                   Protecting the Router Control Plane

Abstract

   This memo provides a method for protecting a router's control plane
   from undesired or malicious traffic. In this approach, all
   legitimate router control plane traffic is identified. Once
   legitimate traffic has been identified, a filter is deployed in the
   router's forwarding plane. That filter prevents traffic not
   specifically identified as legitimate from reaching the router's
   control plane, or rate-limits such traffic to an acceptable level.

   Note that the filters described in this memo are applied only to
   traffic that is destined for the router, and not to all traffic that
   is passing through the router.

Table of Contents

1. Introduction

Modern router architecture design maintains a strict separation of
forwarding and router control plane hardware and software. The
router control plane supports routing and management functions. It
is generally described as the router architecture hardware and
software components for handling packets destined to the device
itself as well as building and sending packets originated locally on
the device. The forwarding plane is typically described as the
router architecture hardware and software components responsible for
receiving a packet on an incoming interface, performing a lookup to
identify the packet's IP next hop and determine the best outgoing
interface towards the destination, and forwarding the packet out
through the appropriate outgoing interface.

Visually, this architecture can be represented as the router's
control plane hardware sitting on top of, and interfacing with, the
forwarding plane hardware with interfaces connecting to other network
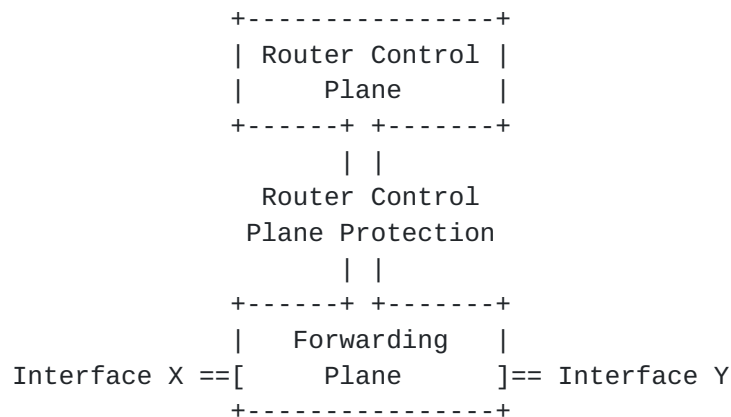devices. See Figure 1.

```
                    +----------------+
                    | Router Control |
                    |     Plane      |
                    +------+ +-------+
                        | |
                     Router Control
                    Plane Protection
                        | |
                    +------+ +-------+
                    |   Forwarding   |
         Interface X ==[    Plane      ]== Interface Y
                    +----------------+
```

                  Figure 1: Router Control Plane Protection

   Typically, forwarding plane functionality is realized in high-
   performance Application Specific Integrated Circuits (ASICs) that are
   capable of handling very high packet rates. By contrast, the router
   control plane is generally realized in software on general-purpose
   processors. While software instructions run on both planes, the
   router control plane hardware is usually not optimized for high-speed
   packet handling. Given their differences in packet-handling
   capabilities, the router's control plane hardware is more susceptible
   to being overwhelmed by a Denial-of-Service (DoS) attack than the
   forwarding plane's ASICs. It is imperative that the router control
   plane remain stable regardless of traffic load to and from the device
   because the router control plane is what drives the programming of
   the forwarding plane.

   The router control plane also processes traffic destined to the
   router, and because of the wider range of functionality is more
   susceptible to security vulnerabilities and a more likely target for
   a DoS attack than the forwarding plane.

   It is advisable 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). "Router control
   plane protection" is the concept of filtering or rate-limiting
   unwanted traffic that would be diverted from the forwarding plane up
   to the router control plane. The closer the filters and rate
   limiters are to the forwarding plane and line-rate hardware, the more
   effective the protection is and the more resistant the system is to
   DoS attacks. This memo demonstrates one example of how to deploy a
   policy filter that satisfies a set of sample traffic-matching,
   filtering, and rate-limiting criteria.

Note that the filters described in this memo are applied only to
traffic that is destined for the router, and not to all traffic that
is passing through the router.

2. Applicability Statement

The method described in Section 3 and depicted in Figure 1
illustrates how to protect the router control plane from unwanted
traffic. Recognizing that deployment scenarios will vary, the exact
implementation is not generally applicable in all situations. The
categorization of legitimate router control plane traffic is
critically important in a successful implementation.

The examples given in this memo are simplified and minimalistic,
designed to illustrate the concept of protecting the router's control
plane. From them, operators can extrapolate specifics based on their
unique configuration and environment. This document is about
semantics, and Appendix A exemplifies syntax. For additional router
vendor implementations, or other converged devices, the syntax should
be translated to the respective language in a manner that preserves
the semantics.

Additionally, the need to provide the router control plane with
isolation, stability, and protection against rogue packets has been
incorporated into router designs for some time. Consequently, there
may be other vendor or implementation specific router control plane
protection mechanisms that are active by default or always active.
Those approaches may apply in conjunction with, or in addition to,
the method described in Section 3 and illustrated in Appendices A.1
and A.2. Those implementations should be considered as part of an
overall traffic management plan but are outside the scope of this
document.

This method is applicable for IPv4 as well as IPv6 address families,
and the legitimate traffic example in Section 3.1 provides examples
for both.

3. Method

In this memo, the authors demonstrate how a filter protecting the
router control plane can be deployed. In Section 3.1, a sample
router is introduced, and all traffic that its control plane must
process is identified. In Section 3.2, filter design concepts are
discussed. Cisco (Cisco IOS software) and Juniper (JUNOS)
implementations are provided in Appendices A.1 and A.2, respectively.

3.1. Legitimate Traffic

In this example, the router control plane must process traffic (i.e.,
traffic destined to the router and not through the router) per the
following criteria:

o Drop all IP packets that are fragments (see Section 3.3)

o Permit ICMP and ICMPv6 traffic from any source, rate-limited to
   500 kbps for each category

o Permit OSPF traffic from routers within subnet 192.0.2.0/24 and
   OSPFv3 traffic from IPv6 Link-Local unicast addresses (fe80::/10)

o Permit internal BGP (iBGP) traffic from routers within subnets
   192.0.2.0/24 and 2001:db8:1::/48

o Permit external BGP (eBGP) traffic from eBGP peers 198.51.100.25,
   198.51.100.27, 198.51.100.29, and 198.51.100.31; and IPv6 peers
   2001:db8:100::25, 2001:db8:100::27, 2001:db8:100::29, and
   2001:db8:100::31

o Permit DNS traffic from DNS servers within subnet 198.51.100.0/30
   and 2001:db8:100:1::/64

o Permit NTP traffic from NTP servers within subnet 198.51.100.4/30
   and 2001:db8:100:2::/64

o Permit Secure SHell (SSH) traffic from network management stations
   within subnet 198.51.100.128/25 and 2001:db8:100:3::/64

o Permit Simple Network Management Protocol (SNMP) traffic from
   network management stations within subnet 198.51.100.128/25 and
   2001:db8:100:3::/64

o Permit RADIUS authentication and accounting replies from RADIUS
   servers 198.51.100.9, 198.51.100.10, 2001:db8:100::9, and
   2001:db8:100::10 that are listening on UDP ports 1812 and 1813
   (Internet Assigned Numbers Authority (IANA) RADIUS ports). Note
   that this does not accommodate a server using the original UDP
   ports of 1645 and 1646

o Permit all other IPv4 and IPv6 traffic that was not explicitly
   matched in a class above, rate-limited to 500 kbps, and drop above
   that rate for each category

o Permit non-IP traffic (e.g., Connectionless Network Service
   (CLNS), Internetwork Packet Exchange (IPX), PPP Link Control
   Protocol (LCP), etc.), rate-limited to 250 kbps, and drop all
   remaining traffic above that rate

The characteristics of legitimate traffic will vary from network to
network. To illustrate this, a router implementing the DHCP relay
function can rate-limit inbound DHCP traffic from clients and
restrict traffic from servers to a list of known DHCP servers. The
list of criteria above is provided for example only.

3.2. Filter Design

A filter is installed on the forwarding plane. This filter counts
and applies the actions to the categories of traffic described in
Section 3.1. Because the filter is enforced in the forwarding plane,
it prevents traffic from consuming bandwidth on the interface that
connects the forwarding plane to the router control plane. The
counters serve as an important forensic tool for the analysis of
potential attacks, and as an invaluable debugging and troubleshooting
aid. By adjusting the granularity and order of the filters, more
granular forensics can be performed (i.e., 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 would allow for counters to be monitored for the
allowed protocol filter, as well as any traffic matching the specific
protocol that didn't originate from the explicitly allowed hosts.

In addition to the filters, rate limiters for certain classes of
traffic are also installed in the forwarding plane as defined in
Section 3.1. These rate limiters help further control the traffic
that will reach the router control plane for each filtered class as
well as all traffic not matching an explicit class. The actual rates
selected for various classes are network deployment specific;
analysis of the rates required for stability should be done
periodically. It is important to note that the most significant
factor to consider regarding the traffic profile going to the router
control plane is the packets per second (pps) rate. Therefore,
careful consideration must be given to determine the maximum pps rate
that could be generated from a given set of packet size and bandwidth
usage scenarios.

Syntactically, these filters explicitly define "allowed" traffic
(including IP addresses, protocols, and ports), define acceptable
actions for these acceptable traffic profiles (e.g., rate-limit or
simply permit the traffic), and then discard all traffic destined to
the router control plane that is not within the specifications of the
policy definition.

In an actual production environment, predicting a complete and
exhaustive list of traffic necessary to reach the router's control
plane for day-to-day operation may not be as obvious as the example
described herein. One recommended method to gauge this set of
traffic is to allow all traffic initially, and audit the traffic that
reaches the router control plane before applying any explicit filters
or rate limits. See Section 3.3 below for more discussion of this
topic.

The filter design provided in this document is intentionally limited
to attachment at the local router in question (e.g., a "service-
policy" attached to the "control-plane" in Cisco IOS, or a firewall
filter attached to the "lo0" interface in JUNOS). While virtually
all production environments utilize and rely heavily upon edge
protection or interface filtering, these methods of router protection
are beyond the intended scope of this document. Additionally, the
protocols themselves that are allowed to reach the router control
plane (e.g., OSPF, RSVP, TCP, SNMP, DNS, NTP, and inherently, SSH,
TLS, ESP, etc.) may have cryptographic security methods applied to
them, and the method of router control plane protection provided
herein is not a replacement for those cryptographic methods.

3.3. Design Trade-Offs

In designing the protection method, there are two independent parts
to consider: the classification of traffic (i.e., which traffic is
matched by the filters), and the policy actions taken on the
classified traffic (i.e., drop, permit, rate-limit, etc.).

There are different levels of granularity utilized for traffic
classification. For example, allowing all traffic from specific
source IP addresses versus allowing only a specific set of protocols
from those specific source IP addresses will each affect a different
subset of traffic.

Similarly, the policy actions taken on the classified traffic have
degrees of impact that may not become immediately obvious. For
example, discarding all ICMP traffic will have a negative impact on
the operational use of ICMP tools such as ping or traceroute to debug
network issues or to test deployment of a new circuit. Expanding on
this, in a real production network, an astute operator could define

varying rate limits for ICMP such that internal traffic is granted
uninhibited access to the router control plane, while traffic from
external addresses is rate-limited. Operators should pay special
attention to the new functionality and roles that ICMPv6 has in the
overall operation of IPv6 when designing the rate-limit policies.
Example functions include Neighbor Discovery (ND) and Multicast
Listener Discovery version 2 (MLDv2).

It is important to note that both classification and policy action
decisions are accompanied by respective trade-offs. Two examples of
these trade-off decisions are operational complexity at the expense
of policy and statistics-gathering detail, and tighter protection at
the expense of network supportability and troubleshooting ability.

Two types of traffic that need special consideration are IP fragments
and IP optioned packets:

o For network deployments where IP fragmentation is necessary, a
   blanket policy of dropping all fragments destined to the router
   control plane may not be feasible. However, many deployments
   allow network configurations such that the router control plane
   should never see a fragmented datagram. Since many attacks rely
   on IP fragmentation, the example policy included herein drops all
   fragments destined to the router control plane.

o Similarly, some deployments may choose to drop all IP optioned
   packets. Others may need to loosen the constraint to allow for
   protocols that require IP optioned packets such as the Resource
   Reservation Protocol (RSVP). The design trade-off is that
   dropping all IP optioned packets protects the router from attacks
   that leverage malformed options, as well as attacks that rely on
   the slow-path processing (i.e., software processing path) of IP
   optioned packets. For network deployments where the protocols do
   not use IP options, the filter is simpler to design in that it can
   drop all packets with any IP option set. However, for networks
   utilizing protocols relying on IP options, the filter to identify
   the legitimate packets is more complex. If the filter is not
   designed correctly, it could result in the inadvertent blackholing
   of traffic for those protocols. This document does not include
   filter configurations for IP optioned packets; additional
   explanations regarding the filtering of packets based on the IP
   options they contain can be found in [IP-OPTIONS-FILTER].

The goal of the method for protecting the router control plane is to
minimize the possibility for disruptions by reducing the vulnerable
surface, which is inversely proportional to the granularity of the
filter design. The finer the granularity of the filter design (e.g.,
filtering a more targeted subset of traffic from the rest of the
policed traffic, or isolating valid source addresses into a different
class or classes), the smaller the probability of disruption.

In addition to the traffic that matches explicit classes, care should
be taken on the policy decision that governs the handling of traffic
that would fall through the classification. Typically, that traffic
is referred to as traffic that gets matched in a default class. It
may also be traffic that matches a blanket protocol specific class
where previous classes that have more granular classification did not
match all packets for that specific protocol. The ideal policy would
have explicit classes to match only the traffic specifically required
at the router control plane and would drop all other traffic that
does not match a predefined class. As most vendor implementations
permit all traffic hitting the default class, an explicit drop action
would need to be configured in the policy such that the traffic
hitting that default class would be dropped, versus being permitted
and delivered to the router control plane. This approach requires
rigorous traffic pattern identification such that a default drop
policy does not break existing device functionality. The approach
defined in this document allows the default traffic and rate-limits
it as opposed to dropping it. This approach was chosen as a way to
give the operator time to evaluate and characterize traffic in a
production scenario prior to dropping all traffic not explicitly
matched and permitted. However, it is highly recommended that after
monitoring the traffic matching the default class, explicit classes
be defined to catch the legitimate traffic. After all legitimate
traffic has been identified and explicitly allowed, the default class
should be configured to drop any remaining traffic.

Additionally, the baselining and monitoring of traffic flows to the
router's control plane are critical in determining both the rates and
granularity of the policies being applied. It is also important to
validate the existing policies and rules or update them as the
network evolves and its traffic dynamics change. Some possible ways
to achieve this include individual policy counters that can be
exported or retrieved, for example via SNMP, and logging of filtering
actions.

Finally, the use of flow-based behavioral analysis or command-line
interface (CLI) functions to identify what client/server functions a
given router's control plane handles would be very useful during
initial policy development phases, and certainly for ongoing forensic
analysis.

3.4. Additional Protection Considerations

   In addition to the design described in this document of defining
   "allowed" traffic (i.e., identifying traffic that the control plane
   must process) and limiting (e.g., rate-limiting or blocking) the
   rest, the router control plane protection method can be applied to
   thwart specific attacks. In particular, it can be used to protect
   against TCP SYN flooding attacks and other Denial-of-Service attacks
   that starve router control plane resources.

4. Security Considerations

   The filters described in this document leave the router susceptible
   to discovery from any host in the Internet. If network operators
   find this risk objectionable, they can reduce the exposure to
   discovery with ICMP by restricting the sub-networks from which ICMP
   Echo requests and potential traceroute packets (i.e., packets that
   would trigger an ICMP Time Exceeded reply) are accepted, and
   therefore to which sub-networks ICMP responses (ICMP Echo Reply and
   Time Exceeded) are sent. A similar concern exists for ICMPv6 traffic
   but on a broader level due to the additional functionalities
   implemented in ICMPv6. Filtering recommendations for ICMPv6 can be
   found in [RFC4890]. Moreover, different rate-limiting policies may
   be defined for internally (e.g., from the Network Operations Center
   (NOC)) versus externally sourced traffic. Note that this document is
   not targeted at the specifics of ICMP filtering or traffic filtering
   designed to prevent device discovery.

   The filters described in this document do not block unwanted traffic
   having spoofed source addresses that match a defined traffic profile
   as discussed in Section 3.1. Network operators can mitigate this
   risk by preventing source address spoofing with filters applied at
   the network edge. Refer to Section 5.3.8 of [RFC1812] for more
   information regarding source address validation. Other methods also
   exist for limiting exposure to packet spoofing, such as the
   Generalized Time to Live (TTL) Security Mechanism (GTSM) [RFC5082]
   and Ingress Filtering [RFC2827] [RFC3704].

   The ICMP rate limiter specified for the filters described in this
   document protects the router from floods of ICMP traffic; see
   Sections 3.1 and 3.3 for details. However, during an ICMP flood,
   some legitimate ICMP traffic may be dropped. Because of this, when
   operators discover a flood of ICMP traffic, they are highly motivated
   to stop it at the source where the traffic is being originated.

Additional considerations pertaining to the usage and handling of
traffic that utilizes the IP Router Alert Options can be found in
[RTR-ALERT-CONS], and additional IP options filtering explanations
can be found in [IP-OPTIONS-FILTER].

The treatment of exception traffic in the forwarding plane and the
generation of specific messages by the router control plane also
require protection from a DoS attack. Specifically, the generation
of ICMP Unreachable messages by the router control plane needs to be
rate-limited, either implicitly within the router's architecture or
explicitly through configuration. When possible, different ICMP
Destination Unreachable codes (e.g., "fragmentation needed and DF
set") or "Packet Too Big" messages can receive a different rate-
limiting treatment. Continuous benchmarking of router-generated ICMP
traffic should be done before applying rate limits such that
sufficient headroom is included to prevent inadvertent Path Maximum
Transmission Unit Discovery (PMTUD) blackhole scenarios during normal
operation. It is also recommended to deploy explicit rate limiters
where possible to improve troubleshooting and monitoring capability.
The explicit rate limiters in a class allow for monitoring tools to
detect and report when these rate limiters become active (i.e., when
traffic is policed). This in turn serves as an indicator that either
the normal traffic rates have increased or "out of policy" traffic
rates have been detected. More thorough analysis of the traffic
flows and rate-limited traffic is needed to identify which of these
two cases triggered the rate limiters. For additional information
regarding specific ICMP rate-limiting, see Section 4.3.2.8 of
[RFC1812].

Additionally, the handling of TTL / Hop Limit expired traffic needs
protection. This traffic is not necessarily addressed to the device,
but it can get sent to the router control plane to process the TTL /
Hop Limit expiration. For example, rate-limiting the TTL / Hop Limit
expired traffic before sending the packets to the router control
plane component that will generate the ICMP error, and distributing
the sending of ICMP errors to Line Card CPUs, are protection
mechanisms that mitigate attacks before they can negatively affect a
rate-limited router control plane component.

5. Acknowledgements

The authors would like to thank Ron Bonica for providing initial and
ongoing review, suggestions, and valuable input. Pekka Savola,
Warren Kumari, and Xu Chen provided very thorough and useful feedback
that improved the document. Many thanks to John Kristoff,
Christopher Morrow, and Donald Smith for a fruitful discussion around
the operational and manageability aspects of router control plane
protection techniques. The authors would also like to thank

Joel Jaeggli, Richard Graveman, Danny McPherson, Gregg Schudel, Eddie
Parra, Seo Boon Ng, Manav Bhatia, German Martinez, Wen Zhang, Roni
Even, Acee Lindem, Glen Zorn, Joe Abley, Ralph Droms, and Stewart
Bryant for providing thorough review, useful suggestions, and
valuable input. Assistance from Jim Bailey and Raphan Han in
providing technical direction and sample configuration guidance on
the IPv6 sections was also very much appreciated. Finally, the
authors extend kudos to Andrew Yourtchenko for his review, comments,
and willingness to present this document at IETF 78 (July 2010,
Maastricht, The Netherlands) on behalf of the authors.

6. Informative References

[IP-OPTIONS-FILTER]
          Gont, F. and S. Fouant, "IP Options Filtering
          Recommendations", Work in Progress, February 2010.

[RFC1812] Baker, F., Ed., "Requirements for IP Version 4 Routers",
          RFC 1812, June 1995.

[RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering:
          Defeating Denial of Service Attacks which employ IP Source
          Address Spoofing", BCP 38, RFC 2827, May 2000.

[RFC3704] Baker, F. and P. Savola, "Ingress Filtering for Multihomed
          Networks", BCP 84, RFC 3704, March 2004.

[RFC4890] Davies, E. and J. Mohacsi, "Recommendations for Filtering
          ICMPv6 Messages in Firewalls", RFC 4890, May 2007.

[RFC5082] Gill, V., Heasley, J., Meyer, D., Savola, P., Ed., and C.
          Pignataro, "The Generalized TTL Security Mechanism
          (GTSM)", RFC 5082, October 2007.

[RTR-ALERT-CONS]
          Le Faucheur, F., Ed., "IP Router Alert Considerations and
          Usage", Work in Progress, March 2011.

Appendix A. Configuration Examples

   The configurations provided below are syntactical representations of
   the semantics described in the document and should be treated as
   non-normative.

A.1. Cisco Configuration

   Refer to the Control Plane Policing (CoPP) document in the Cisco IOS
   Software Feature Guides (available at <http://www.cisco.com/>) for
   more information on the syntax and options available when configuring
   Control Plane Policing.

```
   !Start: Protecting The Router Control Plane
   !
   !Control Plane Policing (CoPP) Configuration
   !
   !Access Control List Definitions
   !
   ip access-list extended ICMP
    permit icmp any any
   ipv6 access-list ICMPv6
    permit icmp any any
   ip access-list extended OSPF
    permit ospf 192.0.2.0 0.0.0.255 any
   ipv6 access-list OSPFv3
    permit 89 FE80::/10 any
   ip access-list extended IBGP
    permit tcp 192.0.2.0 0.0.0.255 eq bgp any
    permit tcp 192.0.2.0 0.0.0.255 any eq bgp
   ipv6 access-list IBGPv6
    permit tcp 2001:DB8:1::/48 eq bgp any
    permit tcp 2001:DB8:1::/48 any eq bgp
   ip access-list extended EBGP
    permit tcp host 198.51.100.25 eq bgp any
    permit tcp host 198.51.100.25 any eq bgp
    permit tcp host 198.51.100.27 eq bgp any
    permit tcp host 198.51.100.27 any eq bgp
    permit tcp host 198.51.100.29 eq bgp any
    permit tcp host 198.51.100.29 any eq bgp
    permit tcp host 198.51.100.31 eq bgp any
    permit tcp host 198.51.100.31 any eq bgp
```

```
   ipv6 access-list EBGPv6
    permit tcp host 2001:DB8:100::25 eq bgp any
    permit tcp host 2001:DB8:100::25 any eq bgp
    permit tcp host 2001:DB8:100::27 eq bgp any
    permit tcp host 2001:DB8:100::27 any eq bgp
    permit tcp host 2001:DB8:100::29 eq bgp any
    permit tcp host 2001:DB8:100::29 any eq bgp
    permit tcp host 2001:DB8:100::31 eq bgp any
    permit tcp host 2001:DB8:100::31 any eq bgp
   ip access-list extended DNS
    permit udp 198.51.100.0 0.0.0.252 eq domain any
   ipv6 access-list DNSv6
    permit udp 2001:DB8:100:1::/64 eq domain any
    permit tcp 2001:DB8:100:1::/64 eq domain any
   ip access-list extended NTP
    permit udp 198.51.100.4 255.255.255.252 any eq ntp
   ipv6 access-list NTPv6
    permit udp 2001:DB8:100:2::/64 any eq ntp
   ip access-list extended SSH
    permit tcp 198.51.100.128 0.0.0.128 any eq 22
   ipv6 access-list SSHv6
    permit tcp 2001:DB8:100:3::/64 any eq 22
   ip access-list extended SNMP
    permit udp 198.51.100.128 0.0.0.128 any eq snmp
   ipv6 access-list SNMPv6
    permit udp 2001:DB8:100:3::/64 any eq snmp
   ip access-list extended RADIUS
    permit udp host 198.51.100.9 eq 1812 any
    permit udp host 198.51.100.9 eq 1813 any
    permit udp host 198.51.100.10 eq 1812 any
    permit udp host 198.51.100.10 eq 1813 any
   ipv6 access-list RADIUSv6
    permit udp host 2001:DB8:100::9 eq 1812 any
    permit udp host 2001:DB8:100::9 eq 1813 any
    permit udp host 2001:DB8:100::10 eq 1812 any
    permit udp host 2001:DB8:100::10 eq 1813 any
   ip access-list extended FRAGMENTS
    permit ip any any fragments
   ipv6 access-list FRAGMENTSv6
    permit ipv6 any any fragments
   ip access-list extended ALLOTHERIP
    permit ip any any
   ipv6 access-list ALLOTHERIPv6
    permit ipv6 any any
```

```
!
!Class Definitions
!
class-map match-any ICMP
 match access-group name ICMP
class-map match-any ICMPv6
 match access-group name ICMPv6
class-map match-any OSPF
 match access-group name OSPF
 match access-group name OSPFv3
class-map match-any IBGP
 match access-group name IBGP
 match access-group name IBGPv6
class-map match-any EBGP
 match access-group name EBGP
 match access-group name EBGPv6
class-map match-any DNS
 match access-group name DNS
 match access-group name DNSv6
class-map match-any NTP
 match access-group name NTP
 match access-group name NTPv6
class-map match-any SSH
 match access-group name SSH
 match access-group name SSHv6
class-map match-any SNMP
 match access-group name SNMP
 match access-group name SNMPv6
class-map match-any RADIUS
 match access-group name RADIUS
 match access-group name RADIUSv6
class-map match-any FRAGMENTS
 match access-group name FRAGMENTS
 match access-group name FRAGMENTSv6
class-map match-any ALLOTHERIP
 match access-group name ALLOTHERIP
class-map match-any ALLOTHERIPv6
 match access-group name ALLOTHERIPv6
```

```
!
!Policy Definition
!
policy-map COPP
 class FRAGMENTS
   drop
 class ICMP
  police 500000
      conform-action transmit
      exceed-action drop
      violate-action drop
 class ICMPv6
  police 500000
      conform-action transmit
      exceed-action drop
      violate-action drop
 class OSPF
 class IBGP
 class EBGP
 class DNS
 class NTP
 class SSH
 class SNMP
 class RADIUS
 class ALLOTHERIP
    police cir 500000
      conform-action transmit
      exceed-action drop
      violate-action drop
 class ALLOTHERIPv6
    police cir 500000
      conform-action transmit
      exceed-action drop
      violate-action drop
 class class-default
    police cir 250000
      conform-action transmit
      exceed-action drop
      violate-action drop
!
!Control Plane Configuration
!
control-plane
 service-policy input COPP
!
!End: Protecting The Router Control Plane
```

A.2. **Juniper Configuration**

   Refer to the Firewall Filter Configuration section of the Junos
   Software Policy Framework Configuration Guide (available at
   <http://www.juniper.net/>) for more information on the syntax and
   options available when configuring Junos firewall filters.

```
   policy-options {
       prefix-list IBGP-NEIGHBORS {
           192.0.2.0/24;
       }
       prefix-list EBGP-NEIGHBORS {
           198.51.100.25/32;
           198.51.100.27/32;
           198.51.100.29/32;
           198.51.100.31/32;
       }
       prefix-list RADIUS-SERVERS {
           198.51.100.9/32;
           198.51.100.10/32;
       }
       prefix-list IBGPv6-NEIGHBORS {
           2001:DB8:1::/48;
       }
       prefix-list EBGPv6-NEIGHBORS {
           2001:DB8:100::25/128;
           2001:DB8:100::27/128;
           2001:DB8:100::29/128;
           2001:DB8:100::31/128;
       }
       prefix-list RADIUSv6-SERVERS {
           2001:DB8:100::9/128;
           2001:DB8:100::10/128;
       }
   }
```

```
    firewall {
        policer 500kbps {
            if-exceeding {
                bandwidth-limit 500k;
                burst-size-limit 1500;
            }
            then discard;
        }
        policer 250kbps {
            if-exceeding {
                bandwidth-limit 250k;
                burst-size-limit 1500;
            }
            then discard;
        }
        family inet {
            filter protect-router-control-plane {
                term first-frag {
                    from {
                        first-fragment;
                    }
                    then {
                        count frag-discards;
                        log;
                        discard;
                    }
                }
                term next-frag {
                    from {
                        is-fragment;
                    }
                    then {
                        count frag-discards;
                        log;
                        discard;
                    }
                }
                term icmp {
                    from {
                        protocol icmp;
                    }
                    then {
                        policer 500kbps;
                        accept;
                    }
                }
```

```
            term ospf {
                from {
                    source-address {
                        192.0.2.0/24;
                    }
                    protocol ospf;
                }
                then accept;
            }
            term ibgp-connect {
                from {
                    source-prefix-list {
                        IBGP-NEIGHBORS;
                    }
                    protocol tcp;
                    destination-port bgp;
                }
                then accept;
            }
            term ibgp-reply {
                from {
                    source-prefix-list {
                        IBGP-NEIGHBORS;
                    }
                    protocol tcp;
                    port bgp;
                }
                then accept;
            }
            term ebgp-connect {
                from {
                    source-prefix-list {
                        EBGP-NEIGHBORS;
                    }
                    protocol tcp;
                    destination-port bgp;
                }
                then accept;
            }
```

```
            term ebgp-reply {
                from {
                    source-prefix-list {
                        EBGP-NEIGHBORS;
                    }
                    protocol tcp;
                    port bgp;
                }
                then accept;
            }
            term dns {
                from {
                    source-address {
                        198.51.100.0/30;
                    }
                    protocol udp;
                    port domain;
                }
                then accept;
            }
            term ntp {
                from {
                    source-address {
                        198.51.100.4/30;
                    }
                    protocol udp;
                    destination-port ntp;
                }
                then accept;
            }
            term ssh {
                from {
                    source-address {
                        198.51.100.128/25;
                    }
                    protocol tcp;
                    destination-port ssh;
                }
                then accept;
            }
```

```
            term snmp {
                from {
                    source-address {
                        198.51.100.128/25;
                    }
                    protocol udp;
                    destination-port snmp;
                }
                then accept;
            }
            term radius {
                from {
                    source-prefix-list {
                        RADIUS-SERVERS;
                    }
                    protocol udp;
                    port [ 1812 1813 ];
                }
                then accept;
            }
            term default-term {
                then {
                    count copp-exceptions;
                    log;
                    policer 500kbps;
                    accept;
                }
            }
        }
    }

    family inet6 {
        filter protect-router-control-plane-v6 {
            term fragv6 {
                from {
                    next-header fragment;
                }
                then {
                    count frag-v6-discards;
                    log;
                    discard;
                }
            }
```

```
            term icmpv6 {
                from {
                    next-header icmpv6;
                }
                then {
                    policer 500kbps;
                    accept;
                }
            }
            term ospfv3 {
                from {
                    source-address {
                        FE80::/10;
                    }
                    next-header ospf;
                }
                then accept;
            }
            term ibgpv6-connect {
                from {
                    source-prefix-list {
                        IBGPv6-NEIGHBORS;
                    }
                    next-header tcp;
                    destination-port bgp;
                }
                then accept;
            }
            term ibgpv6-reply {
                from {
                    source-prefix-list {
                        IBGPv6-NEIGHBORS;
                    }
                    next-header tcp;
                    port bgp;
                }
                then accept;
            }
            term ebgpv6-connect {
                from {
                    source-prefix-list {
                        EBGPv6-NEIGHBORS;
                    }
                    next-header tcp;
                    destination-port bgp;
                }
                then accept;
            }
```

```
                term ebgpv6-reply {
                    from {
                        source-prefix-list {
                            EBGPv6-NEIGHBORS;
                        }
                        next-header tcp;
                        port bgp;
                    }
                    then accept;
                }
                term dnsv6 {
                    from {
                        source-address {
                           2001:DB8:100:1::/64;
                            }
                        next-header [ udp tcp ];
                        port domain;
                    }
                    then accept;
                }
                term ntpv6 {
                    from {
                        source-address {
                            2001:DB8:100:2::/64;
                        }
                        next-header udp;
                        destination-port ntp;
                    }
                    then accept;
                }
                term sshv6 {
                    from {
                        source-address {
                            2001:DB8:100:3::/64;
                        }
                        next-header tcp;
                        destination-port ssh;
                    }
                    then accept;
                }
```

```
            term snmpv6 {
                from {
                    source-address {
                        2001:DB8:100:3::/64;
                    }
                    next-header udp;
                    destination-port snmp;
                }
                then accept;
            }
            term radiusv6 {
                from {
                    source-prefix-list {
                        RADIUSv6-SERVERS;
                    }
                    next-header udp;
                    port [ 1812 1813 ];
                }
                then accept;
            }
            term default-term-v6 {
                then {
                    policer 500kbps;
                    count copp-exceptions-v6;
                    log;
                    accept;
                }
            }
        }
    }

    family any {
        filter protect-router-control-plane-non-ip {
            term rate-limit-non-ip {
                then {
                    policer 250kbps;
                    accept;
                }
            }
        }
    }
}
```

```
    interfaces {
        lo0 {
            unit 0 {
                family inet {
                    filter input protect-router-control-plane;
                }
                family inet6 {
                    filter input protect-router-control-plane-v6;
                }
                family any {
                    filter input protect-router-control-plane-non-ip;
                }
            }
        }
    }
```

Authors' Addresses

    Dave Dugal
    Juniper Networks
    10 Technology Park Drive
    Westford, MA 01886
    US

    EMail: dave@juniper.net

    Carlos Pignataro
    Cisco Systems
    7200-12 Kit Creek Road
    Research Triangle Park, NC 27709
    US

    EMail: cpignata@cisco.com

    Rodney Dunn
    Cisco Systems
    7200-12 Kit Creek Road
    Research Triangle Park, NC 27709
    US

    EMail: rodunn@cisco.com