Hosts use only stateless address autoconfiguration. The default is <b>false</b> .
<b>max-interval</b> <i>interval</i>	The maximum time, in seconds, allowed between sending unsolicited multicast router advertisements from the interface. The range of supported values is 4 to 1800. The default is 600 (10 minutes).
<b>min-interval</b> <i>interval</i>	The minimum time, in seconds, allowed between sending unsolicited multicast router advertisements from the interface. The range of supported values is 3 to $0.75 * \text{max-interval}$ . The default is $0.33 * \text{max-interval}$ .
<b>other-config-flag</b> <i>state</i>	The interfaces uses the administered (stateful) protocol for autoconfiguration of non-address information, as defined in RFC 4862. Supported values are as follows: <b>true:</b> Hosts use the administered protocol for autoconfiguration of non-address information. <b>false:</b> Hosts use stateless autoconfiguration of non-address information. The default is <b>false</b> .
<b>prefix</b> <i>ipv6net</i>	Multi-node. The IPv6 prefix to be advertised on the IPv6 interface, in the format <i>ipv6-address/prefix</i> .  You can define more than one IPv6 prefix by configuring multiple <b>prefix</b> configuration nodes.
<b>autonomous-flag</b> <i>state</i>	Specifies whether the prefix can be used for autonomous address configuration as defined in RFC 4862. Supported values are as follows: <b>true:</b> The prefix can be used for autonomous address configuration. <b>false:</b> The prefix cannot be used for autonomous address configuration. The default is <b>true</b> .

---

<b>on-link-flag</b> <i>state</i>	Specifies whether the prefix can be used for on-link determination, as defined in RFC 4862. Supported values are as follows: <b>true:</b> The prefix can be used for on-link determination. <b>false:</b> The advertisement makes no statement about on-link or off-link properties of the prefix. For instance, the prefix might be used for address configuration with some addresses belonging to the prefix being on-link and others being off-link. The default is <b>true</b> .
<b>preferred-lifetime</b> <i>lifetime</i>	The length of time, in seconds, that the addresses generated from the prefix via stateless address autoconfiguration (SLAAC) is to remain preferred, as defined in RFC 4862. The interval is with respect to the time the packet is sent. The range is 1 to 4294967296 plus the keyword <b>infinity</b> , which represents forever. (The actual value of <b>infinity</b> is a byte where all bits are set to ones: 0xFFFFFFFF.) The default is 604800 (seven days).
<b>valid-lifetime</b> <i>lifetime</i>	The length of time, in seconds, that the prefix is valid for the purpose of on-link determination, as defined in RFC 4862. The interval is with respect to the time the packet is sent. The range is 1 to 4294967296 plus the keyword <b>infinity</b> , which represents forever. (The actual value of <b>infinity</b> is a byte where all bits are set to ones: 0xFFFFFFFF.) The default is 2592000 (30 days).
<b>reachable-time</b> <i>time</i>	The length of time, in milliseconds, for which the system assumes a neighbor is reachable after having received a reachability confirmation. This value is used by address resolution and the Neighbor Unreachability Detection algorithm (see Section 7.3 of RFC 2461). The range is 0 to 3600000, where a value of 0 means the reachable time is not specified in the router advertisement message. The default is 0.
<b>retrans-timer</b> <i>time</i>	The length of time, in milliseconds, between retransmitted NS messages. This value is used by address resolution and the Neighbor Unreachability Detection algorithm (see Sections 7.2 and 7.3 of RFC 2461). The range of supported values is 0 to 4294967295, where a value of 0 means the retransmit time is not specified in the router advertisement message. The default is 0.

---

---

<b>send-advert</b> <i>state</i>	Specifies whether router advertisements are to be sent from this interface. Supported values are as follows: <b>true</b> : Sends router advertisements from this interface. <b>false</b> : Does not send router advertisements from this interface. If this value is in effect, parameters in this configuration subtree are still used to configure the local implementation parameters. The default is <b>true</b> .
---------------------------------	---

---

---

### Default

Router advertisements are not sent on an interface.

---

### Usage Guidelines

Use this command to configure router advertisements (RAs) to be sent out of the interface being configured.

Router advertisements are sent out by IPv6 routers in order to advertise their existence to hosts on the network. IPv6 hosts do not send out router advertisements.

If the **router-advert** node of the configuration tree is missing, router advertisements are not sent out. Also, if IPv6 forwarding is disabled either globally (using the **system ipv6 disable-forwarding** command ) or on the interface (using the [interfaces wirelessmodem <wlmx> ipv6 disable-forwarding](#) command), router advertisements are not sent out.

Most router advertisement parameters are required by either the Neighbor Discovery (ND) protocol or the Stateless Address Auto-Configuration (SLAAC) protocol. These parameters are used both locally for the IPv6 implementation and become part of the RA messages sent to hosts on the network so that they can be configured appropriately.

Use the **set** form of this command to create the **router-advert** configuration node and begin to send router advertisements.

Use the **delete** form of this command to remove **router-advert** configuration node and stop sending router advertisements.

Use the **show** form of this command to view router advertisement 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 wlmx {
    mtu mtu
  }
}
```

### Parameters

<i>wlmx</i>	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are using. This may be <b>wlm0</b> to <b>wlm999</b> .
<i>mtu</i>	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 wireless modem interface.

---

### Syntax

```
set interfaces wirelessmodem wlmx network [scriptfile | att | sc1 | verizon]
delete interfaces wirelessmodem wlmx network
show interfaces wirelessmodem wlmx network
```

---

### Command Mode

Configuration mode.

---

### Configuration Statement

```
interfaces {
  wirelessmodem wlmx {
    network scriptfile
  }
}
```

---

### Parameters

<i>wlmx</i>	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are using. This may be <b>wlm0</b> to <b>wlm999</b> .
<i>scriptfile</i>	Specifies the full path to a user-supplied chat script file used for a specific network.
<b>att</b>	Specifies that the AT&T chat script is to be used.
<b>sc1</b>	Specifies that the sc1 chat script is to be used.
<b>verizon</b>	Specifies that the Verizon chat script is to be used.

---

### 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 `/config/scripts`. In order to access it, you will need to specify the full path to the script.

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 wlmx {
        no-dns
    }
}
```

---

### Parameters

<i>wlmx</i>	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are using. This may be <b>wlm0</b> to <b>wlm999</b> .
-------------	---

---

### 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 wlmx {
        ondemand
    }
}
```

---

### Parameters

<i>wlmx</i>	Mandatory. Multi-node. The identifier for the wirelessmodem interface you are using. This may be <b>wlm0</b> to <b>wlm999</b> .
-------------	---

---

### 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 | statistics]]
```

---

### Command Mode

Operational mode.

---

### Parameters

---

<i>wlmx</i>	The name of a wirelessmodem interface. This may be <b>wlm0</b> to <b>wlm999</b> .
<b>debug</b>	Shows the startup debug log for the specified wirelessmodem interface.
<b>statistics</b>	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 4-5](#) shows output for **show interfaces** with the wlm0 interface visible at the bottom of the output.

Example 4-5 “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 4-6](#) shows output for `show interfaces wirelessmodem wlmx`.

Example 4-6 “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 4-7](#) shows output for `show interfaces wirelessmodem wlmx debug`.

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

Example 4-7 “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> <acomp>] 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 4-8](#) shows the output for `show interfaces wirelessmodem wlmx` statistics.

Example 4-8 “show interfaces wirelessmodem wlmx statistics”: Displaying statistics for the wirelessmodem interface

---

```

vyatta@R1> show interfaces wirelessmodem wlm0 statistics
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
AMI	Amazon Machine Image
API	Application Programming Interface
AS	autonomous system
ARP	Address Resolution Protocol
AWS	Amazon Web Services
BGP	Border Gateway Protocol
BIOS	Basic Input Output System
BPDU	Bridge Protocol Data Unit
CA	certificate authority
CCMP	AES in counter mode with CBC-MAC
CHAP	Challenge Handshake Authentication Protocol
CLI	command-line interface
DDNS	dynamic DNS
DHCP	Dynamic Host Configuration Protocol
DHCPv6	Dynamic Host Configuration Protocol version 6

---

DLCI	data-link connection identifier
DMI	desktop management interface
DMZ	demilitarized zone
DN	distinguished name
DNS	Domain Name System
DSCP	Differentiated Services Code Point
DSL	Digital Subscriber Line
eBGP	external BGP
EBS	Amazon Elastic Block Storage
EC2	Amazon Elastic Compute Cloud
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/Output
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
KVM	Kernel-Based Virtual Machine
L2TP	Layer 2 Tunneling Protocol
LACP	Link Aggregation Control Protocol
LAN	local area network
LDAP	Lightweight Directory Access Protocol
LLDP	Link Layer Discovery Protocol
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
PAT	Port Address Translation
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
PTMU	Path Maximum Transfer Unit
PVC	permanent virtual circuit
QoS	quality of service
RADIUS	Remote Authentication Dial-In User Service
RHEL	Red Hat Enterprise Linux
RIB	Routing Information Base
RIP	Routing Information Protocol
RIPng	RIP next generation
Rx	receive
S3	Amazon Simple Storage Service
SLAAC	Stateless Address Auto-Configuration
SNMP	Simple Network Management Protocol
SMTP	Simple Mail Transfer Protocol
SONET	Synchronous Optical Network
SSH	Secure Shell
SSID	Service Set Identifier
STP	Spanning Tree Protocol
TACACS+	Terminal Access Controller Access Control System Plus
TBF	Token Bucket Filter
TCP	Transmission Control Protocol
TKIP	Temporal Key Integrity Protocol

---

---

ToS	Type of Service
TSS	TCP Maximum Segment Size
Tx	transmit
UDP	User Datagram Protocol
VHD	virtual hard disk
vif	virtual interface
VLAN	virtual LAN
VPC	Amazon virtual private cloud
VPN	Virtual Private Network
VRRP	Virtual Router Redundancy Protocol
WAN	wide area network
WAP	wireless access point
WPA	Wired Protected Access

---