DATUM SYSTEMS

# I5-I7 Application Guide

# Networks with Uni-Directional Links and Point to Multi-Point Links

**Revision History**

| | | |
|---|---|---|
| Rev 0.1 | March-14-2014 | Initial Release |
| Rev 0.2 | April -4-2014 | |

## 1.0 I5-I7 Uni-Directional and Point to Multi-Point Links Overview

This Tech Note described the resources and procedures necessary to use the Datum Systems' I5-I7 Ethernet IP Interface within networks containing uni-directional links, including point to multi-point networks. The basic concepts here are applicable to networks in general, but several significant technical difficulties associated with creating this type of network architecture have been solved using both unique I5-I7 software and procedures plus specific external hardware.

### 1.0.1 Base Definitions

For the purposes of the remainder of this document the conventions for naming the different objects that are part of common networks plus elements of the Linux operating system are:

- "***LAN***" The acronym Local Area Network. This normally means a local wired or wireless network of computers and network devices like switches and routers. It may also have a gateway to the full Internet.
- "***WAN***" The Acronym for Wide Area Network.
- "***Switch***" A network device that connects multiple other network elements together. Normally used in an Ethernet network element. A switch allows multiple simultaneous packet transfers between ports by building a table which associates port with MAC addresses. It is considered a Layer 2 device since it operates on hardware (MAC) addresses. Normally switches are implemented in hardware for speed.
- "***Bridge***" Could be considered a software switch. In our use it is a Linux program that can bridge between two ports connected to different parts of the same LAN. Bridging can be used between the eth0 and hldc0 ports and over a link with two I5-I7s in bridging mode will act like a single bridge.
- "***Router***" A device that interconnects two or more computer networks, and selectively interchanges packets of data between them. Routers connect two or more logical subnets which do not share a common network address. Routers use either static routing rules to build a routing table for controlling packet flow between networks, or can use various dynamic routing protocols where routers talk to each other, exchanging information and automatically build their routing tables. A router is normally considered a Layer 3 device operating on IP Addresses as opposed to a switch which is a Layer 2 device.
- "**I5-I7**" These are modem specific programmable Ethernet IP interfaces that run a Linux based Operating System, usually Vyatta (or the newer VyOS). The I5 is an option in the M500 series modems and the I7 is an option in the M7 series modems. They are a more powerful version of the SnIP interface.

### 1.0.2 Why are Uni-Directional Links a Problem?

If you look around at common Ethernet/TCP/IP equipment and connections used in virtually all networks the physical connections are bi-directional. That is every interface on a computer, router, switch, modem, etc is capable of transmitting and receiving. Do not confuse uni-directional with half duplex. Half duplex can either transmit or receive but not both at the same time.

From a protocol standpoint TCP/IP demands a return path for acknowledgements as a minimum, so the full network from end to end cannot be truly unit-directional and handle TCP/IP traffic, only UDP types. Networks however can be built which transfer TCP/IP traffic but contain segments that are unidirectional with the return path via another route.

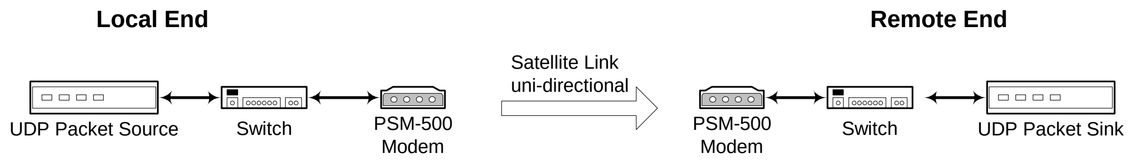
Problems may arise from multiple sources, but the main one is the inclusion of software or equipment that expects responses from sent packets, or expects responses on the same interface port that the outgoing packet was sent on. These mainly include “bridges, switches or routers in the network. Examples will be shown in the various network types where this can occur.

### 1.0.3 Types of Satellite Networks that May Contain Uni-Directional Links.

There are 3 basic network types or architectures that could have a uni-directional component. In order of complexity:

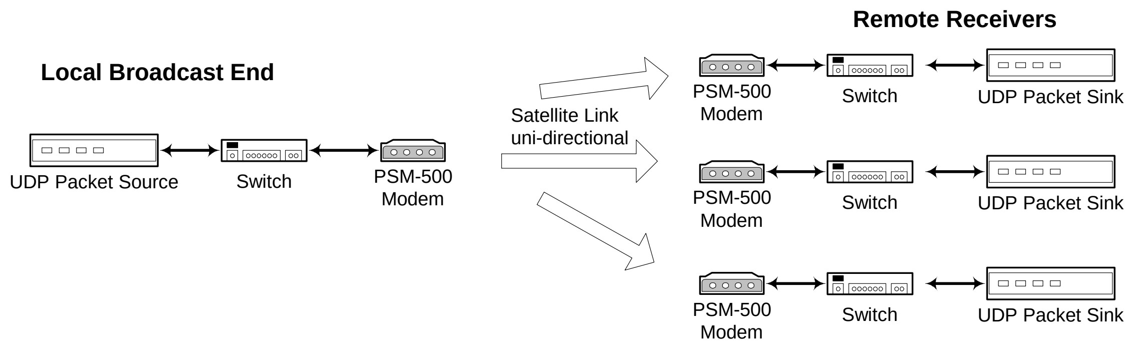
#### 1.0.3.1 Point to Point Uni-directional Link

For example if one station wants to transmit UDP type data from a source to a sink. This might be video or music that is being distributed, and no return path is available or needed. This Network link is NOT capable of handling TCP/IP traffic because the lack of a return path will cause the packet originator to cease sending. However read the following section on multicast since this may be possible with a single group member.



#### 1.0.3.2 Point to Multi-Point Broadcast and Multicast System:

The same example of sending video or audio UDP packets could apply here but the transmission is intended to be received by more than one station. This Network link is also NOT capable of handling TCP/IP traffic because the possible lack of a return path will cause the packet originator to cease sending.

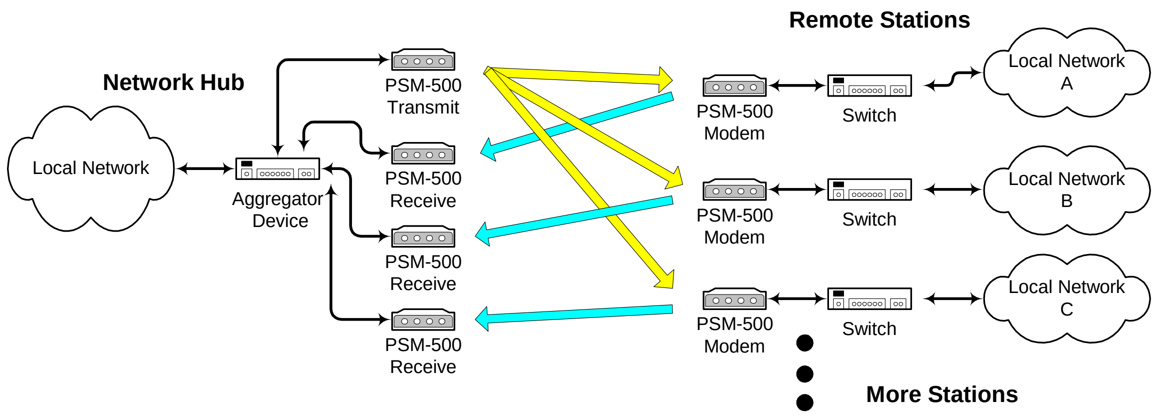


Typical network techniques designed for this specific type of connection are termed “broadcast” and “multicast” The difference is that broadcast is sent to all hosts while multicast is sent to a group of hosts – those who have joined a specific multicast group.

#### 1.0.3.3 Point to Multi-Point with Common Outbound and Separate Receive

An example of this might be a Star topology network with a single outbound carrier from a main "Hub" site that is received by all or multiple remote stations. The remotes take only the traffic from the single outbound transmission stream that is destined for their local network. The difference between this and the preceding types is that a return path is provided; either by another media or into separate receive equipment at the main "Hub" site. Because the network is bi-directional from end to end it is capable of handling all standard network traffic including UDP and TCP/IP types. But, it consists of multiple unidirectional links.

What would normally be considered a "switch" at the hub site is now called for our reference an "aggregator" device. This is because a switch by itself will not work in this configuration. The return packets on different ports from those transmitted will cause the switch to block after its initial "flooding" stage. In the working point to multi-point configuration shown later the aggregator is composed of a Cisco router and a switch.



Specifics of configuring the equipment for this type of Point to Multi-point network topology are given in Section 2.03 later. That section also shows example IP addressing, router and I5-I7 configuration information. This topology offers many advantages but requires very specific configuration.

### 1.0.4 Equipment and Processes that Will Not Function within the Uni-Directional Segment of a Link.

Standard network devices are not normally designed to work on a uni-directional link. The items listed below are common devices found in most transmission networks that may cause problems.

- **Dynamic Routers** – Dynamic routers depend on bi-directional communications between routers to determine if a link is good and to transfer resource information back and forth. As a matter of fact, dynamic routing will not operate over any link that contains a segment that is uni-directional whether it is within that segment or not. A router in this type link must use static routing.

  By router definition the link is "Down" or broken and therefore is not a valid route to take in getting a packet through. Consequently Cisco style routers that use the standard WAN keep-alive protocols may still report a link as Down even if set up for static routing.

- **Standard Switches** – If a standard switch device is used at the hub of a point to multi-point system where the received packets come in on a different port than the one transmitted on, then every outbound packet after the first will be blocked by the switch. It is potentially possible to avoid this by using smart or managed switches, but normally a router is required in conjunction with the switch.

- **Bridges** - A bridge is basically a two port switch, and therefore the failure shown in item 2 is also applicable to bridge devices.

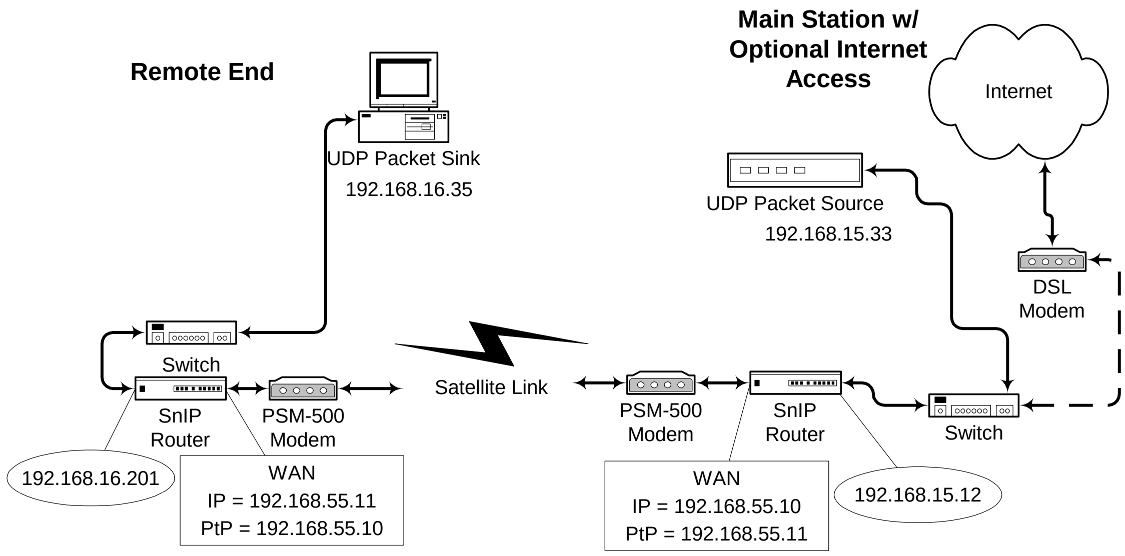
## 2.0 Specific Network Configuration

Since customers want to create these types of networks, we try to show specific network configurations that can be created to allow them to operate. The exact IP addresses are not a necessary part of the configurations, but used mainly as an example. Users should use addresses compatible to their systems.

### *2.0.1 Configuring a Point to Point Uni-directional Link*

A point to point unidirectional link is only capable of transferring UDP type Ethernet packets. It cannot handle TCP/IP simply because a return path is required for TCP/IP packet acknowledgements. A totally separate channel however could be used for the return, for example a satellite system could be used to push information out in one direction with a telephone line for the return path. That would allow a wider band transfer in one direction.

No special configuration is required to achieve the connection other than insuring that the traffic is only UDP.



### *2.0.2 Configuring a Point to Multi-Point Broadcast or Multicast System*

A point to multi-point broadcast or multicast system requires additional configuration over a simple link carrying UDP packets. Both types will require that routing be accomplished using static-routes.

Broadcasting sends packets with a broadcast address to everyone on the network and uses the top address within the mask for the purpose. The current broadcast address for a given IP Mask can be viewed by looking at the response to the “ifconfig” command or alternately testing potential masks using the “ipcalc” command.

Multicast packets have addresses in the range of 224.0.0.0 to 239.255.255.255. A I5-I7 or other transmission equipment must be set to recognize these addresses. In a bi-directional network this is accomplished by the multicast router named “mrouted”. This router cannot operate unidirectionally because it passes information back and forth with other routers. Consequently the Linux kernel must be set to accept this address range and static-routes must be added to pass

No content here yet.

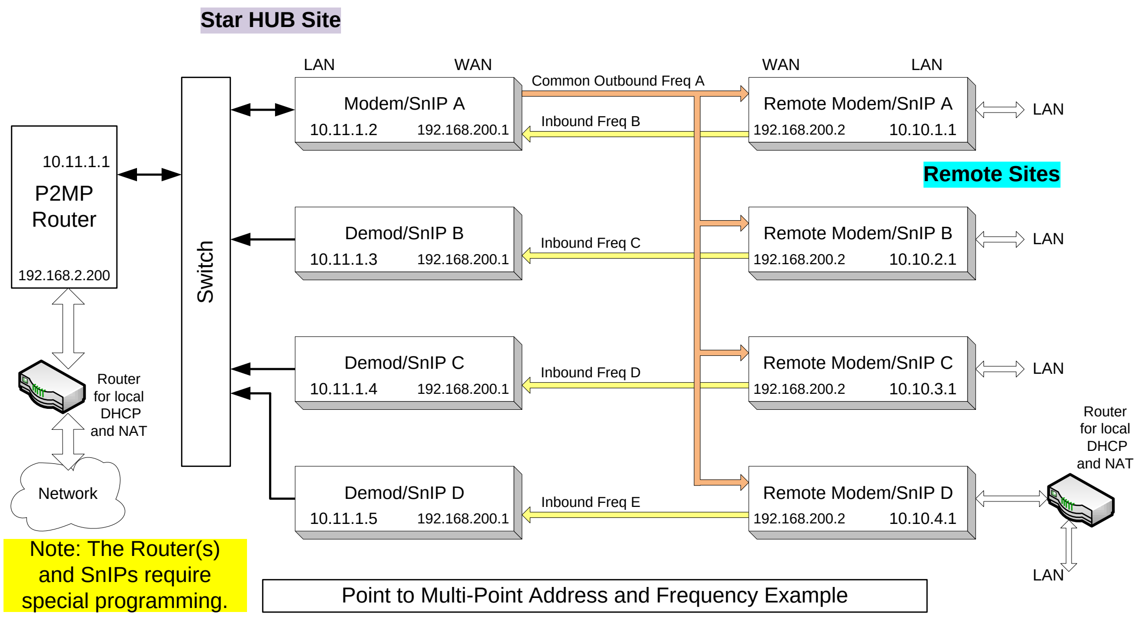
### *2.0.3 Configuring a Point to Multi-Point with Common Outbound and Separate Receive*

This is the most complex of this type system. The network requires a router, a hub based modem or modulator with I5-I7 for the common outbound transmit, one hub based demodulator with I5-I7 for each remote and one full modem with I5-I7 located at each remote site. The advantage is the savings in equipment at the hub and potential spectrum reduction for the single outbound carrier versus multiple carriers.

The key to making this network topology work is the router setup at the hub and the use of specific firewall rule sets, combined with static ARP entries for the I5-I7s. The configuration diagram and setups shown below also contain several specifics that must be followed correctly for this type network to work correctly. Specifically note the all of the hub WAN (peth1) IP addresses are the same (192.168.200.1) and all of the remote WAN (peth1) IP addresses are the same (192.168.200.2).

All of the I5-I7s in this network use the “router” link mode with static routing. There are however very minimal static routes required.

You must also have a predetermined frequency plan with sufficient segment allocated for the common outbound transmit and each return carrier. Normally the outbound will be a higher data rate to carry the traffic of all the remotes while each return carrier only has to carry its own return traffic. There is one common outbound frequency assignment plus one inbound for each remote.



## 2.0.3.1 HUB Equipment Configuration

### *2.0.3.1.1 Cisco Router Setup.*

This should be a router similar to a Cisco 2600 or newer router with two fast Ethernet ports. As shown in the diagram one port is connected to the local LAN and the other port is connected to the “Star Hub” equipment consisting of the common transmit modulator and the per-remote receive demodulators. The connection between the router and I5-I7 on each modem device is a standard switch.

Connect the router to network as shown and connect the console port to a computer a terminal emulator with serial port access, such as Hyperterminal in Windows or GTK Term in Linux. For a Cisco router you will also need the console cable for this connection.

1. **Set terminal emulator to 9600,N,8,1 with no flow control**
2. **Power up router - wait for completion and a ">" prompt.**
3. **Enter "enable" and the password when prompted.**
4. **Remove existing configuration if it is incompatible.**

```
 erase startup-config      [Note a full erase would be nvram]
 reload
```

5. **Set base configuration:**

```
ip subnet-zero
no voice hpi capture buffer
no voice hpi capture destination
mta receive maximum-recipients 0
```

6. **Set the Fast Ethernet Interface 0 to connect to the Modem/I5-I7 bank:**

```
interface FastEthernet0/0
 description Link to MyStarHub
 ip address 10.11.1.1 255.255.255.0  <address and mask in star hub>
 ip nat inside
 duplex auto
 speed auto
```

7. **Set the Fast Ethernet Interface 1 to connect to the local LAN and Internet access:**

```
interface FastEthernet0/1
 description Link to Internet
 ip address 192.168.2.200 255.255.255.0 <address and mask in local LAN>
 ip nat outside
 duplex auto
 speed auto
```

8. **Setup remainder – NAT, ACL and static routes**

```
ip nat inside source list 10 interface FastEthernet0/1 overload
ip nat inside source list 20 interface FastEthernet0/1 overload
ip http server
ip classless
ip route 0.0.0.0 0.0.0.0 192.168.2.1
ip route 10.10.0.0 255.255.0.0 10.11.1.2
!
access-list 10 permit 10.10.0.0 0.0.255.255
access-list 20 permit 10.0.0.0 0.255.255.255
!
call rsvp-sync
!
mgcp profile default
!
dial-peer cor custom
!
line con 0
line aux 0
line vty 0 4
 password cisco
 login
```

9. **Save the configuration using the Cisco command**
   copy running-config startup-config

### *2.0.3.1.2 HUB Common Outbound Modulator & I5-I7 Setup:*

The common outbound is typically at a higher data rate than any of the individual receive demodulators since it is carrying all of the traffic for all connected remote sites.

Select and set the modem transmit frequency, data rate, FEC parameters, etc. These of course must match the receive demodulator settings at each remote site.

Chose the I5-I7 as the data interface and set its IP address either via the front panel or from a console connection. Once set a telnet session can be used to set the remaining configuration items below:

The I5-I7 configuration is set via the Vyatta/VyOS configure command. We are basically setting the outgoing I5-I7 configuration as a static router. Following is a snippet of the configuration showing those items needed in the configuration:

```
interfaces {
    ethernet eth0 {
        address 10.11.1.2/24
        description "LAN Interface"
        duplex auto
        smp_affinity auto
        speed auto
    }
    ethernet eth1 {
        duplex auto
        smp_affinity auto
        speed auto
    }
    loopback lo {
    }
    pseudo-ethernet peth1 {
        address 192.168.200.1/24
        description "Satellite WAN Interface"
        link eth1
        mac aa:bb:cc:dd:ee:01
        mode private
    }
}
protocols {
    static {
        route 0.0.0.0/0 {
            next-hop 10.11.1.1 {
            }
        }
        route 10.10.0.0/16 {
            next-hop 192.168.200.2 {
            }
        }
    }
}
```

Note here that eth0 is the LAN side interface and peth1 is the WAN (modem) side interface. There is really no configuration applied to the two interfaces other than the static-routes.

If the interfaces are set by default as part of a bridge interface, the bridge must be removed prior to configuring the interfaces in routed mode. To remove interfaces eth0 and eth1 from bridge mode, issue the following commands:

```
configure
del interfaces ethernet eth0 bridge-group
del interfaces ethernet eth1 bridge-group
del interfaces bridge br0
commit
save
```

Next, we need to apply the following configuration to interface eth0:

```
ethernet eth0 {
    address 10.11.1.2/24
    description "LAN Interface"
    duplex auto
    smp_affinity auto
    speed auto
}
```

The following set commands are used to set the configuration shown above for interface eth0:

```
set interfaces ethernet eth0 address '10.11.1.2/24'
set interfaces ethernet eth0 description 'LAN Interface'
set interfaces ethernet eth0 duplex 'auto'
set interfaces ethernet eth0 smp_affinity 'auto'
set interfaces ethernet eth0 speed 'auto'
```

Next, we need to apply the following configuration to interface peth1:

```
pseudo-ethernet peth1 {
    address 192.168.200.1/24
    description "Satellite WAN Interface"
    link eth1
    mac aa:bb:cc:dd:ee:01
    mode private
}
```

As shown above, interface peth1 is actually a pseudo-ethernet interface that is linked to physical Ethernet interface eth1.

The following set commands are used to set the configuration shown above for interface peth1:

```
set interfaces pseudo-ethernet peth0 address '10.172.192.202/24'
set interfaces pseudo-ethernet peth0 link 'eth0'
set interfaces pseudo-ethernet peth0 mode 'private'
set interfaces pseudo-ethernet peth1 address '192.168.200.1/24'
set interfaces pseudo-ethernet peth1 description 'Satellite WAN Interface'
set interfaces pseudo-ethernet peth1 link 'eth1'
set interfaces pseudo-ethernet peth1 mac 'aa:bb:cc:dd:ee:01'
set interfaces pseudo-ethernet peth1 mode 'private'
```

Next, we need to apply the following static routes:

```
protocols {
    static {
        route 0.0.0.0/0 {
            next-hop 10.11.1.1 {
            }
        }
        route 10.10.0.0/16 {
            next-hop 192.168.200.2 {
            }
        }
    }
}
```

For consistency, all static routes (including the system gateway) are set in protocols static route. The following set commands are used to add the routes shown above:

```
set protocols static route 0.0.0.0/0 next-hop '10.11.1.1'
set protocols static route 10.10.0.0/16 next-hop '192.168.200.2'
delete system gateway-address
```

At this point, we need to commit and save the configuration using the following commands:

```
commit
save
exit
```

Finally, we need to create a static ARP entry for IP address 192.168.200.2 that points to MAC address AA:BB:CC:DD:EE:02.  To achieve this, we must execute the following commands:

```
sudo su -
echo "aa:bb:cc:dd:ee:02  remotes" >>/etc/ethers
echo "192.168.200.2  remotes" >>/etc/hosts
echo "/usr/sbin/arp -f" >>/config/scripts/vyatta-postconfig-bootup.script
/usr/sbin/arp -f
exit
```

Verify that the static ARP entry has been created by issuing the following command:

```
show arp
```

The output should contain the highlighted line:

```
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.200.2            ether   aa:bb:cc:dd:ee:02   CM                    peth1
```

The other values can be set as desired. For example if desired the SNMP daemon can be enabled if SNMP control is required.

There are no firewall rules applied for HUB Common Outbound Modulator.

To verify the configuration, issue the “show ip route” command.  The routing table should be populated as follows:

```
vyatta@vyatta:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

S>* 0.0.0.0/0 [1/0] via 10.11.1.1, eth0
S>* 10.10.0.0/16 [1/0] via 192.168.200.2, peth1
C>* 10.11.1.0/24 is directly connected, eth0
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.168.200.0/24 is directly connected, peth1
```

No access-lists or “iptables” rules are required for the Hub Modulator unless the demodulator side of a modem is used as one of the receive channels. See the following Section 2.0.3.1.3

### *2.0.3.1.3 Alternate HUB Outbound Modulator/Demodulator & I5-I7 Setup:*

A modulator only unit is very rare. So you might as well use a full standard modem for the common transmit and the demod side becomes one of the receivers needed. The I5-I7 configwiz setup has the following relevant parameters set.

```
firewall {
    name FORWARDING {
        default-action drop
        rule 10 {
            action accept
            destination {
                address 0.0.0.0/0
            }
            log disable
            protocol all
            source {
                address 10.10.0.0/16
            }
        }
        rule 15 {
            action accept
            destination {
                address 10.10.0.0/16
            }
            log disable
            protocol all
            source {
                address 0.0.0.0/0
            }
        }
        rule 20 {
            action accept
            destination {
                address 0.0.0.0/0
            }
            log disable
            protocol all
            source {
                address 10.11.1.0/24
            }
        }
        rule 25 {
            action accept
            destination {
                address 10.11.1.0/24
            }
            log disable
            protocol all
            source {
                address 0.0.0.0/0
            }
        }
    }
}
interfaces {
    ethernet eth0 {
        address 10.11.1.2/24
        description "LAN Interface"
        duplex auto
        firewall {
            in {
                name FORWARDING
            }
            out {
                name FORWARDING
            }
        }
        smp_affinity auto
        speed auto
    }
    ethernet eth1 {
        duplex auto
        smp_affinity auto
        speed auto
    }
    loopback lo {
    }
    pseudo-ethernet peth1 {
        address 192.168.200.1/24
        description "Satellite WAN Interface"
        firewall {
            in {
                name FORWARDING
            }
            out {
                name FORWARDING
            }
        }
        link eth1
        mac aa:bb:cc:dd:ee:01
        mode private
    }
}
protocols {
    static {
        route 0.0.0.0/0 {
            next-hop 10.11.1.1 {
            }
        }
        route 10.10.0.0/16 {
            next-hop 192.168.200.2 {
            }
        }
    }
}
```

First, we need to set the following firewall rules.  They must exist before they can be applied to an interface.

```
firewall {
    name FORWARDING {
        default-action drop
        rule 10 {
            action accept
            destination {
                address 0.0.0.0/0
            }
            log disable
            protocol all
            source {
                address 10.10.0.0/16
            }
        }
        rule 15 {
            action accept
            destination {
                address 10.10.0.0/16
            }
            log disable
            protocol all
            source {
                address 0.0.0.0/0
            }
        }
        rule 20 {
            action accept
            destination {
                address 0.0.0.0/0
            }
            log disable
            protocol all
            source {
                address 10.11.1.0/24
            }
        }
        rule 25 {
            action accept
            destination {
                address 10.11.1.0/24
            }
            log disable
            protocol all
            source {
                address 0.0.0.0/0
            }
        }
    }
}
```

To configure the firewall rules shown above, we use the following set commands:

```
set firewall name FORWARDING default-action 'drop'
set firewall name FORWARDING rule 10 action 'accept'
set firewall name FORWARDING rule 10 destination address '0.0.0.0/0'
set firewall name FORWARDING rule 10 log 'disable'
set firewall name FORWARDING rule 10 protocol 'all'
set firewall name FORWARDING rule 10 source address '10.10.0.0/16'
set firewall name FORWARDING rule 15 action 'accept'
set firewall name FORWARDING rule 15 destination address '10.10.0.0/16'
set firewall name FORWARDING rule 15 log 'disable'
set firewall name FORWARDING rule 15 protocol 'all'
set firewall name FORWARDING rule 15 source address '0.0.0.0/0'
set firewall name FORWARDING rule 20 action 'accept'
set firewall name FORWARDING rule 20 destination address '0.0.0.0/0'
set firewall name FORWARDING rule 20 log 'disable'
set firewall name FORWARDING rule 20 protocol 'all'
set firewall name FORWARDING rule 20 source address '10.11.1.0/24'
set firewall name FORWARDING rule 25 action 'accept'
set firewall name FORWARDING rule 25 destination address '10.11.1.0/24'
set firewall name FORWARDING rule 25 log 'disable'
set firewall name FORWARDING rule 25 protocol 'all'
set firewall name FORWARDING rule 25 source address '0.0.0.0/0'
```

Next, we configure our interfaces.

Note here that eth0 is the LAN side interface and peth1 is the WAN (modem) side interface. There is really no configuration applied to the two interfaces other than the static-routes.

If the interfaces are set by default as part of a bridge interface, the bridge must be removed prior to configuring the interfaces in routed mode. To remove interfaces eth0 and eth1 from bridge mode, issue the following commands:

```
configure
del interfaces ethernet eth0 bridge-group
del interfaces ethernet eth1 bridge-group
del interfaces bridge br0
commit
save
```

Next, we need to apply the following configuration to interface eth0:

```
ethernet eth0 {
    address 10.11.1.2/24
    description "LAN Interface"
    duplex auto
    firewall {
        in {
            name FORWARDING
        }
        out {
            name FORWARDING
        }
    }
    smp_affinity auto
    speed auto
}
```

The following set commands are used to set the configuration shown above for interface eth0:

```
set interfaces ethernet eth0 address '10.11.1.2/24'
set interfaces ethernet eth0 description 'LAN Interface'
set interfaces ethernet eth0 duplex 'auto'
set interfaces ethernet eth0 firewall in name 'FORWARDING'
set interfaces ethernet eth0 firewall out name 'FORWARDING'
set interfaces ethernet eth0 smp_affinity 'auto'
set interfaces ethernet eth0 speed 'auto'
```

Next, we need to apply the following configuration to interface peth1:

```
pseudo-ethernet peth1 {
    address 192.168.200.1/24
    description "Satellite WAN Interface"
```

```
        firewall {
            in {
                name FORWARDING
            }
            out {
                name FORWARDING
            }
        }
        link eth1
        mac aa:bb:cc:dd:ee:01
        mode private
    }
```

As shown above, interface peth1 is actually a pseudo-ethernet interface that is linked to physical Ethernet interface eth1.

The following set commands are used to set the configuration shown above for interface peth1:

```
set interfaces pseudo-ethernet peth1 address '192.168.200.1/24'
set interfaces pseudo-ethernet peth1 description 'Satellite WAN Interface'
set interfaces pseudo-ethernet peth1 firewall in name 'FORWARDING'
set interfaces pseudo-ethernet peth1 firewall out name 'FORWARDING'
set interfaces pseudo-ethernet peth1 link 'eth1'
set interfaces pseudo-ethernet peth1 mac 'aa:bb:cc:dd:ee:01'
set interfaces pseudo-ethernet peth1 mode 'private'
```

Next, we need to apply the following static routes:

```
protocols {
    static {
        route 0.0.0.0/0 {
            next-hop 10.11.1.1 {
            }
        }
        route 10.10.0.0/16 {
            next-hop 192.168.200.2 {
            }
        }
    }
}
```

For consistency, all static routes (including the system gateway) are set in protocols static route. The following set commands are used to add the routes shown above:

```
set protocols static route 0.0.0.0/0 next-hop '10.11.1.1'
set protocols static route 10.10.0.0/16 next-hop '192.168.200.2'
delete system gateway-address
```

At this point, we need to commit and save the configuration using the following commands:

```
commit
save
exit
```

Finally, we need to create a static ARP entry for IP address 192.168.200.2 that points to MAC address AA:BB:CC:DD:EE:02. To achieve this, we must execute the following commands:

```
sudo su -
echo "aa:bb:cc:dd:ee:02  remotes" >>/etc/ethers
echo "192.168.200.2  remotes" >>/etc/hosts
echo "/usr/sbin/arp -f" >>/config/scripts/vyatta-postconfig-bootup.script
/usr/sbin/arp -f
exit
```

Verify that the static ARP entry has been created by issuing the following command:

```
show arp
```

The output should contain the highlighted line:

```
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.200.2            ether   aa:bb:cc:dd:ee:02   CM                    peth1
```

The “iptables” rules are set up automatically by the firewall commands in the previous sections. The resulting iptables chains and rules are the result:

```
 vyatta@vyatta:~$ sudo iptables -n -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
VYATTA_PRE_FW_IN_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_FW_LOCAL_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_POST_FW_IN_HOOK  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
VYATTA_PRE_FW_FWD_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_FW_IN_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_FW_OUT_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_POST_FW_FWD_HOOK  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
VYATTA_PRE_FW_OUT_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_POST_FW_OUT_HOOK  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARDING (4 references)
target     prot opt source               destination
RETURN     all  --  10.10.0.0/16         0.0.0.0/0            /* FORWARDING-10 */
RETURN     all  --  0.0.0.0/0            10.10.0.0/16         /* FORWARDING-15 */
RETURN     all  --  10.11.1.0/24         0.0.0.0/0            /* FORWARDING-20 */
RETURN     all  --  0.0.0.0/0            10.11.1.0/24         /* FORWARDING-25 */
DROP       all  --  0.0.0.0/0            0.0.0.0/0            /* FORWARDING-10000
default-action drop */

Chain VYATTA_FW_IN_HOOK (1 references)
target     prot opt source               destination
FORWARDING  all  --  0.0.0.0/0            0.0.0.0/0
FORWARDING  all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_FW_LOCAL_HOOK (1 references)
target     prot opt source               destination

Chain VYATTA_FW_OUT_HOOK (1 references)
target     prot opt source               destination
FORWARDING  all  --  0.0.0.0/0            0.0.0.0/0
FORWARDING  all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_POST_FW_FWD_HOOK (1 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_POST_FW_IN_HOOK (1 references)
```

```
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_POST_FW_OUT_HOOK (1 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_PRE_FW_FWD_HOOK (1 references)
target     prot opt source               destination
RETURN     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_PRE_FW_IN_HOOK (1 references)
target     prot opt source               destination
RETURN     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_PRE_FW_OUT_HOOK (1 references)
target     prot opt source               destination
RETURN     all  --  0.0.0.0/0            0.0.0.0/0
```

To verify the configuration, issue the “show ip route” command.  The routing table should be populated as follows:

```
vyatta@vyatta:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

S>* 0.0.0.0/0 [1/0] via 10.11.1.1, eth0
S>* 10.10.0.0/16 [1/0] via 192.168.200.2, peth1
C>* 10.11.1.0/24 is directly connected, eth0
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.168.200.0/24 is directly connected, peth1
```

The other values can be set as desired. For example if desired the SNMP daemon can be enabled if SNMP control is required.

### *2.0.3.1.4 HUB Receive only Demodulator & I5-I7 Setup*

Each hub based receive only demodulator requires setting its receive frequency and parameters to match those for the corresponding remote transmit modulator frequency and parameters.

Note in the following parameters that a variable “REMx” is given and is replaced by the remote number as desired. Each remote must have a unique IP address. In this case the remote number plus 2 is used because the router and common transmit use the first 2 addresses.

The I5-I7 configwiz has the following relevant parameters set.

```
firewall {
    name FORWARDING {
        default-action drop
        rule 10 {
            action accept
            destination {
                address 0.0.0.0/0
            }
            log disable
            protocol all
            source {
                address 10.10.0.0/16
            }
        }
        rule 15 {
            action accept
            destination {
                address 10.10.0.0/16
            }
            log disable
            protocol all
            source {
                address 0.0.0.0/0
            }
        }
        rule 20 {
            action accept
            destination {
                address 0.0.0.0/0
            }
            log disable
            protocol all
            source {
                address 10.11.1.0/24
            }
        }
        rule 25 {
            action accept
            destination {
                address 10.11.1.0/24
            }
            log disable
            protocol all
            source {
                address 0.0.0.0/0
            }
        }
    }
}
interfaces {
    ethernet eth0 {
        address 10.11.1.[REMx+2]/24
        description "LAN Interface"
        duplex auto
        firewall {
            in {
                name FORWARDING
            }
            out {
                name FORWARDING
            }
        }
        smp_affinity auto
        speed auto
    }
    ethernet eth1 {
        duplex auto
        smp_affinity auto
        speed auto
    }
    loopback lo {
    }
    pseudo-ethernet peth1 {
        address 192.168.200.1/24
        description "Satellite WAN Interface"
        firewall {
            in {
                name FORWARDING
            }
            out {
                name FORWARDING
            }
        }
        link eth1
        mac aa:bb:cc:dd:ee:01
        mode private
    }
}
protocols {
    static {
        route 0.0.0.0/0 {
            next-hop 10.11.1.1 {
            }
        }
        route 10.10.0.0/16 {
            next-hop 192.168.200.2 {
            }
        }
    }
}
```

First, we need to set the following firewall rules. They must exist before they can be applied to an interface.

```
firewall {
    name FORWARDING {
        default-action drop
        rule 10 {
            action accept
            destination {
                address 0.0.0.0/0
            }
            log disable
            protocol all
            source {
                address 10.10.0.0/16
            }
        }
        rule 15 {
            action accept
            destination {
                address 10.10.0.0/16
            }
            log disable
            protocol all
            source {
                address 0.0.0.0/0
            }
        }
        rule 20 {
            action accept
            destination {
                address 0.0.0.0/0
            }
            log disable
            protocol all
            source {
                address 10.11.1.0/24
            }
        }
        rule 25 {
            action accept
            destination {
                address 10.11.1.0/24
            }
            log disable
            protocol all
            source {
                address 0.0.0.0/0
            }
        }
    }
}
```

To configure the firewall rules shown above, we use the following set commands:

```
set firewall name FORWARDING default-action 'drop'
set firewall name FORWARDING rule 10 action 'accept'
set firewall name FORWARDING rule 10 destination address '0.0.0.0/0'
set firewall name FORWARDING rule 10 log 'disable'
set firewall name FORWARDING rule 10 protocol 'all'
set firewall name FORWARDING rule 10 source address '10.10.0.0/16'
set firewall name FORWARDING rule 15 action 'accept'
set firewall name FORWARDING rule 15 destination address '10.10.0.0/16'
set firewall name FORWARDING rule 15 log 'disable'
set firewall name FORWARDING rule 15 protocol 'all'
set firewall name FORWARDING rule 15 source address '0.0.0.0/0'
set firewall name FORWARDING rule 20 action 'accept'
set firewall name FORWARDING rule 20 destination address '0.0.0.0/0'
set firewall name FORWARDING rule 20 log 'disable'
set firewall name FORWARDING rule 20 protocol 'all'
set firewall name FORWARDING rule 20 source address '10.11.1.0/24'
set firewall name FORWARDING rule 25 action 'accept'
set firewall name FORWARDING rule 25 destination address '10.11.1.0/24'
set firewall name FORWARDING rule 25 log 'disable'
set firewall name FORWARDING rule 25 protocol 'all'
set firewall name FORWARDING rule 25 source address '0.0.0.0/0'
```

Next, we configure our interfaces.

Note here that eth0 is the LAN side interface and peth1 is the WAN (modem) side interface. There is really no configuration applied to the two interfaces other than the static-routes.

If the interfaces are set by default as part of a bridge interface, the bridge must be removed prior to configuring the interfaces in routed mode. To remove interfaces eth0 and eth1 from bridge mode, issue the following commands:

```
configure
del interfaces ethernet eth0 bridge-group
del interfaces ethernet eth1 bridge-group
del interfaces bridge br0
commit
save
```

Next, we need to apply the following configuration to interface eth0:

```
ethernet eth0 {
    address 10.11.1.[REMx+2]/24
    description "LAN Interface"
    duplex auto
    firewall {
        in {
            name FORWARDING
        }
        out {
            name FORWARDING
        }
    }
    smp_affinity auto
    speed auto
}
```

The following set commands are used to set the configuration shown above for interface eth0:

```
set interfaces ethernet eth0 address '10.11.1.[REMx+2]/24'
set interfaces ethernet eth0 description 'LAN Interface'
set interfaces ethernet eth0 duplex 'auto'
set interfaces ethernet eth0 firewall in name 'FORWARDING'
set interfaces ethernet eth0 firewall out name 'FORWARDING'
set interfaces ethernet eth0 smp_affinity 'auto'
set interfaces ethernet eth0 speed 'auto'
```

Next, we need to apply the following configuration to interface peth1:

```
  pseudo-ethernet peth1 {
    address 192.168.200.1/24
    description "Satellite WAN Interface"
    firewall {
      in {
        name FORWARDING
      }
      out {
        name FORWARDING
      }
    }
    link eth1
    mac aa:bb:cc:dd:ee:01
    mode private
  }
```

As shown above, interface peth1 is actually a pseudo-ethernet interface that is linked to physical Ethernet interface eth1.

The following set commands are used to set the configuration shown above for interface peth1:

```
set interfaces pseudo-ethernet peth1 address '192.168.200.1/24'
set interfaces pseudo-ethernet peth1 description 'Satellite WAN Interface'
set interfaces pseudo-ethernet peth1 firewall in name 'FORWARDING'
set interfaces pseudo-ethernet peth1 firewall out name 'FORWARDING'
set interfaces pseudo-ethernet peth1 link 'eth1'
set interfaces pseudo-ethernet peth1 mac 'aa:bb:cc:dd:ee:01'
set interfaces pseudo-ethernet peth1 mode 'private'
```

Next, we need to apply the following static routes:

```
protocols {
  static {
    route 0.0.0.0/0 {
      next-hop 10.11.1.1 {
      }
    }
    route 10.10.0.0/16 {
      next-hop 192.168.200.2 {
      }
    }
  }
}
```

For consistency, all static routes (including the system gateway) are set in protocols static route. The following set commands are used to add the routes shown above:

```
set protocols static route 0.0.0.0/0 next-hop '10.11.1.1'
set protocols static route 10.10.0.0/16 next-hop '192.168.200.2'
delete system gateway-address
```

At this point, we need to commit and save the configuration using the following commands:

```
commit
save
exit
```

Finally, we need to create a static ARP entry for IP address 192.168.200.2 that points to MAC address AA:BB:CC:DD:EE:02. To achieve this, we must execute the following commands:

```
sudo su -
echo "aa:bb:cc:dd:ee:02  remotes" >>/etc/ethers
echo "192.168.200.2  remotes" >>/etc/hosts
echo "/usr/sbin/arp -f" >>/config/scripts/vyatta-postconfig-bootup.script
/usr/sbin/arp -f
exit
```

Verify that the static ARP entry has been created by issuing the following command:

```
show arp
```

The output should contain the highlighted line:

```
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.200.2            ether   aa:bb:cc:dd:ee:02   CM                    peth1
```

The “iptables” rules are set up automatically by the firewall commands in the previous sections. The resulting iptables chains and rules are the result:

```
 vyatta@vyatta:~$ sudo iptables -n -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
VYATTA_PRE_FW_IN_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_FW_LOCAL_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_POST_FW_IN_HOOK  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
VYATTA_PRE_FW_FWD_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_FW_IN_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_FW_OUT_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_POST_FW_FWD_HOOK  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
VYATTA_PRE_FW_OUT_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_POST_FW_OUT_HOOK  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARDING (4 references)
target     prot opt source               destination
RETURN     all  --  10.10.0.0/16         0.0.0.0/0            /* FORWARDING-10 */
RETURN     all  --  0.0.0.0/0            10.10.0.0/16         /* FORWARDING-15 */
RETURN     all  --  10.11.1.0/24         0.0.0.0/0            /* FORWARDING-20 */
RETURN     all  --  0.0.0.0/0            10.11.1.0/24         /* FORWARDING-25 */
DROP       all  --  0.0.0.0/0            0.0.0.0/0            /* FORWARDING-10000
default-action drop */

Chain VYATTA_FW_IN_HOOK (1 references)
```

```
target     prot opt source               destination
FORWARDING  all  --  0.0.0.0/0            0.0.0.0/0
FORWARDING  all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_FW_LOCAL_HOOK (1 references)
target     prot opt source               destination

Chain VYATTA_FW_OUT_HOOK (1 references)
target     prot opt source               destination
FORWARDING  all  --  0.0.0.0/0            0.0.0.0/0
FORWARDING  all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_POST_FW_FWD_HOOK (1 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_POST_FW_IN_HOOK (1 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_POST_FW_OUT_HOOK (1 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_PRE_FW_FWD_HOOK (1 references)
target     prot opt source               destination
RETURN     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_PRE_FW_IN_HOOK (1 references)
target     prot opt source               destination
RETURN     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_PRE_FW_OUT_HOOK (1 references)
target     prot opt source               destination
RETURN     all  --  0.0.0.0/0            0.0.0.0/0
```

To verify the configuration, issue the “show ip route” command. The routing table should be populated as follows:

```
vyatta@vyatta:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

S>* 0.0.0.0/0 [1/0] via 10.11.1.1, eth0
S>* 10.10.0.0/16 [1/0] via 192.168.200.2, peth1
C>* 10.11.1.0/24 is directly connected, eth0
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.168.200.0/24 is directly connected, peth1
```

The other values can be set as desired. For example if desired the SNMP daemon can be enabled if SNMP control is required.

### 2.0.3.2 Remote Equipment Configuration

#### *2.0.3.2.1 Remote Modem & I5-I7 Setup:*

Each remote requires setting its receive frequency and parameters to the same as the common hub outbound and the transmit frequency and parameters to match those for the corresponding demodulator at the hub.

Note in the following parameters that a variable “[REMx]” is given and is replaced by the remote number as desired. Each remote must have a unique LAN IP Network.

The I5-I7 configwiz has the following relevant parameters set.

```
firewall {
    name LAN-FORWARDING {
        default-action drop
        rule 10 {
            action accept
            log disable
            protocol all
            source {
                address 10.10.[REMx].0/24
            }
        }
        rule 15 {
            action accept
            destination {
                address 10.10.[REMx].0/24
            }
            log disable
            protocol all
        }
    }
    name WAN-IN {
        default-action drop
        rule 10 {
            action accept
            destination {
                address 10.10.[REMx].0/24
            }
            log disable
            protocol all
        }
        rule 15 {
            action drop
            log disable
            protocol all
            source {
                address 10.10.[REMx].0/24
            }
        }
    }
    name WAN-OUT {
        default-action drop
        rule 10 {
            action accept
            log disable
            protocol all
            source {
                address 10.10.[REMx].0/24
            }
        }
    }
}
interfaces {
    ethernet eth0 {
        address 10.10.[REMx].1/24
        description "LAN Interface"
        duplex auto
        firewall {
            in {
                name LAN-FORWARDING
            }
            out {
                name LAN-FORWARDING
            }
        }
        smp_affinity auto
        speed auto
    }
    ethernet eth1 {
        duplex auto
        smp_affinity auto
        speed auto
    }
    loopback lo {
    }
    pseudo-ethernet peth1 {
        address 192.168.200.2/24
        description "Satellite WAN Interface"
        firewall {
            in {
                name WAN-IN
            }
            out {
                name WAN-OUT
            }
        }
        link eth1
        mac aa:bb:cc:dd:ee:02
        mode private
    }
}
protocols {
    static {
        route 0.0.0.0/0 {
            next-hop 192.168.200.1 {
            }
        }
    }
}
```

First, we need to set the following firewall rules.  They must exist before they can be applied to an interface.

```
firewall {
    name LAN-FORWARDING {
        default-action drop
        rule 10 {
            action accept
            log disable
            protocol all
            source {
                address 10.10.[REMx].0/24
            }
        }
        rule 15 {
            action accept
            destination {
                address 10.10.[REMx].0/24
            }
            log disable
            protocol all
        }
    }
    name WAN-IN {
        default-action drop
        rule 10 {
            action accept
            destination {
                address 10.10.[REMx].0/24
            }
            log disable
            protocol all
        }
        rule 15 {
            action drop
            log disable
            protocol all
            source {
                address 10.10.[REMx].0/24
            }
        }
    }
    name WAN-OUT {
        default-action drop
        rule 10 {
            action accept
            log disable
            protocol all
            source {
                address 10.10.[REMx].0/24
            }
        }
    }
}
```

To configure the firewall rules shown above, we use the following set commands:

```
set firewall name LAN-FORWARDING default-action 'drop'
set firewall name LAN-FORWARDING rule 10 action 'accept'
set firewall name LAN-FORWARDING rule 10 log 'disable'
set firewall name LAN-FORWARDING rule 10 protocol 'all'
set firewall name LAN-FORWARDING rule 10 source address '10.10.[REMx].0/24'
set firewall name LAN-FORWARDING rule 15 action 'accept'
set firewall name LAN-FORWARDING rule 15 destination address '10.10.[REMx].0/24'
set firewall name LAN-FORWARDING rule 15 log 'disable'
set firewall name LAN-FORWARDING rule 15 protocol 'all'
set firewall name WAN-IN default-action 'drop'
set firewall name WAN-IN rule 10 action 'accept'
set firewall name WAN-IN rule 10 destination address '10.10.[REMx].0/24'
set firewall name WAN-IN rule 10 log 'disable'
set firewall name WAN-IN rule 10 protocol 'all'
set firewall name WAN-IN rule 15 action 'drop'
set firewall name WAN-IN rule 15 log 'disable'
set firewall name WAN-IN rule 15 protocol 'all'
set firewall name WAN-IN rule 15 source address '10.10.[REMx].0/24'
set firewall name WAN-OUT default-action 'drop'
set firewall name WAN-OUT rule 10 action 'accept'
set firewall name WAN-OUT rule 10 log 'disable'
set firewall name WAN-OUT rule 10 protocol 'all'
set firewall name WAN-OUT rule 10 source address '10.10.[REMx].0/24'
```

Next, we configure our interfaces.

Note here that eth0 is the LAN side interface and peth1 is the WAN (modem) side interface. There is really no configuration applied to the two interfaces other than the static-routes.

If the interfaces are set by default as part of a bridge interface, the bridge must be removed prior to configuring the interfaces in routed mode. To remove interfaces eth0 and eth1 from bridge mode, issue the following commands:

```
configure
del interfaces ethernet eth0 bridge-group del interfaces ethernet eth1 bridge-group
del interfaces bridge br0
commit
save
```

Next, we need to apply the following configuration to interface eth0:

```
ethernet eth0 {
    address 10.10.[REMx].1/24
    description "LAN Interface"
    duplex auto
    firewall {
        in {
            name LAN-FORWARDING
        }
        out {
            name LAN-FORWARDING
        }
    }
    smp_affinity auto
    speed auto
}
```

The following set commands are used to set the configuration shown above for interface eth0:

```
set interfaces ethernet eth0 address '10.10. [REMx].1/24'
set interfaces ethernet eth0 description 'LAN Interface'
set interfaces ethernet eth0 duplex 'auto'
set interfaces ethernet eth0 firewall in name 'LAN-FORWARDING'
set interfaces ethernet eth0 firewall out name 'LAN-FORWARDING'
set interfaces ethernet eth0 smp_affinity 'auto'
set interfaces ethernet eth0 speed 'auto'
```

Next, we need to apply the following configuration to interface peth1:

```
pseudo-ethernet peth1 {
    address 192.168.200.2/24
    description "Satellite WAN Interface"
    firewall {
        in {
            name WAN -IN
        }
        out {
            name WAN-OUT
        }
    }
    link eth1
    mac aa:bb:cc:dd:ee:02
    mode private
}
```

As shown above, interface peth1 is actually a pseudo-ethernet interface that is linked to physical Ethernet interface eth1.

The following set commands are used to set the configuration shown above for interface peth1:

```
set interfaces pseudo-ethernet peth1 address '192.168.200.2/24'
set interfaces pseudo-ethernet peth1 description 'Satellite WAN Interface'
set interfaces pseudo-ethernet peth1 firewall in name WAN-IN
set interfaces pseudo-ethernet peth1 firewall out name WAN-OUT
set interfaces pseudo-ethernet peth1 link 'eth1'
set interfaces pseudo-ethernet peth1 mac 'aa:bb:cc:dd:ee:02'
set interfaces pseudo-ethernet peth1 mode 'private'
```

Next, we need to apply the following static routes:

```
protocols {
    static {
        route 0.0.0.0/0 {
            next-hop 192.168.200.1 {
            }
        }
    }
}
```

For consistency, all static routes (including the system gateway) are set in protocols static route. The following set commands are used to add the routes shown above:

```
set protocols static route 0.0.0.0/0 next-hop '192.168.200.1'
delete system gateway-address
```

At this point, we need to commit and save the configuration using the following commands:

```
commit
save
exit
```

Finally, we need to create a static ARP entry for IP address 192.168.200.1 that points to MAC address AA:BB:CC:DD:EE:01. To achieve this, we must execute the following commands:

```
sudo su -
echo "aa:bb:cc:dd:ee:01  hub" >>/etc/ethers
echo "192.168.200.1  hub" >>/etc/hosts
echo "/usr/sbin/arp -f" >>/config/scripts/vyatta-postconfig-bootup.script
/usr/sbin/arp -f
exit
```

Verify that the static ARP entry has been created by issuing the following command:

```
show arp
```

The output should contain the highlighted line:

```
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.200.1            ether   aa:bb:cc:dd:ee:01   CM                    peth1
```

The "iptables" rules are set up automatically by the firewall commands in the previous sections. The resulting iptables chains and rules are the result:

```
 vyatta@vyatta:~$ sudo iptables -n –L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
VYATTA_PRE_FW_IN_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_FW_LOCAL_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_POST_FW_IN_HOOK  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
VYATTA_PRE_FW_FWD_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_FW_IN_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_FW_OUT_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_POST_FW_FWD_HOOK  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
```

```
target     prot opt source               destination
VYATTA_PRE_FW_OUT_HOOK  all  --  0.0.0.0/0            0.0.0.0/0
VYATTA_POST_FW_OUT_HOOK  all  --  0.0.0.0/0            0.0.0.0/0

Chain LAN-FORWARDING (2 references)
target     prot opt source               destination
RETURN     all  --  10.10.[REMx].0/24        0.0.0.0/0            /* LAN-FORWARDING-
10 */
RETURN     all  --  0.0.0.0/0            10.10.[REMx].0/24         /* LAN-FORWARDING-
15 */
DROP       all  --  0.0.0.0/0            0.0.0.0/0            /* LAN-FORWARDING-10000
default-action drop */

Chain VYATTA_FW_IN_HOOK (1 references)
target     prot opt source               destination
WAN-IN     all  --  0.0.0.0/0            0.0.0.0/0
LAN-FORWARDING  all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_FW_LOCAL_HOOK (1 references)
target     prot opt source               destination

Chain VYATTA_FW_OUT_HOOK (1 references)
target     prot opt source               destination
WAN-OUT    all  --  0.0.0.0/0            0.0.0.0/0
LAN-FORWARDING  all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_POST_FW_FWD_HOOK (1 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_POST_FW_IN_HOOK (1 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_POST_FW_OUT_HOOK (1 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_PRE_FW_FWD_HOOK (1 references)
target     prot opt source               destination
RETURN     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_PRE_FW_IN_HOOK (1 references)
target     prot opt source               destination
RETURN     all  --  0.0.0.0/0            0.0.0.0/0

Chain VYATTA_PRE_FW_OUT_HOOK (1 references)
target     prot opt source               destination
RETURN     all  --  0.0.0.0/0            0.0.0.0/0

Chain WAN-IN (1 references)
target     prot opt source               destination
RETURN     all  --  0.0.0.0/0            10.10.[REMx].0/24         /* WAN-IN-10 */
DROP       all  --  10.10.[REMx].0/24        0.0.0.0/0            /* WAN-IN-15 */
DROP       all  --  0.0.0.0/0            0.0.0.0/0            /* WAN-IN-10000
default-action drop */

Chain WAN-OUT (1 references)
target     prot opt source               destination
RETURN     all  --  10.10.[REMx].0/24        0.0.0.0/0            /* WAN-OUT-10 */
DROP       all  --  0.0.0.0/0            0.0.0.0/0            /* WAN-OUT-10000
default-action drop */
```

To verify the configuration, issue the “show ip route” command. The routing table should be populated as follows:

```
vyatta@vyatta:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route
```

```
S>* 0.0.0.0/0 [1/0] via 192.168.200.1, peth1
C>* 10.10.[REMx].0/24 is directly connected, eth0
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.168.200.0/24 is directly connected, peth1
```

The other values can be set as desired. For example if desired the SNMP daemon can be enabled if SNMP control is required.

### 2.0.3.3 Local Computer Setup:

Local computers at the remotes and at the star hub node (switch connecting the router and I5-I7s) do not have DHCP service available in the configuration as shown. Connection into those LAN segment would therefore require that computers use static addressing of the IP Address, IP Mask, Gateway device and DNS server names. If the remote connections will go to another router before computers are added this may be fine, but if end terminal computers are intended to be connected directly then a small router, even a common home type router/gateway with DHCP and a following switch can be placed at the I5-I7 connection point. That connection is shown in the diagram for Remote D. Although it is potentially possible to have the I5-I7 act as a DHCP server, this exposes the transmission system to users able to log in and make changes. Those users would also need to know Linux procedures.

### 2.0.3.4 Point to Multi-Point Notes:

One good advantage to this system configuration is that it is possible for computers at any of the remotes to talk to computers at other remotes. The remote to remote connection does involve a “double hop” and will therefore have round trip delay over 1 second, magnifying the TCP/IP slowdown waiting for acknowledgement packets and also making possible Voice over IP communications very difficult.

Due to the numerous unidirectional links it is not always possible to ping from a local telnet session within one the the I5-I7s to other I5-I7s or computers. For example if you telnet into the I5-I7 in the common outbound transmit and attempt to ping one of the remote I5-I7s, it will fail.

Broadcasting and multicasting will also work over the point to multi-point network if it uses full TCP/IP protocols and no specific filters which would inhibit multicast packets. Multicasting must be enabled and operated in accordance with standard Linux rules as described in the separate document titled “*I5-I7 Broadcast and Multicast Guide*”.

### 2.0.3.5 Redundancy in Point to Multi-Point Systems

The common transmit presents a risk in that it represents a single point whose failure would take down the entire network of remotes. Users may desire to have a redundant unit at that and possibly other points in the setup. Below is a brief discussion of redundancy with I5-I7s, while a more detailed explanation and setups can be found in the separate document titled “***I5-I7 Redundancy Guide***”.

The M500 series of modems are designed for 1:1 redundancy with very little external equipment. Redundancy however is complicated by the use of the internal I5-I7 and requires a special setup to make it work as described below. Communications between the two redundant modems normally occurs over a “Y” data cable which also carries the user data, but in this case that cable is not present.

When the I5-I7 is the active data interface in a redundant modem a different method is used to allow the modems to communicate status information since there is no Y cable. This is accomplished using a special ribbon cable (Datum part number DRF08-084) to connect between the two modem's Remote Control Ports at J6. This cable simply swaps pins 2 and 3 between the two connector and also includes the common on pin 5.

| **Minimal Redundancy Cable Wiring** | |
|---|---|
| **Male DB9 A** | **Male DB9 B** |
| 2 | 3 |
| 3 | 2 |
| 5 | 5 |

The two modems recognize the “redundancy with I5-I7 installed” case and uses the alternate cable for redundancy status communications. The modems handle redundancy switching as normal and the I5-I7's modem polling routine detects that the modems are in redundancy mode and which one is currently “on-line”. The I5-I7s call a redundancy script located in “/etc/red-switch” each time the redundancy status changes.

The red-switch script is made separate so that it can be modified if necessary to handle special cases. In the case of redundancy with a router attached it is convenient to have a separate common IP address used by both I5-I7s for data connection and maintain the standard IP addresses for communicating with the I5-I7 for control purposes. The on-line I5-I7 creates a data address at for example 10.11.1.2 at a virtual interface port named “eth0:1” which shares the MAC and other port parameters with eth0. The off-line unit removes that virtual interface port and also uses “ebtables” to drop all packets going to the modem. There are also arp commands used by the script to update the router of the change in MAC address relative to the IP address.

The red-switch script keeps its configuration addresses in a redundancy configuration file named “/etc/config/red-config”. This file contains the addresses and masks required for redundancy. In the case of switching relative to a router as above it might have the following contents:

```
INT="eth0:1"
COMMON_IP="10.11.1.2"
NETMASK="255.255.255.0"
BROADCAST="10.11.1.255"
ROUTER_IP="10.11.1.1"
```

End of Document.