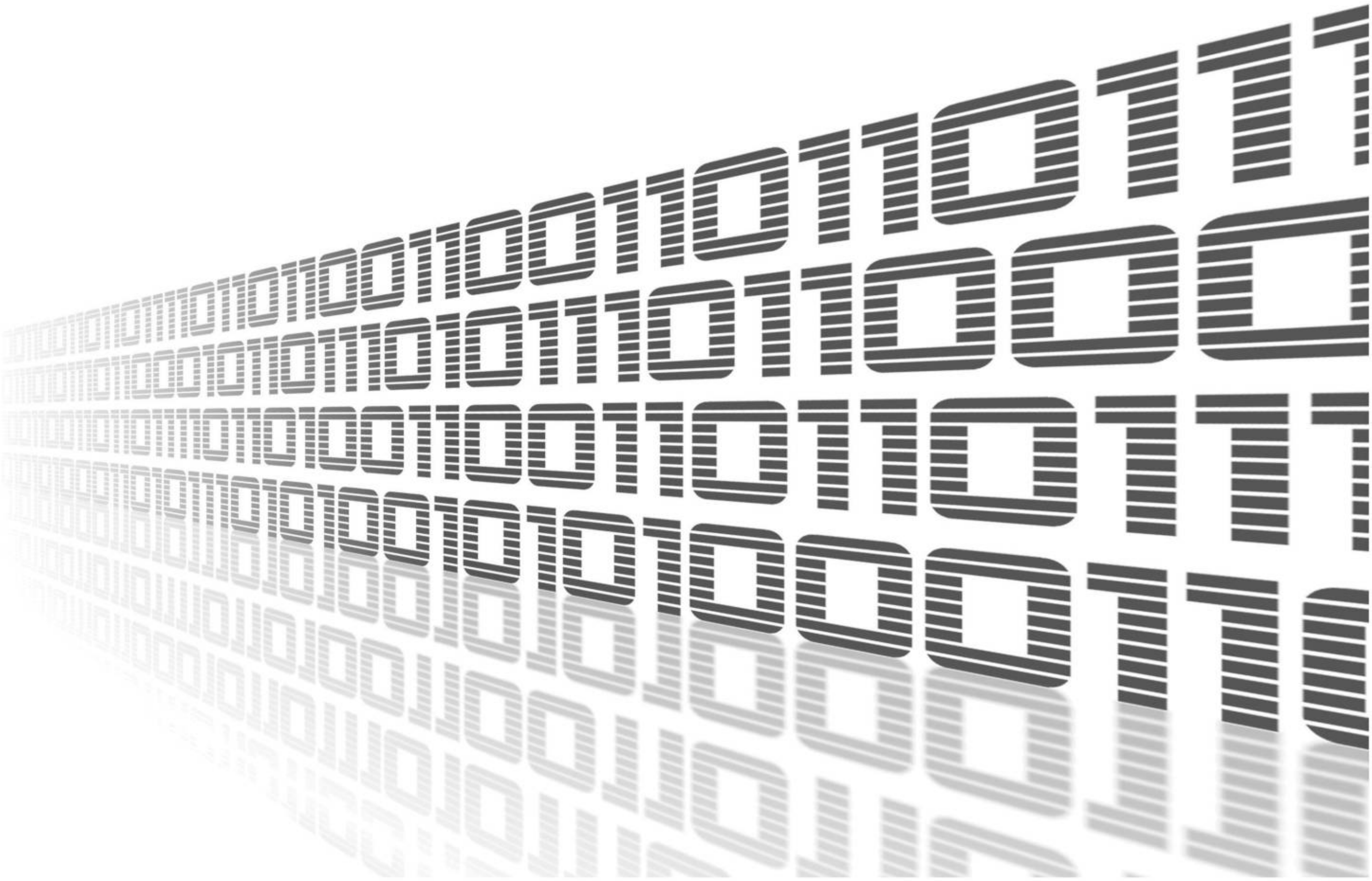
FRR



Advantech Czech s.r.o., Sokolska 71, 562 04 Usti nad Orlici, Czech Republic
Document No. APP-0100-EN, revision from 20th November, 2023.

Advantech Czech s.r.o. shall not be liable for incidental or consequential damages resulting from the furnishing, performance, or use of this manual.

All brand names used in this manual are the registered trademarks of their respective owners. The use of trademarks or other designations in this publication is for reference purposes only and does not constitute an endorsement by the trademark holder.

Used symbols

⚠ *Danger* – Information regarding user safety or potential damage to the router.

❗ *Attention* – Problems that can arise in specific situations.

ℹ *Information* – Useful tips or information of special interest.

✎ *Example* – Example of function, command or script.

Contents

List of Figures

List of Tables

1. Changelog

1.1 FRR Changelog

v1.0.0 (2020-11-20)

- First release

v1.0.1 (2021-01-19)

- Added staticd

v1.1.0 (2021-12-07)

- Upgraded to version 7.5.1
- Added LDP/MPLS support
- Aded VRF support
- For proprer MPLS function is FW 6.3.3+ needed

v1.1.1 (2022-02-01)

- Fixed FRR routing daemons stopping/restarting
- Fixed MPLS init script

v1.2.0 (2022-06-16)

- Updated FRR to version 8.2.2

2. Router App Description

2.1 Introduction

FRRouting (FRR) is an IP routing protocol suite for Linux and Unix platforms that includes protocol daemons to support some routing protocols.

Advantech has developed the *FRR* router app to extend the router's functionalities by supporting these routing protocols: BGP, IS-IS, LDP, MPLS, NHRP, OSPF, OSFP6, RIP, RIPNG, Static, VFR, and Zebra.

2.2 Installation

This router app is not installed on *Advantech* routers by default. However, you can get the `*.tgz` installation file on the *Engineering Portal*[1].

This router app can be installed to the router in the router's GUI by clicking *Customization -> Router Apps -> Add or Update* button.

[1] https://icr.advantech.com/products/software/user-modules#frr

3. Web Interface

Once the installation of the *FRR* Router App is complete, its GUI can be invoked by clicking the module name on the Router apps page of router's web interface.

Left part of this GUI contains menu with *Status* menu section, *Configuration* menu section and *Information* menu section. *Customization* menu section contains only the *Return* item, which switches back from the app's web page to the router's web configuration pages. The main menu of app's GUI is shown on Figure 2.

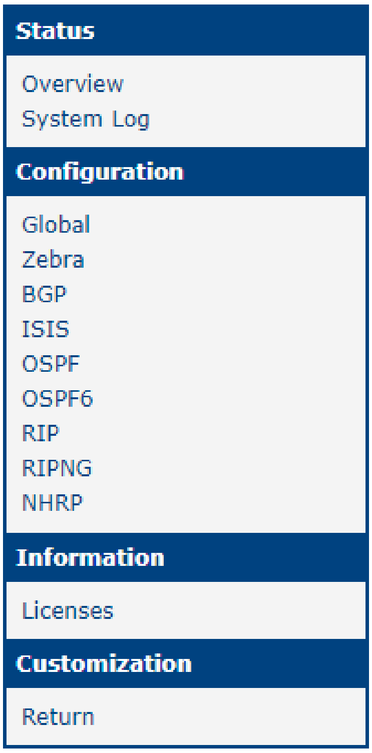
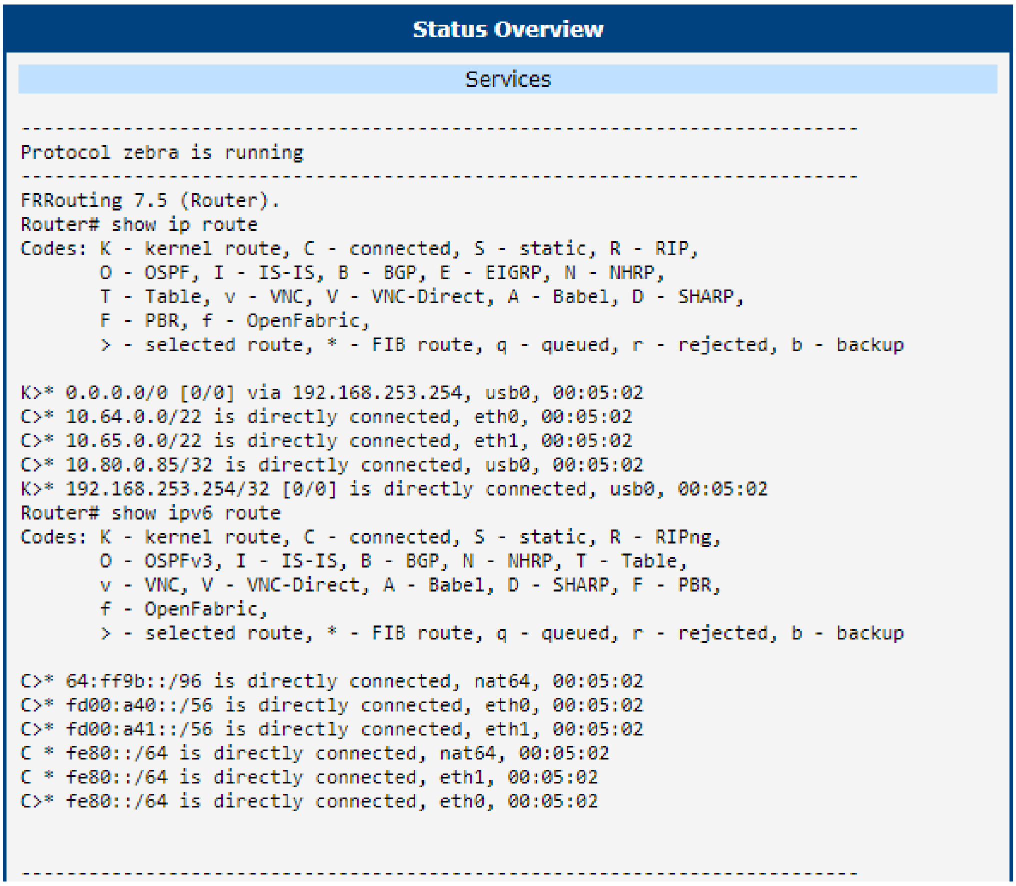


Figure 1: Menu

4. Status

In this section, in the *Overview* part, you can see the status of all protocols which can be configured via the *FRR* Router App. The figure below is an example of the Zebra protocol running.



```
---------------------------------------------------------------------------
Protocol zebra is running
---------------------------------------------------------------------------
FRRouting 7.5 (Router).
Router# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup

K>* 0.0.0.0/0 [0/0] via 192.168.253.254, usb0, 00:05:02
C>* 10.64.0.0/22 is directly connected, eth0, 00:05:02
C>* 10.65.0.0/22 is directly connected, eth1, 00:05:02
C>* 10.80.0.85/32 is directly connected, usb0, 00:05:02
K>* 192.168.253.254/32 [0/0] is directly connected, usb0, 00:05:02
Router# show ipv6 route
Codes: K - kernel route, C - connected, S - static, R - RIPng,
       O - OSPFv3, I - IS-IS, B - BGP, N - NHRP, T - Table,
       v - VNC, V - VNC-Direct, A - Babel, D - SHARP, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup

C>* 64:ff9b::/96 is directly connected, nat64, 00:05:02
C>* fd00:a40::/56 is directly connected, eth0, 00:05:02
C>* fd00:a41::/56 is directly connected, eth1, 00:05:02
C * fe80::/64 is directly connected, nat64, 00:05:02
C * fe80::/64 is directly connected, eth1, 00:05:02
C>* fe80::/64 is directly connected, eth0, 00:05:02


---------------------------------------------------------------------------
```

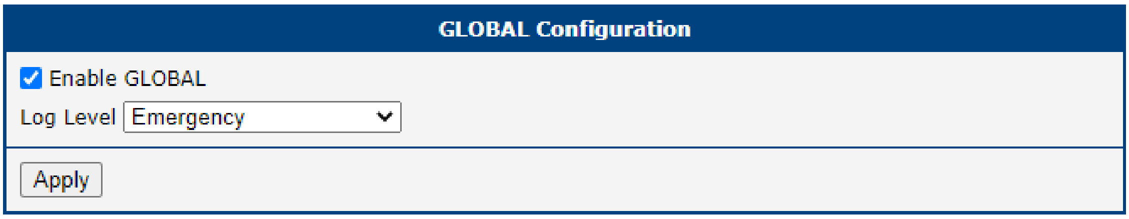
Figure 2: Status Overview Example

In the *System Log* part, you can see a copy of the system log, also available in the router *Status -> System Log*.

5. Configuration

5.1 Global

All Secure Syslog router app settings can be configured by clicking on the *Global* item in the main menu of module web interface. An overview of configurable items is given below.



GLOBAL Configuration

☑ Enable GLOBAL

Log Level: Emergency

Figure 3: Global Configuration

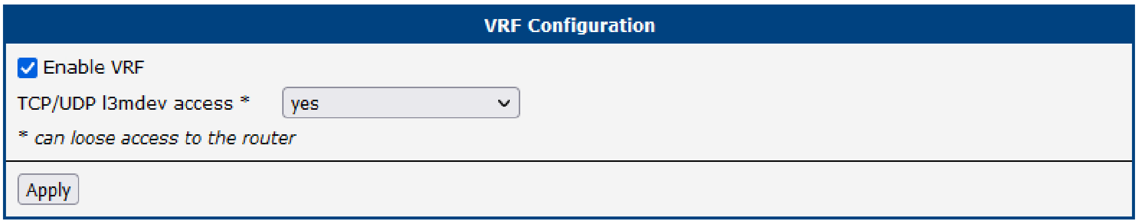
Item	Description
Enable GLOBAL	Enables FRR functionality.
Log Level	Select what level of information will appear in log

Table 1: GLOBAL Configuration items description

5.2 VRF

In IP-based computer networks, virtual routing and forwarding (VRF) is a technology that allows multiple instances of a routing table to co-exist within the same router at the same time. More about this protocol and examples can be found in the FRR online documentation[1].

There are more configuration pages for the VRF configuration under *Customization -> Router Apps -> FRR -> Configuration -> VRF* menu item. The first, see Figure 4, is for the global VRF configuration. You can enable/disable the VRF globally and enable the TCP/UD l3mdev (the L3 master device) access here as well.



VRF Configuration

☑ Enable VRF

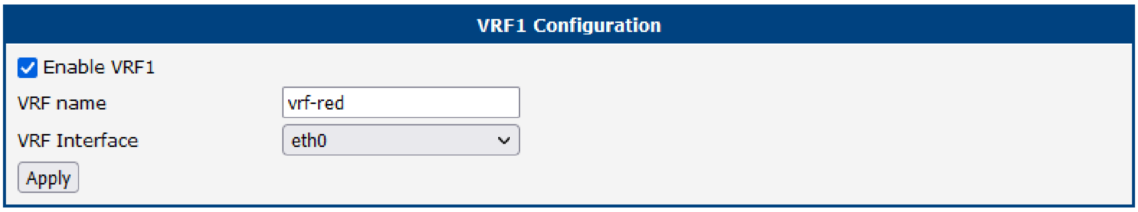
TCP/UDP l3mdev access *: yes

* can loose access to the router

Figure 4: VRF Global Configuration

[1] http://docs.frrouting.org/en/latest/zebra.html?highlight=vrf#clicmd-vrf-VRF

Next are configuration pages for individual VRF interface configurations; see Figure 5.



VRF1 Configuration

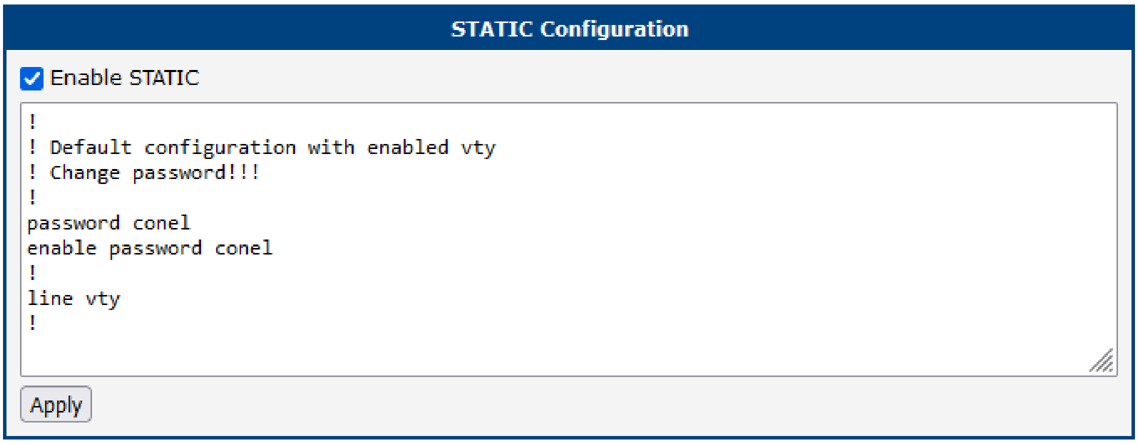
☑ Enable VRF1

VRF name	vrf-red
VRF Interface	eth0

Figure 5: VRF Interface Configuration

5.3 Static

Static routing is a form of routing that occurs when a router uses a manually-configured routing entry, rather than information from dynamic routing traffic. More about configuring and examples can be found in the FRR online documentation[1].



STATIC Configuration

☑ Enable STATIC

```
!
! Default configuration with enabled vty
! Change password!!!
!
password conel
enable password conel
!
line vty
!
```

Figure 6: Static Configuration

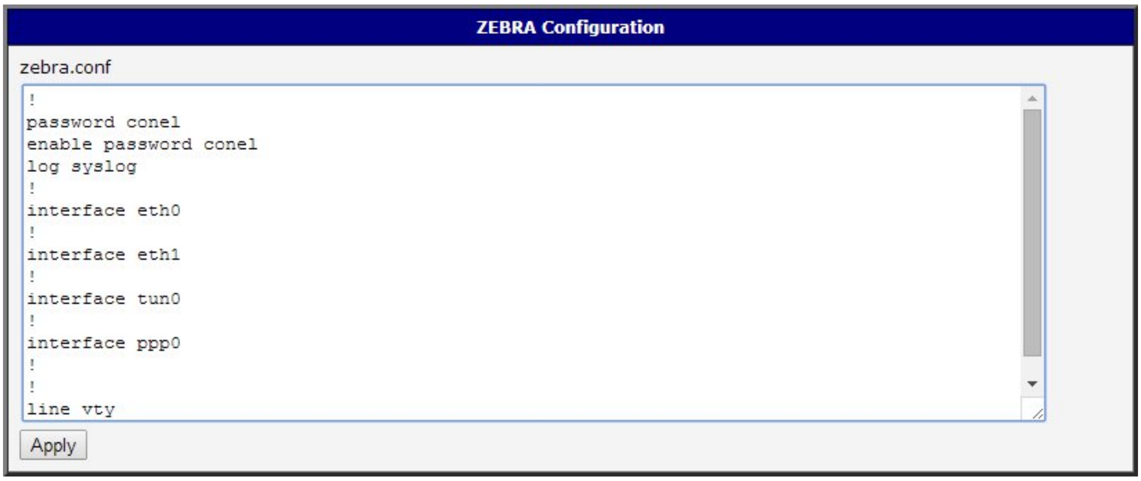
[1] http://docs.frrouting.org/en/latest/static.html

5.4 Zebra

Zebra is an IP routing manager, It provides kernel routing table updates, interface lookups, and redistribution of routes between different routing protocols. More about configuring and examples can be found in the text below or in the FRR online documentation[1].

An example of the zebra configuration file (*zebra.conf*):

```
!
password conel
enable password conel
log syslog
!
interface eth0
!
interface eth1
!
interface tun0
!
interface ppp0
!
!
line vty
!
```



```
!
password conel
enable password conel
log syslog
!
interface eth0
!
interface eth1
!
interface tun0
!
interface ppp0
!
!
line vty
```

Figure 7: Configuration of zebra deamon

[1] http://docs.frrouting.org/en/latest/zebra.html

5.5 BGP

Border Gateway Protocol (BGP) is a standardized exterior gateway protocol designed to exchange routing and reachability information between autonomous systems (AS) on the Internet. More about configuring and examples can be found in the text below or in the FRR online documentation[1].

Due to this module it is possible to used the routing between autonomous systems. These systems might be perceived as a group of IP networks and routers under the control of one or more network operators that presents a common clearly defined routing policy (only one of interior gateway protocols). The routing information is exchanged between autonomous systems via border gateway. The BGP router app is based on software called Quagga. It is a routing software package that provides TCP/IP based routing services with routing protocols support RIP, OSPF and BGP.

The Quagga is composed of several deamons. The most important is the *zebra* deamon, which collects routing information, cooperates with the system core and adjusts its routing tables. The rest of deamons including the *bgpd* deamon serves as an interface of the central deamon (zebra) for routing protocols (RIP, OSPF, BGP). Each deamon has its own configuration file.

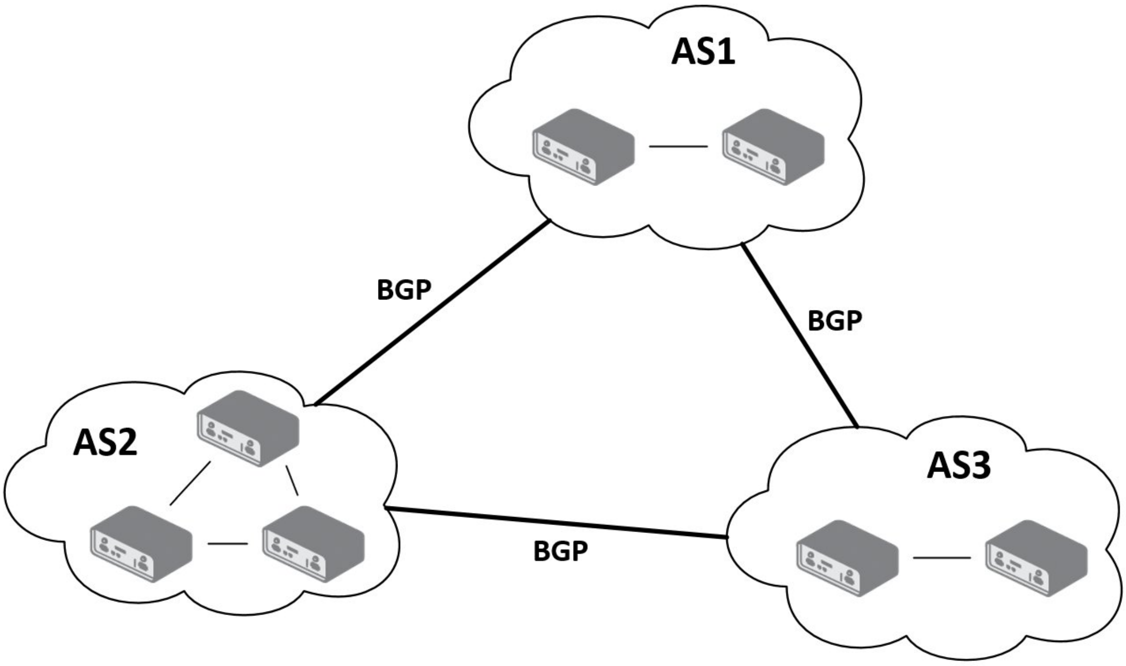


Figure 8: Model scheme

Important notices:

- Using telnet is vty interface of zebra and bgpd deamons available only via the loopback interface 127.0.0.1.
- New configuration files should be created only by an experienced user!

[1] http://docs.frrouting.org/en/latest/bgp.html

5.5.1 Example of Configuration

The figure below shows a model situation of using the *BGP* router app. Then there are mentioned examples of configuration files of *zebra* and *bgpd* deamons. In this form are entered in the configuration form in the web interface *BGP* or *ZEBRA*.

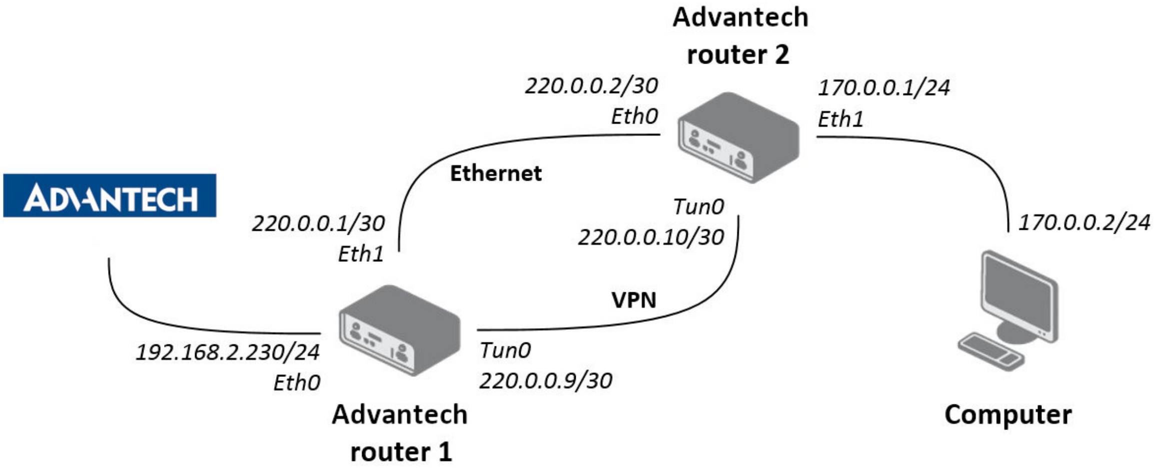


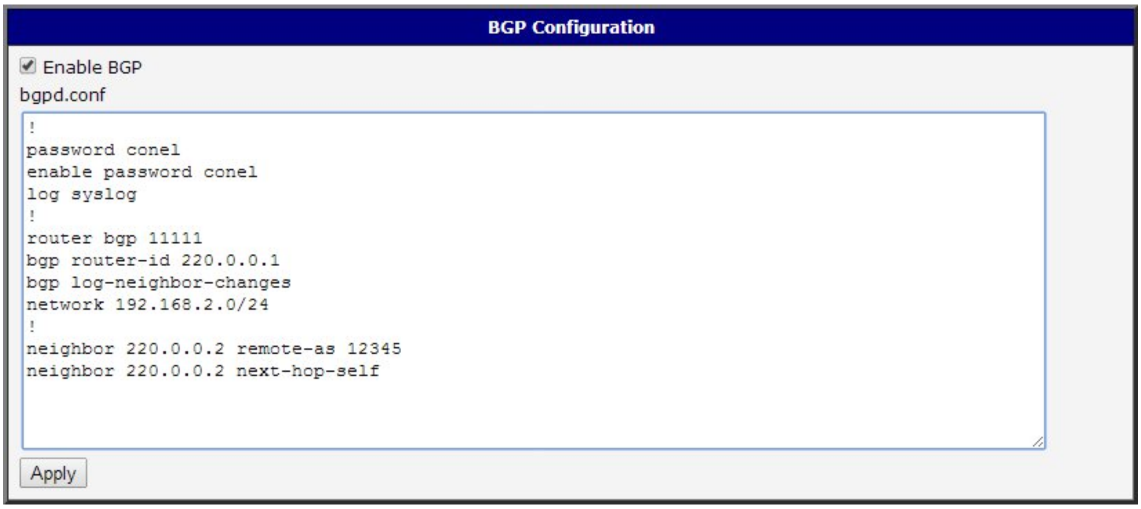
Figure 9: Example of configuration

An example of the *bgpd.conf* configuration file for a device which is referred to as *Advantech router 1* in the figure above:

```
!
password conel
enable password conel
log syslog
!
router bgp 11111
bgp router-id 220.0.0.1
bgp log-neighbor-changes
network 192.168.2.0/24
!
neighbor 220.0.0.2 remote-as 12345
neighbor 220.0.0.2 next-hop-self
```

An example of the *bgpd.conf* configuration file for a device which is referred to as *Advantech router 2* in the figure above:

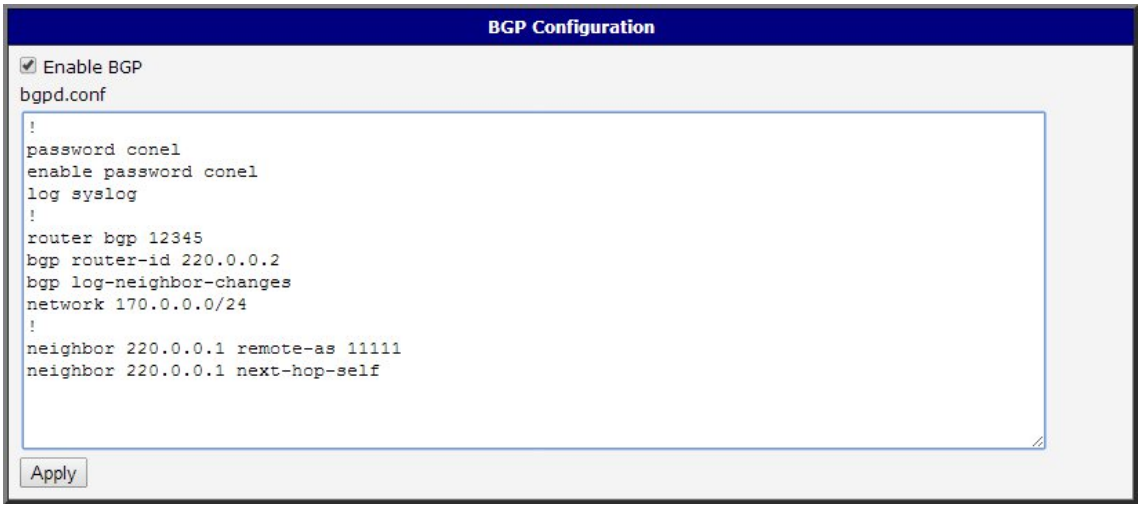
```
!
password conel
enable password conel
log syslog
!
router bgp 12345
bgp router-id 220.0.0.2
bgp log-neighbor-changes
network 170.0.0.0/24
!
neighbor 220.0.0.1 remote-as 11111
```



```
!
password conel
enable password conel
log syslog
!
router bgp 11111
bgp router-id 220.0.0.1
bgp log-neighbor-changes
network 192.168.2.0/24
!
neighbor 220.0.0.2 remote-as 12345
neighbor 220.0.0.2 next-hop-self
```

Figure 10: Configuration of bgpd deamon 1

```
neighbor 220.0.0.1 next-hop-self
```



```
!
password conel
enable password conel
log syslog
!
router bgp 12345
bgp router-id 220.0.0.2
bgp log-neighbor-changes
network 170.0.0.0/24
!
neighbor 220.0.0.1 remote-as 11111
neighbor 220.0.0.1 next-hop-self
```

Figure 11: Configuration of bgpd deamon 2

5.5.2 BGP Basic commands

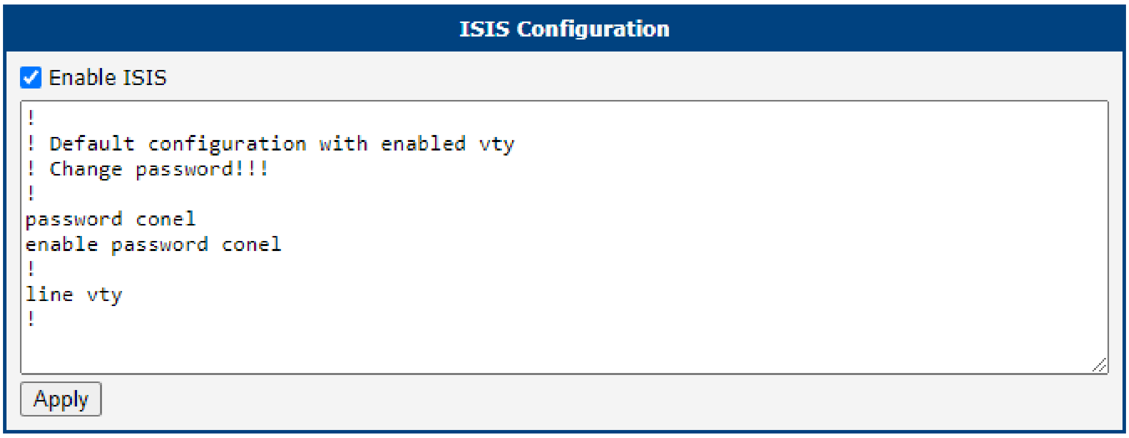
The following table lists basic commands which can be used when editing *bgpd.conf* file and description of these commands:

Item	**Description**
router bgp *<ASN>*	Configures the BGP routing process for ASN (autonomous system number)
no router bgp *<ASN>*	Removes a routing process from ASN
bgp router-id *<ip-address>*	Configures a fixed router ID for a BGP-speaking router
no bgp router-id *<ip-address>*	Removes the *bgp router-id* command from the configuration file and restore the default value of the router ID
distance bgp *<1-255><1-255> <1-255>*	Allows the use of external, internal, and local distances that could be a better route to a node
no distance bgp	Returns distances to the default values (20, 200, 200)
network *<network-number>*	Specifies the list of networks for the BGP routing process
no network *<network-number>*	Removes network from the list
aggregate-address *<address>*	Creates an aggregate entry in a BGP routing table
no aggregate-address *<address>*	Disables this function
bgp log-neighbor-changes	Enables logging of BGP neighbor resets
no bgp log-neighbor-changes	Disables logging of changes
neighbor *<ip-address/peer>* remote-as *<number>*	Adds an entry to the BGP neighbor table
no neighbor *<ip-address/peer>* remote-as *<number>*	Removes an entry from the BGP neighbor table
neighbor *<ip-address/peer>* next-hop-self	Disables next-hop processing of BGP updates on the router
no neighbor *<ip-address/peer>* next-hop-self	Disables this feature
neighbor *<ip-address/peer>* version *<version>*	Sets up the neighbor's BGP version (4, 4+, 4-)
neighbor *<name>* peer-group	Defines a new BGP peer group
no neighbor *<name>* peer-group	Removes the peer group and all of its members
show ip bgp	Displays entries in the BGP routing table

Table 2: BGP Basic commands

5.6 ISIS

IS-IS (Intermediate System – Intermediate System) is routing protocol, which is designed for the exchange of routing information between routers. More about this protocol and examples can be found in *IS-IS Application Note* [1] or in the FRR online documentation[1].



```
!
! Default configuration with enabled vty
! Change password!!!
!
password conel
enable password conel
!
line vty
!
```

Figure 12: IS-IS Configuration

5.7 OSPF & OSPF6

OSPF and OSPF6 which is IPv6 version of this protocol, are designed for exchanging routing information within an autonomous system. The OSPF is a link state protocol, which means that routers maintain a map of the network (link state database) that is updated after any change to the network topology. To computes the shortest (least cost) path between the router and all the networks is used Dijkstra's algorithm. Then these data are filled in the routing table. More about this protocol and examples can be found in the text below or in the FRR online documentation[1][2].

Due to this module the OSPF routing protocol is available. This protocol is designed for exchanging routing information within an autonomous system. The OSPF is a link state protocol, which means that routers maintain a map of the network (link state database) that is updated after any change to the network topology. To computes the shortest (least cost) path between the router and all the networks is used Dijkstra's algorithm. Then these data are filled in the routing table.

OSPF router app is based on software called Quagga. It is a routing software package that provides TCP/IP based routing services. The Quagga is composed of several deamons. The most important is the *zebra* deamon, which collects routing information, cooperates with the system core and adjusts its routing tables. The rest of deamons including the *ospfd* deamon serves as an interface of the central deamon (zebra) for routing protocols. Each deamon has its own configuration file.

[1] http://docs.frrouting.org/en/latest/isisd.html

[1] http://docs.frrouting.org/en/latest/ospfd.html

[2] http://docs.frrouting.org/en/latest/ospf6d.html

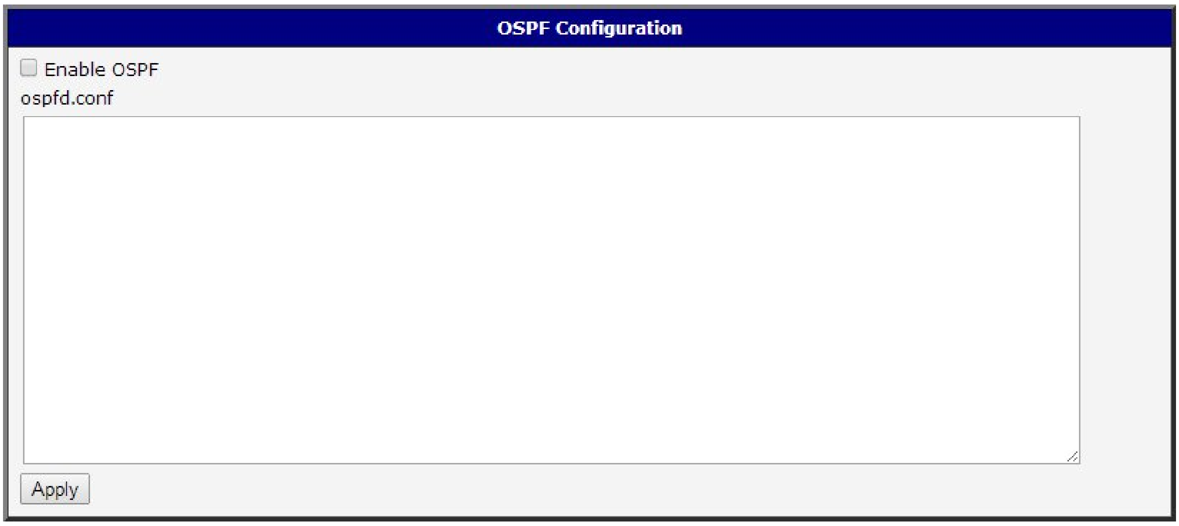


Figure 13: OSPF web interface

Important notices:

- Using telnet is vty interface of zebra and ospfd deamons available only via the loopback interface 127.0.0.1.
- New configuration files should be created only by an experienced user!

5.7.1 Example of configuration

The figure below shows a model situation of using the *OSPF* router app. Then there are mentioned examples of configuration files of *zebra* and *ospfd* deamons. In this form are entered in the configuration form in the web interface *OSPF* or *ZEBRA*.

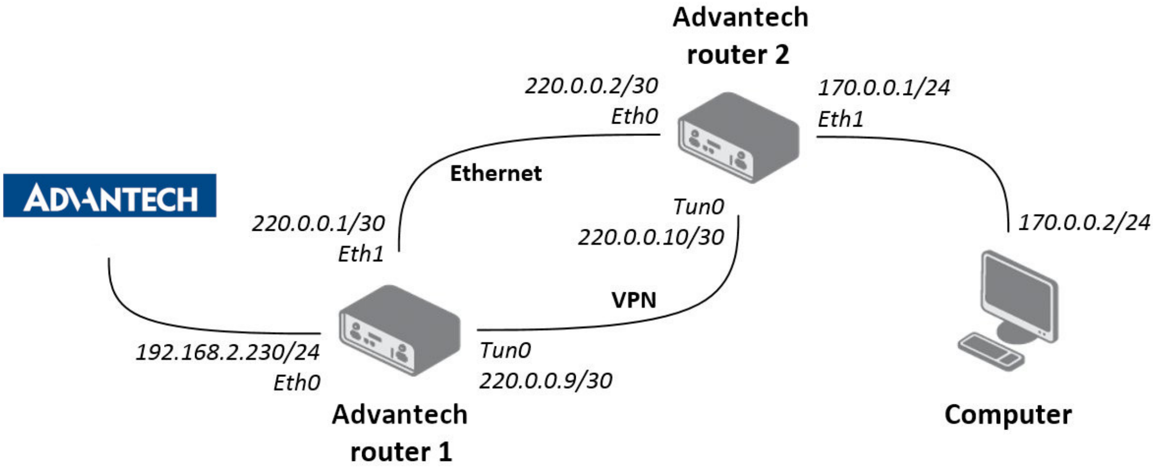


Figure 14: Example of configuration

5.7.2 IPv4 Configuration

An example of the *ospfd.conf* configuration file for a device which is referred to as *Advantech router 1* in the figure above:

```
!
password conel
enable password conel
!
log syslog
!
! interface ven
! interface eth0
! interface ppp0
! po eth
interface eth1
ip ospf cost 1
ip ospf dead-interval 40
ip ospf hello-interval 10
!
! tunelem
interface tun0
ip ospf cost 100
ip ospf dead-interval 40
ip ospf hello-interval 30
!
!
router ospf
ospf router-id 220.0.0.1
redistribute connected metric-type 1
redistribute static metric-type 1
!
network 220.0.0.0/24 area 0
!
line vty
!
```

An example of the *ospfd.conf* configuration file for a device which is referred to as *Advantech router 2* in the figure above:

```
!
password conel
enable password conel
!
log syslog
!
! interface ven
! interface eth0
! interface ppp0
! po eth
interface eth0
ip ospf cost 1
ip ospf dead-interval 40
ip ospf hello-interval 10
!
! tunelem
interface tun0
ip ospf cost 100
ip ospf dead-interval 40
ip ospf hello-interval 30
!
!
router ospf
ospf router-id 220.0.0.2
redistribute connected metric-type 1
redistribute static metric-type 1
!
network 220.0.0.0/24 area 0
!
line vty
!
```

5.7.3 IPv6 Configuration

An example of the *ospf6d.conf* configuration file for a device which is referred to as *Advantech router 1* in the figure above:

```
!
password conel
enable password conel
!
log syslog
!
interface eth1
ipv6 ospf6 instance-id 1
ipv6 ospf6 cost 1
ipv6 ospf6 dead-interval 40
ipv6 ospf6 hello-interval 10
ipv6 ospf6 retransmit-interval 5
!
interface tun0
ipv6 ospf6 instance-id 2
ipv6 ospf6 cost 1
ipv6 ospf6 dead-interval 40
ipv6 ospf6 hello-interval 10
ipv6 ospf6 retransmit-interval 5
!
!
router ospf6
router-id 220.0.0.1
redistribute connected
redistribute static
interface eth0 area 0.0.0.0
interface eth1 area 0.0.0.0
```

An example of the *ospf6d.conf* configuration file for a device which is referred to as *Advantech router 2* in the figure above:

```
!
password conel
enable password conel
!
log syslog
!
interface eth0
ipv6 ospf6 instance-id 1
ipv6 ospf6 cost 1
ipv6 ospf6 dead-interval 40
ipv6 ospf6 hello-interval 10
ipv6 ospf6 retransmit-interval 5
!
interface tun0
ipv6 ospf6 instance-id 2
ipv6 ospf6 cost 1
ipv6 ospf6 dead-interval 40
ipv6 ospf6 hello-interval 10
ipv6 ospf6 retransmit-interval 5
!
!
router ospf6
router-id 220.0.0.2
redistribute connected
redistribute static
interface eth0 area 0.0.0.0
interface eth1 area 0.0.0.0
```

5.7.4 OSPF Basic commands

The following table lists basic commands which can be used when editing *ospfd.conf* and *ospf6d.conf* files and description of these commands:

Command	Description
router ospf	Enables the OSPF process
no router ospf	Disables the OSPF process
ospf router-id <*ip-address*>	Sets the router-ID of the OSPF process
no ospf router-id	Forces OSPF to use the previous OSPF router-id behavior
log-adjacency-changes	Configures the router to send a syslog message when an OSPF neighbor goes up or down
no log-adjacency-changes	Turns off *log-adjacency-changes* function
network <*address*> area <*areaid*>	Defines the interfaces on which OSPF runs and defines the area ID for those interfaces
no network <*address*> area <*area-id*>	Disables OSPF routing for interfaces defined with *address*
area <*area-id*> range <*address mask*>	Consolidates and summarizes routes at an area boundary
no area <*area-id*> range <*address mask*>	Disables this function
area <*area-id*> authentication	Enables authentication for an OSPF area
no area <*area-id*> authentication	Removes an area's authentication
ip ospf authentication-key <*password*>	Assigns a password to be used by neighboring routers that are using OSPF's simple password authentication
no ip ospf authentication-key <*password*>	Removes a previously assigned OSPF password
ip ospf cost <*cost*>	Specifies the cost of sending packet on an interface
no ip ospf cost	Resets the path cost to the default value
ip ospf dead-interval <*seconds*>	Sets how long hello packets must not have been seen before its neighbors declare the router down
no ip ospf dead-interval	Returns to the default time
ip ospf hello-interval <*seconds*>	Specifies the interval between hello packets that are sending on the interface
no ip ospf hello-interval	Returns to the default time
ip ospf priority <*number*>	Sets the router priority (0-255)
redistribute <*protocol*>	Redistributes routes from one routing domain into another domain
no redistribute <*protocol*>	Disables redistribution
default-metric	Sets default metric values for the OSPF routing protocol

Continued on next page

Continued from previous page

Command	Description
no default-metric	Returns to the default state
show ip ospf	Displays general information about OSPF routing processes
show ip ospf interface	Displays OSPF-related interface information
show ip ospf neighbor	Displays OSPF-neighbor information

Table 3: OSPF Basic commands

5.8 RIP & RIPNG

RIP and RIPNG which is an IPv6 version of RIP, allows the routers to communicate with each other and react to changes in network topology. The RIP is a distance-vector protocol, which means that routers send each other updated routing tables (don't know the entire network topology). More about this protocol and examples can be found in the text below or in the FRR online documentation[1][2].

Due to this module the RIP routing protocol is available. Allows the routers to communicate with each other and react to changes in network topology. The RIP is a distance-vector protocol, which means that routers send each other updated routing tables (don't know the entire network topology). Searching the shortest paths in the network is based on the Bellman-Ford's algorithm. The decisive factor is the number of routers leading to the destination network. In terms of safety (protection against routing loops), this number is limited to 15. However, this maximum also limits the size of a network.

RIP router app is based on software called Quagga. It is a routing software package that provides TCP/IP based routing services. The Quagga is composed of several deamons. The most important is the *zebra* deamon, which collects routing information, cooperates with the system core and adjusts its routing tables. The rest of deamons including the *ripd* deamon serves as an interface of the central deamon (zebra) for routing protocols. Each deamon has its own configuration file.

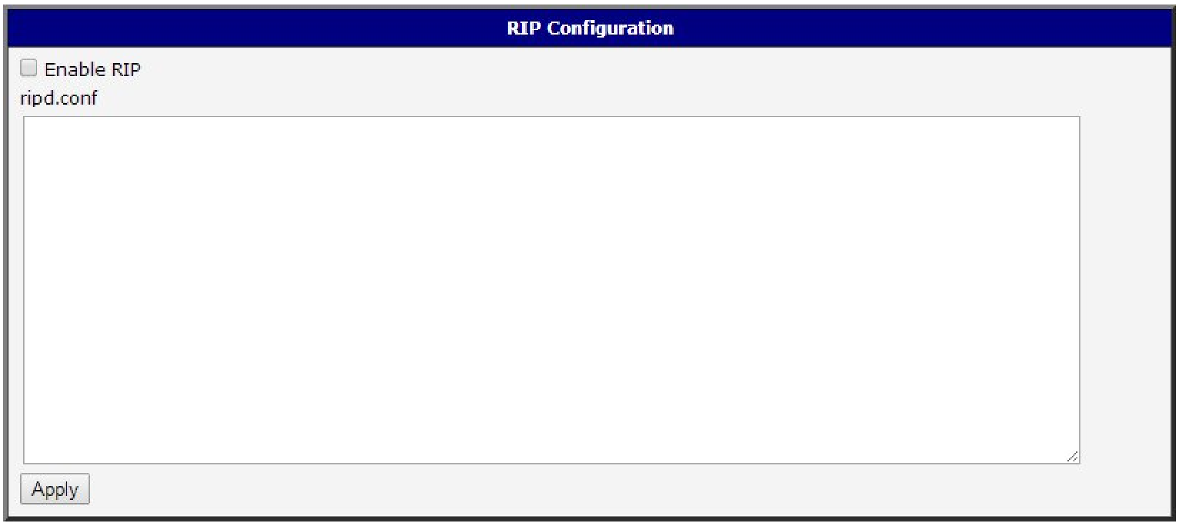


Figure 15: RIP web interface

Important notices:

- Using telnet is vty interface of zebra and ripd deamons available only via the loopback interface 127.0.0.1.
- New configuration files should be created only by an experienced user!

5.8.1 Example of configuration

The figure below shows a model situation of using the *RIP* router app. Then there are mentioned examples of configuration files of *zebra* and *ripd* deamons. In this form are entered in the configuration form in

[1] http://docs.frrouting.org/en/latest/ripd.html

[2] http://docs.frrouting.org/en/latest/ripngd.html

the web interface *RIP* or *ZEBRA*.

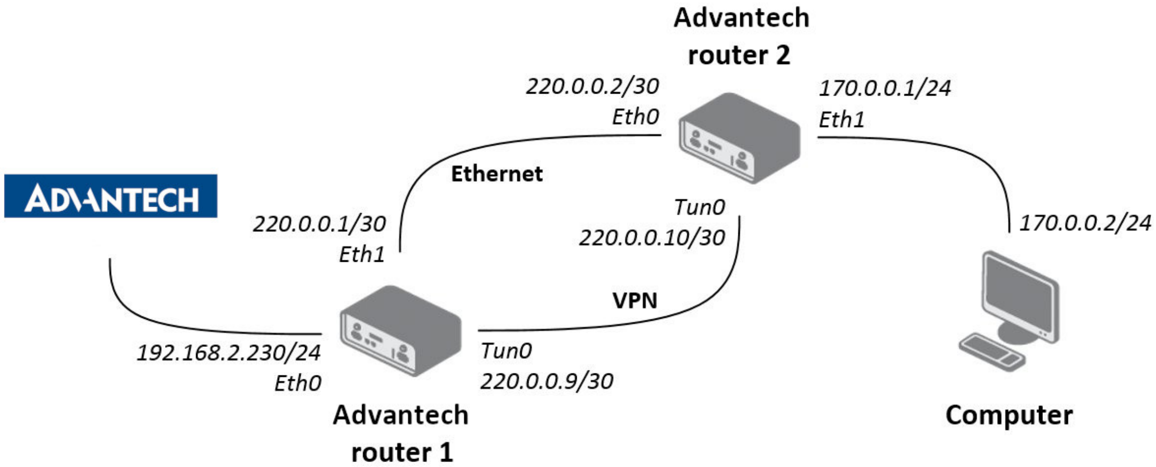


Figure 16: Example of configuration

5.8.2 IPv4 Configuration

An example of the *ripd.conf* configuration file for a device which is referred to as *Advantech router 1* in the figure above:

```
!
password conel
enable password conel
log syslog
!
interface eth0
!
interface eth1
!
interface ppp0
!
interface tun0
!
router rip
version 2
network eth0
network eth1
network tun0
passive-interface eth0
!
line vty
!
```

An example of the *ripd.conf* configuration file for a device which is referred to as *Advantech router 2* in the figure above:

```
!
password conel
enable password conel
log syslog
!
interface eth0
!
interface eth1
!
interface ppp0
!
interface tun0
!
router rip
version 2
network eth0
network eth1
network tun0
! passive-interface eth1
!
line vty
!
```

5.8.3 IPv6 Configuration

An example of the *ripngd.conf* configuration file for a device which is referred to as *Advantech router 1* in the figure above:

```
!
password conel
enable password conel
log syslog
!
router ripng
!
network eth0
network eth1
!
passive-interface eth0
!
```

An example of the *ripngd.conf* configuration file for a device which is referred to as *Advantech router 2* in the figure above:

```
!
password conel
enable password conel
log syslog
!
router ripng
!
network eth0
network eth1
!
! passive-interface eth1
!
```

5.8.4 RIP Basic commands

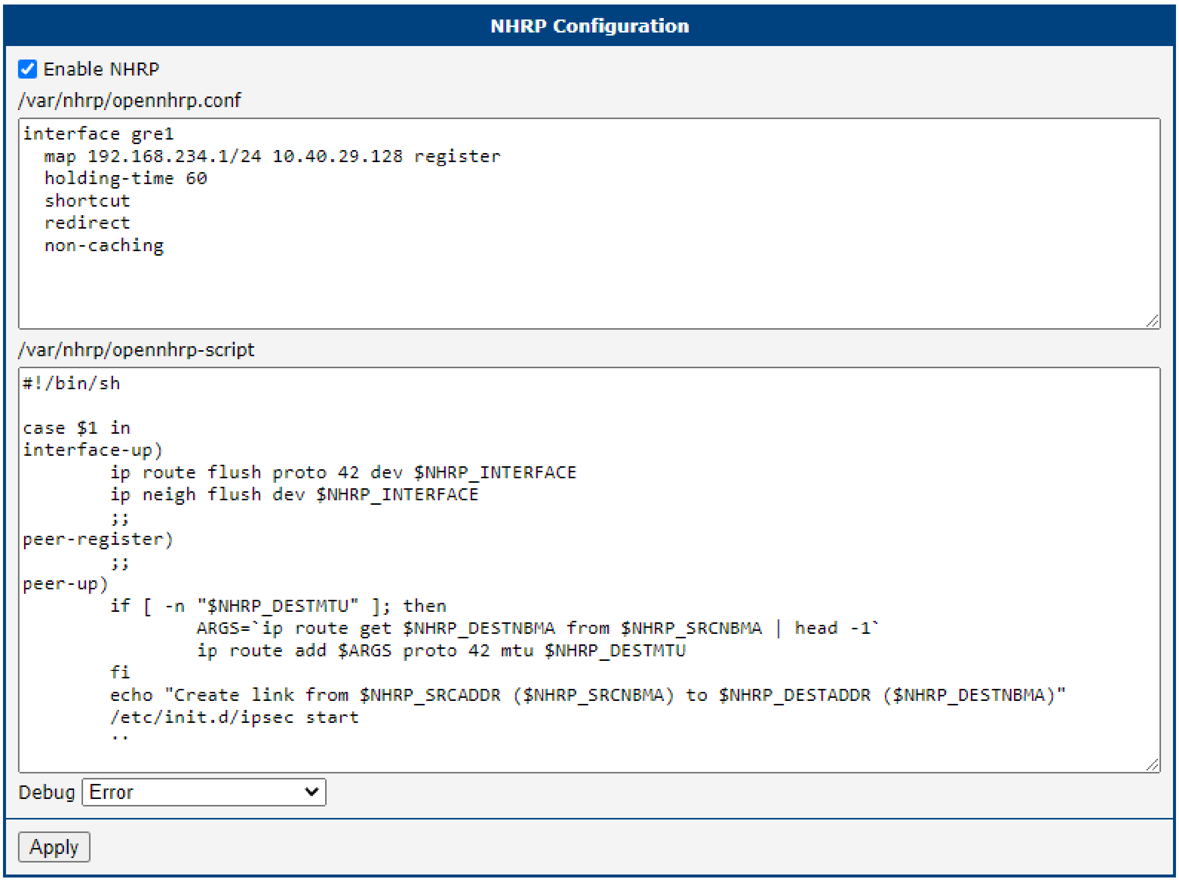
The following table lists basic commands which can be used when editing *ripd.conf* and *ripngd.conf* files and description of these commands:

Command	Description
router rip	necessary command to enable RIP
no router rip	disables RIP
network <network>	sets the RIP enable interface by specified network
no network <network>	disables RIP for the specified network
network <ifname>	both the sending and receiving of RIP packets will be enabled on the port specified in this command
no network <ifname>	disables RIP on the specified interface
neighbor <ip-address>	defines a neighboring router with which to exchange routing information
no neighbor <ip-address>	disables the RIP neighbor
passive-interface <ifname>	sets the specified interface to passive mode, i.e. disables sending routing updates on an interface
passive-interface default	sets all inerfaces to passive mode
no passive-interface <ifname>	sets the specified interface to normal mode
ip split-horizon	enables the split horizon mechanism (information about the routing is never sent back on the same interface)
no ip split-horizon	disables the split horizon mechanism (enabled on each interface by default)
version <version>	specifies a RIP version used globally by the router (it can be either 1 or 2)
no version	resets the global version setting back to the default
ip rip send version <version>	specifies a RIP version to send on an interface basis
ip rip receive version <version>	specifies a RIP version to receive on an interface basis
show ip rip	shows RIP routes
show ip protocols	displays the parameters and current state of the active routing protocol process

Table 4: RIP Basic commands

5.9 NHRP

The Next Hop Resolution Protocol (NHRP) is an extension of the ATM ARP routing mechanism that is sometimes used to improve the efficiency of routing computer network traffic over Non-Broadcast, Multiple Access (NBMA) Networks. It can be used by a sender to determine a route with the fewest hops to a receiver. More about this protocol and configuration can be found in the text below or in the FRR online documentation[1].



NHRP Configuration

☑ Enable NHRP

/var/nhrp/opennhrp.conf

```
interface gre1
  map 192.168.234.1/24 10.40.29.128 register
  holding-time 60
  shortcut
  redirect
  non-caching
```

/var/nhrp/opennhrp-script

```
#!/bin/sh

case $1 in
interface-up)
        ip route flush proto 42 dev $NHRP_INTERFACE
        ip neigh flush dev $NHRP_INTERFACE
        ;;
peer-register)
        ;;
peer-up)
        if [ -n "$NHRP_DESTMTU" ]; then
                ARGS=`ip route get $NHRP_DESTNBMA from $NHRP_SRCNBMA | head -1`
                ip route add $ARGS proto 42 mtu $NHRP_DESTMTU
        fi
        echo "Create link from $NHRP_SRCADDR ($NHRP_SRCNBMA) to $NHRP_DESTADDR ($NHRP_DESTNBMA)"
        /etc/init.d/ipsec start
        ..
```

Debug: Error

Figure 17: NHRP Configuration

[1] http://docs.frrouting.org/en/latest/nhrpd.html

Field */var/nhrp/opennhrp.conf* – insert the following configuration. It is to register the proper interface to the NHRP headquarter hub router and other needed parameters (edit to your own needs).

```
interface gre1
map 192.168.234.1/24 10.40.29.128 register
holding-time 60
shortcut
redirect
non-caching
```

Field */var/nhrp/opennhrp-script* – this is the *OpenNHRP* script to define the behavior in various situations. You can left it unchanged. If you accidentally edit it, you can copy it from the next page.

Press the *Apply* button to save the changes. Use the same procedure for all spokes – the *NHRP Configuration* remains the same for all the spoke routers.

Field */var/nhrp/opennhrp-script*

```
#!/bin/sh

case $1 in
interface-up)
ip route flush proto 42 dev $NHRP_INTERFACE
ip neigh flush dev $NHRP_INTERFACE
;;
peer-register)
;;
peer-up)
if [ -n "$NHRP_DESTMTU" ]; then
ARGS=‘ip route get $NHRP_DESTNBMA from $NHRP_SRCNBMA | head -1‘
ip route add $ARGS proto 42 mtu $NHRP_DESTMTU
fi
echo "Create link from $NHRP_SRCADDR ($NHRP_SRCNBMA) to $NHRP_DESTADDR ($NHRP_DESTNBMA)"
/etc/init.d/ipsec start
;;
peer-down)
echo "Delete link from $NHRP_SRCADDR ($NHRP_SRCNBMA) to $NHRP_DESTADDR ($NHRP_DESTNBMA)"
if [ "$NHRP_PEER_DOWN_REASON" != "lower-down" ]; then
/etc/init.d/ipsec stop
fi
ip route del $NHRP_DESTNBMA src $NHRP_SRCNBMA proto 42
;;
route-up)
echo "Route $NHRP_DESTADDR/$NHRP_DESTPREFIX is up"
ip route replace $NHRP_DESTADDR/$NHRP_DESTPREFIX proto 42 via $NHRP_NEXTHOP dev
    $NHRP_INTERFACE
ip route flush cache
;;
route-down)
echo "Route $NHRP_DESTADDR/$NHRP_DESTPREFIX is down"
ip route del $NHRP_DESTADDR/$NHRP_DESTPREFIX proto 42
ip route flush cache
;;
esac

exit 0
```

5.10 MPLS

Multiprotocol Label Switching (MPLS) is a routing technique in telecommunications networks that directs data from one node to the next based on labels rather than network addresses. Whereas network addresses identify endpoints, the labels identify established paths between endpoints. MPLS can encapsulate packets of various network protocols, hence the multiprotocol component of the name. MPLS supports a range of access technologies, including T1/E1, ATM, Frame Relay, and DSL.

Figure 18 shows a simplified version of an MPLS domain. There are routers that exist within the MPLS network or domain, and they communicate with each other via a specific label distribution protocol to set up the LSPs. There are other routers that are outside of the MPLS domain that simply forwards IP traffic like a normal router.

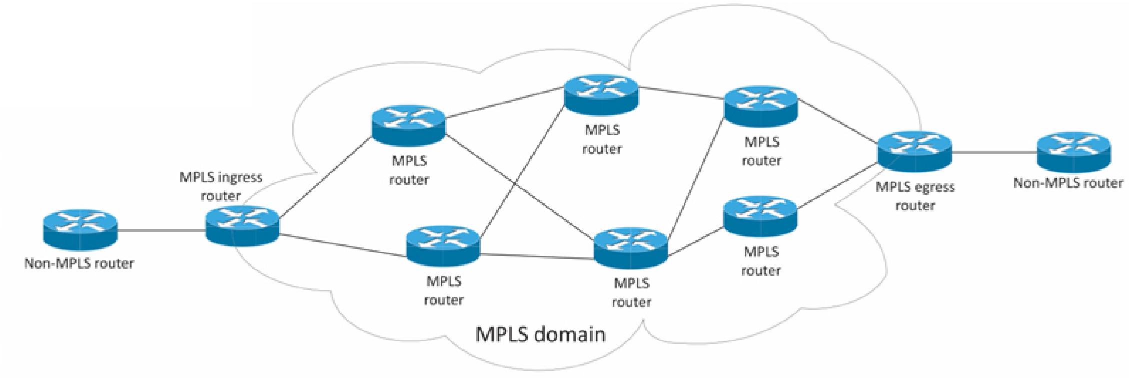
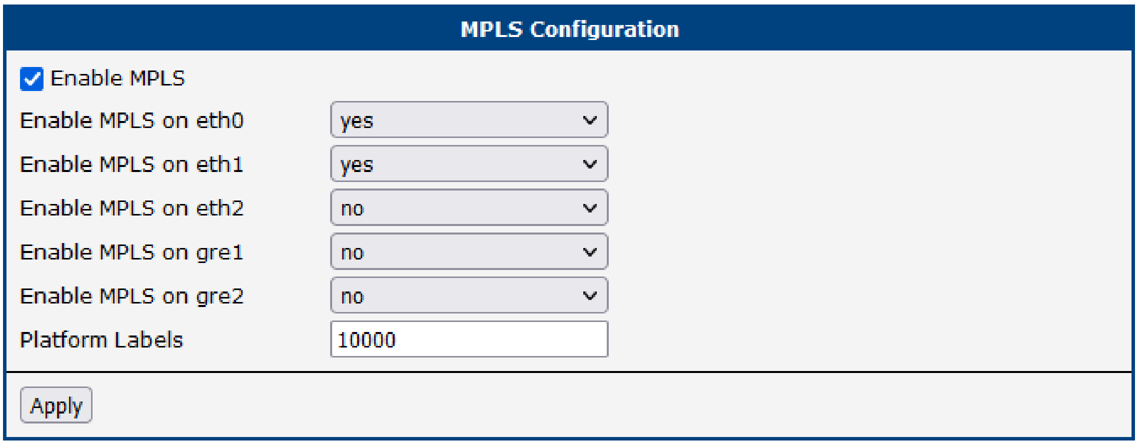


Figure 18: Simplified MPLS Domain Example

To enable the MPLS routing on an Advantech router, install the *FRR* router app first (see Chapter 2.2). Go to the router app's configuration GUI, select the *Customization -> Router Apps -> FRR -> Configuration -> MPLS* configuration page. Here, enable the MPLS service and choose which interfaces to enable for the MPLS, as shown in Figure 19. You can set the *Platform Labels* value here as well.



MPLS Configuration

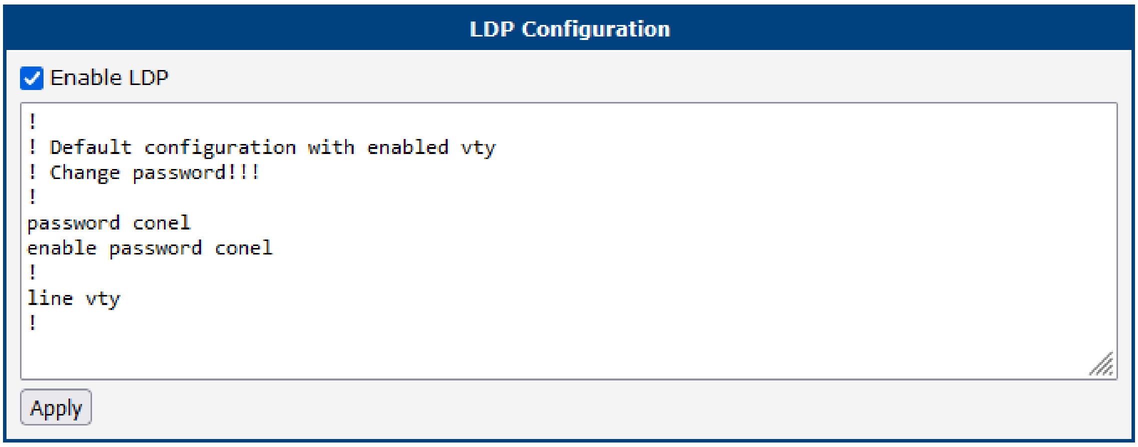
☑ Enable MPLS

Enable MPLS on eth0	yes
Enable MPLS on eth1	yes
Enable MPLS on eth2	no
Enable MPLS on gre1	no
Enable MPLS on gre2	no
Platform Labels	10000

Figure 19: MPLS Configuration

5.11 LDP

Label Distribution Protocol (LDP) is a protocol in which routers capable of Multiprotocol Label Switching (MPLS) exchange label mapping information. Two routers with an established session are called LDP peers and the exchange of information is bi-directional. LDP is used to build and maintain LSP databases that are used to forward traffic through MPLS networks. More about this protocol and examples can be found in the FRR online documentation[1].



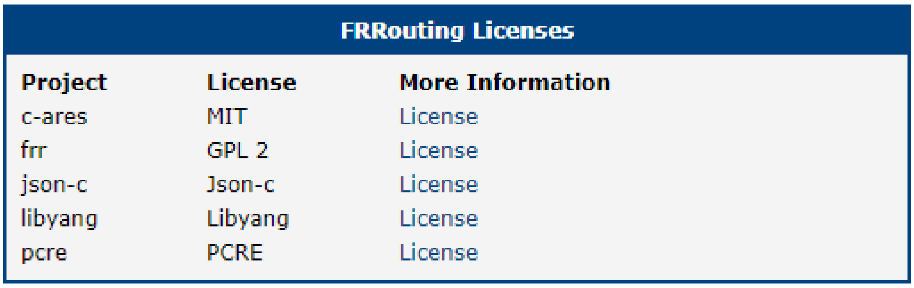
```
!
! Default configuration with enabled vty
! Change password!!!
!
password conel
enable password conel
!
line vty
!
```

Figure 20: LDP Configuration

[1] http://docs.frrouting.org/en/latest/ldpd.html?highlight=ldp

6. Licenses

Summarizes Open-Source Software (OSS) licenses used by this module.



FRRouting Licenses

Project	License	More Information
c-ares	MIT	License
frr	GPL 2	License
json-c	Json-c	License
libyang	Libyang	License
pcre	PCRE	License

Figure 21: Licenses Window

7. Related Documents

[1] Protocol IS-IS Application Note
[2] DMVPN Application Note

You can obtain product-related documents on *Engineering Portal* at *icr.advantech.com* address.

To get your router's *Quick Start Guide*, *User Manual*, *Configuration Manual*, or *Firmware* go to the *Router Models* page, find the required model, and switch to the *Manuals* or *Firmware* tab, respectively.

The *Router Apps* installation packages and manuals are available on the *Router Apps* page.

For the *Development Documents*, go to the *DevZone* page.