

CALM DURING THE STORM

Best Practices in Multicast Security

Table of Contents

Introduction	1
Description and Deployment Scenario	1
Design Considerations	2
Scope	2
Throttling Multicast Source Discovery Protocol	2
Per-Source Limits	2
Per-Peer Limits	3
Per-Instance Limits	4
Disabling Multicast Source Discovery Protocol Data Encapsulation	4
Multicast Source Discovery Protocol Filters	4
Multicast Boundaries	5
Protocol Independent Multicast Bootstrap Router Filters	6
Protocol Independent Multicast Join Filters	6
PIM Register Filters	7
PIM Passive Interfaces	8
Forwarding Cache Limit	9
Packet Filtering	9
Routing Engine Filtering	10
Session Announcement Protocol Storms	11
Putting It All Together	12
Summary	15
References	15
About Juniper Networks	15

Introduction

Internet multicast introduces a range of new security threats to a network. These threats are not necessarily any more or less destructive than those found in unicast-only networks, but they represent a new class of vulnerability that may be unfamiliar to those with minimal multicast experience. Juniper Networks® JUNOS® Software offers the most comprehensive set of features in the industry for securing a multicast infrastructure. This expertise comes from lessons learned after more than a decade of deployment experience in the world's largest Internet backbones. The following is a detailed set of recommended best practices for securing a multicast infrastructure of Juniper Networks routers.

Description and Deployment Scenario

Originally, multicast deployment recommendations focused on filtering certain groups that should not be leaked in or out of a network. Some of these groups are protocol related, some are from legacy applications, and some are administratively scoped. Examples of these groups include:

```

.....
224.0.1.2/32 SGI-Dogfight
224.0.1.3/32 RWHOD
224.0.1.8/32 SUN-NIS
224.0.1.22/32 SRVLOC
224.0.1.24/32 MICROSOFT-DS
224.0.1.25/32 NBC-PRO
224.0.1.35/32 SVRLOC-DA
224.0.1.39/32 AUTORP-Announce
224.0.1.40/32 AUTORP-Discovery
224.0.1.60/32 HP-Device-Discovery
224.0.2.1/32 RWHO
224.0.2.2/32 SUN-RPC
224.77.0.0/16 Norton-Ghost
226.77.0.0/16 Norton-Ghost
225.1.2.3/32 Altiris
229.55.150.208/32 Norton-Ghost
234.42.42.40/30 Phoenix/StorageSoft ImageCast
239.0.0.0/8 Administratively Scoped
.....

```

This list is added to periodically as providers discover traffic that should not be on the Internet. You can find the updated list of inappropriate group addresses in the Internet Engineering Task Force (IETF) Internet draft document, draft-ietf-mboned-ipv4-mcast-bcp. Over the past few years, the primary threat seen on multicast networks has been Multicast Source Discovery Protocol (MSDP) storms. Internet worms, such as Ramen, Slammer, and Sasser, have most often caused these storms. Typically, these worms infect a host and propagate by using these infected hosts to discover and attack other vulnerable hosts. To discover other vulnerable hosts, they usually select a large block of addresses (such as a /16) at random and port scan all hosts in that block. In just a few minutes, these infected hosts can scan up to tens or even hundreds of thousands of other hosts.

The coders of these worms often randomly select any IP address range, inadvertently including IP multicast addresses (224/4) in the possible range. When packets are transmitted with a destination address in 224/4 on a multicast-enabled network, multicast protocols use this data to drive the creation of multicast state. For example, when a Protocol Independent Multicast-Sparse Mode (PIM-SM) router detects that one of its locally connected hosts is transmitting multicast packets, it encapsulates these packets in PIM register messages and sends these messages to the PIM-SM Rendezvous Point (RP), which keeps the state of these local sources. The RP then creates an MSDP Source Active (SA) message for each source-group pair and floods it to all its MSDP peers, which cache these messages and flood them to all their respective peers, until MSDP speakers across the Internet have populated their SA caches. In this way, a single infected host that port scans only a single /16 of multicast addresses will generate more than 65,000 MSDP SAs that are flooded and cached by all MSDP speaking routers on the Internet. As more hosts attack and scan larger blocks of multicast addresses, this state explosion quickly causes MSDP-speaking routers to run out of memory and crash. This type of attack, which is not even launched to intentionally affect the multicast infrastructure, is known as an MSDP SA storm. These storms are quite easy to launch and have been by far the most common type of attack seen on multicast networks to date. Juniper has introduced a number of

features in its implementation of MSDP to mitigate and even eliminate the damage caused by MSDP storms. These features are described extensively in the following sections. Additionally, it should be noted that the damage done by MSDP storms is primarily accidental. That is, attackers damage critical components of the network infrastructure without even targeting these components. Juniper recommends a two-fold approach to hardening a multicast deployment: first, by enabling features that will reduce or eliminate the damage caused by attacks, and second, by reducing the size of the target for accidental attacks by using the concept of whitelisting. The most common use of filtering in network security involves blacklists. Blacklists simply contain a list of hosts or services that need to be denied, while all other hosts or services are permitted. Whitelists, on the other hand, contain a list of hosts or services that are permitted, while all other hosts or services are denied. So blacklists allow everything except what is on the list, while whitelists block everything except what is on the list.

According to RFC 3171, the Internet Assigned Numbers Authority (IANA) has allocated addresses for global use only from the 224/8, 232/8, and 233/8 address ranges. Multicast groups outside this range can be considered reserved from global use, and there is no legitimate reason to see this traffic on the Internet. Accordingly, Juniper strongly recommends allowing forwarding and control traffic only for groups in these whitelisted ranges. By permitting only 3/16 of the possible multicast address range to be routed, 81 percent of the accidental attacks should be prevented.

Design Considerations

The recommendations listed in this document are supported on all routers that run JUNOS Software. All features and commands mentioned have been available since JUNOS version 7.6 or earlier.

Scope

This document assumes a multicast deployment that uses static anycast RP as the RP-mapping mechanism. An in-depth discussion of the operation of PIM-SM and MSDP is outside of the scope of this document. Additionally, for full documentation on the JUNOS commands listed, consult the JUNOS Software multicast manual, *JUNOS Internet Software for J Series, M Series, and T Series Routing Platforms Multicast Protocols Configuration Guide*, at www.juniper.net/techpubs/software/junos.

Throttling Multicast Source Discovery Protocol

The most obvious method to prevent damage caused by MSDP storms is to rate limit. However, whenever you rate limit control traffic, you inadvertently introduce new vulnerability since it is usually impossible to tell the difference between a good control message and a bad one. For example, if you blindly rate limit all MSDP (TCP port 639) traffic to 1 Mbps, you simply lower the bar for attack. An attacker must send only 1 Mbps of MSDP traffic to engage the rate limiter, and all legitimate MSDP traffic is also blocked. Therefore, it is crucial that rate limiting be as intelligent as possible to deny the malicious, but permit the legitimate.

Per-Source Limits

The most important feature used to achieve intelligent rate limiting is per-source SA limits. Per-source limits prevent an MSDP speaker from allowing any single host to generate more than a configured maximum number of SAs. For example, a per-source limit of 1000 will not allow any individual host to generate more than 1000 SAs. A worm-infected host that port scanned a /16 of multicast addresses would then generate only 1000 SAs instead of more than 65,000. The configuration includes both a maximum and a threshold. SAs are randomly dropped using the random early detection (RED) algorithm after they exceed the threshold value. MSDP per-peer and per-instance rate limits use the same threshold and maximum syntax and behavior. It is difficult to imagine a host that legitimately sources more than 1000 multicast streams. In the event that there are known hosts that legitimately transmit more than 1000 multicast streams, these sources can be given larger limits. The following configuration allows the host 10.1.1.1 to generate up to 5000 MSDP SAs, while all other hosts on the Internet are rate limited to 1000 SAs.

```
.....  
msdp {  
    source 0.0.0.0/0 {  
        active-source-limit {  
            maximum 1000;  
            threshold 900;  
        }  
    }  
}
```

```

source 10.1.1.1/32 {
    active-source-limit {
        maximum 5000;
        threshold 4500;
    }
}

```

To see the per-source limits as they apply to all known multicast sources, use the **show msdp source** command:

```

lenny@paix> show msdp source

```

Source address	/Len	Type	Maximum	Threshold	Exceeded
0.0.0.0	/0	Configured	1000	900	0
10.0.32.32	/32	Dynamic	1000	900	0
10.14.59.8	/32	Dynamic	1000	900	0
10.14.59.57	/32	Dynamic	1000	900	1243
10.39.0.144	/32	Dynamic	1000	900	0
10.89.2.245	/32	Dynamic	1000	900	13

To see only the hosts that exceed the per-source limits, use the “show msdp source | except “\ 0” regular expression:

```

lenny@paix> show msdp source | except "\ 0"

```

Source address	/Len	Type	Maximum	Threshold	Exceeded
10.14.68.99	/32	Dynamic	1000	900	81099693
10.51.171.149	/32	Dynamic	1000	900	540763
10.88.60.37	/32	Dynamic	1000	900	9080
10.88.176.175	/32	Dynamic	1000	900	1144711
10.58.5.249	/32	Dynamic	1000	900	425

Per-Peer Limits

Per-peer SA limits rate limit the number of SAs received from an MSDP peer. These limits are functionally analogous to BGP maximum-prefix limits. The following configuration prevents the router from accepting more than 100,000 SAs from peer 1.1.1.1.

```

msdp {
    group eMSDP-peer {
        local-address 2.2.2.2;
        peer 1.1.1.1 {
            active-source-limit {
                maximum 100000;
                threshold 90000;
            }
        }
    }
}

```

To apply the same per-peer limit to all MSDP peers, use **apply-groups**.

Per-Instance Limits

Per-instance SA limits rate limit the maximum number of SAs that a router allows in the entire routing instance. The following configuration prevents the router from installing more than 200,000 SAs in its default routing instance.

```
msdp {
  active-source-limit {
    maximum 200000;
    threshold 190000;
  }
}
```

Disabling Multicast Source Discovery Protocol Data Encapsulation

To support bursty sources, MSDP SAs may also contain actual multicast data packets. Typically, the first packet in a multicast stream is added to the SA. MSDP SAs are placed in inet.4, the routing table that contains the MSDP SA cache. Additionally, SAs that include encapsulated data are placed in the forwarding table. The forwarding table generally has less capacity than the routing table. Therefore, it is recommended to prevent the creation of unnecessary forwarding table entries, as the forwarding table is a finite resource that also contains all the unicast forwarding entries. The following configuration prevents originating RPs from placing encapsulated data in SAs and ignores the encapsulated data in SAs received by peers. This sample configuration prevents MSDP from creating forwarding table entries. Note, however, that the configuration may have an impact on bursty source applications—for instance, when the source transmits one multicast packet every 20 minutes. If support for bursty source applications is critical, leave MSDP data encapsulation enabled.

```
msdp {
  data-encapsulation disable;
}
```

Multicast Source Discovery Protocol Filters

It is important to apply MSDP SA filters on all external MSDP sessions, inbound and outbound. MSDP SA filters prevent SAs for groups and sources that should remain inside a network from leaking in or out. At a minimum, these filters should be applied to all external MSDP peerings. The following configuration will prevent SAs (for sources and groups that do not belong on the Internet) from being transmitted or received by all MSDP peers.

```
protocols {
  msdp {
    export sa-filter;
    import sa-filter;
  }
}
policy-options {
  policy-statement sa-filter {
    term bad-groups {
      from {
        route-filter 224.0.1.2/32 exact;
        route-filter 224.0.1.3/32 exact;
        route-filter 224.0.1.8/32 exact;
        route-filter 224.0.1.22/32 exact;
        route-filter 224.0.1.24/32 exact;
        route-filter 224.0.1.25/32 exact;
        route-filter 224.0.1.35/32 exact;
        route-filter 224.0.1.39/32 exact;
        route-filter 224.0.1.40/32 exact;
        route-filter 224.0.1.60/32 exact;
        route-filter 224.0.2.1/32 exact;
      }
    }
  }
}
```

```

        route-filter 224.0.2.2/32 exact;
        route-filter 224.77.0.0/16 orlonger;
        route-filter 226.77.0.0/16 orlonger;
        route-filter 225.1.2.3/32 exact;
        route-filter 229.55.150.208/32 exact;
        route-filter 232.0.0.0/8 orlonger;
        route-filter 234.42.42.40/30 orlonger;
        route-filter 239.0.0.0/8 orlonger;
    }
    then reject;
}
term bad-sources {
    from {
        source-address-filter 10.0.0.0/8 orlonger;
        source-address-filter 127.0.0.0/8 orlonger;
        source-address-filter 172.16.0.0/12 orlonger;
        source-address-filter 192.168.0.0/16 orlonger;
    }
    then reject;
}
term accept-everything-else {
    then accept;
}
}
}
}

```

Multicast Boundaries

Apply multicast boundary filters on all customer-facing interfaces by using multicast scoping. Multicast scoping prevents multicast packets from flowing into or out of an interface. Apply these scopes on all interfaces and on all routers in your network, since there is usually no good reason for these groups to flow on backbone links. The following configuration prevents multicast data packets from flowing into or out of all interfaces on the router for groups that do not belong on the Internet. This configuration also permits data packets in 239/8 to flow over backbone links, which is necessary on networks that allow 239/8 traffic for internal purposes.

```

routing-options {
    multicast {
        scope-policy boundary-filter;
    }
}
policy-options {
    policy-statement boundary-filter {
        term permit-239-on-backbone {
            from {
                interface [ so-0/0/0.0 so-1/0/0.0 ];
                route-filter 239.0.0.0/8 orlonger;
            }
            then accept;
        }
    }
    term bad-groups {
        from {
            route-filter 224.0.1.2/32 exact;
            route-filter 224.0.1.3/32 exact;
            route-filter 224.0.1.8/32 exact;
            route-filter 224.0.1.22/32 exact;
            route-filter 224.0.1.24/32 exact;
            route-filter 224.0.1.25/32 exact;
            route-filter 224.0.1.35/32 exact;
        }
    }
}

```



```

term permit-239-on-backbone {
    from {
        interface [ so-0/0/0.0 so-1/0/0.0 ];
        route-filter 239.0.0.0/8 orlonger;
    }
    then accept;
}
term bad-groups {
    from {
        route-filter 224.0.1.2/32 exact;
        route-filter 224.0.1.3/32 exact;
        route-filter 224.0.1.8/32 exact;
        route-filter 224.0.1.22/32 exact;
        route-filter 224.0.1.24/32 exact;
        route-filter 224.0.1.25/32 exact;
        route-filter 224.0.1.35/32 exact;
        route-filter 224.0.1.39/32 exact;
        route-filter 224.0.1.40/32 exact;
        route-filter 224.0.1.60/32 exact;
        route-filter 224.0.2.1/32 exact;
        route-filter 224.0.2.2/32 exact;
        route-filter 224.77.0.0/16 orlonger;
        route-filter 226.77.0.0/16 orlonger;
        route-filter 225.1.2.3/32 exact;
        route-filter 229.55.150.208/32 exact;
        route-filter 234.42.42.40/30 orlonger;
        route-filter 239.0.0.0/8 orlonger;
    }
    then reject;
}
term bad-sources {
    from {
        source-address-filter 10.0.0.0/8 orlonger;
        source-address-filter 127.0.0.0/8 orlonger;
        source-address-filter 172.16.0.0/12 orlonger;
        source-address-filter 192.168.0.0/16 orlonger;
    }
    then reject;
}
term accept-everything-else {
    then accept;
}
}
}

```

PIM Register Filters

PIM register filters can be used to prevent unwanted traffic from creating PIM register state on the RP. These filters can be applied on the RP as well as on the source's designated router (DR). By filtering at the DR, the register messages are never sent to the RP. By filtering at the RP, unwanted register messages sent from any DR will be discarded.

PIM registers can be filtered based on the source address or group address. The following example shows a DR filter that allows local sources in 10.1.1.0/24 to register for groups in 224.1.0.0/16. All PIM registers for traffic outside these sources and groups will be prevented from sending registers to the RP.

```
.....  
protocols {  
    pim {  
        rp {  
            dr-register-policy dr-filter;  
        }  
    }  
}  
policy-options {  
    policy-statement dr-filter {  
        term permitted-sources-groups {  
            from {  
                route-filter 224.1.0.0/16 orlonger;  
                source-address-filter 10.1.1.0/24 orlonger;  
            }  
            then accept;  
        }  
        term deny-everything-else {  
            then reject;  
        }  
    }  
}
```

The following example shows an RP filter that also allows sources in 10.1.1.0/24 to create register state on the RP for groups in 224.1.0.0/16. All PIM registers for traffic outside these sources and groups from all DRs will be filtered on this RP.

```
.....  
protocols {  
    pim {  
        rp {  
            rp-register-policy rp-filter;  
        }  
    }  
}  
policy-options {  
    policy-statement rp-filter {  
        term permitted-sources-groups {  
            from {  
                route-filter 224.1.0.0/16 orlonger;  
                source-address-filter 10.1.1.0/24 orlonger;  
            }  
            then accept;  
        }  
        term deny-everything-else {  
            then reject;  
        }  
    }  
}
```

PIM Passive Interfaces

When more than one router on a LAN is running PIM, DR election is performed. The router with the highest DR priority is elected as the DR. If the priorities are the same, the router with the highest IP address is usually elected as the DR. On a stub network with only one router, it may be desirable to ensure that the router is always elected as the DR. This approach prevents a malicious host or misconfigured router from hijacking the DR assignment, which can prevent all hosts on the LAN from sourcing or joining multicast traffic outside the LAN.

To ensure that the stub router becomes the DR and ignores all other PIM neighbors on the LAN, you can set the **hello-interval** value to 0. This setting will simulate a passive interface, where the protocol is enabled on the interface, but no hellos are transmitted or received on this interface. Passive interfaces are commonly used with interior gateway protocols (IGPs), such as OSPF and IS-IS, to accomplish a similar result.

```

.....
protocols {
  pim {
    interface ge-0/0/0.0 {
      hello-interval 0;
    }
  }
}
.....

```

Forwarding Cache Limit

Inet.1 is the table that contains the multicast forwarding cache, which includes all PIM entries as well as MSDP SAs with encapsulated data. You limit the number of inet.1 entries on the router by configuring a forwarding cache limit. The multicast forwarding cache is added to the forwarding table, which the Packet Forwarding Engine (PFE) uses for unicast and multicast forwarding decisions. The following configuration limits the number of entries in the forwarding-cache to 150,000. After this limit is reached, the router cannot add entries until the forwarding cache drops to 149,000.

```

.....
routing-options {
  multicast {
    forwarding-cache {
      threshold {
        suppress 150000;
        reuse 149000;
      }
    }
  }
}
.....

```

Packet Filtering

You can use packet filters to deny certain multicast traffic based on values in the IP header. While scoping blocks multicast data packets based on group address, firewall filters block data packets based on values such as source address or protocol. For example, many worms use TCP packets when they probe. There is no legitimate reason for multicast TCP packets, so you can deny them on all routers. Filtering illegitimate multicast traffic such as TCP packets with firewall filters prevents data forwarding as well as state creation of unwanted traffic.

```

.....
interfaces {
  fe-0/0/0 {
    unit 0 {
      family inet {
        filter {
          input mcast-packet-filter;
        }
      }
    }
  }
}
firewall {
  filter mcast-packet-filter {
    term deny-tcp {
.....

```

```
        from {
            destination-address {
                224.0.0.0/4;
            }
            protocol tcp;
        }
        then {
            count mcast-tcp;
            discard;
        }
    }
    term allow-everything-else {
        then accept;
    }
}
}
```

Routing Engine Filtering

Filtering of control traffic to the Routing Engine (RE) is a critical step in securing routers. Firewall filters on loopback interfaces provide this function. In the case of MSDP, only configured peers should be allowed to send MSDP packets to a router. The following configuration allows only MSDP packets (TCP port 639) to be sent to the RE if the source address is one of its configured MSDP peers. Note that the **apply-path** command enables this filter to be applied automatically to all peers configured under MSDP, so this filter does not need to be edited when MSDP peers are added or removed.

```
interfaces {
    lo0 {
        unit 0 {
            family inet {
                filter {
                    input Protect-RE;
                }
            }
        }
    }
}
policy-options {
    prefix-list MSDP-Peers {
        apply-path "protocols msdp group <*> peer <*>";
    }
}
firewall {
    filter Protect-RE {
        term MSDP-Allow {
            from {
                source-prefix-list {
                    MSDP-Peers;
                }
                protocol tcp;
                port msdp;
            }
            then accept;
        }
    }
}
}
```

Also use the **apply-path** command for BGP peers. The loopback filter should include all other protocols required to reach the router's control plane such as PIM, OSPF, Internet Group Management Protocol (IGMP), SSH, Domain Name System (DNS), Internet Control Message Protocol (ICMP), and SNMP.

Session Announcement Protocol Storms

Another common source of trouble on multicast networks has been the Session Announcement Protocol (SAP). SAP is an advertisement protocol used by sources to inform receivers of an available multicast session. Session Description Protocol (SDP) messages are sent over the well-known SAP group, 224.2.127.254, and contain the pertinent information describing a multicast session. Receiving hosts can join this SAP group and learn of available multicast sessions, in a manner similar to viewing a program guide.

It has been a common practice on many networks to enable routers to join and listen to the SAP group and cache these SDP messages, instead of merely forwarding this group to interested receivers. With SAP listening enabled, you can view the SAP/SDP cache as seen by the router with the `show multicast sessions` command. This functionality has been used by operators as a quick and easy way to see whether multicast is working properly on a network. If the router sees a large number of sessions, then multicast is probably working properly on the network; if not, something is amiss and further investigation is needed.

Unfortunately, SAP has been a source of many problems. Misbehaving sources have been known to originate malformed SDP packets that have crashed routers and receiving hosts. Additionally, misconfigured SAP servers have been known to multicast the actual data session accidentally over the SAP group, also crashing routers, firewalls, and receiving hosts, which were not expecting high-data-rate flows on this well-known announcement group.

These SAP storms are another example of successful attacks upon the multicast infrastructure without even trying. Whatever limited, vague benefits may be gained from enabling routers to cache these messages have proven in practice to not be worth the risk. Thus, it is recommended that routers should not listen to the SAP group or cache these messages. By default, SAP listening is not enabled (`set protocols sap`) in JUNOS software, so removing this configuration (`delete protocols sap`) or not configuring it in the first place will ensure that routers will not be affected by SAP storms.

Disabling SAP caching on the routers will have no effect on hosts interested in receiving SAP announcements. Routers will continue to happily forward SAP packets blindly to interested receivers. Some have suggested policing 224.2.127.254 to some low value such as 1 Mbps to protect firewalls and hosts as well as downstream routers that still listen to SAP. As mentioned earlier, simply policing control traffic actually makes DoS easier. With a 1-Mbps policer on all traffic to 224.2.127.254, only one malicious host is needed to send 1 Mbps of bad traffic to the SAP group to engage this policer and block all legitimate SAP traffic, effectively killing the SAP service for all hosts.

Note that seeing unwanted traffic on a given multicast group is an inherent weakness of the any-source multicast (ASM) service model. The only real way to prevent traffic from unwanted sources is to use the source-specific multicast (SSM) service model, in which traffic is forwarded only from explicitly requested sources. The SAP group is a particularly attractive target for attack since it can be reliably assumed to have many persistent receivers. In fact, of all the well-known ASM groups, it is probably the best known and most commonly joined group. If you still want to protect receivers and firewalls from unwanted traffic on the SAP group without making it easier to cripple the SAP service, you can monitor the traffic statistics for each source for the SAP group to see whether any particular source is sending too much traffic via the `show multicast route group 224.2.127.254 detail` command. This monitoring can be automated using a JUNOS software event script. If one source is found to be sending excess traffic, a simple policer can be applied to that source to protect the rest of the SAP service.

```

firewall {
  policer sap-policer {
    if-exceeding {
      bandwidth-limit 1m;
      burst-size-limit 10k;
    }
    then discard;
  }
  filter bad-sap-source {
    term one {
      from {

```



```
protocols {
  msdp {
    apply-groups MSDP-Per-Peer-Limit;
    data-encapsulation disable;
    active-source-limit {
      maximum 200000;
      threshold 190000;
    }
    export [ Bogon-Sources ASM-Allow ];
    import [ Bogon-Sources ASM-Allow ];
    source 0.0.0.0/0 {
      active-source-limit {
        maximum 1000;
        threshold 900;
      }
    }
    group EMSDP-Peer1 {
      local-address 1.1.1.1;
      peer 2.2.2.2;
    }
  }
  pim {
    import [ Bogon-Sources SSM-Allow ASM-Allow ];
    rp {
      bootstrap-import BSR-Deny;
      bootstrap-export BSR-Deny;
      rp-register-policy [ Bogon-Sources ASM-Allow ];
      local {
        address 1.1.1.1;
      }
    }
    interface all {
      mode sparse;
      version 2;
    }
  }
}
policy-options {
  prefix-list MSDP-Peers {
    apply-path "protocols msdp group <*> peer <*>";
  }
  prefix-list BGP-Peers {
    apply-path "protocols bgp group <*> neighbor <*>";
  }
  policy-statement ASM-Allow {
    term Bogon-Groups {
      from {
        route-filter 224.0.1.2/32 exact;
        route-filter 224.0.1.3/32 exact;
        route-filter 224.0.1.8/32 exact;
        route-filter 224.0.1.22/32 exact;
        route-filter 224.0.1.24/32 exact;
        route-filter 224.0.1.25/32 exact;
        route-filter 224.0.1.35/32 exact;
        route-filter 224.0.1.39/32 exact;
        route-filter 224.0.1.40/32 exact;
        route-filter 224.0.1.60/32 exact;
        route-filter 224.0.2.1/32 exact;
        route-filter 224.0.2.2/32 exact;
        route-filter 224.77.0.0/16 orlonger;
      }
    }
  }
}
```

```
    }
    then reject;
  }
  term ASM-Whitelist {
    from {
      route-filter 224.0.0.0/8 orlonger;
      route-filter 233.0.0.0/8 orlonger;
    }
    then accept;
  }
  term Deny-Everything-Else {
    then reject;
  }
}
policy-statement BSR-Deny {
  then reject;
}
policy-statement Bogon-Sources {
  term RFC-1918-Addresses {
    from {
      source-address-filter 10.0.0.0/8 orlonger;
      source-address-filter 127.0.0.0/8 orlonger;
      source-address-filter 172.16.0.0/12 orlonger;
      source-address-filter 192.168.0.0/16 orlonger;
    }
    then reject;
  }
}
policy-statement SSM-Allow {
  term SSM-Whitelist {
    from {
      route-filter 232.0.0.0/8 orlonger;
    }
    then accept;
  }
}
}
firewall {
  filter Protect-RE {
    term MSDP-Allow {
      from {
        source-prefix-list {
          MSDP-Peers;
        }
        protocol tcp;
        port msdp;
      }
      then accept;
    }
    term BGP-Allow {
      from {
        source-prefix-list {
          BGP-Peers;
        }
        protocol tcp;
        port bgp;
      }
      then accept;
    }
    term PIM-Allow {
```

