61200860L1-29.6A
November 2005

Configuration Guide

Configuring T1 and E1 WAN Interfaces

This configuration guide explains the processes for configuring your ADTRAN Operating System (AOS) T1/E1 product for some common applications. This guide discusses configuring the T1/E1 interfaces, various Layer 2 (L2) protocols, many-to-one Network Address Translation (NAT), and adding static routes to the route table. For more detailed information regarding specific command syntax, refer to the *AOS Command Reference Guide* on your *ADTRAN OS System Documentation* CD.

This guide consists of the following sections:

Overview of T1/E1 WAN Applications

Wide area networks (WANs) provide the mechanism for connecting remote sites together and connecting your local network to the Internet through a connection to an ISP. WANs use a variety of physical transports; T1/E1 connections are a common means of transport.T1 circuits are generally used in domestic applications, while E1 circuits are widely deployed internationally. T1/E1 circuits are a relatively inexpensive investment because they allow remote sites to share corporate resources at other locations and thus eliminate the need for redundant equipment at multiple locations. For example, a corporation with many small branch offices can consolidate their Internet access through a single interface and avoid paying for Internet connectivity at each small office. Not only does this provide a cost savings, but it also gives the corporation's IT department more control over Internet usage and protection at each of the branch offices. Many companies have centralized databases that must be used by employees at remote locations. Creating a WAN connection between the centralized database and the remote location allows the remote users full access to the resources.

Configuring T1/E1 WAN applications includes six steps:

1. Configure the physical interfaces (Ethernet and WAN interfaces)
2. Configure the L2 protocol(s)
3. Cross-connect the physical and virtual (L2) interfaces
4. Create access lists and policies (including NAT parameters)
5. Apply the policies to interfaces
6. Configure the routing information (static routes, OSPF, RIP, etc.)

These configuration steps are explained on the following pages. Each step includes a brief discussion of available settings, but does not elaborate on parameters that are normally left in the default state. In addition, each step provides a sample command listing for a generic configuration. Specific example configurations (with configuration scripts) are provided at the end of this document. For detailed information regarding WAN configuration parameters, refer to the documentation provided on your *ADTRAN OS System Documentation* CD.

Physical Interface Configurations (T1, E1, and Ethernet)

NOTE *The NetVanta Network Interface Modules (NIMs) use a* **slot/port** *notation for interface identification (e.g.,* **t1 1/1***). All non-modular interfaces built into the base unit are identified using* **0** *as the slot number (e.g.* **eth 0/1***).*

To begin configuring physical interfaces, you must first activate the appropriate interface configuration mode from the Global configuration prompt. For example, enter the following commands to activate the interface configuration mode for the first T1 interface on a T1 module inserted in slot 1:

```
>enable
#config terminal
(config)#interface t1 1/1
(config-t1 1/1)#
```

All interfaces are disabled by default and must be activated using the **no shutdown** command. Interfaces will not be able to pass data until this command is entered.

Configuring Ethernet Interfaces

Ethernet interface configuration can range from assigning an IP address and activating the interface to activating the DHCP client to poll the network DHCP server to gain an IP address. Standard Ethernet configurations generally contain an IP address, a speed, and a duplex setting. By default, all NetVanta Ethernet interfaces are configured to auto-detect the speed (as 10 or 100 Mbps) and are set to full-duplex. For most cases, these settings should suffice and will not be changed from the default state.

The following example commands configure an IP address of (10.10.0.7/24) and activates the interface for the eth 0/1 interface:

```
>enable
#config terminal
(config)#interface eth 0/1
(config-eth 0/1)#ip address 10.10.0.7 255.255.255.0
(config-eth 0/1)#no shutdown
(config-eth 0/1)#exit
(config)#
```

Configuring T1 Interfaces

There are four main settings to consider when configuring T1 network interfaces. The line coding (**coding**), framing format (**framing**), active channels (**tdm-group**), and clock source (**clock source**) must all be configured to match the circuit supplied by your network provider. By default, all NetVanta T1 interfaces are configured for ESF (**framing esf**) and B8ZS (**coding b8zs**), and to recover clocking from the network circuit (**clock source line**). Generally, the line coding, framing format, and clock source default values will be the correct ones for your application and should not be changed.

Each configured T1 interface must have the active channels specified using the **tdm-group** command because there are no default TDM groups defined. The active channels are entered as a single number representing 1 of the 24 T1 channel timeslots or as a contiguous group of channels.

The following example commands specify the configuration parameters required for a standard T1 interface:

```
>enable
#config terminal
(config)#interface t1 1/1
(config-t1 1/1)#tdm-group 1 timeslots 1-24
(config-t1 1/1)#no shutdown
(config-t1 1/1)#exit
```

Configuring E1 Interfaces

There are four main settings to consider when configuring E1 network interfaces. The line coding (**coding**), framing format (**framing**), active channels (**tdm-group**), and clock source (**clock source**) must all be configured to match the circuit supplied by your network provider. By default, all NetVanta E1 interfaces are configured for standard multi-frame without the optional CRC4 error correction (**no framing crc4**), and to recover clocking from the network circuit (**clock source line**). Generally, the line coding, framing format, and clock source default values will be the correct ones for your application and should not be changed.

Each configured E1 interface must have the active channels specified using the **tdm-group** command because there are no default TDM groups. The active channels are entered as a single number representing 1 of the 31 E1 channel timeslots or as a contiguous group of channels.

The following example commands specify the configuration parameters required for a standard E1 interface:

```
>enable
#config terminal
(config)#interface e1 1/1
(config-e1 1/1)#tdm-group 1 timeslots 1-31
(config-e1 1/1)#no shutdown
(config-e1 1/1)#exit
```

Configuring Layer 2 Protocols (Frame Relay, PPP, HDLC)

Each WAN connection in your AOS product must contain a physical interface (T1, E1, ADSL, etc.) and a Layer 2 protocol (ATM, frame relay/multilink frame relay, PPP/multilink PPP, or HDLC). The physical interface provides the actual bandwidth between your device and the network provider. The Layer 2 protocol defines how the data is packaged and presented on the physical interface. Layer 2 protocols must be configured to match the protocol provided on the circuit. For example, configuring the AOS product for PPP operation on a frame relay circuit would not be successful.

AOS currently supports the following Layer 2 protocols for T1/E1 physical links:

- frame relay, including multilink frame relay (FRF.16)
- point-to-point protocol (PPP), including multilink PPP
- high-level data link control (HDLC) protocol

Configuring the Frame Relay Interfaces (and Sub-Interfaces)

There are two settings to consider when configuring frame relay interfaces. The interface type (**frame-relay intf-type**) and signaling type (**frame-relay lmi-type**) must be configured to match the specifications supplied on your frame relay circuit by your network provider. By default, all NetVanta frame relay interfaces are configured as a DTE interface (**frame-relay intf-type dte**) with Annex D signaling (**frame-relay lmi-type ansi**).

Frame relay interfaces have a sub-interface component for each PVC which must also be configured. Each frame relay sub-interface contains a DLCI (**frame-relay interface-dlci**) and IP address (**ip address**). You must manually configure the frame relay sub-interface DLCI and IP address because there are no default DLCIs or IP addresses defined. Access policies are also applied at the sub-interface level (see *Creating Access Lists and Policies* on page 9).

Each PVC should also have a configured committed burst value (**frame-relay bc**) which is equivalent to the committed information rate (CIR) given to you by your network provider. PVCs will also have a negotiated burst rate (**frame-relay be**) which is equivalent to the excess information rate (EIR) given to you by your network provider. Both the CIR and EIR should be decided on by you and your service provider when defining your service agreement. To determine the appropriate committed burst value and EIR, you need to know the CIR and physical bandwidth for both the local and remote connections. If one side transmits data at a rate much higher than the other side's CIR (or physical bandwidth), packets will be dropped causing a decrease in efficiency. A general rule is to provision the committed burst value with the remote side CIR and configure the EIR with the difference between the CIR and the actual physical bandwidth at the location. The committed burst value plus the EIR should not be greater than the physical bandwidth.

The following commands specify the configuration parameters required for a standard frame relay interface:

```
>enable
#config terminal
(config)#interface fr 1
(config-fr 1)#no shutdown
(config-fr 1)#exit
```

The following commands specify the configuration parameters required for a standard frame relay sub-interfaces

```
(config)#interface fr 1.16
(config-fr 1.16)#frame-relay interface-dlci 16
(config-fr 1.16)#frame-relay bc 768000
(config-fr 1.16)#frame-relay be 768000
(config-fr 1.16)#ip address 192.168.72.1 /30
(config-fr 1.16)#no shutdown
(config-fr 1.16)#exit
```

NOTE *Labeling the frame relay sub-interfaces using the DLCI (such as* **1.16** *indicating a DLCI of 16) is useful for quickly determining (from a configuration printout) which sub-interface corresponds to which PVC.*

Multilink Frame Relay Operation

Multilink frame relay operation increases bandwidth on your frame relay service by aggregating multiple physical links into a single logical bundle. All the physical links in a multilink bundle are treated as a single entity by the system, allowing each PVC on the connection to dynamically share the total bandwidth of the bundle. Single data packets can be fragmented into smaller pieces which may or may not be transmitted to the network over the same physical link. Multilink frame relay devices balance the transmitted information to evenly use all the physical links in a bundle.

AOS products support multilink frame relay (FRF.16), requiring that the multilink operation be supported from the network provider. Remote side frame relay connections are unaffected by multilink operation; the multilink FRF.16 functionality provides an effective way to increase the total bandwidth at a single site between the frame relay device and the network provider.

Physical links can be dynamically added and removed from the logical bundle, so a failure on one physical link does not halt the overall operation of the bundle. Since all PVCs have access to the entire bundle bandwidth, failure of a single physical connection in the bundle does not decrease efficiency.

Multilink frame relay requires minimal configuration in your AOS product. You must first enable multilink operation on the frame relay interface (not sub-interface) and then cross-connect the multiple physical interfaces to the single frame relay interface. Optionally, you can set a bundle ID (label for the bundle), but the AOS will automatically define one based on the specified frame relay interface. For example, if multilink operation is enabled on a frame relay interface labeled **fr 1**, the bundle ID becomes **mfr1** (with the 1 corresponding to the label of the frame relay interface). Bundle IDs can be character strings containing 1 to 48 characters. Manually defining the bundle ID can make it easier to differentiate between bundles in systems with more than one multilink bundle. In systems with a single multilink bundle, leaving the bundle ID to the default value is the easiest solution.

The following commands specify the configuration parameters required for a standard multilink frame relay interface:

```
>enable
#config terminal
(config)#interface fr 1
(config-fr 1)#frame-relay multilink
(config-fr 1)#no shutdown
(config-fr 1)#exit
```

Now, cross connect multiple physical interfaces to the same multilink frame-relay interface:

```
(config)#cross-connect 1 t1 3/1 1 fr 1
(config)#cross-connect 2 t1 3/2 2 fr 1
(config)#cross-connect 3 t1 3/3 3 fr 1
```

Configuring PPP Interfaces

There are two settings to consider when configuring PPP interfaces: the IP address and the maximum transmission unit (MTU). There are no default IP addresses, so each interface must be manually programmed with the appropriate address (**ip address**). All NetVanta PPP interfaces have a default MTU of 1500 bytes, which works for most applications.

The following commands specify the configuration parameters required for a standard PPP interface:

```
>enable
#config terminal
(config)#interface ppp 1
(config-ppp 1)#ip address 68.22.15.2 /30
(config-ppp 1)#no shutdown
(config-ppp 1)#exit
```

Multilink PPP Operation

Multilink PPP operation increases bandwidth on your PPP connection by aggregating multiple physical links into a single logical bundle. All the physical links in a multilink bundle are treated as a single entity by the system, allowing each PPP session on the connection to dynamically share the total bandwidth of the bundle. Single data packets can be fragmented into smaller pieces which may or may not be transmitted to the network over the same physical link. Multilink PPP devices balance the transmitted information to evenly use all the physical links in a bundle.

The multilink bundle will remain active with a minimum of one physical link. Physical links can be dynamically added and removed from the logical bundle with a minor interruption to data flow, so a failure on one physical link does not halt the overall operation of the bundle. Since all PPP sessions have access to the entire bundle bandwidth, failure of a single physical connection in the bundle does not decrease efficiency.

Remote side PPP peers are virtually unaffected by multilink operation; however, they must be aware that multilink PPP operation is occurring and be able to handle the fragmented frames transmitted on multiple physical links. Each PPP fragmented frame will include a sequence number to aid in the reconstruction of PPP frames.

Multilink PPP requires minimal configuration in your AOS product. You must first enable multilink operation on the PPP interface and then cross-connect the multiple physical interfaces to the single PPP interface.

The fragmentation and interleave options can be used to enhance multilink operation. Fragmentation is used to reduce serialization delays during the transmission of large packets. The fragmentation process evenly divides the data among all the links in the bundle with a minimum packet size of 96 bytes. Use the **ppp multilink fragmentation** command (at the Global configuration level) to activate the fragmentation option for all multilink PPP bundles configured on the system. The interleave process is used with streaming protocols to reduce delay by giving priority to packets identified as high priority. Sequential delivery is guaranteed with multilink fragmentation, but is not guaranteed with multilink interleave operation. Use the **ppp multilink interleave** command (at the Global configuration level) to activate the interleave option for all multilink PPP bundles configured on the system.

The following commands specify the configuration parameters required for a standard multilink PPP interface:

```
>enable
#config terminal
(config)#interface ppp 1
(config-ppp 1)#ppp multilink
(config-ppp 1)#no shutdown
(config-ppp 1)#exit
```

Now, cross connect multiple physical interfaces to the same multilink PPP interface:

```
(config)#cross-connect 1 t1 3/1 1 ppp 1
(config)#cross-connect 2 t1 3/2 2 ppp 1
(config)#cross-connect 3 t1 3/3 3 ppp 1
```

Configuring HDLC Interfaces

HDLC is a protocol developed by the International Organization for Standardization (ISO) under standards ISO 3309 and 4335. Originally created for the mainframe environment, HDLC has become popularly used in many network environments because of its flexibility and ease of configuration. HDLC provides synchronous data transmission regardless of the physical layer access. Because HDLC is totally unaware of the physical layer access, it supports both half duplex and full duplex communication lines, can work in both point-to-point and multi-point network configurations, and can be transmitted over switched or non-switched channels. The AOS supports HDLC transmission over T1, E1, and serial interfaces.

HDLC configuration in AOS products consists of creating the HDLC interface and assigning an IP address. There are no protocol-specific configuration parameters for HDLC.

The following commands specify the configuration parameters required for a standard HDLC interface:

```
>enable
#config terminal
(config)#interface hdlc 1
(config-hdlc 1)#ip address 68.22.15.2 /30
(config-hdlc 1)#no shutdown
(config-hdlc 1)#exit
(config)#
```

Cross-Connecting Physical and Virtual Interfaces

Virtual interfaces must be cross-connected to physical interfaces to create a WAN interface where L2 signaling occurs. Use the **cross-connect** command to connect the physical and virtual interfaces. A single virtual interface is assigned to a single physical interface, except in the case of multilink operation, where one virtual interface is connected with multiple physical interfaces. Each created cross-connect has a unique label identifier and specifies a virtual and a physical interface.

The following command listing depicts three cross-connects to a multilink frame relay interface and a single cross-connect to a PPP interface. Each cross-connect has a unique label identifier (1 through 4):

```
>enable
#config terminal
(config)#cross-connect 1 t1 3/1 1 fr 1
(config)#cross-connect 2 t1 3/2 2 fr 1
(config)#cross-connect 3 t1 3/3 3 fr 1
(config)#cross-connect 4 t1 3/8 4 ppp 1
```

Creating Access Lists and Policies

Access lists (ACLs) and access policies (ACPs) are used to regulate traffic through your routed network. ACLs and ACPs can block, filter, and manipulate traffic to make your network more secure.

ACLs are traffic selectors that include a “matching” parameter (to select the traffic) and an action statement (to either permit or deny the matched traffic). Standard ACLs (using the **ip access-list standard** command) provide pattern matching for source IP addresses only. Use extended ACLs (using the **ip access-list extended** command) for more flexible pattern matching (including destination IP addresses).

ACPs use configured ACLs to permit, deny, or manipulate (using NAT) data on each interface where the ACP is applied. When packets are received on an interface, the configured ACPs are applied to determine whether the data will be processed or discarded. Creating access policies is a five-step process:

1. Determine what traffic needs to be regulated.
2. Enable the security features (using the **ip firewall** command).
3. Create an ACL to act as a traffic selector.
4. Create an ACP to either permit, deny, or manipulate (using NAT) the traffic selected using an access list.
5. Apply the ACP to an interface (or multiple interfaces).

Access List Traffic Selectors

ACLs include a matching parameter (to select traffic) and an action statement (to either permit or deny the matched traffic). Standard ACLs provide pattern matching for source IP addresses only. To create a standard ACL (labeled MYLIST), use the following command:

```
(config)#ip access-list standard MYLIST
(config-std-nacl)#
```

The following outlines the syntax for creating a standard ACL entry:

permit **|** deny <*source address*>

Select the traffic into the list using the **permit** keyword, or block the traffic from the list using the **deny** keyword. The source IP addresses can be entered in one of three ways:

1. Using the keyword **any** to match any IP address. For example, entering **deny any** will effectively shut down the interface that uses the access list because all traffic will match the **any** keyword.
2. Using the **host** *<A.B.C.D>* to specify a single host address. For example, entering **permit host 196.173.22.253** will allow all traffic from the host with an IP address of 196.173.22.253.
3. Using the *<A.B.C.D> <wildcard>* format to match all IP addresses in a "range." Wildcard masks work in reverse logic from subnet mask. Specifying a one in the wildcard mask equates to a "don't care." For example, entering **permit 192.168.0.0 0.0.0.255** will permit all traffic from the 192.168.0.0/24 network.

Extended ACLs provide flexible pattern matching on various different parameters. The following lists the complete syntax for the **ip access-list extended** commands:

<action> <protocol> <source IP> <source port> <destination ip> <destination port>

For example:

Source IP Address

[**permit** | **deny**] [**ip | tcp | udp**] [**any | host** *<A.B.C.D>* **|** *<A.B.C.D> <W.W.W.W>*]

*<source port>** [**any | host** *<A.B.C.D>* **|** *<A.B.C.D> <W.W.W.W>*] *<destination port>**

Destination IP Address

or:

Source IP Address

[**permit** | **deny icmp** [**any | host** *<A.B.C.D>* **|** *<A.B.C.D> <W.W.W.W>*]

[**any | host** *<A.B.C.D>* **|** *<A.B.C.D> <W.W.W.W>*] *<icmp-type>* <icmp-code>* <icmp-message>**

Destination IP Address

* = optional

For detailed information regarding the extended ACL matching parameters, refer to the *AOS Command Reference Guide* on your *ADTRAN OS System Documentation* CD.

Access Policy Action Statements

AOS access policies are used to permit, deny, or manipulate (using NAT) data for each interface. Each ACP consists of a selector (access list) and an action (allow, discard, NAT). When packets are received on an interface, the configured ACPs are applied to determine whether the data will be processed or discarded. Possible actions performed by the access policy are as follows:

allow list *<access list names>*
All packets passed by the access list(s) entered will be allowed to enter the router system.

discard list *<access list names>*
All packets passed by the access list(s) entered will be dropped from the router system.

allow list *<access list names>* **policy** *<access policy name>*

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be permitted to enter the router system. This allows for configurations to permit packets to a single interface and not the entire system.

discard list *<access list names>* **policy** *<access policy name>*

All packets passed by the access list(s) entered and destined for the interface using the access policy listed will be blocked from the router system. This allows for configurations to deny packets on a specified interface.

allow list *<access list names>* **self**

All packets passed by the access list(s) entered and destined for any local interface on the unit will be permitted to enter the router system. These packets are terminated by the unit and are not routed or forwarded to other destinations. This access list can be used for external access to Telnet and the Web GUI.

discard list *<access list names>* **self**

All packets passed by the access list(s) entered and destined for any local interface on the unit will be blocked from the router system. This allows for configurations to deny external access to the router on a specified interface.

nat source list *<access list names>* **address** *<IP address>* **overload policy** *<access policy name>*

All packets passed by the access list(s) entered will be modified to replace the source IP address with the entered IP address. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address entered. This hides private IP addresses from outside the local network. The optional **policy** keyword specifies that all packets passed by the access list(s) and destined for the interface using the listed access policy will be modified to replace the source IP address with the entered IP address. This hides private IP addresses from outside the local network.

nat source list *<access list names>* **interface** *<interface>* **overload policy** *<access policy name>*

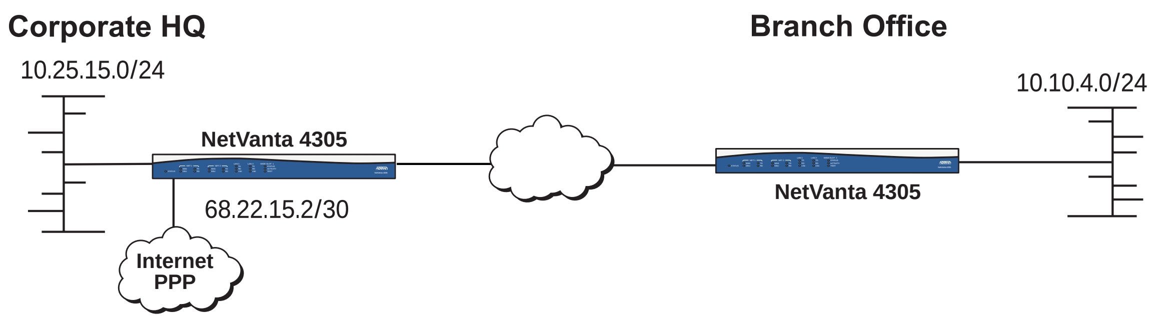
All packets passed by the access list(s) entered will be modified to replace the source IP address with the primary IP address of the listed interface. The **overload** keyword allows multiple source IP addresses to be replaced with the single IP address of the specified interface. The optional **policy** keyword specifies that all packets passed by the access list(s) and destined for the interface using the listed access policy will be modified to replace the source IP address with the primary IP address of the listed interface. This hides private IP addresses from outside the local network.

nat destination list *<access list names>* **address** *<IP address>*

All packets passed by the access list(s) entered will be modified to replace the destination IP address with the entered IP address. The **overload** keyword is not an option when performing NAT on the destination IP address; each private address must have a unique public address. This hides private IP addresses from outside the local network.

Access List and Access Policy Example

Let's review the following example to illustrate the ACL and ACP creation process.



For our example, evaluate the incoming and outgoing traffic on the WAN and local Ethernet interfaces. Use ACLs and ACPs to provide connectivity for traffic between the private LANs (branch site 10.10.4.0 network and corporate HQ 10.25.15.0 network), grant access to the public internet connection for all users (branch site and corporate HQ), and hide private IP addresses for all traffic transmitted to the public domain over the PPP connection (to protect the network). The following table outlines our traffic concerns:

Interface	Traffic to Select
Connection to Branch Office	traffic from remote LAN (10.10.4.0/24) destined for the local LAN (10.25.15.0/24)
	traffic from remote LAN (10.10.4.0/24) to the Internet through the PPP interface
Local Network (Ethernet Interface)	traffic destined for the remote LAN (10.10.4.0/24)
	traffic to the Internet through the PPP interface

Begin by planning the ACL selectors for the traffic received on the connection to the branch office. Use extended ACLs to use source and destination IP addresses to sort the traffic received from the remote LANs into two categories – traffic destined for the corporate LAN or traffic destined for the public Internet. Each category requires an extended ACL to select the appropriate traffic. All traffic destined for the public Internet requires a many-to-one NAT configuration to hide the private IP addresses and to allow a single, public IP address for access to the Internet.

Next, plan the ACL selectors for the traffic received on the local network (Ethernet interface). Use extended ACLs to use source and destination IP addresses to sort the traffic received from the local network into two categories – traffic destined for the branch office LAN or traffic destined for the public Internet. Each category requires an extended ACL to select the appropriate traffic. All traffic destined for the public Internet requires a many-to-one NAT configuration to hide private IP addresses and to allow a single, public IP address for access to the Internet.

The following table provides sample ACL commands for the various traffic on our sample network.

Action	Description	Command(s)
Allow	traffic between LANs (10.10.4.0/16 to 10.25.15.0/24)	**permit ip 10.10.4.0 0.0.0.255 10.25.15.0 0.0.0.255**
Allow	traffic from remote LAN (10.10.4.0/24) to the Internet through the PPP interface	**permit ip 10.10.4.0 0.0.0.255 any**
Allow	traffic from Corp LAN (10.25.15.0/24) to remote LAN (10.10.4.0/24)	**permit ip 10.25.15.0 0.0.0.255 10.10.4.0 0.0.0.255**
Allow	traffic from Corp LAN (10.25.15.0/24) to the Internet	**permit ip 10.10.0.0 0.0.0.255 any**

The traffic selectors required for traffic on the connection to the branch office and the local network are basically identical but contain different IP subnets on the 10.0.0.0 network. If we modify the IP addresses listed in the **permit** statements to encompass the entire 10.0.0.0 network (by using **10.0.0.0** with wildcard bits **0.255.255.255**), we can create a single set of ACLs that can be used in ACPs on both interfaces. All traffic on IP subnets of the 10.0.0.0 network will be allowed to transmit data to one another and the Internet.

The following activates the security features in the NetVanta and creates two extended ACLs to select our traffic:

```
>enable
#config terminal
(config)#ip firewall
(config)#ip access-list extended INTERLAN
(config-ext-nacl)#permit ip 10.0.0.0 0.255.255.255 10.0.0.0 0.255.255.255
(config-ext-nacl)#exit
(config)#ip access-list extended INTERNET
(config-ext-nacl)#permit ip 10.0.0.0 0.255.255.255 any
(config-ext-nacl)#exit
(config)#
```

Now, create the access policy to allow the traffic between the LANs and to NAT traffic bound for the public Internet.

```
(config)#ip policy-class INTERLANwNAT
(config-policy-class)#allow list INTERLAN
(config-policy-class)#nat source list INTERNET interface ppp 1 overload
(config-policy-class)#exit
(config)#
```

Apply the ACPs to the interface(s) to complete the configuration. For our example, the **INTERLANwNAT** ACP should be applied to **fr 1.16** (PVC to the branch office) and **eth 0/1** (local network connection).

The following command listing applies the ACP to the **fr 1.16** and **eth 0/1** interfaces:

```
>enable
#config terminal
(config)#interface fr 1.16
(config-fr 1.16)#access-policy INTERLANwNAT
(config-fr 1.16)#exit
(config)#interface eth 0/1
(config-eth 0/1)#access-policy INTERLANwNAT
(config-eth 0/1)#exit
(config)#
```

Configuring Routing Information (Static Routes)

AOS products support various routing protocols including static routes, RIP, OSPF, and BGP.
RIP, OPSF, and BGP are all routing protocols which allow routers to share the information contained in their route tables with other routers in the network. These routing protocols are generally used on networks that frequently change or contain a large number of nodes. For small applications, manually adding static routes to the router's route table is the easiest method of configuration.

Manually adding static routes to the route table requires two steps:

1. Determine the routes needed (destination address and subnet mask as well as the next-hop address or forwarding interface). Be sure to plan the default route.
2. Use the **ip route** command to add the route to the route table.

The following lists the complete syntax for the **ip route** command:

ip route *<ip address> <subnet mask> <interface or ip address> <administrative distance>*

<ip address>
Specifies the network address (in dotted decimal notation) to add to the route table.

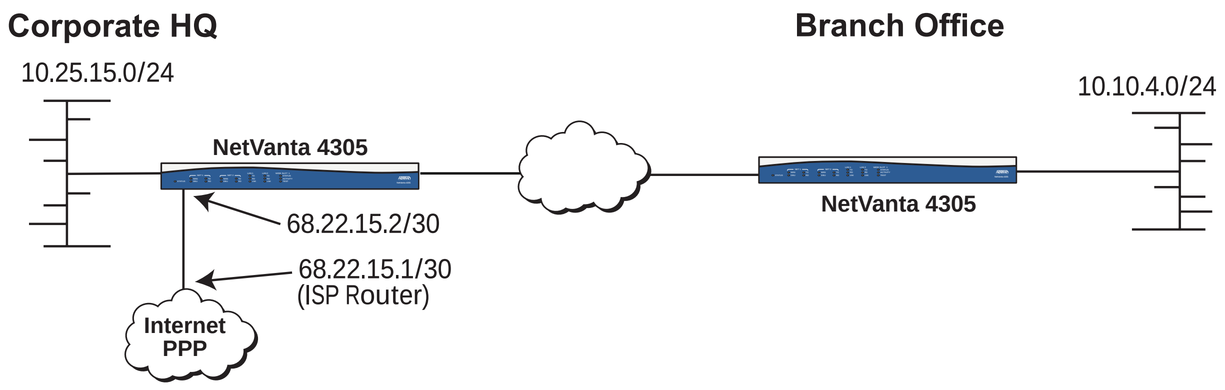
<subnet mask>
Specifies the subnet mask (in dotted decimal notation) associated with the listed network IP address.

<interface or ip address>
Specifies the gateway peer IP address (in dotted decimal notation) or a configured interface in the unit.

<administrative distance>
Specifies an administrative distance associated with this router (**1** to **255**). The administrative distance provides a way for a router to determine the best route when multiple routes to the same destination exist. The smaller the administrative distance, the more reliable the route.

Static Route Example

Let’s review the following example to illustrate the static route creation process.



The following table outlines the static routes needed in the Corporate HQ router.

Destination Address	Subnet Mask	Next-Hop Address/Forwarding Interface
10.10.10.0	255.255.255.0	fr 1.19 (WAN connection to the branch office)
Default Route (0.0.0.0)	0.0.0.0	ppp 1 (public Internet)

The following command listing adds the necessary static routes to the Corporate router’s route table:

```
>enable
#config terminal
(config)#ip route 10.10.10.0 255.255.255.0 fr 1.19
(config)#ip route 0.0.0.0 0.0.0.0 ppp 1
(config)#
```

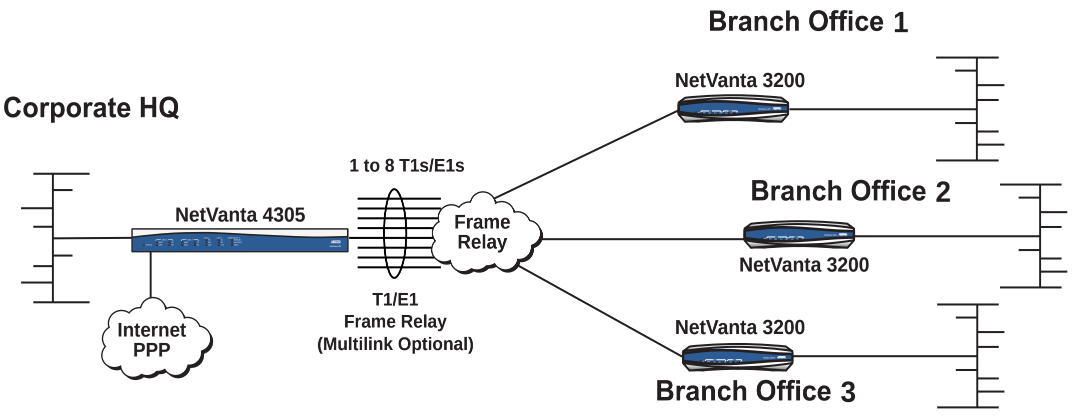
Configuration Examples

This guide contains four examples for basic WAN applications.

- Frame relay/multilink frame relay between sites
- PPP/multilink PPP in a point-to-point scenario between sites
- PPP/multilink PPP for a public connection to an ISP
- HDLC for a public connection to an ISP

Each example provides a network diagram, configuration parameters (to explain complicated network diagrams), and the sample script. Brief explanations are provided beside the example scripts when needed to differentiate between Layer 2 protocol configurations. For more details on configuring Layer 2 protocols, refer to *Configuring Layer 2 Protocols (Frame Relay, PPP, HDLC)* on page 5.

Frame Relay/Multilink Frame Relay Application



Configuration Parameters

Parameter	Corporate HQ	Branch 1	Branch 2	Branch 3
WAN Interface(s)	t1 3/1 (1-24) t1 3/2 (1-24) t1 3/3 (1-24)	t1 1/1 (1-24)	t1 1/1 (1-24)	t1 1/1 (1-24)
LAN: Eth 0/1 IP	10.10.0.7/24	10.10.10.1/24	10.10.20.1/24	10.10.30.1/24
Frame Relay Interface(s)	fr 1.16 fr 1.17 fr 1.18	fr 1.19	fr 1.20	fr 1.21
DLCI	16 17 18	19	20	21
Signaling Type	Annex D	Annex D	Annex D	Annex D
IP Address	192.168.72.1/30 (fr 1.16) 192.168.72.5/30 (fr 1.17) 192.168.72.9/30 (fr 1.18)	192.168.72.2/30	192.168.72.6/30	192.168.72.10/30
Internet Connection	t1 3/8 (1-18)	N/A	N/A	N/A
PPP Interface	ppp 1			
IP Address	68.22.15.2/30			

Example Configuration Script

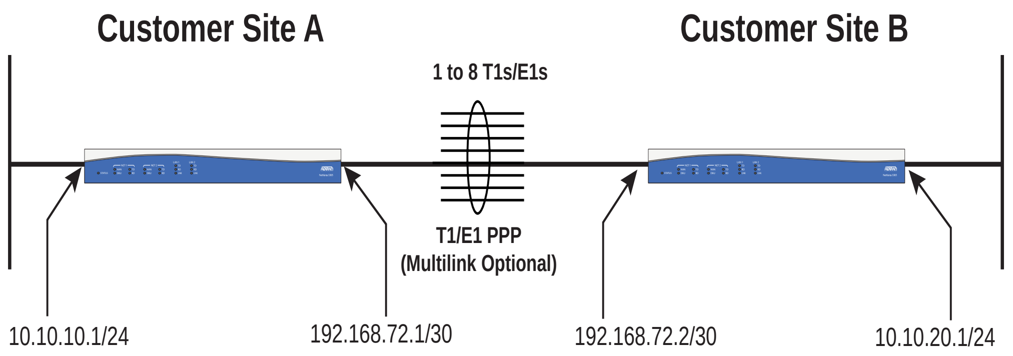
```
!
!
hostname "Corporate HQ"
enable password md5 encrypted 7f1a02ddd2cf3df129eb99c8408a5e28
!
!
ip firewall
no ip firewall alg h323
ip firewall alg sip
!
!
interface eth 0/1
  ip address  10.10.0.7  255.255.255.0
  access-policy INTERLANwNAT
  no shutdown
!
!
interface t1 3/1
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface t1 3/2
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface t1 3/3
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface t1 3/8
  tdm-group 1 timeslots 1-18 speed 64
  no shutdown
!
!
interface fr 1 point-to-point
  frame-relay lmi-type ansi
  frame-relay multilink
  no shutdown
  cross-connect 1 t1 3/1 1 frame-relay 1
  cross-connect 2 t1 3/2 1 frame-relay 1
  cross-connect 3 t1 3/3 1 frame-relay 1
!
interface fr 1.16 point-to-point
  frame-relay interface-dlci 16
  frame-relay bc 768000
  frame-relay be 768000
  ip address  192.168.72.1  255.255.255.252
  access-policy INTERLANwNAT
!
interface fr 1.17 point-to-point
  frame-relay interface-dlci 17
  frame-relay bc 768000
  frame-relay be 768000
  ip address  192.168.72.5  255.255.255.252
```

Enables multilink frame relay on the interface. For regular frame relay applications, remove this statement.

Multiple cross-connects for multilink frame relay. Regular frame relay applications have a single cross-connect for each frame relay interface.

```
!
interface fr 1.18 point-to-point
  frame-relay interface-dlci 18
  frame-relay bc 768000
  frame-relay be 768000
  ip address  192.168.72.9  255.255.255.252
!
interface ppp 1
  ip address  68.22.15.2  255.255.255.252
  no shutdown
  cross-connect 4 t1 3/8 1 ppp 1
!
!
ip access-list extended InterLAN
  permit ip 10.10.0.0 0.0.255.255  10.10.0.0 0.0.255.255
!
ip access-list extended INTERNET
  permit ip 10.10.0.0 0.0.255.255  any
!
!
ip policy-class INTERLANwNAT
  allow list InterLAN
  nat source list INTERNET interface ppp 1 overload
!
!
!
ip route 0.0.0.0 0.0.0.0 ppp 1
ip route 10.10.10.0 255.255.255.0 fr 1.16
ip route 10.10.20.0 255.255.255.0 fr 1.17
ip route 10.10.30.0 255.255.255.0 fr 1.18
!
!
end
```

PPP/Multilink PPP (Point-to-Point) Example



Example Configuration Script

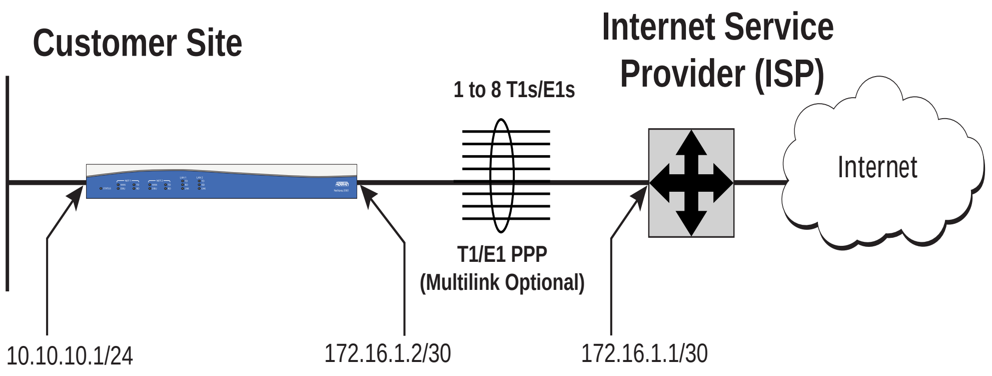
```
!
!
hostname "Customer Site A"
enable password md5 encrypted 7f1a02ddd2cf3df129eb99c8408a5e28
!
!
ip firewall
no ip firewall alg h323
ip firewall alg sip
!
interface eth 0/1
  ip address  10.10.10.1  255.255.255.0
  no shutdown
!
!
interface t1 3/1
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface t1 3/2
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface t1 3/3
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
!
interface ppp 1
  ip address  192.168.72.1  255.255.255.252
  access-policy INTERLAN
  ppp multilink
  no shutdown
  cross-connect 1 t1 3/1 1 ppp 1
  cross-connect 2 t1 3/2 1 ppp 1
  cross-connect 3 t1 3/3 1 ppp 1
!
!
```

Enables multilink PPP on the interface. For regular PPP applications, remove this statement.

Multiple cross-connects for multilink PPP. Regular PPP applications have a single cross-connect for each PPP interface.

```
!
ip access-list extended INTERLAN
  permit ip 10.10.20.0 0.0.0.255  10.10.10.0 0.0.0.255
!
ip policy-class INTERLAN
  allow list INTERLAN
!
!
ip route 10.10.20.0 255.255.255.0 ppp 1
!
!
end
```

PPP/Multilink PPP (Public Internet) Example



Example Configuration Script

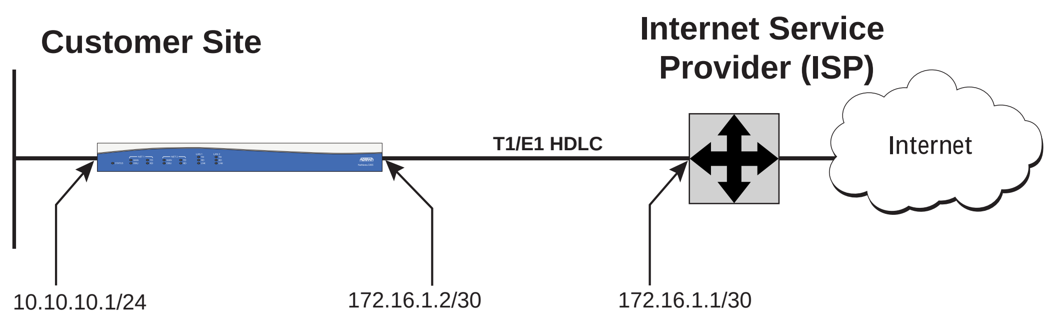
```
!
!
hostname "Customer Site"
enable password md5 encrypted 7f1a02ddd2cf3df129eb99c8408a5e28
!
!
ip firewall
no ip firewall alg h323
ip firewall alg sip
!
!
!
interface eth 0/1
  ip address  10.10.10.1  255.255.255.0
  access-policy NAT
  no shutdown
!
!
!
interface t1 3/1
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
```

```
!
interface t1 3/2
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
interface t1 3/3
  tdm-group 1 timeslots 1-241 speed 64
  no shutdown
!
!
interface ppp 1
  ip address  172.16.1.2  255.255.255.252
  ppp multilink
  no shutdown
  cross-connect 1 t1 3/1 1 ppp 1
  cross-connect 2 t1 3/2 1 ppp 1
  cross-connect 3 t1 3/3 1 ppp 1
!
!
!
ip access-list extended MATCHALL
  permit ip any  any
!
ip policy-class NAT
  nat source list MATCHALL interface ppp 1 overload
!
!
end
```

Enables multilink PPP on the interface. For regular PPP applications, remove this statement.

Multiple cross-connects for multilink PPP. Regular PPP applications have a single cross-connect for each PPP interface.

HDLC (Public Internet) Example



Example Configuration Script

```
!
!
hostname "Customer Site"
enable password md5 encrypted 7f1a02ddd2cf3df129eb99c8408a5e28
!
!
ip firewall
no ip firewall alg h323
ip firewall alg sip
```

```
!
interface eth 0/1
  ip address  10.10.10.1  255.255.255.0
  access-policy NAT
  no shutdown
!
!
interface t1 3/1
  tdm-group 1 timeslots 1-24 speed 64
  no shutdown
!
!
interface hdlc 1
  ip address  172.16.1.2  255.255.255.252
  no shutdown
  cross-connect 1 t1 3/1 1 hdlc 1
!
!
!
ip access-list extended MATCHALL
  permit ip any any
!
ip policy-class NAT
  nat source list MATCHALL interface hdlc 1 overload
!
!
end
```