Understand Route Aggregation in BGP

Contents

Introduction

Prerequisites

- Requirements
- Components Used
- Conventions

Background Information

Network Diagram

Aggregate without theas-set Argument

Aggregate with theas-set Argument

Change the Attributes of the Aggregate Route

Useadvertise-map to Aggregate a Subset of Specific Routes

Impact of the Use ofsuppress-map with Other Configuration Commands

Related Information

Introduction

This document describes how to manipulate the different attributes when you use the **aggregate-address** command and how to influence the propagation.

Prerequisites

Requirements

Cisco recommends that you have knowledge of this topic:

- Basic BGP operation. For more information, see BGP Case Studies.

Components Used

This document is not restricted to specific software and hardware versions. However, the configuration in this document was tested with Cisco IOS® Software Release 12.2(28).

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

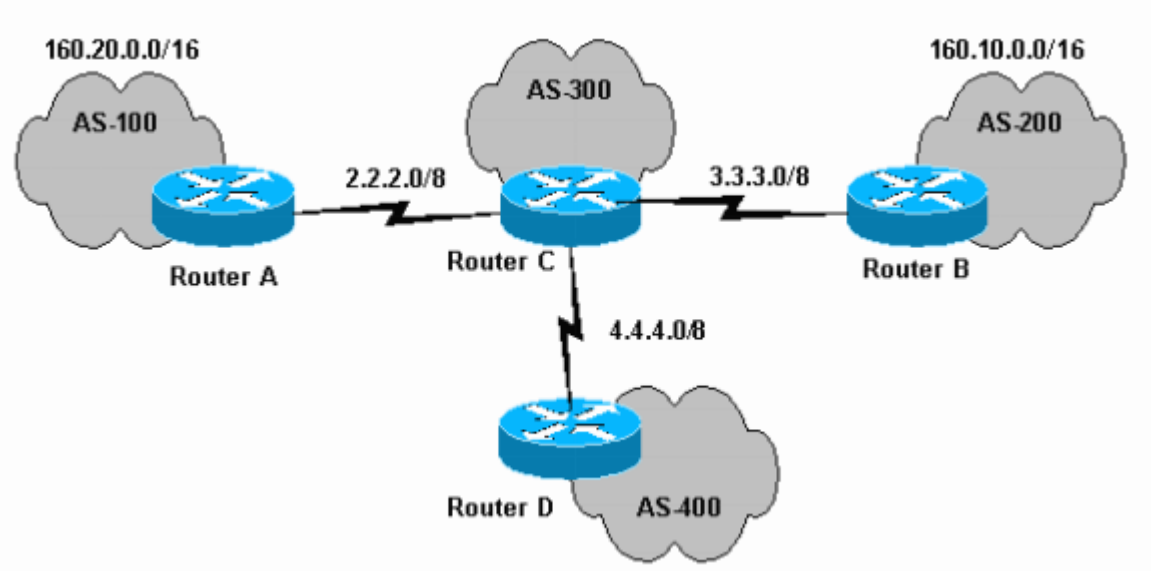
Conventions

Refer to Cisco Technical Tips Conventions for more information on document conventions.

Background Information

Border Gateway Protocol (BGP) allows the aggregation of specific routes into one route with use of the **aggregate-address address mask [as-set] [summary-only] [suppress-map map-name] [advertise-map map-name] [attribute-map map-name]** command. When you issue the **aggregate-address** command without any arguments, there is no inheritance of the individual route attributes (such as AS_PATH or community), which causes a loss of granularity. This document illustrates how to manipulate the different attributes when you use the **aggregate-address** command and how to influence the propagation.

Network Diagram



Aggregate without the as-set Argument

Use of the **as-set** argument creates an aggregate address with a mathematical set of autonomous systems (AS). This **as-set** argument summarizes the AS_PATH attributes of all the individual routes. These sample

configurations enable you to examine this feature and how this argument helps BGP detect and avoid loops.

Router A

```
Current configuration:

 hostname RouterA
 !
 interface Serial1
  ip address 2.2.2.2 255.0.0.0
 !
interface Loopback0
  ip address 160.20.1.1 255.255.0.0
 !
 router bgp 100
  network 160.20.0.0

!--- Router A advertises network 160.20.0.0/16.

  neighbor 2.2.2.1 remote-as 300
 !
 end
```

Router B

```
Current configuration:

 hostname RouterB
 !
 interface Serial0
  ip address 3.3.3.3 255.0.0.0
 !
interface Loopback0
  ip address 160.10.1.1 255.255.0.0
 !
 router bgp 200
  network 160.10.0.0

!--- Router B advertises network 160.10.0.0/16.

  neighbor 3.3.3.1 remote-as 300
 !
 end
```

Router C

```
Current configuration:

 hostname RouterC
 !
 interface Serial0
  ip address 2.2.2.1 255.0.0.0
 !
 interface Serial1
  ip address 3.3.3.1 255.0.0.0
 !
```

```
interface Serial2
 ip address 4.4.4.1 255.0.0.0
!
router bgp 300
 neighbor 2.2.2.2 remote-as 100
 neighbor 2.3.3.3 remote-as 200
 neighbor 4.4.4.4 remote-as 400
 aggregate-address 160.0.0.0 255.0.0.0 summary-only

!--- The network is summarized, and Router C only
!--- advertises 160.0.0.0/8.

!
end
```

Router D

```
Current configuration:

hostname RouterD
!
interface Serial0
 ip address 4.4.4.4 255.0.0.0
!
router bgp 400
 neighbor 4.4.4.1 remote-as 300
!
end
```

Router C (AS-300) aggregates the routes 160.20.0.0/16 and 160.10.0.0/16 that come from AS-100 and AS-200, respectively. This action occurs because you have configured the **summary-only** argument on Router C. Router C only announces the aggregate 160.0.0.0/8 to Router D. The aggregate 160.0.0.0/8 is the classless inter-domain routing (CIDR) route. The more specific 160.10.0.0/16 and 160.20.0.0/16 routes are suppressed, as this BGP table on Router C shows:

<#root>

```
RouterC#
```

show ip bgp

```
BGP table version is 6, local router ID is 4.4.4.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 10.160.0.0/8      0.0.0.0                              32768 i
```

s>

```
 10.160.10.0       3.3.3.3                  0             0 200 i
```

s>

```
 10.160.20.0       2.2.2.2                  0             0 100 i
```

Here is the BGP table of Router D. Observe the path information of the aggregate route:

<#root>

```
RouterD#
```

show ip bgp

```
BGP table version is 6, local router ID is 4.4.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 160.0.0.0/8      4.4.4.1                                0 300 i
```

The aggregate route 160.0.0.0/8 is considered to have originated from AS-300 with origin code IGP. The route has lost all the specific AS_PATH information of the individual prefixes 160.10.0.0/16, of AS-200, and 160.20.0.0/16, of AS-100.

Aggregate with the as-set Argument

Now configure the **as-set** argument in the aggregate-address command on Router C. Here is the new configuration:

Router C

<#root>

```
Current configuration:

 hostname RouterC
 !
 interface Serial0
  ip address 2.2.2.1 255.0.0.0
 !
 interface Serial1
  ip address 3.3.3.1 255.0.0.0
 !
 interface Serial2
  ip address 4.4.4.1 255.0.0.0
 !
 router bgp 300
  neighbor 2.2.2.2 remote-as 100
  neighbor 3.3.3.3 remote-as 200
  neighbor 4.4.4.4 remote-as 400
  aggregate-address 160.0.0.0 255.0.0.0 summary-only
```

as-set

```
!--- With the
as-set
 configuration command, the aggregate
!--- inherits the attributes of the more-specific routes.
 !
 end
```

NReview how this argument influences the **show ip bgp** output on Router:

<#root>

```
RouterD#
show ip bgp

BGP table version is 2, local router ID is 4.4.4.4
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 160.0.0.0/8      4.4.4.1                                0 300 {200,100} i
```

With the **as-set** argument, the path information in the BGP table for the aggregate route changes to include a set from 300 {200,100}. This set indicates that the aggregate actually summarizes routes that have passed through AS-200 and AS-100. The **as-set** nformation becomes important in the avoidance of routing loops because the information records where the route has been.

In any closed network, this aggregate information propagates through BGP and back to one of the AS that the **as-set** lists. This propagation creates the possibility of a loop. The loop detection behavior of BGP notes its own AS number in the **as-set** of the aggregate update and drops the aggregate. This action prevents a loop.

Note: The as-set argument contains information about each individual route that the aggregate summarizes. Changes in the individual route cause an update of the aggregate. In the example, if 160.10.0.0/16 goes down, the path information of the aggregate changes from 300 {200,100} to 300 {200}. The aggregate is updated. If the aggregate summarizes tens or hundreds of routes and the routes that form the aggregate have problems, there can be a constant flap.

Change the Attributes of the Aggregate Route

The Aggregate with the as-set Argument section shows you how to use **as-set** to save the AS_PATH attributes with a specific route. In some cases, you can require a change in the attributes of the aggregate route.

Examples of such attributes include metric, community, and origin.

Impact

This section shows how you can use the **attribute-map** argument to manipulate the **aggregate-address** attributes. In this case, you configure one or more of the specific aggregated routes with the **no-export** community attribute. Router A sets the community attribute **no-export** to network 160.20.0.0/16 and announces the network to Router C. This section shows the configuration. Router C inherits the community attribute **no-export** while the router aggregates 160.0.0.8. Therefore, there is no advertisement of 160.0.0.0/8 to Router D. The configuration of Routers B, C, and D do not change. Here is the new configuration for Router A:

Router A

```
<#root>

Current configuration:

 hostname RouterA

 !
 interface Serial1
  ip address 2.2.2.2 255.0.0.0
 !
 router bgp 100
  network 160.20.0.0

!--- Router A advertises network 160.20.0.0/16.

  neighbor 2.2.2.1 remote-as 300
  neighbor 2.2.2.1 send-community
  neighbor 2.2.2.1 route-map SET_NO_EXPORT out
 !
 access-list 1 permit 160.20.0.0 0.0.255.255
 route-map SET_NO_EXPORT permit 10
  match ip address 1
  set community no-export

!--- This sets the community attribute

no-export

.  at Router A for route 160.20.0.0/16 ! end
```

Here is the BGP table of Router C for 160.0.0.0/8:

```
<#root>

RouterC#

show ip bgp 160.0.0.0


BGP routing table entry for 160.0.0.0/8, version 9
Paths: (1 available, best #1, not advertised to EBGP peer)
  Not advertised to any peer
  {200,100}, (aggregated by 300 4.4.4.1)
```

```
    0.0.0.0 from 0.0.0.0 (4.4.4.1)
      Origin IGP, localpref 100, weight 32768, valid, aggregated, local, atomic-
aggregate, best, ref 2
      Community: no-export
```

The community **no-export** stops the Router C announcement of the aggregate route to eBGP peer Router D. Router D shows that it has not learned 160.0.0.0 from Router C:

<#root>

```
RouterD#
```

show ip bgp 160.0.0.0

```
% Network not in table
```

You can configure the **attribute-map** argument at Router C in order to manipulate the community attribute of the aggregate route from **no-export** to **none**. This configuration allows the advertisement of the aggregate to Router D.

Router C

<#root>

```
Current configuration:

 hostname RouterC
 !
 interface Serial0
  ip address 2.2.2.1 255.0.0.0
 !
 interface Serial1
  ip address 3.3.3.1 255.0.0.0
 !
 interface Serial2
  ip address 4.4.4.1 255.0.0.0
 !
 router bgp 300
  neighbor 2.2.2.2 remote-as 100
  neighbor 3.3.3.3 remote-as 200
  neighbor 4.4.4.4 remote-as 400
  aggregate-address 160.0.0.0 255.0.0.0
  as-set summary-only attribute-map Map

!--- Use of the
```

attribute-map

```
 argument allows
!--- you to change the community of the aggregate.

 !
 route-map Map permit 10
  set community none

!--- This sets the community of the aggregate to
```

```
none

.

 end
```

Now, look at the BGP table of Router C for 160.0.0.0/8. Because there is no community set for the aggregate route, Router C advertises 160.0.0.0/8 to Router D.

<#root>

```
RouterC#

show ip bgp 160.0.0.0


BGP routing table entry for 10.160.0.0/8, version 6
Paths: (1 available, best #1)
  Advertised to non peer-group peers:
  2.2.2.2  3.3.3.3  4.4.4.4
  {200,100}, (aggregated by 300 4.4.4.1)
    0.0.0.0 from 0.0.0.0 (4.4.4.1)
      Origin IGP, localpref 100, weight 32768, valid, aggregated, local, atomic-
aggregate, best, ref 2
```

The **show ip bgp 160.0.0.0** output at Router D shows that Router D has learned the aggregate route 160.0.0.0/8 from Router C.

<#root>

```
RouterD#

show ip bgp 160.0.0.0


BGP routing table entry for 160.0.0.0/8, version 10
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Not advertised to any peer
  300 {200,100}, (aggregated by 300 4.4.4.1)
    4.4.4.1 from 4.4.4.1 (4.4.4.1)
      Origin IGP, localpref 100, valid, external, best
```

Use advertise-map to Aggregate a Subset of Specific Routes

If you have control over the individual prefixes that form the aggregate route, you can more easily decide which attributes the aggregate can carry. Exclude prefix 160.20.0.0 from the aggregate route in the example in the Change the Attributes of the Aggregate Route section. In this case, the aggregate 160.0.0.0/8 does not inherit the community attribute **no-export** . In order to make this change, configure the **advertise-map** argument at Router C.

Router C

```
<#root>

Current configuration:

 hostname RouterC
 !
 interface Serial0
  ip address 2.2.2.1 255.0.0.0
 !
 interface Serial1
  ip address 3.3.3.1 255.0.0.0
 !
 interface Serial2
  ip address 4.4.4.1 255.0.0.0
 !
 router bgp 300
  neighbor 2.2.2.2 remote-as 100
  neighbor 3.3.3.3 remote-as 200
  neighbor 4.4.4.4 remote-as 400
  aggregate-address 160.0.0.0 255.0.0.0
  as-set summary-only advertise-map SELECT_SP_ROUTE

!--- You exclude a particular prefix with the
!--- use of

advertise-map

.

 !
 access-list 1 permit 160.10.0.0 0.0.255.255
 !
 route-map SELECT_SP_ROUTE permit 10
  match ip address 1
 !
 end
```

Now, look at the BGP table of Router C for 160.0.0.0/8:

```
<#root>

RouterC#

show ip bgp 160.0.0.0


BGP routing table entry for 160.0.0.0/8, version 15
Paths: (1 available, best #1)
  Advertised to non peer-group peers:
  2.2.2.2 4.4.4.4
  200, (aggregated by 300 2.2.2.1)
    0.0.0.0 from 0.0.0.0 (2.2.2.1)
      Origin IGP, localpref 100, weight 32768, valid, aggregated, local, atomic-
aggregate, best, ref 2
```

Only AS-200 is part of the AS_PATH information of the aggregate; AS-100 is not part of the information. Also, there is no inheritance of the community **no-export** from 160.20.0.0/16. Therefore, the aggregate route is announced to Router D. The **show ip bgp 160.0.0.0** output shows the announcement:

<#root>

```
RouterD#
```

show ip bgp 160.0.0.0

```
BGP routing table entry for 160.0.0.0/8, version 7
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Not advertised to any peer
  300 200, (aggregated by 300 4.4.4.1)
    4.4.4.1 from 4.4.4.1 (4.4.4.1)
      Origin IGP, localpref 100, valid, external, atomic-aggregate, best
ip bgp 160.0.0.0
```



Note: Because the aggregate as-set has AS-200 only, Router A in AS-100 accepts the aggregate route and installs the route in the routing table. The BGP loop detection mechanism causes this route acceptance. The BGP loop detection mechanism does not detect its own AS in as-set.

<#root>

```
RouterA#
```

show ip bgp

```
BGP table version is 3, local router ID is 160.20.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 10.160.0.0/8      2.2.2.1                                0 300 200 i
*> 10.160.20.0       0.0.0.0                  0         32768 i
```

Impact of the Use of suppress-map with Other Configuration Commands

The **aggregate-address** command includes other configuration commands, such as **suppress-map** . In order to understand the impact of the use of all the configuration commands in combination, note that **aggregate-address** only inherits the attributes from the more-specific routes when you use the **as-set** configuration command. Examples of the attributes that **aggregate-address** can inherit include **no-export** and **no-advertise .**

- When you use the **suppress-map** configuration command along with the **summary-only** configuration command, the **summary-only** configuration command does not have any effect. With use of the **suppress-**

map configuration command, the more-specific routes that the **suppress-map** suppresses are not advertised. However, the routes that the **suppress-map** does not cover are advertised in addition to the aggregated route. Therefore, the notes in this section apply to the use of **suppress-map** either with or without the **summary-only configuration** command.

- When you use **as-set** with **suppress-map** , although the suppressed routes are not advertised, the aggregated route inherits the attributes of all the suppressed routes. But you can override the inherited attributes with the use of other configuration commands, such as **attribute-map** . The Change the Attributes of the Aggregate Route section describes the use of **attribute-map** .

- When you use the **as-set** and **suppress-map** configuration commands with **advertise-map** , the aggregate forms. The aggregate inherits the attributes only out of the routes that are selected in the **advertise-map** , irrespective of whether **suppress-map** suppresses the route. See the Use advertise-map to Aggregate a Subset of Specific Routes section.

- When you use **advertise-map** and **attribute-map** along with **as-set** and other configuration commands, the **attribute-map** overrides the attributes that are chosen in the **advertise-map** .

In general, when you use **advertise-map** , only the **advertise-map** influences the aggregate. In the absence of **advertise-map** , the aggregate inherits the attributes of the more-specific routes, both suppressed and unsuppressed. In both the cases, you can use the **attribute-map** configuration command to override the chosen attributes.

Related Information

- **BGP: Frequently Asked Questions**
- **Troubleshooting BGP**
- **BGP Support Page**
- **Cisco Technical Support & Downloads**