Securing the Control Plane

Version 20.8.1, 10 August 2020

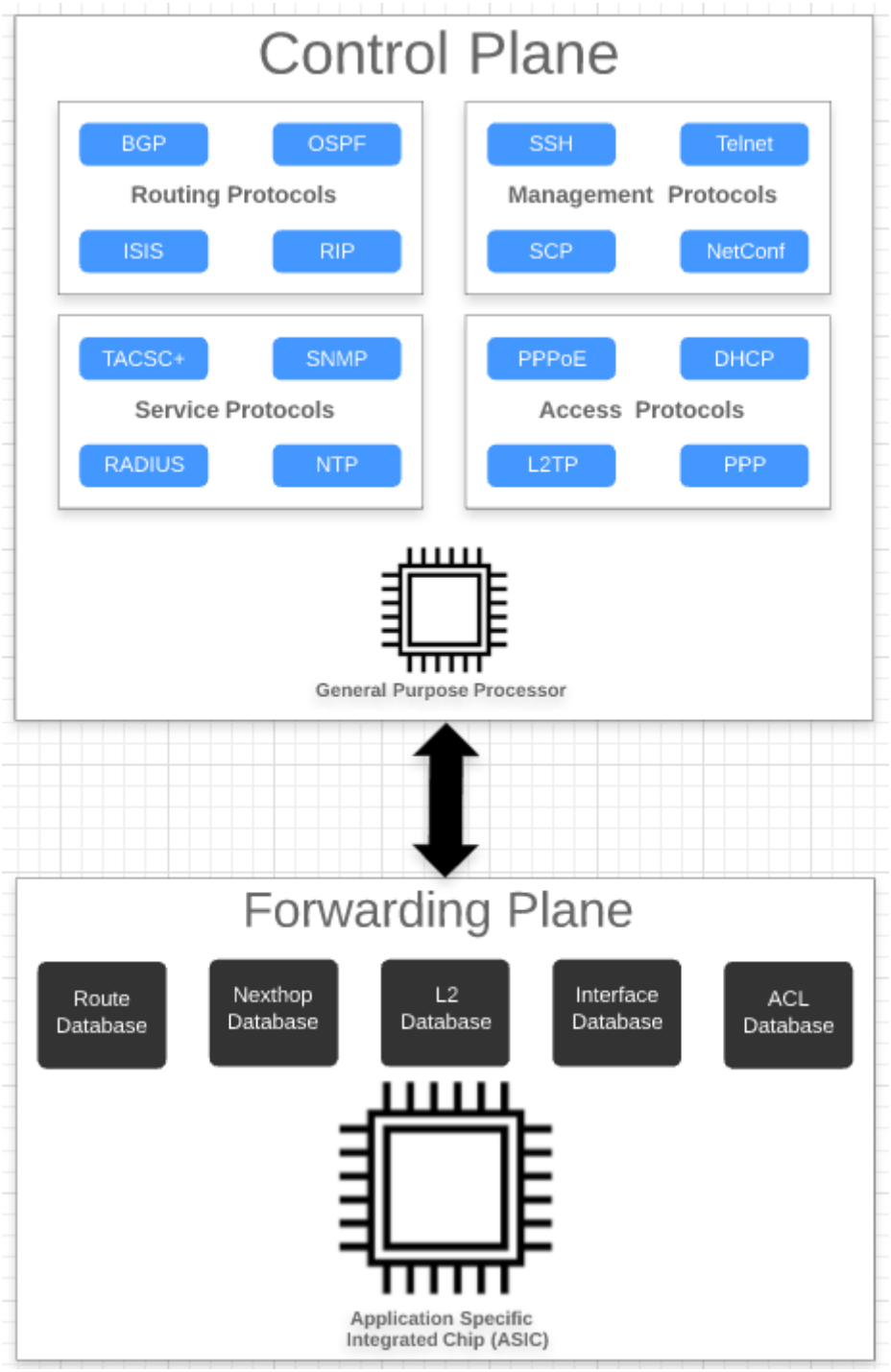
Registered Address	Support	Sales
26, Kingston Terrace, Princeton, New Jersey 08540, United States		
		+91 80 4850 5445
http://www.rtbrick.com	support@rtbrick.com	sales@rtbrick.com

Table of Contents

1. Introduction

Control plane security enables you to filter or rate-limit the unwanted traffic that is transmitted from the forwarding plane to the control plane. You need a filter and a rate-limiter to prevent this kind of traffic. When the filters and rate-limiters are closer to the forwarding plane and line-rate hardware, the protection will be effective and the system will be more resistant to DoS attacks.

All the routing protocols, management protocols, service protocols run in the control plane. The output of these protocols result in certain databases like routing table, MAC table, ARP table, etc., which essentially get programmed in the forwarding plane.



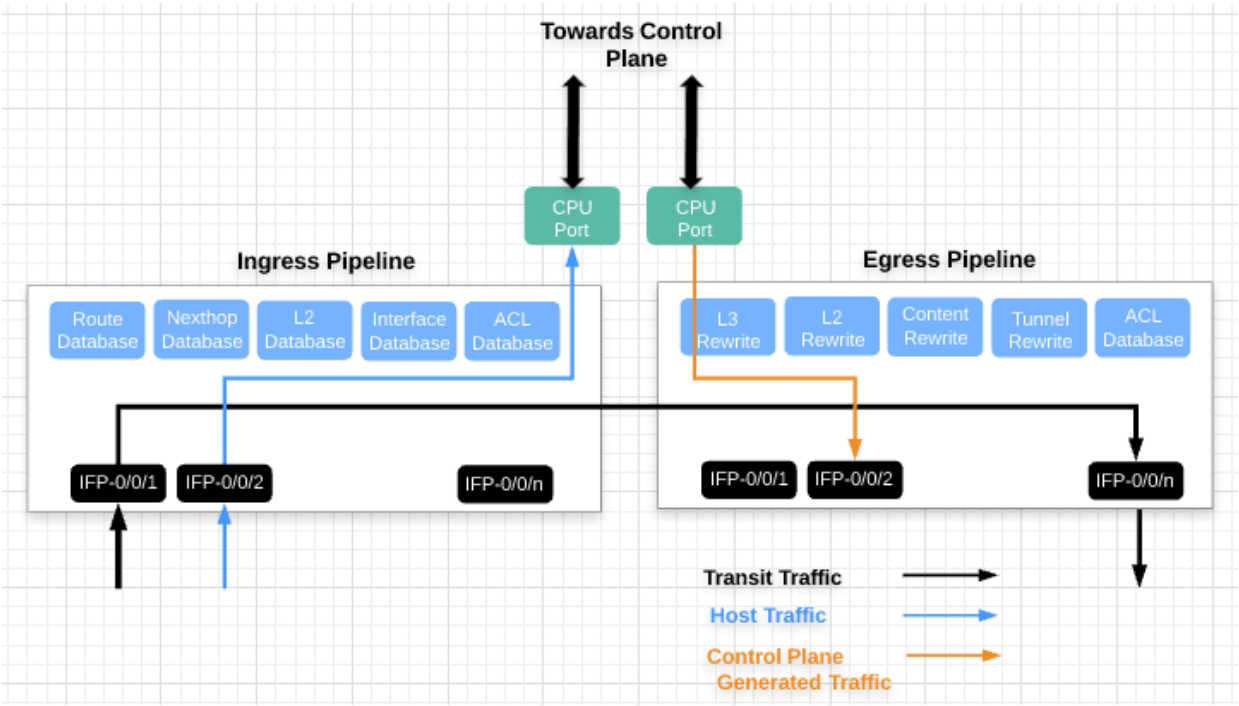
In the diagram above, you can see the group of routing protocols (BGP, OSPF, ISIS, RIP, etc, Management protocols (SSH, Telnet, SNMP, Netconf, etc), service protocols (Radius, NTP, TACACS+), and access protocols (PPPoE. DHCP, L2TP, PPP) associated with control plane. The control plane is generally implemented in software by using general-purpose processors. These protocols typically build a large number of databases like routing, switching, and ACL engines.

In contrast, a forwarding plane is associated with a copy of the databases (Routing, Switching, ACL, etc) built by the control plane. These entries typically contain the match and action which decide the packet flow in the forwarding plane. The forwarding plane functionality is realized in high performance Application Specific

Integrated Circuits (ASICs) that are capable of handling very high packet rates.

There are two kinds of traffic:

1. **Control traffic**: This traffic (packets) is destined to the device itself, that means, the packet is handled by the router itself. The traffic is classified as control traffic based on matching destination IP or because of the ACL rules that are programmed or might be because of some kind of exception that occurred while parsing the packet (could be Non acceptable fields, TTL expiry, etc).
2. **Transit traffic**: This traffic not destined to the device itself. These packets to be sent out on one of the routers physical interfaces.



All the control traffic packets will be destined to the CPU port. These packets go to the control plane for further processing. But as mentioned, the general purpose processor in the control plane is not designed for this. As there are a lot of conditions for packets to be classified as control traffic. This could be a genuine control plane packet or it could be any kind of DDoS attack (Because it matched some of the criteria like MY IP, or ACL which redirects to CPU). So you need to protect the router control plane by implementing mechanisms to filter completely or rate-limit traffic not required at the control plane level (i.e., unwanted traffic).

1.1. Limitations

- The match that is based on the source IPv6 and the destination IPv6 addresses are not be supported in the current release because of the space in the current ACL match field in the Broadcom Qumran chipset. The address includes both global and link local.

- Multi-level classifier filtering is not supported.

1.2. Supported Hardware

The Control Plane security is supported on the following hardware:

- Broadcom Qumran-MX

2. Securing Control Plane

2.1. Access Control List (ACL) Framework

Access Control List (ACL) framework is the building block for creating a customized security profile for the control plane. It allows only specific traffic. The ACL typically defines a rule, which contains a match and an action that are associated with each rule. These rules are programmed in the ACL engine in the hardware, and a packet which matches the rule performs the action associated with the rule.

The following example shows a sample rule.

```
"attribute": {
      "rule_name"                     : "default_bgp_l4_trap_10.1.1.1:10.1.1.2_src",
      "match_source_ipv4_address"     : "10.1.1.1",
      "match_destination_ipv4_address" : "10.1.1.2",
      "match_ip_protocol"             : "TCP",
      "match_source_l4_port"          : 179,
      "match_instance_name"           : "default",
      "action_redirect_to_cpu"        : true,
      "acl_type"                      : "l3v4"
}
```

The ACL type (L3V4) defines that the rule should be applied to all the incoming packets whose type is IPv4. There are two aspects inside this ACL rule.

If all the conditions inside the rule match, then the packet is redirected to the CPU port, that means the packet is punted to the control plane for further processing. The example above shows how an ACL typically looks like and how action can be programmed.

Any control plane protocols like BGP, OSPF, PPPoE, etc. will have rules or set of rules to classify their corresponding protocol packet, and set action to “TRAP” or “PUNT” them to the CPU. The framework is not just confined to traffic to CPU, rather it can be used to control the traffic as well. There are two options:

- Filtering
- Rate Limiting

2.1.1. ACL Extension for Filtering

Security extensions cannot be designed keeping a typical deployment scenario. The set of rules defined in one deployment might not be adequate in another deployment. The first level of defence is to filter or drop the packets which are matching a specific set of conditions. The below example shows how a typical filter drops all the packets matching the source IP “10.200.210.1”

```
"attribute": {
          "rule_name"                    : "filter_from_unknown_ip",
          "match_source_ipv4_address"    : "10.200.210.1",
          "match_instance_name"          : "default",
          "action_drop"                  : true,
          "action_counter"               : true,
          "acl_type"                     : "l3v4"
}
```

Let us assume that the host 10.200.210.1 is flooding with a lot of packets, which are destined to a router. As this filter is programmed in hardware, it prevents traffic from consuming bandwidth on the interface that connects the forwarding plane to the control plane. Another important action is the counter, which is associated with the ACL action (action_counter). The counters serve as an important tool for the analysis of potential attacks, and as an debugging and troubleshooting aid. By adjusting the granularity and the order of the filters, more granular investigations can be performed. For example, you can create a filter that matches only traffic allowed from a group of IP addresses for a given protocol followed by a filter that denies all traffic for that protocol. This allows you to monitor the counters for the allowed protocol filter and any traffic that matches the specific protocol that did not originate from the explicitly allowed hosts.

2.1.2. ACL Extension for Rate Limiting

In certain scenarios, filtering or hard stopping a specific flow might not be helpful. But you need to rate limit the packets that are punted to the CPU. In any hardware devices, there will be a backplane or some sort of dedicated backplane port which connects to the host path or control plane. Note that the most significant factor to consider regarding the traffic profile that goes to the router control plane is the packets per second (pps) rate. With the various profiling enabled, it is calculated that it is 40pps. This means that traffic redirecting to the CPU should never cross more than this number. If it exceeds, tail drop occurs.

There are various number of physical ports and traffic (Control plane protocols traffic) which need to be processed at the control plane. Rate limiting these traffic which are redirected to the CPU is required. This is performed using a rate limiting tool called POLICER.

The ACL can define a set of rules and associate a POLICER and CLASS as an action for it. These POLICERS for certain classes of traffic are also installed in the forwarding plane. These policers help to further control the traffic that reaches the router control plane for each filtered class and all traffic that do not match an explicit class. The actual rates selected for various classes are specific to a network deployment.

2.1.3. Defining the ACL rule for Rate Limiting

Let us take a simple example of rate limiting the SSH packets destined for the MY IP address (172.100.100.20). The below example shows a sample ACL rule.

```
##############################################################################
#        ACL Rule for Matching SSH Packets Destined to MY IP          #
##############################################################################
"attribute":
{
      "rule_name"                    : "default_ssh",
      "match_destination_ipv4_address" : "172.100.100.20",
      "match_ip_protocol"            : "TCP",
      "match_source_l4_port"         : 22,
      "match_instance_name"          : "default",
      "action_redirect_to_cpu"       : true,
      "action_policer_profile_name" : "policer_20mb",
      "acl_type" : "l3v4"
}
```

In the above example, the action is associated with a Quality-Of-Service (QoS) profile.

2.1.4. Defining the Policer

Quality-Of-Service (QoS) is associated with various entities like Classifier, Queue, Policer, etc. For host path rate limiting, you need to define a policer which does policing. Policing is a method of applying the hard limit to the rate which packets are expected to receive. Any excess packets received beyond the mentioned limit are marked RED, which is dropped before redirecting it to CPU Port.

RtBrick QoS implementation supports three kinds of POLICING. For more information on QoS and supported capability, please refer to the [HQoS] document mentioned in the reference. A sample configuration of defining the POLICER configuration is shown below.

```
##########################################################################
# Sample configuration of a Two-Rate Three Color Policer #
##########################################################################
{
    "table": {
            "table_name": "global.qos.policer.config"
     },
     "objects": [
     {
            "attribute": {
                    "policer_name"  : "policer_5mb",
                    "type"          : "Two-rate-three-color",
                    "levels"        : 1,
                    "level1_cir"    : 5000,
                    "level1_pir"    : 5500,
                    "level1_cbs"    : 700,
                    "level1_pbs"    : 800
             }
      }
      ]
}
```

The example above defines a Two-rate-three-color marking. A committed information rate (CIR) of 5MB is defined with a committed burst size (CBS) of 700, and a peak burst size (PBS) of 700. Traffic within 5MB is GREEN. Traffic above 5MB but within 5.5MB is YELLOW. Traffic above 5.5MB is considered RED.

2.2. Protect the control plane

Before getting into a specific example of how to protect the control plane, let us generalize: What kind of packets get punted to the CPU ? An interface would be configured with an IPv4/IPv6 address. In general, these addresses are called “MY IP”. Any packets whose destination address matches to “MY IP” are packets destined to the router. These packets need to be processed in the control plane. So the first level of defence would be to block all the packets that match the “MY IP” and protocol category which the control plane is not interested in.

```
##########################################################################
#                  Discarding all the TCP Packet                         #
##########################################################################
"attribute": {
      "rule_name"                          : "default_discard_tcp",
      "match_destination_ipv4_address"     : "172.100.100.20",
      "match_ip_protocol"                  : "TCP",
      "match_instance_name"                : "default",
      "action_drop"                        : "true",
      "acl_type"                           : "l3v4"
}
"attribute": {
      "rule_name"                          : "default_discard_tcp",
      "match_destination_ipv4_address"     : "182.100.100.20",
      "match_ip_protocol"                  : "TCP",
      "match_instance_name"                : "default",
      "action_drop"                        : "true",
      "acl_type"                            : "l3v4"
}
```

```
##########################################################################
#                  Discarding all the UDP Packet                         #
##########################################################################
"attribute": {
      "rule_name"                          : "default_discard_udp",
      "match_destination_ipv4_address"     : "172.100.100.20",
      "match_ip_protocol"                  : "UDP",
      "match_instance_name"                : "default",
      "action_drop"                        : "true",
      "acl_type"                           : "l3v4"
}
"attribute": {
      "rule_name"                          : "default_discard_udp",
      "match_destination_ipv4_address"     : "182.100.100.20",
      "match_ip_protocol"                  : "UDP",
      "match_instance_name"                : "default",
      "action_drop"                        : "true",
      "acl_type"                           : "l3v4"
}
```

These rules are not auto-generated, and hence you need to install them based on the requirement for MY IP.

2.3. Management Protocols

Management protocols typically involve accessing or servicing the device. There is a wide variety of protocols which fall under this category. Some of them which are very notable are:

- Telnet

- SSH
- SCP
- TACACS+
- RADIUS
- Logging (syslog, graylog)

Not all the protocols listed above need higher bandwidth. So the entire management protocol's profiling can categorized into two main categories which are higher and lower. So the corresponding policing and profiling for these two categories are defined below.

```
###############################################################################
#         Default Policer Configuration for Mgmt Protocols                    #
###############################################################################
{
    "table": {
            "table_name": "global.qos.policer.config"
     },
     "objects": [ {
            "attribute": {
                    "policer_name"  : "_DEFAULT_POLICER_20_MB",
                    "type"          : "Two-rate-three-color",
                    "levels"        : 1,
                    "level1_cir"    : 20000,
                    "level1_pir"    : 20000,
                    "level1_cbs"    : 33000,
                    "level1_pbs"    : 33000
             }
      },
     {
            "attribute": {
                    "policer_name"  : "_DEFAULT_POLICER_5_MB",
                    "type"          : "Two-rate-three-color",
                    "levels"        : 1,
                    "level1_cir"    : 5000,
                    "level1_pir"    : 5000,
                    "level1_cbs"    : 33000,
                    "level1_pbs"    : 33000
             }
                  },  ]
}
```

2.3.1. Telnet/SSH/SCP

There are various ways to access the devices. One of the best practices is to use SSH. They run on the standard port 22/23. The below example shows the ACL rules which are installed by default. Note that the rule mentioned below will be matched only when the access is coming through inband management port.

```
##########################################################################
#                  Acl rules for allowing Telnet                         #
##########################################################################
"attribute": {
      "rule_name"                          : "default_telnet",
      "match_destination_ipv4_address"     : "172.100.100.20",
      "match_ip_protocol"                  : "TCP",
      "match_source_l4_port"               : 22,
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_name"        : "_DEFAULT_POLICER_5_MB",
      "acl_type"                           : "l3v4"
}
"attribute": {
      "rule_name"                          : "default_telnet",
      "match_destination_ipv4_address"     : "182.100.100.20",
      "match_ip_protocol"                  : "TCP",
      "match_source_l4_port"               : 22,
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_name"        : "_DEFAULT_POLICER_5_MB",
      "acl_type"                           : "l3v4"
}
```

```
##########################################################################
#                  Acl rules for allowing SSH                            #
##########################################################################
"attribute": {
      "rule_name"                          : "default_ssh",
      "match_destination_ipv4_address"     : "172.100.100.20",
      "match_ip_protocol"                  : "TCP",
      "match_source_l4_port"               : 23,
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_name"        : "_DEFAULT_POLICER_20_MB",
      "acl_type"                           : "l3v4"
}
"attribute": {
      "rule_name"                          : "default_ssh",
      "match_destination_ipv4_address"     : "182.100.100.20",
      "match_ip_protocol"                  : "TCP",
      "match_source_l4_port"               : 23,
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_name"        : "_DEFAULT_POLICER_20_MB",
      "acl_type"                           : "l3v4"
}
```

2.3.2. TACACS+

TACACS allows a client to accept a username and password and send a query to a TACACS authentication server, sometimes called a TACACS+ daemon or simply TACACSD. It runs on either TCP or UDP port 49.

The following example shows the ACL rules for allowing TACACS+. Based on your application requirements, you can programme the ACLs.

```
#########################################################################
#                  Acl rules for allowing TACACS+                       #
#########################################################################
"attribute": {
      "rule_name"                          : "default_tacacs",
      "match_destination_ipv4_address"     : "172.100.100.20",
      "match_ip_protocol"                  : "TCP",
      "match_source_l4_port"               : 49,
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_namer"       : "policer_5mb",
      "acl_type"                           : "l3v4"
}
"attribute": {
      "rule_name"                          : "default_tacacs",
      "match_destination_ipv4_address"     : "182.100.100.20",
      "match_ip_protocol"                  : "TCP",
      "match_source_l4_port"               : 49,
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_name"        : "policer_5mb",
      "acl_type"                           : "l3v4"
}
"attribute": {
      "rule_name"                          : "default_tacacs",
      "match_destination_ipv4_address"     : "172.100.100.20",
      "match_ip_protocol"                  : "UDP",
      "match_source_l4_port"               : 49,
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_name"        : "policer_5mb",
      "acl_type"                           : "l3v4"
}
"attribute": {
      "rule_name"                          : "default_tacacs",
      "match_destination_ipv4_address"     : "182.100.100.20",
      "match_ip_protocol"                  : "UDP",
      "match_source_l4_port"               : 49,
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_name"        : "policer_5mb",
      "acl_type"                           : "l3v4"
}
```

2.4. Securing the Service Protocols

Service protocols typically involve generating/processing query, error, alert messages. There are a wide variety of protocols which need different rate limiting.

2.4.1. Internet Control Message Protocol (ICMP)

The Internet Control Message Protocol (ICMP) is one of the IP protocols which is used to signal errors and query messages. As it is designed for signal and query messages, it should not consume a lot of resources. But as a security reason, the ICMP packets need to be rate limited.

These rules are not auto-generated, and hence you need to install them based on the requirement.

```
#########################################################################
#                    Acl rules for allowing ICMP                        #
#########################################################################
"attribute": {
      "rule_name"                          : "default_icmp",
      "match_destination_ipv4_address"     : "172.100.100.20",
      "match_ip_protocol"                  : "ICMP",
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_name"        : "policer_10mb",
      "acl_type"                           : "l3v4"
}
"attribute": {
      "rule_name"                          : "default_icmp",
      "match_destination_ipv4_address"     : "182.100.100.20",
      "match_ip_protocol"                  : "ICMP",
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_name"        : "policer_10mb",
      "acl_type"                           : "l3v4"
}
```

2.4.2. Hyper Text Transport Protocol (HTTP)

RBFS supports REST-based configuration. In this regard, GET/SET actions are supported. The REST-based operations involve using the existing HTTP protocol. As the number of accesses is more, this needs a higher bandwidth.

These rules are not auto-generated, and hence you need to install them based on the requirement.

```
##############################################################################
#                    Acl rules for allowing HTTP                             #
##############################################################################
"attribute": {
      "rule_name"                          : "default_http",
      "match_destination_ipv4_address"     : "172.100.100.20",
      "match_ip_protocol"                  : "TCP",
      "match_destination_l4_port"          : 80,
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_name"        : "policer_50mb",
      "acl_type"                           : "l3v4"
}
"attribute": {
      "rule_name"                          : "default_icmp",
      "match_destination_ipv4_address"     : "182.100.100.20",
      "match_ip_protocol"                  : "TCP",
      "match_destination_l4_port"          : 80,
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_name"        : "policer_50mb",
      "acl_type"                           : "l3v4"
}
```

2.4.3. RADIUS

RBFS supports running Broadband Remote Access Server protocols which typically include PPPoE, DHCP etc. As a part of subscriber bring up, the subscribers might be remotely authenticated by sending the request to RADIUS server. Subscriber accounting can also be performed on remote radius servers. This might involve a higher bandwidth. Enabling the remote authentication and accounting is enabled as per the ACCESS configuration. These rules are auto-generated. The below sample shows how the ACL entry looks like.

```
###############################################################################
#                    Acl rules for allowing RADIUS                            #
###############################################################################
"attribute": {
      "rule_name"                          : "radius-srv1-v4-auth-trap",
      "match_destination_ipv4_address"     : "172.100.100.20",
      "match_ip_protocol"                  : "UDP",
      "match_source_l4_port"               : 1812,
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_name"        : "_DEFAULT_POLICER_20_MB",
      "acl_type"                           : "l3v4"
}
"attribute": {
      "rule_name"                          : "radius-srv1-v4-acct-trap",
      "match_destination_ipv4_address"     : "172.100.100.20",
      "match_ip_protocol"                  : "UDP",
      "match_source_l4_port"               : 1813,
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_name"        : "_DEFAULT_POLICER_20_MB",
      "acl_type"                           : "l3v4"
}
"attribute": {
      "rule_name"                          : "radius-srv1-v4-cao-trap",
      "match_destination_ipv4_address"     : "172.100.100.20",
      "match_ip_protocol"                  : "UDP",
      "match_source_l4_port"               : 3799,
      "match_instance_name"                : "default",
      "action_redirect_to_cpu"             : true,
      "action_policer_profile_name"        : "_DEFAULT_POLICER_20_MB",
      "acl_type"                           : "l3v4"
}
```

2.5. Securing the Routing Protocols

RBFS supports various routing protocols like BGP, OSPF and ISIS. The bandwidth usage for the routing protocols will be significantly higher than the management and service protocols. To accommodate the higher bandwidth, three policer profiles are defined by default.

```
###############################################################################
#         Default Policer Configuration for Routing Protocols                 #
###############################################################################
{
    "table": {
            "table_name": "global.qos.policer.config"
     },
     "objects": [
     {
            "attribute": {
                    "policer_name"  : "_DEFAULT_POLICER_500_MB",
                    "type"          : "Two-rate-three-color",
                    "levels"        : 1,
                    "level1_cir"    : 500000,
                    "level1_pir"    : 500000,
                    "level1_cbs"    : 33000,
                    "level1_pbs"    : 33000
             }
      },
     {
            "attribute": {
                    "policer_name"  : "_DEFAULT_POLICER_250_MB",
                    "type"          : "Two-rate-three-color",
                    "levels"        : 1,
                    "level1_cir"    : 250000,
                    "level1_pir"    : 250000,
                    "level1_cbs"    : 33000,
                    "level1_pbs"    : 33000
             }
      },
      {
            "attribute": {
                    "policer_name"  : "_DEFAULT_POLICER_50_MB",
                    "type"          : "Two-rate-three-color",
                    "levels"        : 1,
                    "level1_cir"    : 50000,
                    "level1_pir"    : 50000,
                    "level1_cbs"    : 33000,
                    "level1_pbs"    : 33000
             }
      },
      ]
}
```

2.5.1. Border Gateway Protocol (BGP)

Border Gateway Protocol (BGP) runs on top of the TCP port 179 and is designed to carry complete internet feed. BGPd installs the trap rules for punting BGP packets. This trap rule is associated with a higher bandwidth QoS profile. The below example shows a sample ACL entry programmed for the BGP.

```
##############################################################################
#                    Acl rules for allowing BGP                              #
##############################################################################
"attribute": {
      "rule_name"                    : "default_bgp_l4_trap_destination",
      "match_ip_protocol"            : "TCP",
      "match_destination_l4_port"    : 179,
      "match_instance_name"          : "default",
      "action_redirect_to_cpu"       : true,
      "action_policer_profile_name" : "_DEFAULT_POLICER_50_MB",
      "acl_type"                     : "l3v4"
}

"attribute": {
      "rule_name"                    : "default_bgp_l4_trap_source",
      "match_ip_protocol"            : "TCP",
      "match_source_l4_port"         : 179,
      "match_instance_name"          : "default",
      "action_redirect_to_cpu"       : true,
      "action_policer_profile_name" : "_DEFAULT_POLICER_50_MBb",
      "acl_type"                     : "l3v4"
}
```

2.5.2. Intermediate System to Intermediate System (IS-IS)

The IS-IS protocol runs at layer 2 and helps to build the Layer 3 routing table. IS-IS protocol is enabled on an L3 interface. An ACL entry is required to punt the IS-IS packet punting per L3 interface. There are two ways of handling trap rules.

1. Generic ACL rule to punt all ISIS packets matching destination mac: 09:00:2b:00:00:05
2. Specific ACL rule to punt all ISIS packets matching destination mac: 09:00:2b:00:00:05 and incoming L3 IFL on which ISIS is enabled

[1] has an advantage that a single entry is given the limited ACL space in hardware. But has a flaw of undesired ISIS packets punting even the ISIS protocol is not enabled.

[2] has the disadvantage of ACL entry per L3 IFL on which ISIS is enabled. But only ISIS packets which need to be handled by the control plane will get punted. If policing is to be done per IFL, care should be taken that it does not exceed the overall bandwidth of CPU port.

```
##############################################################################
#                Acl rules for allowing ISIS Per IFL                         #
##############################################################################
"attribute": {
      "rule_name"                   : "isis_acl_ifl-0/2/1/12_all_isis",
      "match_destination_mac"       : "09:00:2b:00:00:05",
      "match_ifl_name"              : "ifl-0/2/1/12",,
      "action_redirect_to_cpu"      : true,
      "action_policer_profile_name" : "_DEFAULT_POLICER_50_MB",
      "acl_type"                    : "l2"
}
"attribute": {
      "rule_name"                   : "isis_acl_ifl-0/5/1/51_all_isis",
      "match_destination_mac"       : "09:00:2b:00:00:05",
      "match_ifl_name"              : "ifl-0/5/1/51",,
      "action_redirect_to_cpu"      : true,
      "action_policer_profile_namer": "_DEFAULT_POLICER_50_MB",
      "acl_type"                    : "l2"
}
```

```
##############################################################################
#                Acl rules for allowing Global ISIS                          #
##############################################################################
"attribute": {
      "rule_name"                   : "isis_acl_all_isis",
      "match_destination_mac"       : "09:00:2b:00:00:05",
      "action_redirect_to_cpu"      : true,
      "action_policer_profile_name" : "_DEFAULT_POLICER_50_MB",
      "acl_type"                    : "l2"
}
```

2.6. Securing the Access Protocols

RBFS supports running Broadband Remote Access Server protocols which typically include PPPoE, DHCP etc. Similar to routing protocols, access protocols start their machineries upon configuration. Typically the bandwidth usage of access protocols are that high as any routing protocols and not as low as management protocols. To meet the requirements of the access protocols, RBFS defines two policer profiles. The below example shows a sample of the default policer created for access.

```
###############################################################################
#         Default Policer Configuration for Access Protocols                  #
###############################################################################
{
    "table": {
            "table_name": "global.qos.policer.config"
     },
     "objects": [
     {
            "attribute": {
                    "policer_name"  : "_DEFAULT_POLICER_50_MB",
                    "type"          : "Two-rate-three-color",
                    "levels"        : 1,
                    "level1_cir"    : 50000,
                    "level1_pir"    : 50000,
                    "level1_cbs"    : 33000,
                    "level1_pbs"    : 33000
             }
      },
      ]
}
```

2.7. Point-To-Point Protocol

Point-to-Point Protocol (PPP) is a data link layer (layer 2) communications protocol between two routers directly without any host or any other networking in between. Point-To-Point protocol machinery starts with the discovery phase where the initial set of packets get exchanged with Ether Type be (0x8863) and (0x8864). There can be certain deployments where RBFS stack needs to support port-based cross connect (L2x) with transparent PPPoE forwarding and local terminated PPPoE on the same physical interface (IFP). Currently BCM Qumran based chipsets do not support ACL's with the VLAN range. Therefore, PPPoE installs ACL which accepts all the PPPoE control plane traffic and discards the unwanted requests. To accommodate the higher bandwidth utilization, PPP Acl entries are associated with "policer_200mb" policer profile. The below example shows the default ACL rule installed for a single interface:

```
"attribute": {
      "rule_name"             : "pppoed_ifp-0/0/1_1-4094-1-4094_8864",
      "match_ifp_name"        : "ifp-0/0/1",
      "match_ether_type"      : 34916,
      "match_inner_vlan_min"  : 1,
      "match_inner_vlan_max"  : 4094,
      "match_outer_vlan_min"  : 1,
      "match_outer_vlan_max"  : 4094
      "action_redirect_to_cpu": true,
      "Access Protocols"      : "_DEFAULT_POLICER_50_MB",
      "acl_type"              : "PPPOE"
}
```

2.8. Enabling Control Plane Security

The control plane security feature is disabled by default.

To enable the control plane security feature, enter the following command:

```
rtb confd set forwarding-options control-plane-security enable
```

To disable the control plane security feature, enter the following command:

```
rtb confd set forwarding-options control-plane-security disable
```

By enabling this feature, the default policer configurations are installed and all control plane ACLs are reprogrammed to update the action_policer_profile_name.

The host-path-qos enable feature is disabled by default. Once it is enabled, you cannot disable it.

To enable the `host-path-qos' feature, enter the following command:

```
rtb confd set forwarding-options host-path-qos enable
```

By enabling this feature, the default scheduler and queue configurations are installed and the CPU ports queue mapping will change. Also, all control plane ACLs will be reprogrammed to update action_forward_class.

The example below shows the running configuration when **Control Plane Security** is enabled.

```
    "forwarding-options": [
      {
        "class-of-service": [
          {
            "policer": [
              {
                "policer-name:_DEFAULT_POLICER_100_MB": {
                  "type": "two-rate-three-color",
                  "levels": 1,
                  "level1_cir": 100000,
                  "level1_pir": 100000,
                  "level1_cbs": 33000,
                  "level1_pbs": 33000
                },
                "policer-name:_DEFAULT_POLICER_1_MB": {
```

```
                "type": "two-rate-three-color",
                "levels": 1,
                "level1_cir": 1000,
                "level1_pir": 1000,
                "level1_cbs": 33000,
                "level1_pbs": 33000
              },
              "policer-name:_DEFAULT_POLICER_20_MB": {
                "type": "two-rate-three-color",
                "levels": 1,
                "level1_cir": 20000,
                "level1_pir": 20000,
                "level1_cbs": 33000,
                "level1_pbs": 33000
              },
              "policer-name:_DEFAULT_POLICER_250_MB": {
                "type": "two-rate-three-color",
                "levels": 1,
                "level1_cir": 250000,
                "level1_pir": 250000,
                "level1_cbs": 33000,
                "level1_pbs": 33000
              },
              "policer-name:_DEFAULT_POLICER_500_MB": {
                "type": "two-rate-three-color",
                "levels": 1,
                "level1_cir": 500000,
                "level1_pir": 500000,
                "level1_cbs": 33000,
                "level1_pbs": 33000
              },
              "policer-name:_DEFAULT_POLICER_50_MB": {
                "type": "two-rate-three-color",
                "levels": 1,
                "level1_cir": 50000,
                "level1_pir": 50000,
                "level1_cbs": 33000,
                "level1_pbs": 33000
              },
              "policer-name:_DEFAULT_POLICER_5_MB": {
                "type": "two-rate-three-color",
                "levels": 1,
                "level1_cir": 5000,
                "level1_pir": 5000,
                "level1_cbs": 33000,
                "level1_pbs": 33000
              }
            }
          ]
        }
      ]
```

The example below shows the running configuration when **host-path-qos** is enabled.

```
  "forwarding-options": [
       {
         "class-of-service": [
           {
             "queue": [
               {
                 "queue-name:_DEFAULT_HOSTPATH_QUEUE_0": {
                   "queue_size": 96000
                 },
                 "queue-name:_DEFAULT_HOSTPATH_QUEUE_1": {
                   "queue_size": 96000
                 },
                 "queue-name:_DEFAULT_HOSTPATH_QUEUE_2": {
                   "queue_size": 64000
                 },
                 "queue-name:_DEFAULT_HOSTPATH_QUEUE_3": {
                   "queue_size": 64000
                 },
                 "queue-name:_DEFAULT_HOSTPATH_QUEUE_4": {
                   "queue_size": 64000
                 },
                 "queue-name:_DEFAULT_HOSTPATH_QUEUE_5": {
                   "queue_size": 32000
                 },
                 "queue-name:_DEFAULT_HOSTPATH_QUEUE_6": {
                   "queue_size": 32000
                 },
                 "queue-name:_DEFAULT_HOSTPATH_QUEUE_7": {
                   "queue_size": 32000
                 }
               }
             ],
             "class-queue-map": [
               {
                 "class-queue-map-name:_DEFAULT_HOSTPATH_CLASS_QUEUE_MAP_": {
                   "class:class-0": {
                     "queue_name": "_DEFAULT_HOSTPATH_QUEUE_0"
                   },
                   "class:class-1": {
                     "queue_name": "_DEFAULT_HOSTPATH_QUEUE_1"
                   },
                   "class:class-2": {
                     "queue_name": "_DEFAULT_HOSTPATH_QUEUE_2"
                   },
                   "class:class-3": {
                     "queue_name": "_DEFAULT_HOSTPATH_QUEUE_3"
                   },
                   "class:class-4": {
                     "queue_name": "_DEFAULT_HOSTPATH_QUEUE_4"
                   },
                   "class:class-5": {
                     "queue_name": "_DEFAULT_HOSTPATH_QUEUE_5"
                   },
                   "class:class-6": {
                     "queue_name": "_DEFAULT_HOSTPATH_QUEUE_6"
                   },
                   "class:class-7": {
                     "queue_name": "_DEFAULT_HOSTPATH_QUEUE_7"
                   }
                 }
               }
             ],
             "queue-group": [
               {
```

```
                "queue-group-name:_DEFAULT_HOSTPATH_QUEUE_GROUP_": {
                  "queue_numbers": 8
                }
              }
            ],
            "scheduler": [
              {
                "scheduler-name:_DEFAULT_HOSTPATH_SCHEDULER_": {
                  "type": "weighted_fair_queueing"
                }
              }
            ],
            "profile": [
              {
                "profile-name:_DEFAULT_HOSTPATH_PROFILE_": {
                  "class_queue_map_name": "_DEFAULT_HOSTPATH_CLASS_QUEUE_MAP_",
                  "scheduler_map_name": "_DEFAULT_HOSTPATH_SCHEDULER_MAP_"
                }
              }
            ],
            "interface": [
              {
                "physical:cpu-0/0/200": {
                  "qos_profile_name": "_DEFAULT_HOSTPATH_PROFILE_"
                },
                "physical:cpu-0/0/201": {
                  "qos_profile_name": "_DEFAULT_HOSTPATH_PROFILE_"
                },
                "physical:cpu-0/0/202": {
                  "qos_profile_name": "_DEFAULT_HOSTPATH_PROFILE_"
                },
                "physical:cpu-0/0/203": {
                  "qos_profile_name": "_DEFAULT_HOSTPATH_PROFILE_"
                }
              }
            ],
            "scheduler-map": [
              {
                "scheduler-map-name:_DEFAULT_HOSTPATH_SCHEDULER_MAP_": {
                  "scheduler-map-queue:_DEFAULT_HOSTPATH_QUEUE_GROUP_
_DEFAULT_HOSTPATH_QUEUE_0": {
                    "parent_scheduler_name": "_DEFAULT_HOSTPATH_SCHEDULER_",
                    "connection_point": "strict_priority_0",
                    "weight": 80
                  },
                  "scheduler-map-queue:_DEFAULT_HOSTPATH_QUEUE_GROUP_
_DEFAULT_HOSTPATH_QUEUE_1": {
                    "parent_scheduler_name": "_DEFAULT_HOSTPATH_SCHEDULER_",
                    "connection_point": "strict_priority_0",
                    "weight": 70
                  },
                  "scheduler-map-queue:_DEFAULT_HOSTPATH_QUEUE_GROUP_
_DEFAULT_HOSTPATH_QUEUE_2": {
                    "parent_scheduler_name": "_DEFAULT_HOSTPATH_SCHEDULER_",
                    "connection_point": "strict_priority_0",
                    "weight": 60
                  },
                  "scheduler-map-queue:_DEFAULT_HOSTPATH_QUEUE_GROUP_
_DEFAULT_HOSTPATH_QUEUE_3": {
                    "parent_scheduler_name": "_DEFAULT_HOSTPATH_SCHEDULER_",
                    "connection_point": "strict_priority_0",
                    "weight": 50
                  },
                  "scheduler-map-queue:_DEFAULT_HOSTPATH_QUEUE_GROUP_
```

```
_DEFAULT_HOSTPATH_QUEUE_4": {
                    "parent_scheduler_name": "_DEFAULT_HOSTPATH_SCHEDULER_",
                    "connection_point": "strict_priority_0",
                    "weight": 40
                  },
                  "scheduler-map-queue:_DEFAULT_HOSTPATH_QUEUE_GROUP_
_DEFAULT_HOSTPATH_QUEUE_5": {
                    "parent_scheduler_name": "_DEFAULT_HOSTPATH_SCHEDULER_",
                    "connection_point": "strict_priority_0",
                    "weight": 30
                  },
                  "scheduler-map-queue:_DEFAULT_HOSTPATH_QUEUE_GROUP_
_DEFAULT_HOSTPATH_QUEUE_6": {
                    "parent_scheduler_name": "_DEFAULT_HOSTPATH_SCHEDULER_",
                    "connection_point": "strict_priority_0",
                    "weight": 20
                  },
                  "scheduler-map-queue:_DEFAULT_HOSTPATH_QUEUE_GROUP_
_DEFAULT_HOSTPATH_QUEUE_7": {
                    "parent_scheduler_name": "_DEFAULT_HOSTPATH_SCHEDULER_",
                    "connection_point": "strict_priority_0",
                    "weight": 10
                  },
                  "scheduler-map-scheduler:_DEFAULT_HOSTPATH_SCHEDULER_": {
                    "connection_type": "scheduler_to_port"
                  }
                }
              }
            ]
          }
        ]
      }
    ]
```